

# Simulation and Analysis of Signal Cancellation in Cooperative Wireless Networks

DIPLOMARBEIT

zur Erlangung des akademischen Grades

**Diplom-Ingenieur**

im Rahmen des Studiums

**Computational Intelligence**

an der

**Technischen Universität Wien**

eingereicht von

**Roman Steiner**

Matrikelnummer 0326433

am Institut für Vernetzte und Eingebettete Systeme  
der Alpen-Adria Universität Klagenfurt

Betreuung:

Betreuer: Univ.-Prof. Dr.-Ing. Christian Bettstetter

Mitwirkung: Ass. DDipl.-Ing. Udo Schilcher

Klagenfurt, April 2009

\_\_\_\_\_  
(Unterschrift Verfasser/in)

\_\_\_\_\_  
(Unterschrift Betreuer/in)



## Acknowledgement

First of all, I want to thank Prof. Bettstetter for giving me the opportunity to write this master thesis at the University of Klagenfurt, which is not self-evident, since I am a master student at the Vienna University of Technology. Special thanks go to Udo Schilcher who leaded me well through my work. I am grateful for the relaxing atmosphere at the NES institute and for every single conversation I had there.

Finally, I want to thank everyone who gave me help, tips, distraction, patience, or simply company during my master thesis. Give credit where credit is due, just to mention a few and in no particular order: Walter for providing accommodation, Johannes for mathematical private lessons, Bettina for proof reading, all my friends in Vienna for not being mad on me for my absence, and finally of course my family.



## Abstract

Wireless networks get more and more popular. Wireless communication is applied in different fields ranging from computer networks, mobile phones to sensor networks. The reasons are the reduced installation and maintenance costs as well as the dramatic increase in flexibility.

Nevertheless, wireless networks do have drawbacks in terms of speed, delay and robustness in comparison to wired ones. The aim of the research community is to fight any of these issues with different approaches. One of the biggest challenges nowadays is the fast varying channel condition. Due to the nature of signal propagation, the channel quality can suddenly drop from excellent to unacceptable for a few milliseconds. Redundant transmissions on independent wireless channels reduce the probability of transmission failures. This is either achieved by multiple antennas or the cooperation of devices in virtual antenna arrays. As research work shows a high performance gain is obtained. Recently, the first products hit the market.

Main drawback of redundant relaying are additional transmissions. Obviously, these relay transmissions produce additional interferences at neighboring nodes. This leads to disturbances of their transmissions. The main idea of this project is to cancel out these additional interferences at the neighboring nodes. Therefore, the neighboring nodes encounter lower interferences and the successful transmission probability increases.

The project analyses this idea in a wide range of scenarios, ranging from the commonly known WLAN standard 802.11 to battery driven sensor networks. To obtain results the behavior is simulated in different scenarios, whereas the simulation tool itself is mathematically validated.



## Zusammenfassung

Funknetzwerke gewinnen mehr und mehr an Bedeutung. Das Anwendungsgebiet ist vielfältig und reicht von Computernetzwerken über Mobiltelefone bis hin zu Sensornetzwerken. Hauptgrund des Erfolges sind reduzierte Kosten der Installation und Erhaltung sowie höhere Flexibilität.

Allerdings haben Funknetzwerke auch Nachteile, vor allem im Vergleich mit traditionellen Kabelnetzwerken. Sowohl die Übertragungsgeschwindigkeit als auch Latenz und Robustheit sind geringer. Die Forschung versucht mit unterschiedlichen Ideen und Ansätzen diesen Nachteilen entgegen zu wirken. Eine der größten Herausforderungen ist im Moment die schnell ändernde Kanalqualität. Aufgrund der Übertragungseigenschaften von Funksignalen können plötzlich ausgezeichnete Verbindungen für einige Millisekunden zusammenbrechen. Redundante Übertragungen auf unabhängigen Kanälen können dabei die Fehlerwahrscheinlichkeit reduzieren. Dies kann sowohl durch mehrere Antennen, als auch durch Kooperation von Geräten in virtuellen Antennen-Arrays erreicht werden. Forschungsergebnisse aus diesem Bereiche zeigen bereits jetzt große Verbesserungen.

Hauptnachteil von redundantem Relaying sind zusätzliche Übertragungen. Diese Relay-Übertragungen produzieren zusätzliche Interferenzen bei benachbarten Knoten, was zu Störungen von deren Übertragungen führt. Die Grundidee dieser Arbeit ist das Herausrechnen dieser zusätzlichen Interferenzen. Die Interferenzen bei den benachbarten Knoten sind dadurch geringer und die Wahrscheinlichkeit einer erfolgreichen Übertragung steigt.

Diese Arbeit analysiert diese Idee in einer großen Auswahl von unterschiedlichen Szenarien. Diese reichen von dem häufig verwendeten WLAN Standard 802.11 bis hin zu batteriebetriebenen Sensornetzwerken. Die Resultate der unterschiedlichen Szenarien werden durch Simulationen ermittelt, wobei der Simulator selbst mathematisch validiert wird.





# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>State of the Art</b>	<b>5</b>
2.1	Wireless Networks . . . . .	5
2.2	Propagation Models . . . . .	6
2.2.1	Free Space Propagation Model . . . . .	7
2.2.2	Basic Propagation Effects . . . . .	8
2.2.3	Path Loss Models . . . . .	9
2.2.4	Multipath Fading . . . . .	10
2.3	Cooperative Diversity . . . . .	14
2.3.1	Relay Selection . . . . .	16
2.3.2	Relay Mode . . . . .	17
2.3.3	Diversity Combining Techniques . . . . .	18
2.4	Simulation Tools . . . . .	20
<b>3</b>	<b>The Project</b>	<b>23</b>
3.1	Description . . . . .	23
3.2	Assumptions . . . . .	24
3.2.1	Node Distribution and Propagation Model . . . . .	24

3.2.2	Traffic Model and Relay Selection . . . . .	25
3.2.3	Cooperative Diversity . . . . .	25
3.2.4	Signal Cancellation . . . . .	26
3.3	Scenarios . . . . .	26
<b>4</b>	<b>The Simulator</b>	<b>31</b>
4.1	General . . . . .	31
4.2	Data Structures . . . . .	33
4.3	Network Definition . . . . .	35
4.4	Transmissions . . . . .	35
4.5	Scheduler . . . . .	36
4.6	Network Layers . . . . .	39
4.6.1	General Description . . . . .	40
4.6.2	Application Layer . . . . .	40
4.6.3	Relay Layer . . . . .	40
4.6.4	Physical Layer . . . . .	42
4.7	Graphical Interface . . . . .	45
4.8	Configuration File . . . . .	47
4.9	Simplified Simulator . . . . .	50
<b>5</b>	<b>Mathematical Validation</b>	<b>53</b>
5.1	Statistical Operations . . . . .	55
5.1.1	Functions of One Random Variable . . . . .	55
5.1.2	Functions of Two Random Variables . . . . .	56
5.2	Probability Distributions of Signal Strengths . . . . .	57
5.2.1	PDF of Node Distances . . . . .	58
5.2.2	PDF of Signals Without Multipath Fading . . . . .	58
5.2.3	PDF of Signals With Multipath Fading . . . . .	59
5.2.4	PDF of Multiple Signals . . . . .	60
5.3	SINR Calculation . . . . .	62
5.4	Distribution of Interfering Transmissions . . . . .	65

5.5	Successful Transmission probability $p_s$ . . . . .	67
5.5.1	Scenario One . . . . .	67
5.5.2	Scenario Two . . . . .	69
<b>6</b>	<b>Results and Conclusion</b>	<b>73</b>
6.1	Results . . . . .	73
6.2	Conclusion . . . . .	76
6.3	Future Work . . . . .	79
	<b>List of Figures</b>	<b>81</b>
	<b>Bibliography</b>	<b>83</b>



## Introduction

People move. Networks don't.

---

Gast, Matthew

Wireless communication takes place in different disciplines. The best commonly known are wireless computer networks and mobile phones. But also sensor networks are more and more connected by wireless links. The success is mainly due to reduced cost and simplification in construction as well as maintenance and the great flexibility achieved. Both, wireless computer networks providing internet connection and the wide coverage of mobile phones influenced massively society in the recent years.

In comparison to wired networks, wireless networks often show a lack of speed, response time and robustness. In all three areas research work is undertaken and showed enormous improvements. Nevertheless, this research field still offers unanswered questions and large space for new ideas.

One of the biggest challenges nowadays are fast varying channel conditions. These fluctuations of signal strength are a result from a changing environment. Due to the nature of signal propagation radio waves propagate through the environment containing obstacles over different paths. At the receiver all signals are superimposed which either amplifies or weakens the signal strength. Because of these variations a generally good link can become unacceptable for a few milliseconds.

The solution to this challenge is the usage of redundant wireless links. The main idea is to spread the communication over independent wireless channels. This is achieved by the usage of multiple antennas on the devices.

The data is transmitted by two or more antennas and also received by two or more antennas (MIMO). If the distance between the antennas is large enough the channel variations are independent from each other.

To overcome the necessity of having multiple antennas, different devices are combined to a virtual antenna array. This approach is called *Cooperative Diversity* and one of the current hot topics. Already a lot of research work is carried out in this area and the results are promising.

However, the main drawback of this approach are the additional transmissions. As any other transmission they consume parts of the available bandwidth. On one hand, they increase the successful transmission probability, but on the other hand also the amount of inferences increases.

The main question of this project is how much a network would gain if these signals could be canceled out at the neighboring nodes.

In cooperative relaying diversity is achieved by using surrounding nodes as relay nodes. Relay nodes have the duty to repeat the data sent by the source node. Since the repetition is more or less an exact copy of the source transmission, the content of this rebroadcast is known by other nodes if they have received them. Based on this knowledge it is assumed that it is possible to cancel out the additional relay transmission.

The project simulates and analyses this idea. The behavior is observed in a wide range of simulation scenarios. There is no explicit focus on any particular network scenario. Because of the lack of suitable simulation tools a new one is developed. The main results are obtained from the simulation, whereas the simulation tool is mathematically validated.

The rest of the book is organized as follows:

*Chapter 2* presents the state of the art. In the beginning a brief overview of wireless networks is given, followed by a detailed description of radio wave propagation and propagation models. Next, cooperative diversity is presented and the main ideas are discussed. The chapter ends with an introduction to currently available simulations tools and their pros and cons.

*Chapter 3* gives an introduction to the project and describes its ideas. A detailed presentation of the scope and aim of the project is given. All assumptions, which are taken in the project, are summarized and argued for. Further, the range of scenarios is presented. All necessary parameters are presented and discussed.

*Chapter 4* presents the simulation tool used. The self developed simulator is explained in full detail. All modules and their behaviors are shown. *Chapter 5* validates the previously presented simulations tool. The success transmission probability is deduced step by step and finally compared to the simulation tool. Finally, the results and the conclusion are presented in the last chapter.





## State of the Art

The economic and technological triumphs of the past few years have not solved as many problems as we thought they would, and, in fact, have brought us new problems we did not foresee.

---

Henry Ford II

This chapter gives a brief overview of the state of the art for all related topics. First some general facts about wireless networks are mentioned. The next section analyses signal propagation models in more detail, followed by the introduction of cooperative diversity. At the end of the chapter current developments of simulation tools and their limitations are presented.

### 2.1 Wireless Networks

Wireless networks differ from wired networks in many aspects. Most design principals and properties for wired networks cannot be applied to wireless. The most obvious difference is the transmission medium. Whereas in wired networks the signals are transmitted in copper or fiber optic, in wireless networks the data is transmitted by electromagnetic (EM) waves. If the world is assumed to be flat free space environment propagation models are relatively easy to handle. As soon as it comes to more realistic environments signal dispersion becomes more and more complex. Diffraction, scattering and reflection make it difficult to predict exact signal strengths. Moreover wireless channel characteristics are time variant since it cannot be assumed that the

environment stands still. Even the flicker of leaves in trees changes the paths of signals. The signal strength at the destination changes in milliseconds without devices moving, purely due to changes in the environment.

Unlike in wired networks, the available resources are limited. While more bandwidth can be acquired in wired networks by running more cables, this approach is not suitable in wireless networks. Additional bandwidth can be acquired by using different frequencies or, more important, higher data rates can be achieved by proceedings in technology.

## 2.2 Propagation Models

An accurate prediction of signal strength at the receiver is a very challenging task. In most cases radio propagation is far too complex to give exact results. Only in a few cases, where a line-of-sight exists and no surrounding obstacles are present a precise prediction is possible. This is e.g. satisfied in satellite communication.

In almost all other cases, radio propagation is influenced by surrounding obstacles like mountains, buildings and other smaller objects. Figure 2.1 shows the three most important propagation mechanisms, which are reflection, diffraction and scattering. A detailed description is given in section 2.2.2.

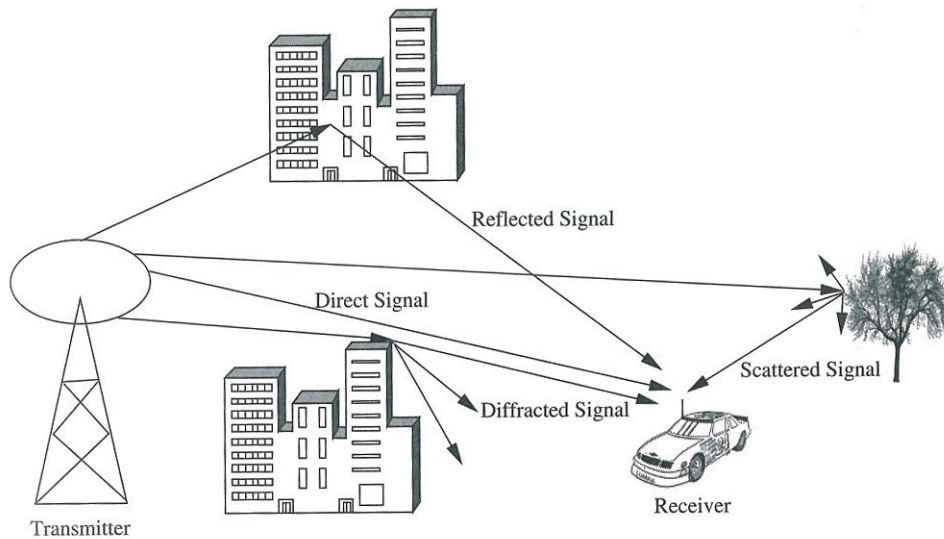


Figure 2.1: The three basic propagation mechanisms, [1] page 31.

In theory it is possible to exactly predict the signal strength at the receiver, but only under the assumption that an exact model of the environment exists. Since this is not accomplishable, statistical models are used.

Traditional propagation models predict an average signal strength at a certain location. The average is typically taken over a few (5-10) wavelengths. These types of models are categorized under the term *large-scale models*. The other category, *small-scale models*, contains all models describing instantaneous signal strengths. Instantaneous signal strengths can show spikes up to 30-40dB or even more. The fluctuation comes from changes in the signal paths in the dimension of less than a wavelength. This is a result from movements of the sender and/or the receiver. Moreover a changing environment produces similar effects.

### 2.2.1 Free Space Propagation Model

In the free space propagation the simplest possible scenario is assumed: a free space without any obstacles in reach. Hence there is a line of sight between the antennas, which is the only propagation path from the sender that reaches the receiver. This scenario can e.g. be applied to satellite communication. The signal strength at the receiver is given by Friis' equation

$$P_r(d) = \frac{P_t G_t G_r \lambda^2}{(4\pi)^2 d^2 L} \quad (2.1)$$

with the following parameters

$d$	...	distance between sender and receiver
$P_r$	...	receiving power
$P_t$	...	sending power
$G_t$	...	transmit antenna gain
$G_r$	...	receive antenna gain
$\lambda$	...	wave length
$L$	...	system loss factor

$G_t$  and  $G_r$  are the gains of the antennas at the sender respectively receiver in comparison to a hypothetical ideal antenna that radiates equally in all directions (isotropically) and has no losses.  $L$  is the factor which defines the loss in the system (e.g. cables, connectors).

Friis' equation is only valid in the far-field (Fraunhofer region). Unfortunately the border of this region is only vaguely defined. The distance  $d$  has to be greater than  $d_f$  and the inequations 2.2 and 2.3 must be satisfied [1].

$$d_f \gg \frac{2D^2}{\lambda} \quad (2.2)$$

$$d_f \gg \lambda \quad (2.3)$$

$D$  is the maximum overall dimension of the antenna. Since it can safely be assumed that the antennas are at least a few meters apart, the transmission always takes place in the far-field.

### 2.2.2 Basic Propagation Effects

The propagation mechanisms describe how the signals change during propagation. While in free space the propagation is a straight line, it gets more difficult when hitting obstacles. Depending on how and which obstacles are hit, the signal propagation can be described by the following three mechanisms.

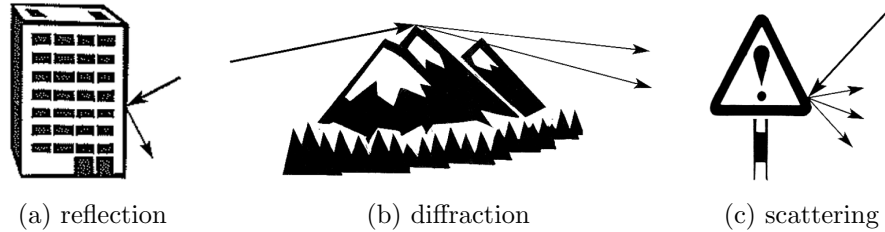


Figure 2.2: The three basic propagation mechanisms, [2] page 38f

**Reflection** occurs when the radio wave hits objects whose dimensions are large in comparison to the wavelength. Obstacles like buildings and the earth result in massive reflections. Therefore the signal not only propagates on a straight line, but also in a zig-zag line depending on the environment. (Figure 2.2a).

**Diffraction** occurs when the radio path between sender and receiver is obstructed by a surface that has sharp irregularities (edges). The

secondary waves resulting from the obstructing surface are present throughout the space and even behind the obstacle, giving rise to bending of waves around the obstacle (Figure 2.2b).

**Scattering** is similar to diffraction as described above. The difference is that the size of the object is comparable to or smaller than the wavelength. Therefore the radio waves are more scattered into different directions. Scattering is the most difficult mechanism to predict (Figure 2.2c).

**Doppler Effect** is observed if sender and / or receiver are moving relatively to each other. If the devices move towards each other, the frequency at the receiver increases. A lower frequency is received, if devices separate from each other and the distance between them increases. The change of the frequency is expressed by

$$\Delta f = \frac{v}{c} f_0 \quad (2.4)$$

where  $f_0$  is the emitted frequency at the sender,  $v$  the velocity with which the sender and receiver move towards each other, and  $c$  is the speed of light. If the devices separate from each other,  $\Delta f$  is negative and the received frequency is lower than the emitted.

### 2.2.3 Path Loss Models

Path loss models are large-scale models. They predict average signal strengths at certain locations. Nevertheless, they try to capture all propagation mechanisms described above. Due to the complexity lots of empirical studies are carried out [3, 4, 5]. All studies focus on an individual scenario and try to examine this in deep detail. The generalization is a challenging task and loses lots of precision. The presented models always try to fit as well as possible to the empirical data. Thus, the models are a combination of analytical and empirical methods.

**Log-distance path loss model** This model is also known under the term *simple path loss model* and given by

$$p_r(d) = p_0 \left( \frac{d}{d_0} \right)^{-n}. \quad (2.5)$$

$p_r(d)$  denotes the signal strength at the receiver at a distance  $d$ .  $p_0$  is the signal strength at a reference distance  $d_0$ . This signal strength can either be determined empirically or by Friis equation (Equation 2.1). Depending on the scenario typical reference distances are 1m, 100m or 1km. The path loss exponent  $n$  defines how fast the signal strength decreases with distance. The parameter mainly depends on the environment but also on the frequency. The lowest suitable value is 2 for free space. Values below 2 are only achieved with waveguide effects. For outdoor, the typical values are between 2-3, and 3-4 for indoor [1, 6, 3].

**Log-normal shadowing** This model is similar to the log-distance path loss model, but also takes into account shadowing. Shadowing is approximated by a zero-mean gaussian random variable with standard derivative  $\sigma$ . The standard derivative  $\sigma$  defines how much shadowing influences the average signal strength. In [7] Seidel collected empirical data for four european cities. In his scenarios the path loss exponent ranges from  $n = 2.5$  to  $n = 3$  and  $\sigma = 8$  to  $\sigma = 13$ .

$$p_r(d) = p_0 \left( \frac{d}{d_0} \right)^{-n} X_\sigma \quad \text{with } X_\sigma \log N \quad (2.6)$$

## 2.2.4 Multipath Fading

The path loss models as described in the previous section, define average signal strengths. This section, in contrast, deals with instantaneous signal strengths which show higher variations. The fluctuation of the signal strength is due to the same reasons as described in section 2.2.2, namely reflection, diffraction, scattering and doppler effect.

As already seen in Figure 2.1, radio waves propagate from the source over different paths to the receiver. The paths are influenced by all objects in the environment. All radio waves reaching the receiver are superimposed. The received power is then the sum of all individual signals. Due to different path lengths each signal arrives with a different phase shift. Superimposed signals may amplify or completely destroy each other. Figure 2.3 shows two extreme cases where two signals are in phase and out of phase. In the second case the signal is completely destroyed.

The phenomenon of receiving multiple copies of the same signal is called *time spreading*. If the time spread gets large in comparison to the symbol

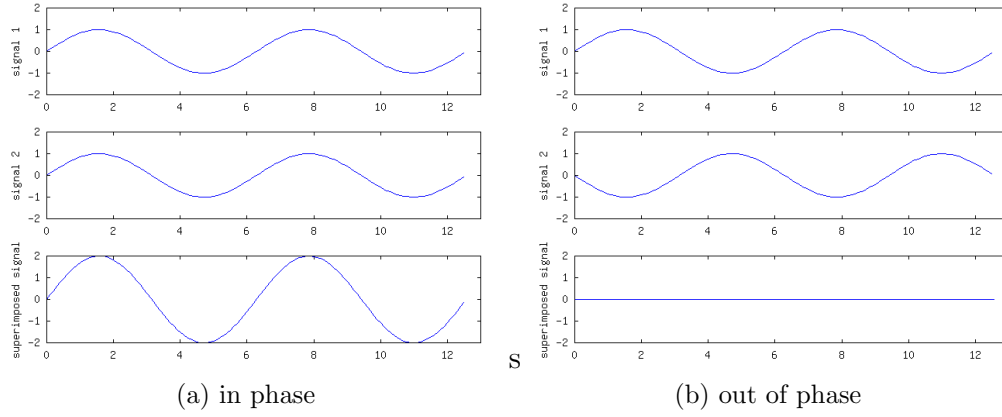


Figure 2.3: Superimposed signals

duration, received symbols are overlapping at the receiver. This is called symbol interferences and causes irreversible damages to the signal.

Since it has to be assumed that the environment is constantly changing, also the propagation paths are changing. An altering in the path length has the same effects as if the devices are moving. Therefore the frequency at the receiver changes according to the doppler effect. This phenomenon applies to each signal path individually. So different paths exhibit different frequency changes which leads to a broadening of the frequency range (frequency spreading).

Assuming there is no line-of-sight or other dominant path, the electrical field at the receiver is given by

$$E_z = E_0 \sum_{n=1}^N C_n \cos(2\pi f_c t + \theta_n), \quad (2.7)$$

where  $E_0$  is the real amplitude of the local average electrical field which is assumed to be constant and  $f_c$  is the carrier frequency.  $C_n$  is a real random variable that denotes the amplitude of the  $n$ th signal under the constraint  $\sum \overline{C_n^2} = 1$ . The random phase of the  $n$ th received signal  $\theta_n$  is given by

$$\theta_n = 2\pi f_n t + \phi_n \quad (2.8)$$

due to the doppler shift and different path lengths.

Based on the analysis by Rice [8] the electrical field can be expressed in

an in-phase and quadrature form

$$E_z = T_c(t) \cos(2\pi f_c t) - T_s(t) \sin(2\pi f_c t) \quad (2.9)$$

where

$$T_c(t) = E_0 \sum_{n=1}^N C_n \cos(2\pi f_n t + \phi_n) \quad (2.10)$$

and

$$T_s(t) = E_0 \sum_{n=1}^N C_n \sin(2\pi f_n t + \phi_n). \quad (2.11)$$

In general it can be assumed that the doppler shift is small in comparison to the carrier frequency  $f_c$ . If the number of signals  $N$  is sufficiently large,  $T_c(t)$  and  $T_s(t)$  can be approximated by Gaussian random processes. Let  $T_c$  and  $T_s$  denote the Gaussian random variables. They are uncorrelated with zero-mean and equal variance given by

$$\overline{T_c^2} = \overline{T_s^2} = \overline{|E_c|^2} = E_0^2/2. \quad (2.12)$$

The envelope of the received electric field,  $E_z(t)$  is given by

$$|E_z(t)| = \sqrt{T_c^2(t) + T_s^2(t)} = r(t). \quad (2.13)$$

Since  $T_c$  and  $T_s$  are Gaussian random variables, it can be shown using the Jacobian transformation that  $E_z(t)$  follows the Rayleigh distribution (Figure 2.4)

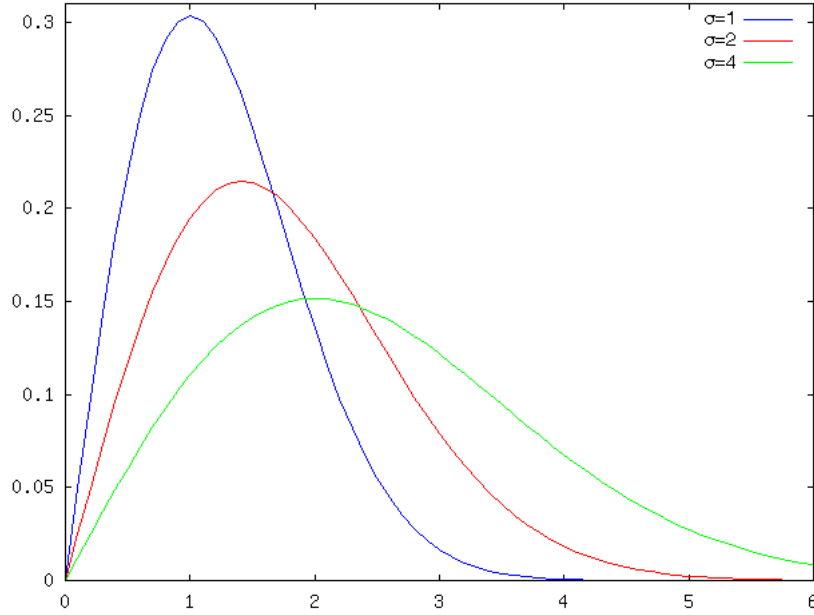
$$p(r) = \frac{r}{2\sigma^2} e^{-\frac{r^2}{2\sigma^2}}, \quad r \in [0, \infty), \quad (2.14)$$

with  $\sigma^2 = E_0^2/2$ .

Note that the rayleigh distribution gives the distribution of the electrical field which also equals the distribution of the received voltage, but not the signal strength. The signal strength is determined by  $v^2/2$  where  $v$  is the received voltage. The distribution of signal strengths due to multipath fading becomes an exponential distribution (Equation 2.15). Further details on distributions and distribution evaluations are presented in Chapter 5.

$$f_{Ex}(x) = \lambda e^{-\lambda x} \quad x \in [0, \infty) \quad (2.15)$$



Figure 2.4: Rayleigh distribution for different  $\sigma$ 

The rate parameter  $\lambda$  is defined with  $\sigma^2$  and therefore the average received power. The mean value of the exponential distribution is  $\sigma^2$ . So on average multipath fading does not impact the signal strength.

As shown in Figure 2.5 the instantaneous signal strength can drop easily by 30-40 dB. In these cases a transmission most likely fails. The rate of the signal changes is called the *channel coherence time*. This rate mainly depends on changes in the environment as well as movements of the receiver and/or sender. Measurements of the channel coherence time in an indoor environment are carried out in [9, 10]. Both papers conclude that the channel coherence time averages approximately 100ms. Most research papers assume that the channel does not change while transmitting a single data packet. For the next transmission an independent multipath fading factor is assumed.

In scenarios where a dominant path between the sender and the receiver exists, this model is inaccurate. In these cases the fluctuations are approximated by the *rice distribution* (Equation 2.16). The factor  $K$  depends on the environment and can take values ranging from 2 to 10dB,  $I_0(z)$  is the Bessel function. [3]

$$f(x) = \frac{x}{\sigma^2} e^{-\frac{x^2 + K^2}{2\sigma^2}} I_0\left(\frac{xK}{\sigma^2}\right) \quad (2.16)$$

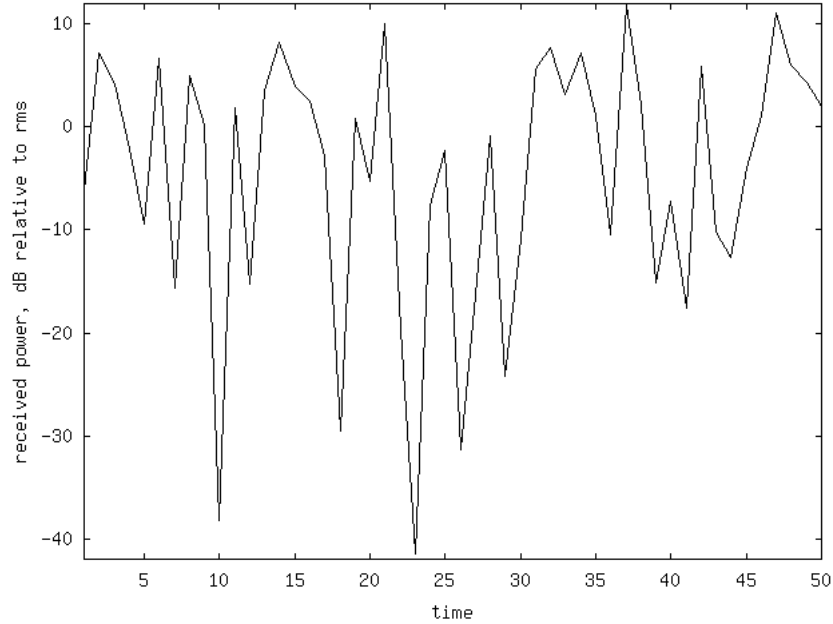


Figure 2.5: Multipath fading

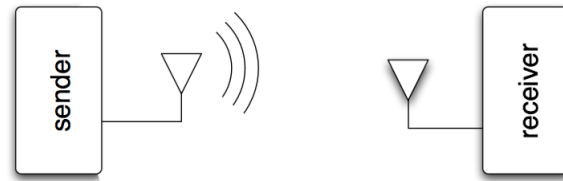
## 2.3 Cooperative Diversity

Multiple-input multiple-output (MIMO) systems have multiple antennas on the sender and the receiver (Figure 2.6b). Because of the spatial distribution of the antennas spatial multiplexing is introduced. This approach is widely acknowledged to give system improvements. It mainly mitigates the negative effect of small scale fading.

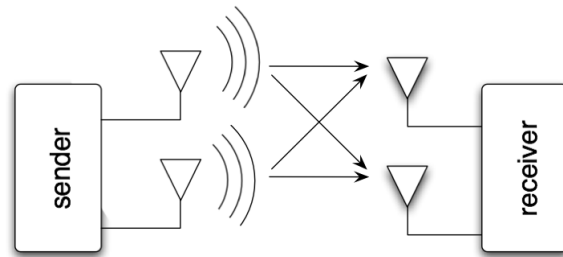
Main disadvantage is the increase in size and power consumption of devices. The distance between two antennas should at least be the size of one wave length. For common 2.4GHz transmissions this would be approx. 12.5 cm. This may not be any problem at stationary devices, but produces significant challenges in mobile ones. If it comes down to tiny devices like in sensor networks, this approach is not practical at all. Both size and power consumption have to be kept as low as possible.

Cooperative diversity tries to overcome this limitation and is a generic term for devices cooperation to achieve diversity. Instead of using multiple antennas on one device, devices *share* their antennas. In this way a virtual antenna array is generated [11, 12].

Figure 2.7 shows two mobile devices, which want to send data to the same



(a) single-input single-output (SISO)



(b) multiple-input multiple-output (MIMO)

Figure 2.6: Difference of SISO and MIMO

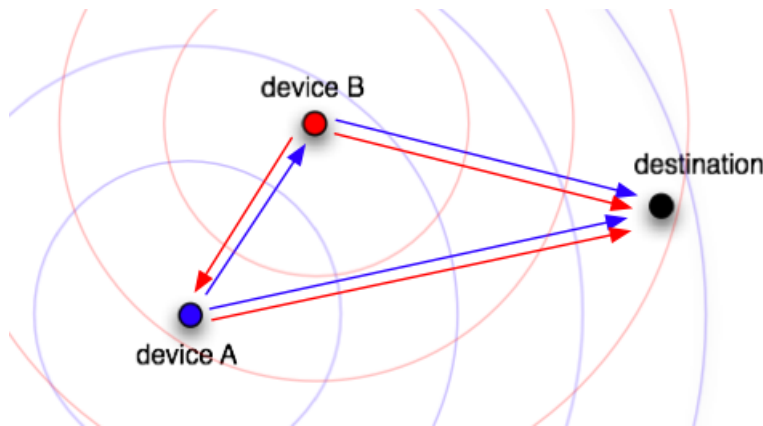


Figure 2.7: Cooperative Diversity

destination (e.g., two mobile phones transmitting data to the base station). It is assumed that all devices are within transmission range and can therefore hear each other, at least most of the time. Because of small scale fading the quality of one channel may accidentally drop in between. In this event a successful transmission is not possible any more. Since the devices are far enough away (more than a wavelength), the channels are uncorrelated.

Device A broadcasts its data (in blue), which is received at the destination D as well as at device B. Now device B can act as antenna in the virtual antenna array and rebroadcast the same data. The destination D receives the data twice and merges both transmissions. In case of a bad channel between node A and D the data is transmitted through the relay B. The event that channel A-D and one of the channels A-B and B-D have bad quality is lower than having bad quality on channel A-D. If device B sends data (in red) to the destination, device A acts as a relay. The procedure is the same with exchanged roles of device A and B.

How and when the relay device is selected depends on which method is used. Also the merging of the two packets can be done differently. Some methods are described in the following sections. Especially relay selection is a hot topic in this research area and still offers open research questions.

### 2.3.1 Relay Selection

Relay selection describes the method of selecting a relay node for a transmission from the source to the destination. Relay selection plays a major role in the final performance of cooperative diversity [13, 14]. Choosing a useless relay node produces only extra interferences without gaining any advantages at the destination node.

A simple selection algorithm just takes a single hop transmission into account. Unfortunately, most of these approaches cannot be easily extended to multi hop transmission, because the selection of the relay massively influences the next transmission and their relay selection [15]. In sensor networks, where data is just transmitted to the next hop (or a few hops at most), simple algorithms may be good enough. For larger networks where data packets are transmitted on long paths through the network, more advanced algorithms give significant advantages. [16, 17]

In general it is possible to use more than one relay node. Zhao et al. showed in their work[18] that taking the 'best' relay is sufficient. Already with one (the best) relay node full diversity is achieved. This result is interesting, as it paves the way for researchers on assigning no more than one relay node.

Relay selection algorithms can be categorized in many ways. On distinction is in pre-selection and on-demand selection.

**Pre-selected** The relay is selected before the transmission is initiated by the source. So all attending nodes know their roles. There is only

one relay node which has to listen to the transmission and buffer the data. All other surrounding nodes can go into a standby mode. If the transmission to the destination fails, the relay node immediately starts its retransmission.

A drawback of this method is that a relay node has always to be selected. In cases where the transmission from the source to the destination is successful anyway, the selection is unnecessary. The main design parameter of such algorithms is how often the relay node is selected. The selection of a relay node for a source-destination pair can be used for multiple transmissions. If the selection of the relay is buffered too long, the wireless channel to and from the relay node may change. Depending on that, the relay may not be suitable anymore.

**On-demand selection** In contrast to pre-selected relays, the relay is selected on demand. So only if the transmission from the source to the destination fails a relay is selected which rebroadcasts the data. This avoids the relay selection in case of successful transmissions. Moreover the current wireless channels to and from the relay can be easily considered in the relay selection algorithm. Therefore it is easier to select the best relay at this point of time.

Drawback of this method is that the retransmission is delayed, because the relay has to be selected first. Depending on the speed of the selection algorithm this results in some delay. Another major drawback is that all possible relays have to listen to the transmission and buffer the data. In case of a failed transmission, each of them may have to rebroadcast the transmission. This may not be any problem for stationary devices, but can show negative effects on battery driven ones.

### 2.3.2 Relay Mode

Independent of how the relay is selected, the behavior of the relay node can be different. The behavior is categorized in two different modes. The simplest way is *Amplify-and-Forward* (AF), a more complex is *Decode-and-Forward* (DF).

**Amplify-and-Forward (AF)** As the name implies, the signal is amplified at the relay device and retransmitted. The relay device does not take any decisions on the relay signal. The signal is amplified as far as the

general power constraints of the relay device allow it. Main drawback is that with the signal amplification also the noise is amplified. The final decision takes place at the destination device. There the two signals are combined together and decoded afterwards. Although noise is amplified by cooperation, the destination device receives two independently faded versions of the signal and can make better decisions on the detection of information.

To make the best decisions at the destination device, it is necessary that the channel conditions between the source and the relay are known to the destination. This information has to be somehow propagated by the cooperation protocol. It is needed to merge the signals at the destination in an appropriate way.

**Decode-and-Forward (DF)** In contrast to the amplify-and-forward method the relay node decodes the data received from the source. The relay may even change the coding to more powerful codes. Decode-and-forward heavily depends on the channel conditions between the source and the relay. If the data cannot be decoded, no retransmission is possible. On the other hand, decoding the data has the advantage that noise is not amplified and propagated to the destination.

**Hybrid** To avoid error propagation, Laneman et al. [19] proposed a hybrid mode of amplify-and-forward and decode-and-forward. If the channel condition between the source and the relay is good enough, decode-and-forward is used. In the opposite case, the relay device switches to amplify-and-forward. Although this sounds as a promising approach, Meng Yu et al. show in [20] that it is not worth the effort. In practical scenarios the performance of Amplify-and-Forward and Decode-and-Forward are similar. A hybrid mode does not gain significant improvements.

### 2.3.3 Diversity Combining Techniques

Diversity combining is a technique which merges multi-branched signals [21]. Depending on the complexity of the receiver there exist different techniques. The simplest ones are *scanning combining* and *selection combining*. Both are easy to implement, but the resulting SNR is not greater than of any of the received signals. *Maximum-ratio combining* and *equal-gain diversity* combine the signals and produce an output which is better than each of the inputs.

All of them are linear diversity combining techniques and can be described with

$$f(t) = \sum_{j=1}^N a_j f_j(t), \quad (2.17)$$

where  $f_j(t)$  is the signal from the  $j$ th source, the result is  $f(t)$ . Depending on the technique the coefficients  $a_j$  are determined differently.

**Scanning Diversity** This is a switched technique, meaning only one  $a_j$  is greater than zero of any time. A selector device scans all channels in a predefined sequence. If the signal quality is greater than a threshold, the signal is selected ( $a_j = 1$ ). If the signal falls below a certain threshold the scanning is resumed until an appropriate signal is found again.

**Selection Diversity** This is a more sophisticated switched technique. At any time, the signal with the best quality is chosen. Let  $k$  denote the signal with  $r_k \geq r_j, \forall j = 1, 2, \dots, N$  where  $r_j$  is the quality of the  $j$ th signal.

$$a_j = \begin{cases} 1 & \text{for } k = j \\ 0 & \text{otherwise} \end{cases} \quad (2.18)$$

**Equal-Gain Diversity** In comparison to the above techniques, equal-gain diversity takes all received signals into account. The signals are equally weighted and then summed together. The signal quality is not considered and bad quality signals may destroy strong ones. Nevertheless the performance is still better than with selection diversity.

$$a_j = 1 \quad \forall j = 1, 2, \dots, N. \quad (2.19)$$

**Maximum-Ratio Diversity [22] [23]** The best performance is achieved by maximum-ratio diversity. As in the equal-gain diversity all signals are considered, but they are weighted according to their quality. Better signals get a higher weight than bad ones. The resulting signal quality  $r$  is given by

$$r = \sum_{j=1}^N r_j. \quad (2.20)$$

Again  $r_j$  defines the signal quality of the  $j$ th signal. The coefficients  $a_j$  in Equation 2.17 are set to

$$a_j = \frac{x_j}{n_j}, \quad (2.21)$$

where  $x_j$  is the signal strength of the  $j$ th signal,  $n_j$  denotes the average noise.

Figure 2.8 shows a comparison of the combining techniques presented above. The plot shows the gain in terms of SNR over the number of signals. Taking more signals into account, the gain increases. It can be seen that maximum-ratio diversity clearly outperforms all other selection techniques.

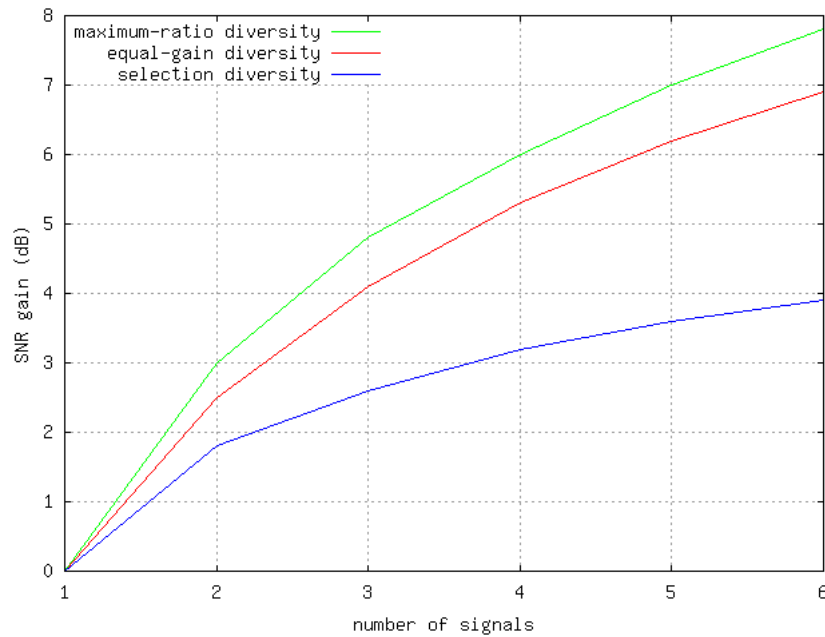


Figure 2.8: SNR gain of different diversity techniques

## 2.4 Simulation Tools

The number of simulation tools for wireless networks increased rapidly in the last years. Some older network simulators for wired networks have got extensions for wireless communication (like NS-2 [24]), while others are developed from scratch.



The following list enumerates a few known and widely used simulation tools. The list is without any specific order and is not complete.

**NS-2** is a discrete event network simulator based on the REAL network simulator [25] and was initially intended for wired networks. Later on the Monarch Group extended NS-2 to support wireless networking such as MANET and wireless LANs [26].

Programming in NS-2 can become a challenging task due to the lack of documentation and consistency in the code base [27]. The learning curve is steep and debugging difficult due to the system architecture. Moreover the memory footprint gets large and it lacks scalability. Simulations of more than a few hundred nodes require huge system resources.

**OmNet++** [28] is not a wireless network simulator, but provides a general framework for discrete event simulations. It can be used to model communication networks, multiprocessors and other distributed or parallel systems.

The TU Berlin designed a Mobility Framework [29] which provides a solid foundation for creating wireless and mobile networks with OM-NeT++. It provides a detailed radio model, several mobility models, MAC models including IEEE 802.11b, and several other components [30].

**GloMoSim** (Global Mobile Information Systems Simulation Library) [31] is a scalable simulation environment for wireless and wired network systems. At the current state, only wireless communication models are implemented. According to the webpage, simulation of hybrid networks (wired and wireless) should be possible in the future. The simulator is free for academic use only. A download of the source-code is only possible while accessing from \*.edu domain. There is also a commercial version based on GloMoSim available, which is called QualNet [32].

**jProwler** [33] is a simulator for communication protocols based on TinyOS. TinyOS is an open source operating system specially designed for wireless embedded sensor networks. jProwler is optimal for prototyping, verifying and analyzing TinyOS applications. Due to different programming languages, it is not possible to share the same source code between TinyOS and jProwler. jProwler is written in Java and highly

optimized for speed. It is able to run simulations up to 5000 nodes in real time. The system architecture is kept simple and straight forward. The simulator comes with a build in Rayleigh fading model and an example implementation of Mica2 nodes.

There are numerous papers analyzing the capabilities and results of simulation tools. Unfortunately, the outcomes of simulation tools differ quite a lot. [27] shows that even a simple flooding algorithm produces different results in different simulation tools.

The main problem is that different tools use different levels of abstraction. In general, the level of abstraction depends on what to simulate. A simulation of higher protocols (e.g. at the application layer) does not need full details in the lower protocols. Nevertheless, simplifications of lower protocols have to be handled with care. The impact of noise and multipath fading shows significant implications on higher protocols, as shown in the papers [34, 35].

Most of the above mentioned examples of simulation tools focus mainly on higher layers. More advanced models for the physical as well as the data link layer are missing.

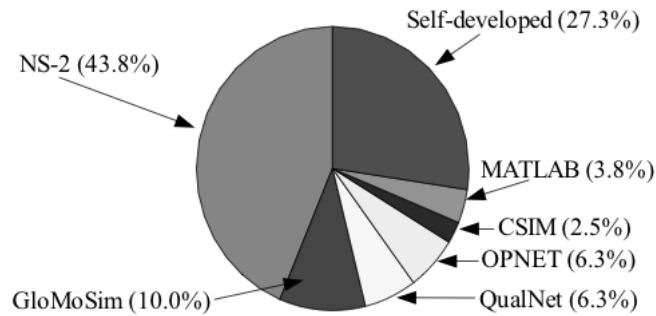


Figure 2.9: Used simulations tools [36]

While comparing simulation tools, NS-2 suffers in most areas. Still, it is mostly used in the research community in the time from 2001 to 2005 (Figure 2.9), followed by self developed ones. Self developed simulations tools have the drawback that comparing the results with other studies is difficult. In general the studies lack reproducibility due to the missing source code. [36]

## The Project

Problems worthy of attack prove  
their worth by hitting back.

---

Piet Hein

### 3.1 Description

Cooperative diversity fights fading induced by multipath propagation in wireless networks. One of the main drawbacks of cooperative diversity is the additionally produced interference. The aim of the project is to show a way how to deal with this additional interference. The basic assumption is that it is possible to cancel out the interference produced by an a priori known transmission. Since relay transmissions are rebroadcasts of the original message, they are usually known beforehand.

The following simplified situation (Figure 3.1) illustrates these ideas.

Suppose node  $S1$  starts sending a packet. All other nodes are within the transmission radius and therefore receive the packet. Node  $R1$  acts as a relay node for the transmission  $S1 - D1$  and repeats the packets. If node  $S2$  starts transmitting its own packet while node  $R1$  is repeating the previous message, there will be a collision at node  $D2$ . Since node  $D2$  exactly knows what node  $R1$  sends, it can cancel  $S2$ 's packet out. The packet from node  $S2$  is therefore successfully received at destination  $D2$ .

The main question of the project is how much a system gains if relay signals are recognized and these signals are canceled out.

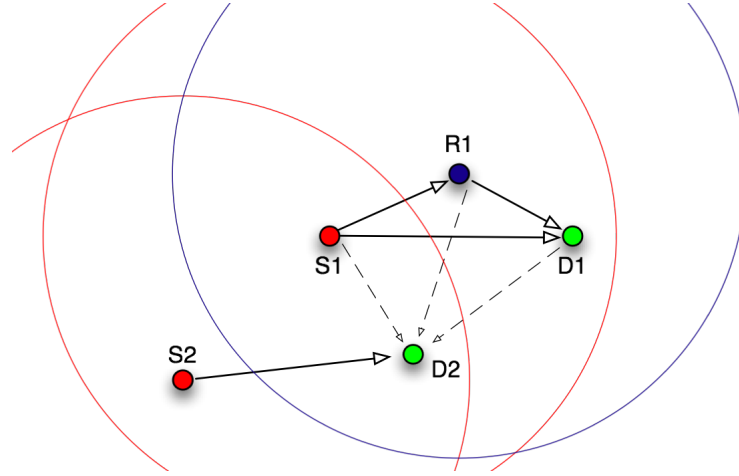


Figure 3.1: Basic example

It is not the aim of the project to perform any real signal cancelation. One of the basic assumptions of the project is that signal cancelation can be performed, no matter how. It is only assumed that transmissions from a relay node can be canceled by a certain factor. This work is mainly considered to be a study whether signal cancelation in this context shows effects or not.

## 3.2 Assumptions

### 3.2.1 Node Distribution and Propagation Model

For the node distribution the most simple model is selected. The nodes are placed by a Poisson process in a 2D infinite area. Since for simulation this is not a practical approach, a large finite area is used instead. To avoid border effects the edges are wrapped around. The area can therefore be seen as a torus which behaves similar to an infinite area. Further a high node density is assumed to avoid the node placement taking influence on the statistics.

The whole project deals with cooperative diversity fighting multipath fading. Therefore, it is absolutely necessary to have multipath fading in the propagation model. In general it is assumed that there is no dominant signal path between transmitter and receiver. Therefore, multipath fading is statistically approximated by the *Rayleigh distribution* as described in 2.2.4.

Since cooperative diversity does not deal with large scale propagation effects, the path loss model is kept simple and the *log-distance path loss*

*model* is used (section 2.2.3). A reference distance of  $d_0 = 1$  is used with the signal strength  $p_0 = p_r(d_0)$  derived by the Friis' equation. It is assumed that there is neither an antenna gain at the sender nor of the receiver and the system loss factor is  $L = 1$ . The wavelength is set to 125mm which is equivalent to a frequency of approx. 2.4GHz, which is a common frequency.

### 3.2.2 Traffic Model and Relay Selection

A simple single hop traffic model is used. All nodes can act as source, relay or destination all the time. The source node is randomly chosen and the destination is a node within transmission range. The transmission range is defined as the area where on average (without multipath fading) a successful transmission is possible. The relay node is the node where the sum of the distances from the source to the relay and from the relay to the destination is shortest. A high node density ensures that the relay is within transmission range of source and destination. This simple approach is chosen in order to avoid any influence of the relay selection algorithm on the results.

The transmission duration is fixed to one time unit. The transmission start times are uniformly distributed in time. Like the area, also the time is seen as infinite. In the simulation this is approximated by a long enough simulation duration. A simulation is long enough when the simulation time is much longer than the duration of one transmission. The simulation duration has to be extended until the results lie in a certain confidence interval.

### 3.2.3 Cooperative Diversity

Relay selection for cooperative diversity is done a priori and described in the above section. Since the relay is within transmission range it should receive the **DIRECT** transmission on average. In cases where the data packet is not successfully received, the **RELAY** transmission is dropped. There is no retransmission from the source or any changes in the source transmission (e.g. more powerful transmission codes).

The relay operates in the decode-and-forward mode. Hence, each message is fully decoded at the relay node. If the data packet could not be decoded successfully, the relay skips the rebroadcast. Therefore, the probability that the relay node rebroadcasts is equal to the transmission probability within

transmission range. Like the source, the relay also does not check if the transmission channel is unoccupied.

The destination waits until both data packets are received or the relay message is timed out. In the case both data packets are received, they are merged. As combination technique maximum-ratio combining is used. This technique produces the best results and is currently the de-facto standard in the research area of cooperative diversity.

### 3.2.4 Signal Cancellation

This work does not investigate any particular signal cancelation method. It is assumed that cancelation can be done under certain circumstances. If the **DIRECT** transmission (initiated by the source node) was received beforehand, the **RELAY** transmission (initiated by the relay node) can be canceled out with a predefined factor.

The basic idea of signal cancelation is based on the maximum-ratio combining technique. As described in section 2.3.3 this technique combines multiple input signals. Consider the example shown in Figure 3.1: For node **D2** there is an interfering transmission from node **S1** to node **D1** with the relay node **R1**. While **S1** broadcasts, node **D2** records and stores the transmission in its buffer  $f_{S1}(t)$ . When node **R1** starts rebroadcasting the message, node **D2** combines its input signals with  $-f_{S1}(t)$ .

Under ideal conditions the whole transmission from **R1** would disappear. Since this is not reasonable a cancelation factor is introduced. Setting the cancelation factor to 1 equals full interference cancelation, setting it to zero performs no cancelation at all.

In this work, the cancelation factor is treated as a parameter. An exact derivation of the cancelation factor is outside the scope of this work.

## 3.3 Scenarios

The aim of the chosen scenarios is to cover a wide range of scenarios to identify different outcomes of signal cancellations. Nevertheless, special interest lies in scenarios that are covered by the 802.11 wireless network standard as well as wireless sensor networks. For wireless sensor networks, ideas from Mica2 [37] are taken. Main reasons for picking them is their wide acceptance.

Since signal cancelation acts on the physical layer outcomes of the simulations are not specific for individual wireless standards. It is completely unimportant which network protocol acts on higher layers and has almost no influence. Moreover, the simulation scenarios are kept simple for better understanding and analysis. The simulation of individual behavior of signal cancelation of a special wireless network protocol is out of scope.

The scenarios have a range of parameters. Most of the parameters are dependent on each other. Basically the list of parameters is split up into two categories. The first describes a set of parameters that does not change throughout different simulation scenarios. Changing them does not change the outcome or their influence is minimal and therefore negligible. Some parameters can also be expressed by others. The second list covers simulation parameters which lead to different simulation results when changed.

**Node density** is measured in nodes per unit area. For significant simulation results at least a minimum node density is necessary. If insufficient, too many transmissions take place at the same nodes and therefore do not result in independent trails. Moreover it might not be possible to select a suitable relay node as described above. The node density can be reduced if the simulation time is short and the area is big enough. Insufficient node density is observed when independent simulation runs result in significant different outcomes.

The node density is set to 6 nodes/km<sup>2</sup> for path loss exponent of 2, 100 nodes/km<sup>2</sup> for a path loss exponent of 3 and 15.000 nodes/km<sup>2</sup> for path loss exponent of 4. In all cases the node density lies far above the minimum.

**Sending power** directly influences the maximum distance for sending. Reducing the sending power has the same influence as reducing the traffic density.

The sending power is set to 20dBm which is the upper power limit allowed by law. Moreover, this is a typical value for 802.11 wireless devices. In general battery driven wireless sensor networks may have lower sending power.

**Noise** in this context is defined as internal thermal noise in the receiver. Together with the SINR value it defines the minimum signal strength for a successful transmission in the case of no other interferences. As

long as the noise stays in a reasonable range, only minimal changes in the outcomes are observed. All influences of the other parameters stay the same. To reduce the amount of varying parameters, the noise is kept constant.

The internal thermal noise is set to -90dBm, which is a common value for receivers at 2.4GHz.

**Transmission time** is the amount of time it takes to transmit a single data packet. Changing the transmission time has a linear effect on the traffic density. Reducing the transmission time by half is equivalent to reducing the traffic density by half.

The transmission time is arbitrarily set to 1ms. This is approximately the time it takes to transmit a standard size IP data packet (around 1500 bytes) over a 12 MBit/s link.

The following list describes all parameters that are varied in different simulation scenarios.

**Traffic density** is the number of transmissions per area and time. A higher transmission density results in more transmissions and therefore higher interferences. Above a certain density the network traffic collapses and almost all transmissions fail. Densities which result in more than 80% packet loss due to interferences are not analyzed. Meaningful transmission densities heavily depend on the path loss factor.

For example for a path loss exponent of  $n = 3$  a reasonable transmission density ranges from 0 to 45 transmission/(km<sup>2</sup> ms).

**Path loss exponent**  $n$  defines the signal mitigation over distance (see equation 2.5).

The path loss exponent ranges from  $n = 2$  for free space over  $n = 3$  for outdoor to  $n = 4$  for indoor environments.

**SINR threshold** is the signal to noise plus interferences ratio threshold. Each transmission which exceeds this threshold can be successfully decoded. The minimal SINR depends on the transmission coding. Higher coding schemes (transmitting more bits per symbol) need higher SINRs. For low data rates in wireless sensor networks, SINRs of 10dB are sufficient. Higher data rates afford SINR thresholds up to 40dB. [38]



**Cancellation factor** is the factor by which relay transmissions can be canceled out. A detailed description was already given in section 3.2.4

A hypothetical full cancelation (factor of 1) gives an upper bound. For comparison the case of 0% cancelation is given as well.



## The Simulator

The purpose of computing is  
insight, not numbers.

---

R. W. Hamming, 1961.

All simulation tools discussed in Section 2.4 have major drawbacks, ranging from scalability to a lack of available models. To overcome this problem, a self developed simulation tool is used. This section first gives an overview of the concept and then describes the simulation tool in detail.

### 4.1 General

The simulation tool used in this project was already used in previous works. Its purpose was to simulate routing algorithms for mesh networks. Therefore, only the physical and the data link layer have been implemented. The simulation scenarios in this work require an implementation of the physical and data link layer with cooperative relaying. Any upper layers are not yet available.

The simulation has to scale up to at least a few thousand nodes running in real time and therefore has to be as fast as possible. However the models used in the simulation should cover as many aspects as possible. The simulation speed should not result from simplified models, but from a good and fast implementation. The implementation is therefore done using C++ with procedural programming. The *GNU Standard C++ Library* [39] and

the XML library *libxml2* are additionally used. If the simulation is compiled with the graphical interface, the *glut* library for OpenGL has to be provided (see 4.7).

The simulator follows the paradigm of a discrete time and event driven simulation. Due to simplicity there is no plugin structure available. All models have to be hard coded into the source code. Although this requires a deeper knowledge of the source code, the possibilities of implementing new behaviors are not restricted. This is specially a great advantage if it comes to creating customized statistics. Moreover, unused parts can be commented out, which results in high simulation speed.

For repeatability, most random decisions of the simulations are done beforehand and written into a configuration file. This mainly includes the definition of the network and the traffic (when and how transmissions are executed). The configuration file is generated with a separate tool and is independent from the simulation tool. The generation of the network definition and the traffic are therefore independent from the simulation. Main advantage of the repeatability is the possibility to analyze the behavior in greater detail. Exactly the same situation can be replayed infinitely times in the simulation. In between the runs, the graphical interface as well as the statistics can be changed to achieve further understanding.

The simulation tool supports in general multiple transmission channels. There is no distinction between multiple channels in frequencies or spread codes. Channels can either be independent or dependent on each other. In the simulation scenarios described in Chapter 3 multiple channels are not used. All transmissions take place on the logical channel 1.

If multiple channels are used, each *node* can send on a subset of the available channels. The nodes access each channel via an *interface*. In the context of the simulation tool, a node is a device with multiple antennas. A node can be, for example, a wireless router and an interface is one antenna. The number of interfaces per node has no upper bound, but must be at least one. If the whole simulation scenario consists of one logical channel, each node has exactly one interface and the terms *node* and *interface* can be more or less interchanged.

## 4.2 Data Structures

There are a few global variables which are used throughout the whole simulation. The most important of them is a vector of all interfaces. The *interface struct* contains all important information concerning one specific interface. Listing 4.1 shows important attributes of the struct, minor details are skipped.

---

```
1 struct interface_t {
2     // general information
3     vector<link_t*> transLinks; // all outgoing links
4     vector<link_t*> incoming;   // all incoming links
5     char* id;                  // internal id
6     char* name;                // display name
7     char* ifname;              // interface name
8     node_t* node;              // to which node the interface belongs
9
10    // information for the physical layer
11    bool isTransmitting;
12    unsigned char channel;      // transmission channel
13    double sendingPower;        // sending power (in 109W = 1nW)
14    double sendingInterferences[MAX_CHANNEL-MIN_CHANNEL];
15                                // which interference the interface produces while sending
16    long long transmissionend;  // end of the ongoing transmission
17
18    long phy_cancelBuf[CANCEL_BUFFER_SIZE];
19                                // seq nr of packet which can be cancelled
20    int phy_cancelBufIndex;     // next free position in the phy_cancelBuffer
21    int poi;                    // point of interest
22
23    // information for the relay layer
24    datapacket_t* relay_buffer; // buffer for the relay layer
25
26    /* minor details are skipped. see source code for complete documentation */
27 };
```

---

Listing 4.1: struct interface

The first part of the struct stores information about the network and basic information like node names. Furthermore, each transmission layer saves individual information into the struct. Details are explained in Section 4.6.

Another important data structure is the **datapacket struct**. It represents a data packet which is going to be, is currently or was transmitted.

---

```
1 enum datapacketType_t {DIRECT, RELAY};
2 struct datapacket_t {
3     // general information
4     datapacketType_t type;
5     long seq;                // unique sequence number
6 };
```

---

```

7  // application layer information
8  interface_t *appl_source, *appl_relay, *appl_dest;

10 // physical layer information
11 interface_t* sourceInterface;
12 interface_t* phy_destInterface;

14 double destInterferencesW[MAX_CHANNEL-MIN_CHANNEL];
15 double destInterferencesO[MAX_CHANNEL-MIN_CHANNEL];
16 double ssFading;           // small scale fading (multipath)
17 double cancellation;       // cancellation factor for the transmission

19 double snr;
20 double snirW;
21 double snirO;

23 bool collidedW;           // if the packet is collided during transmission or not
24 bool collidedO;
25 };

```

---

Listing 4.2: struct datapacket

Each data packet can either be of the type **DIRECT** or **RELAY**. A data packet of the type **DIRECT** is sent from the source node, while a **RELAY** is sent from a relay node. This distinction is important for the receiving node to know where the packet is coming from.

The variables **appl\_\*** may only be used by the application layer. Their purpose is to provide easy access to transmission information for generating statistics.

Note that **phy\_destInterface** is in most cases not the same as **appl\_dest**. When the data packet is broadcasted on the physical layer the data structure of the data packet is copied on each outgoing link of the source interface. If there are 10 outgoing links then there will be 10 copies of the same data packet, each bound to one of the links. The variable **phy\_destInterface** is the interface at the end of the link where it is bound to. The application layer information in **appl\_dest** does not change.

The variables ending with a capital *W* and *O* have the same meaning. The difference between the two is that in one case cancellation is included in the calculation, in the other not. *W* stands for *with* and in this case the variable is calculated with cancellation. *O* stands for *without* and in this case the variable is calculated without cancellation.

## 4.3 Network Definition

The complete network is defined before running the simulation and written into the configuration file. All nodes are placed in a flat, rectangular, two dimensional area of arbitrary size. To avoid border effects the edges of the simulation area are wrapped around. Therefore, the area can be seen as a torus.

All nodes get a name for later identification as well as an exact location in the simulation area. The path loss between any two nodes is defined by the simple path loss model (Section 2.2.3). Links exceeding a certain distance are skipped. All links which would produce an average signal strength less than one fifth of the noise level are defined as irrelevant. By default the simple path loss model is used. For more advanced models it is also possible to define individual path losses between each two nodes. It may also be used to simulate a real world scenario where the path loss between each two nodes was measured empirically.

## 4.4 Transmissions

As mentioned before, the traffic is completely defined before starting the simulation run. This is a major requirement for repeatability of simulation runs. The traffic is written into a configuration file and is loaded at the beginning of the simulation.

The traffic is defined individually for each transmission. Each transmission has the following parameters:

**Sequence number** The sequence number is used to uniquely identify the transmission. The sequence number is an integer, has to be unique but needs not be continuous. It is also possible to use random numbers. The number is used by the simulation tool to recognize the same transmission again as well as helping to analyze the behavior of a special transmission in the graphical interface.

**Start time** It specifies the starting time of the transmission.

**Source node** The name of the node which initiates the transmission.

**Relay node** The name of the node which acts as a relay node. This parameter is optional and can also be omitted if no relay is used. If the relay should be selected on some runtime information this parameter can be neglected. In this case an appropriate relay selection mechanism has to be implemented into the simulation tool (Section 4.6.3).

**Destination node** The destination node of the transmission.

**Transmission time** The transmission time for all data packets is the same and is defined as a global variable.

During the start of the simulation the configuration file is read and all transmissions are loaded into memory. They are stored in a priority queue and ordered by time. A priority queue instead of a simple list is used due to performance reasons. In a priority queue it is possible to push and pop elements in logarithmic time. With an ordered list it would be possible to pop the first element in constant time, but adding an element would take linear time for sorting. It is noted that adding elements does not only take place during the initialization. During the execution of the simulation, transmissions can fail and have to be rescheduled. In this case linear time for rescheduling a transmission results into bad performance, especially for long simulation runs.

After all transmissions are loaded into memory, the first one is scheduled for execution. Transmissions are scheduled under the scheduler event name *TM\_POLL* (Section 4.5).

If a *TM\_POLL* event is executed, the first transmission is popped from the priority queue. The transmission properties are then used to generate a data packet. This data packet is passed to the application layer (Section 4.6.2) for transmission. Afterwards, the time stamp of the next transmission in the priority queue is evaluated and a scheduler event (*TM\_POLL*) for that time point is registered.

## 4.5 Scheduler

The scheduler is the heart of the simulator. As soon as the initialization is over, the scheduler manages all the work. All events which have to be processed must be registered in the scheduler.



The scheduler can run in three different modes. The first mode is the real time mode. In this mode the simulation time equals the real time. The second mode is similar to the first one, but slows the execution down by an arbitrary factor. Normally the real time mode is much too fast to observe individual transmission and their behavior. For better observation the second mode can be used. For convenience three different speed factors are implemented by default in the simulation and can be switched interactively during runtime. For even better observation, the execution can be stopped at any point of time and executed stepwise.

The last available mode is the batch mode. In this mode the simulation time does not match the real in any way. All events are executed in a row as fast as possible. Hence, the simulation time hops from event to event.

Listing 4.3 shows the shared source code for first and second mode. If the real time mode is used, the speed factor `global.scheduler.speed` is set to 1.

Lines 8-12 check if all events are processed, in which case the simulation would end. The function `endSimulation()` is called to clean up and generate the final statistics. The function exits the whole program and never returns. The lines 14-18 are only executed at the first run. The current real time is stored in the variable `global.scheduler.start` in ms. This is the starting time of the simulation and all time stamps in the simulation are relative to this starting time.

Line 21 calculates the real time of the next event to process.

In line 26 the calculated time is compared to the current real time. If the time stamp is in the future, the system waits. The system function `nanosleep(.)` is used such that the simulation does not block any other programs.

Finally, in line 42 the function `scheduler_nextStep()` is called which executes the next pending event.

---

```
1 void scheduler_run() {
2     struct timeval nowRT;
3     struct timeval endRT;

4
5     assert (global.scheduler.speed>0);

6
7     /* check if there are pending actions */
8     if (actions.empty()) {
9         global.scheduler.running = false;
10        endSimulation();
11        assert(false);
12    }
```

```

14  /* get start time of the simulation */
15  if (global.scheduler.start==0) {
16      gettimeofday(&nowRT, 0);
17      global.scheduler.start = nowRT.tv_sec*(long long)1e6 + (long long)nowRT.tv_usec;
18  }

20  /* calculate real time of next action */
21  long long next = global.scheduler.start + (long long)(actions.top()->time * (long
      long)global.scheduler.speed);
22  endRT.tv_sec = next / (long)1e6;
23  endRT.tv_usec = next % (long)1e6;

25  gettimeofday(&nowRT, 0);
26  if(timercmp(&nowRT, &endRT, <)) {

28      struct timeval sleeptime;
29      struct timespec remainder_spec;
30      struct timespec sleeptime_spec;
31      timersub(&endRT, &nowRT, &sleeptime);

33      sleeptime_spec.tv_sec = sleeptime.tv_sec;
34      sleeptime_spec.tv_nsec = sleeptime.tv_usec * 1000;

36      /* wait until time passes */
37      while(nanosleep(&sleeptime_spec, &remainder_spec) < 0) sleeptime_spec =
          remainder_spec;

39  }

41  /* execute action */
42  scheduler_nextStep();

44 }

```

---

Listing 4.3: Scheduler in second mode

If the simulation runs in batch mode, all events are processed as fast as possible. In this case there are now waiting pauses between two events. The function `scheduler_nextStep()` is called in an infinite loop.

The function `scheduler_nextStep()` pops the first element from the list of events and processes it. All events are one of the following types:

`TM_POLL` initiates the next transmission (Section 4.4).

`RELAY_CLEAR` is scheduled 2 time steps after the whole relay transmission is finished. In some cases the transmission from the relay node may not take place or may not be received by the destination node. In these cases the destination node has to process the direct transmission without the relay transmission and clear its buffer (Section 4.6.3).

**PHY\_DISTRIBUTE** ends a transmission. The physical layer removes the data packets from the links and they are distributed to the interfaces (Section 4.6.4).

**RELAY\_SEND** sends a relay data packet from a relay node. This event is scheduled by the relay layer whenever a direct data packet is successfully received at a relay node (Section 4.6.3).

**END\_SIMULATION** calls the function `endSimulation()`. This event can be used to terminate the simulation at a certain time. If now such event exists, the simulation automatically terminates as soon as no more events are available.

Each event consists not only of a type but also of a `void` pointer. This pointer is used to attach any data structure to the event. E.g. to the event **RELAY\_SEND** the data packet which should be sent to the event is attached.

If two events are scheduled for the same point of time, they are ordered by the event type according to the above list. The event **TM\_POLL** is always executed first. In the rare case that the event type of both is the same, the two events are scheduled randomly.

## 4.6 Network Layers

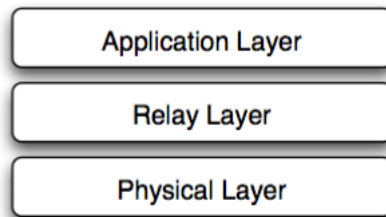


Figure 4.1: Network layers

Each transmission goes through a stack of layers on both transmission ends. The layer model is inspired by the ISO OSI model [40]. At the sending node, the transmitted data packet goes from the topmost to the bottom layer, at all receiving nodes it goes the other way round.

Figure 4.1 shows all three layers which are used in the simulation. The layer model neither matches the ISO OSI model nor the TCP/IP model exactly.

### 4.6.1 General Description

Each layer should act as independently as possible from the upper as well as the lower layer. The only way to exchange information between layers is by storing them into the data packet structure. An example would be the SINR threshold. The SINR value is calculated in the physical layer during the transmission and then stored in the data packet structure. Later on the relay layer decides if the SINR value is high enough or if the packet is collided and therefore not decodable.

Each layer has two primary functions. They serve as entry points for sending and receiving messages and have the endings `_down(interface_t*, datapacket_t*)` and `_up(interface_t*, datapacket_t*)`. The first one is used while sending, the second while receiving data packets.

### 4.6.2 Application Layer

The *application layer* is the top most layer. It applies no changes to the data packets, neither when sending nor when receiving. The main task of the layer is to serve as an entry point for sending messages and collecting statistics.

If a message is transmitted, the function `appl_down(interface_t*, datapacket_t*)` is called. The `interface_t*` is a pointer to the interface which sends the message, and `datapacket_t*` is a pointer to the message to be sent. Since the transmissions are placed randomly in time, it may occur that two transmissions take place at the same time at the same interface. In this case, the second transmission is delayed by 2ms. Another reason for delaying a message is, if the interface acts as a relay node. There is no check if the channel is free before sending. If a transmission is scheduled, it is performed at this point in time. This medium access protocol is called *ALOHA* [41].

### 4.6.3 Relay Layer

The relay layer deals with all aspects of cooperative relaying. The main aspects are relay selection and signal combining.

Apart from the two main functions each layer has to provide there are four helping functions. These functions are used for better structuring the

source code as well as an entry point for the scheduler. First, the four helping functions are described and, second, the two main function for moving data packets up and down in the layer hierarchy.

The function `relay_send(datapacket_t*)` is called by the scheduler, when the relay should send a relay message. The data packet is defined as a relay transmission and for safety all variables concerning the physical layer of the `datapacket_t` struct are reset. Then the data packet is handed over to the physical layer.

`relay_mergePackets(datapacket_t* , datapacket_t*)` merges two data packets. It recalculates the SINR value according to a given combining technique. If the maximum ratio combining is used the resulting SINR value is the sum of the individual data packets. Another possibility is to use selection combining which chooses the best SINR value. It is not possible to use different combining techniques in the same simulation run. Changing the combining technique results in changing the source code and recompiling the simulation tool.

The function `relay_clear(interface_t*, datapacket_t*)` must be called if the relay message is not received by the destination. This function is called by the scheduler when the event `RELAY_CLEAR` occurs. The functions removes the saved direct message from the buffer. If it was received correctly it is forwarded to the upper layer (application layer) without modifications. Now the interface is ready for participating in new transmissions again.

The function `relay_checkCollision(datapacket_t*)` checks the SINR value in the data packet and decides if the packet was received correctly or not. The flags `collidedW` and `collided0` of the data packet are updated accordingly.

The function `relay_down(interface_t*, datapacket_t*)` is called by the application layer. The function itself does nothing but forwarding the packet to the next lower layer (physical layer).

The function `relay_up(interface_t*, datapacket_t*)` is called by the physical layer for each received data packet. First of all it checks if the given interface is either the relay interface or the destination interface for the transmission. If neither applies, the data packet is completely dropped.

If the interface should act as a relay, it is determined if the packet could be decoded successfully. For this task the function `relay_checkCollision(datapacket_t*)` is called. If the data packet is not

corrupted, the event `RELAY_SEND` is registered in the scheduler and the relay message is sent during the next time slot.

If the interface is the destination the behavior depends on what kind of message is received. If a direct data packet is received, the data packet is stored in a buffer. At this time it does not matter if the data packet could be decoded successfully or not. The interface has to wait for the relay data packet for any further processing.

If a relay data packet is received and the data packet sequence matches the sequence number of the data packet in the buffer, both data packets are merged together. Merging data packets is done by the function `relay_mergePackets(datapacket_t* , datapacket_t*)` described above. After merging, it is checked if the data packet is received successfully or if it is collided. In the first case it is forwarded to the next upper layer (application layer), in the second it is dropped.

#### 4.6.4 Physical Layer

The physical layer deals with sending the data packets on the physical medium. It receives data packets from the upper layer (relay layer) and puts them on all available links to neighboring nodes. After the transmission time has elapsed, the data packets are removed from the links and each received data packet is handed over to the next upper layer (relay layer) for further processing.

The behavior of the physical layer is split up into 3 different functions. The function `phy_down(interface_t*, datapacket_t*)` receives a data packet from the upper layer and prepares it for sending. The function `phy_distribute(interface_t*)` is called by the scheduler when the transmission time has elapsed. The function `phy_up(interface_t*, datapacket_t*)` hands the data packet over to the next upper layer. In contrast to the other network layers, this function is not called by a lower layer, since there is no one. More precisely, this function is called by the `phy_distribute()` for each receiving node.

The helper function `calcCollision(interface_t *, int)` recalculates the SINR values of data packets at a given interface and given channel. The functions `inCancelBuffer(interface_t*, long)` and `addCancelBuffer(interface_t*, long)` decide which data packets can be cancelled.

The function `phy_down(interface_t*, datapacket_t*)` takes two arguments. The first is the source interface where the transmission is initiated and the second one is a pointer to the data packet to be sent. Since for simplicity all data packets have the same transmission time, the calculation of the end time of the transmission is simple. More advanced behavior, e.g. transmission time based on data packet length and / or channel bandwidth, can easily be implemented at this point.

Next the points of interest are set. Only at the points of interest it is ensured that the SINR values are calculated correctly. Points of interest are all nodes where the data packet is further processed. This may only happen at the destination and relay node. All other nodes will drop the data packet anyway. The concept of points of interest is described in more detail in the description of the helping function `calcCollision(interface_t *, int)`.

The major task of the function is to put the data packet on all outgoing links of the interface. Therefore the `datapacket_t struct` is cloned for each link. From now on all modification of the data packet just concerns the data packet on an individual link.

The physical destination node is the node at the end of the link. Mind that the physical destination node is in most cases not equal to the destination node of the transmission. The destination node is where the data actually should be delivered to. It was already defined before transmitting the data packet and never changes. The corresponding variable is `appl_dest`. The physical destination node is set after the data packet is cloned and bound to an individual link and is stored in the variable `phy_destination`.

For each link a separate multipath fading coefficient is generated (Listing 4.4). A uniformly distributed random variable `r` between 0 and 1 is generated. This random variable is then transformed with the function  $-\log(1-r)$  to an exponential distributed random variable for the multipath fading coefficient. [42, 35]

---

```
1 // multipath fading
2 float r =(double)rand()/RAND_MAX;
3 link->packet->ssFading = -log(1-r);
```

---

Listing 4.4: Generation of multipath fading coefficient

Next it is checked if the data packet has been received on the physical destination node before. This is done with the function `inCancelBuffer(interface_t*, long)` which is described later. If the data

packet has been received before, it can be cancelled. The cancellation factor is then set to  $1 - \text{global.net.cancellationFactor}$ .

Now the interferences produced at the physical destination node can be calculated. Since the simulation tool supports multiple channels, the interferences have to be calculated at each channel separately. In the simulation scenarios described in 3.3 only channel 1 is used, so the variable `channel` can be assumed to be 1 all the time. Moreover, the interferences have to be calculated twice. Once with taking cancellation into account and once without. The results are stored in the corresponding variables (Listing 4.5).

---

```
1  link->packet->destInterferencesW[channel]
2    = source->sendingInterferences[channel] * link->pathloss * link->packet->ssFading *
      link->packet->cancellation;

4  link->packet->destInterferencesO[channel]
5    = source->sendingInterferences[channel] * link->pathloss * link->packet->ssFading;
```

---

Listing 4.5: Calculation of the interferences

For each link the function `calcCollision(interface_t *, int)` is called with the physical destination interface as the first argument and the channel as second. Afterwards, the scheduler event `PHY_DISTRIBUTE` is scheduled at the end of the transmission. If the data packet is a direct data packet, then also the scheduler event `RELAY_CLEAR` is added. This event is scheduled after the direct and the relay transmission are over.

The helper function `calcCollision(interface_t *, int)` recalculates the SINR values of all data packets arriving at the given interface. Because recalculation of the SINR values is computational heavy, the concept of *points of interests* is introduced. The SINR values need only to be recalculated if the data packet is used later on. This is only the case when the data packet is received at a relay interface or at the destination interface. All other data packets are dropped at the relay layer anyway. The number of dropped data packets is specially high in dense networks. This concept generates a simulation speed up by a factor of 2 or higher in networks with a low path loss coefficient ( $\leq 3$ ).

Each interface has an integer variable `poi`, which is initialized with 0. If an interface becomes a point of interest, `poi` is incremented by one. For all interfaces with a `poi` greater than zero, SINR values are calculated, all other interfaces are skipped. After the transmission is completed, the point of interest is removed by decrementing the variable `poi` by one.



The SINR value is calculated twice because there are two different versions. The value `snirW` is the SINR value of a data packet with interference cancellation turned on, `snir0` is without. After the new SINR values are calculated, they are compared to the ones in the data packet. Only if the new SINR values are worse (smaller) than the old ones, they are taken into account. Otherwise the values in the data packet are not changed.

The scheduler event `PHY_DISTRIBUTE` calls the function `phy_distribute(interface_t*)`. This function first removes the point of interest and then loops through all links. All data packets are removed from the link and are handed over to the function `phy_up(interface_t*, datapacket_t*)`. This function adds the data packet to the cancellation buffer and hands the data packet to the next upper layer (relay layer).

The cancellation buffer is a property of an interface and stores the sequence numbers of received direct data packets. The function `addCancelBuffer(interface_t* interface, long seq)` adds a sequence number to the buffer, the function `bool inCancelBuffer(interface_t* interface, long seq)` tests if the given sequence number is in the buffer or not. The buffer itself is implemented as a ring buffer of arbitrary size. The size has to be fixed at compile time. If the size is chosen too large, unused memory is wasted. More problematic is if the buffer is chosen too small, since then it may happen that sequence number of received packets are overwritten too early. Received data packets would then not be canceled. In this case the simulation terminates with an appropriate error. The required size of the buffer depends on the transmission density and how far data packets can be sent.

## 4.7 Graphical Interface

The aim of the graphical interface is to provide to the user a deeper understanding of what is going on in the simulation. This is most interesting when developing and implementing new protocols. With the help of the graphical interface it is easier to hunt down bugs and design errors.

The graphical interface is split up into three frames. The top most and biggest one shows the network with ongoing transmissions. Below the main frame a small horizontal area with buttons to toggle display properties in the main frame is located. At the bottom a listing of the next events is shown.

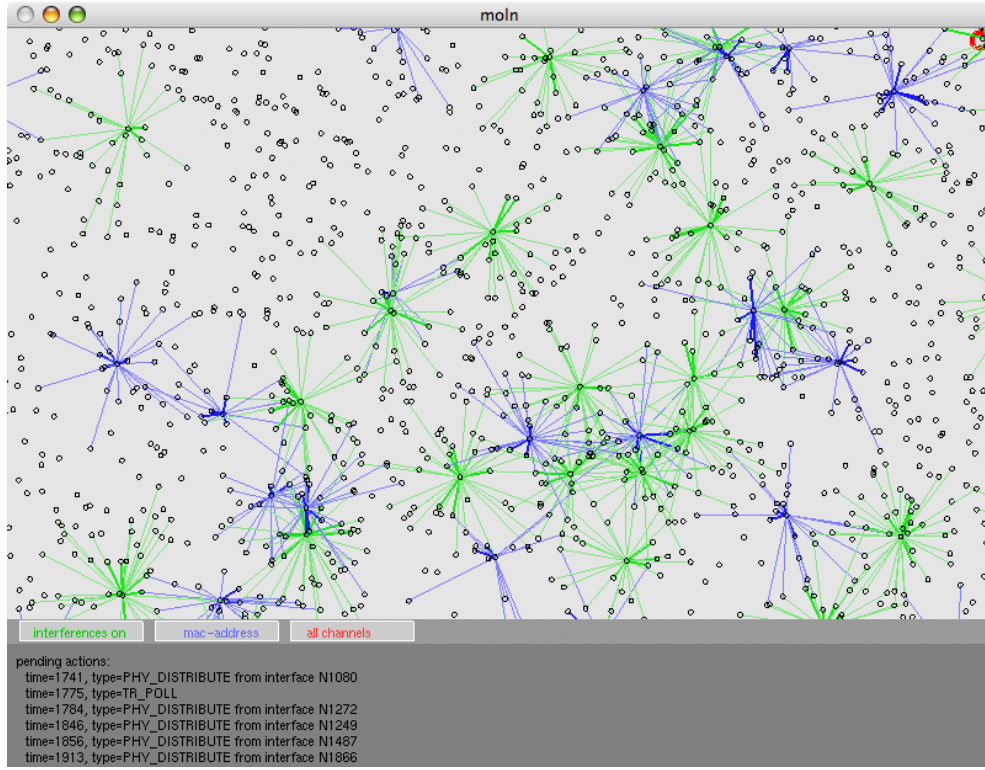


Figure 4.2: Screenshot of the graphical interface

In the main frame all nodes are displayed as small circles. By default only ongoing transmissions are displayed. By pressing the button 'interferences on', all interferences are shown with black lines. In this mode it is easy to find out which node influences which one. For bigger networks this results in a black cloud and therefore, it only makes sense for smaller networks.

Ongoing transmissions are displayed in the colors green and blue. Direct transmissions are displayed in green, whereas relay transmission are displayed in blue. The two different thicknesses of the lines show how good a signal is. The thicker lines indicate that the transmission is successful without taking interferences into account. Hence a transmission can still fail because there is too much interference from neighboring nodes. A thin line indicates that it will for sure not be possible to decode the data packet. Still it produces interferences at the destination node and disturbs other transmissions. Only transmissions that have a signal strength above the noise level are displayed. All other transmissions are skipped in the graphical interface. Note that for the SINR calculation at the physical layer they are still used.

The user interaction is rather limited. Changing the behavior of the graphical interface can only be achieved by changing the source code. The user can control the speed of execution. As already mentioned in Section 4.5, the scheduler has the ability to run in different modes. By pressing the space bar, the simulation starts to run in real time. A thick red circle in the upper right corner is displayed (Figure 4.3). By pressing the numbers 1-3, the simulation is slowed down by the factor of 10, 100 or 1000. The red circle gets thinner as the simulations get slower. For observing some special behavior it is recommended to slow down the execution; otherwise data packets flow around too fast.

The simulation can be paused at any time by pressing the space bar again. The red circle in the corner disappears and the simulation is paused. By pressing the **n**-key only the next event is executed. In this way it is easy to step through some parts of the simulation.



Figure 4.3: Thick red circle indicating the simulation running in real time

The zoom level of the main frame can be changed by pressing **+** and **-**. The network can be panned by using the arrow keys.

At the bottom of the graphical interface a list of the next pending events is displayed. Each entry first shows the time which is followed by the event type. Depending on the event type, some additional information may be provided, e.g., the scheduler event **RELAY\_CLEAR** is followed by the interface name and the packet sequence number (Figure 4.4). If the list gets too long, only the next five events are shown.

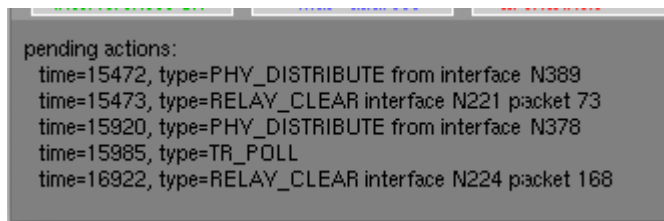


Figure 4.4: Pending events

## 4.8 Configuration File

The configuration file contains all necessary information for one simulation run. No additional input is needed. The configuration file has a XML struc-

ture and has to be provided to the simulation tool when starting. The configuration file can either be written by hand or it is generated with a separate tool, which is the more convenient way. The following paragraphs explain the structure of the configuration file.

---

```
<simulation name="ScenarioA">
  <global seed="1211599682">
    <graphics graphicsScale="1.9" graphicsOffsetX="17" graphicsOffsetY="35" />
    <network pathloss="3" nodeDensity="10000" trafficDensity="10" maxDist="46"
      sizeX="200" sizeY="200" />
    <interface noise="-90" sendingPower="20" snir="20" cancellationFactor="0.9"
      sendingTreshhold="20" />
  </global>
  <nodes>
    <node name="N1" posX="85.3" posY="57.04" >
      <interface name="wth0" channel="1" />
    </node>
    <node name="N2" posX="17.8" posY="197.54" >
      <interface name="wth0" channel="1" />
    </node>
    <node name="N3" posX="85.72" posY="105.21" >
      <interface name="wth0" channel="1" />
    </node>

    [ ... ]

  </nodes>
  <traffic transmissionTime="1">
    <transmission seq="1" time="321.87" source="N139" relay="N164" destination="N297" />
    <transmission seq="2" time="304.34" source="N191" relay="N235" destination="N358" />
    <transmission seq="3" time="472.01" source="N284" relay="N344" destination="N131" />

    [ ... ]

  </traffic>
</simulation>
```

---

Listing 4.6: Sample XML configuration

The root element of the XML file is `<simulation>` with the attribute `name`. The `name` attribute has no special formatting and only helps to identify simulation scenarios. The root element contains three mandatory children named `global`, `nodes` and `traffic`. Moreover, they have to be provided in this order.

**global** This element provides basic global information to the simulation. The attributes of the child elements are stored in global variables in the simulation tool. The following child elements have to be provided.

The element `graphics` controls the initial zoom level and position of

the network in the graphical interface. This element can be omitted, if the simulation is compiled without the graphical interface.

The element **network** holds definitions of the network. The most important attribute is **pathloss**, which is the path loss exponent. The value can be any floating point number greater than zero. The attributes **sizeX** and **sizeY** define the upper right corner of the simulation area. The lower left corner is always assumed to be (0, 0). Nodes outside this area are skipped and not included in the network. Additionally, the boundaries are needed for the correct calculation of links when using a torus.

The element **interface** defines basic information that is valid for all interfaces. The attribute **sendingPower** defines the sending power of the interface. The value is stated in dBm and can be any real number. The attribute **noise** defines the average noise power each interface encounters and is also given in dBm. The attribute **sendingThreshold** defines the maximum interference an interface can encounter while starting a transmission. If the interferences are higher then the transmission is delayed. If this value is set to the same value as the **sendingPower**, ALOHA is used. If a data packet can be successfully decoded i defined by the attribute **snir**. The data packet can be decoded, if the SINR value of the data packet is greater than the stated SINR threshold in the configuration file. The attribute **cancellationFactor** defines the percentage of how much a priori known interfering signal can be canceled out. The value can range from zero (meaning no cancellation at all) to 1 (full cancelation).

**nodes** This element contains all nodes in the simulation. The ordering of the nodes does not matter. Each node has a **name** attribute which uniquely identifies the node. The location of the node is defined by the two attributes **posX** and **posY**. Both attributes hold real numbers in the boundaries of the simulation area defined in the **network** element. Each node must have at least one **interface** element. Each **interface** element has an attribute **name** which is unique within the node and a **channel** attribute. The **channel** attribute defines on which logical channel the interface sends and receives data packets.

**traffic** This element contains all transmissions in the simulation. The attribute **transmissionTime** defines the length of the transmission in time units. If the transmission runs in real time, one time unit is

matched with one millisecond. Each transmission is a separate element. The elements do not have to be in any specific order, because they are sorted by time when loaded into the simulation. The attribute **seq** defines the sequence number of the transmission and has to be unique. The **time** attribute is the starting point in time of the transmission. It has to be always greater than zero. The attributes **source**, **relay** and **destination** define the source node, the relay node and the destination node.

## 4.9 Simplified Simulator

In Chapter 5 also intermediate results are compared to simulation results. Since the simulation tool presented above is rather complex it does not suite very well simple scenarios. Therefore, intermediate results from Chapter 5 are compared to simulation results of this simple simulation tool. The aim of this simplified simulation tool is to observe a single transmission with interfering nodes around.

---

```
1 #define RUNS 1000000
2 #define SIZE 250
3 #define SIGNALS 20

5 int main() {

7     srand(time(NULL));

9     for (long i=0; i<RUNS; i++) {

11         double dists[SIGNALS];
12         double signals[SIGNALS];

14         for (int i=0; i<SIGNALS; i++) { dists[i]=SIZE; signals[i]=0; }

16         for (int sig=0; sig<SIGNALS; sig++) {
17             // generate distances within boundaries
18             double dist=0;
19             do {
20                 double posX=(double)rand()/((double)(RAND_MAX)*SIZE);
21                 double posY=(double)rand()/((double)(RAND_MAX)*SIZE);
22                 dist = pow( pow(posX, 2) + pow(posY, 2), 0.5);
23             } while (((dist>46 && sig==0) || (dist>SIZE && sig>0)) || dist<1);

25             dists[sig]=dist;
26         }

28         // calculate signal strengths
29         signals[0]=1e-5*pow(dists[0], -3);
30         for (int j=1 ; j<11; j++) signals[j]=1e-5*pow(dists[j], -3);
```

```
31     for (int j=11; j<21; j++) signals[j]=0.1*1e-5*pow(dists[j], -3);

33     // add multipath fading
34     for (int i=0; i<SIGNALS; i++) {
35         double fading = -log(1-((double)rand()/RAND_MAX));
36         signals[i] *= fading;
37     }

39     // print out observation
40     printf("%g\n", signals[0]);

42 }
43 }
```

---

Listing 4.7: Simplified simulation tool

The Listing 4.7 shows the simplified simulation tool. It is assumed that the destination node is placed at (0, 0). The array `dists` holds the distances to nodes placed by a Poisson process inside a circle. The first entry is the distance to a node placed within transmission range, which can be interpreted as the source node. All other nodes are placed within the interference range. The array `signals` holds the corresponding values of the signal strength. Entries from index 0 to 10 are non-canceled transmission, from index 11 to 20 are canceled transmission with a cancellation factor of 0.9. Multipath fading can be activated on demand.





## Mathematical Validation

Everything should be made as  
simple as possible, but no simpler.

---

Albert Einstein

In the previous chapter a new simulation tool is introduced and different scenarios are simulated. Obviously, the question of reliability arises. Especially since a new, unknown simulation tool is used. A detailed mathematical analysis of the scenarios is due to complexity impossible. Neither is it possible to give an exact mathematical proof for the outcome of the simulation. This section analyses general transmission behavior and deducts the transmission success probabilities. Both cases of cancelation and non-cancelation are examined. The final as well as the intermediate results are compared to the simulation. The precise match of some sample calculation should confirm the credibility of the simulation.

The rest of the chapter is organized as follows. First, mathematical operations like the transformation and addition of random variables are explained. These operations are used multiple times throughout the rest of the chapter.

The main part deducts the success probability of a single transmission step by step. Based on the distribution of the nodes in space, the pdf of the signal strength is deducted. With that knowledge the SINR value is further analyzed. In the next part the question of the number of interfering neighboring nodes is answered. Finally everything is put together to get the probability of a successful transmission.

The final as well as the intermediate results are checked by simulations. The simpler, intermediate results are compared to the simplified simulation tool (Section 4.9). This tool is specially designed for these kind of simulations that give massive improvements to the execution speed. The successful transmission probability at the end of the chapter is then compared to the main simulation tool (Chapter 4). All simulations are carried out multiple times until a confidence interval of at least 99% is achieved. For simplicity, only the mean values are stated in the rest of the chapter.

**Terminology** Important terms which are used in the rest of the chapter are explained below.

**transmission range** defines a circular area around the node of observation.

If multipath fading is not considered, all nodes in this area are able to successfully transmit data to the node. Note that shadow fading is not considered throughout the project. Taking shadow fading into account, the area would not be circular any more.

**interference range** defines a circular area around the node of observation.

Nodes within this area are considered when calculating the interferences. The interferences of nodes at the border (without multipath fading taken into account) is at least the noise level. Loosely speaking, all nodes outside this area do not disturb the node of observation. Because of multipath fading it is theoretically possible that nodes outside the area produce higher interferences as the noise level. The probability of this case as well as the interference amount are rather small.

**interfering node** is a node within interference range which sends data.

**DIRECT transmission** is initiated by the source node. The transmission is received by the destination as well as the relay node.

**RELAY transmission** is the retransmission by the relay node. This transmission takes place exactly after the **DIRECT** transmission. The data packet can only be retransmitted by the relay node, if the **DIRECT** transmission was successfully decoded.

## 5.1 Statistical Operations

Throughout the rest of this chapter, most variables are random variables. Random variables do not have a certain value, but take values with an assigned probability. The probability of a certain value is given by the *probability density function* (pdf). The *cumulative distribution function* (cdf) is the probability of having a value up to a certain upper bound.

Random variables are written in bold letters (e.g.  $\mathbf{x}$ ). The pdf of  $\mathbf{x}$  is denoted as  $f_x(x)$  and the Cumulative distribution function (cdf) is  $F_x(x)$ . The relation between the pdf and the cdf is mathematically expressed as

$$F_x(x) = \int_{-\infty}^x f_x(y) dy. \quad (5.1)$$

In the following sections operations on random variables are described. The explanations are mainly based on the book *Probability, Random Variables and Stochastic Processes* by A. Papoulis [43].

### 5.1.1 Functions of One Random Variable

Let  $\mathbf{x}$  be a random variable of which we know the pdf. We want to have the pdf of  $y = g(\mathbf{x})$ . The function  $g(\mathbf{x})$  must be defined on the range of  $\mathbf{x}$ , be bounded and be a Borel function. Figure 5.1 shows an example of the function  $g(\mathbf{x})$  and the pdf  $f_x(x)$ . We first derive the pdf of  $\mathbf{y}$  in Figure 5.1 and then discuss the general case.

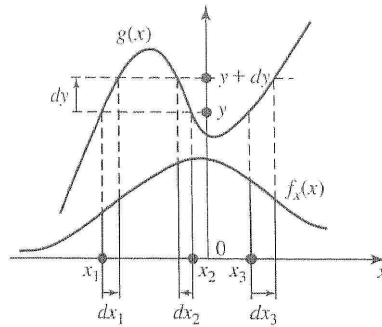


Figure 5.1: pdf  $f_x(x)$  and  $g(x)$  [43] page 130

By definition of the pdf we know that

$$f_y(y)dy = P\{y < \mathbf{y} \leq y + dy\}. \quad (5.2)$$

Now we need to find all values of  $x$  that satisfy  $y < g(x) \leq y + dy$ . As seen from the figure there are three disjoint intervals, namely

$$x_1 < x < x_1 + dx_1 \quad x_2 + dx_2 < x < x_2 \quad x_3 < x < x_3 + dx_3$$

where  $dx_1 > 0, dx_2 < 0, dx_3 > 0$ . From this it follows that

$$\begin{aligned} P\{y < \mathbf{y} \leq y + dy\} &= P\{x_1 < \mathbf{x} < x_1 + dx_1\} + \\ &+ P\{x_2 + dx_2 < \mathbf{x} < x_2\} + P\{x_3 < \mathbf{x} < x_3 + dx_3\} \end{aligned} \quad (5.3)$$

The right hand side can be rewritten with

$$P\{x_1 < \mathbf{x} < x_1 + dx_1\} = f_x(x_1)dx_1 \quad (5.4)$$

$$P\{x_2 + dx_2 < \mathbf{x} < x_2\} = f_x(x_2)|dx_2| \quad (5.5)$$

$$P\{x_3 < \mathbf{x} < x_3 + dx_3\} = f_x(x_3)dx_3 \quad (5.6)$$

where  $dx_i = dy/g'(x)$ . Replacing the right hand side of Equation 5.3 we get

$$f_y(y)dy = \frac{f_x(x_1)}{g'(x_1)}dy + \frac{f_x(x_2)}{|g'(x_2)|}dy + \frac{f_x(x_3)}{g'(x_3)}dy \quad (5.7)$$

Finally we generalize from three real roots to  $n$  roots. Denoting  $y = g(x)$  real roots by  $x_n$ .

$$f_y(y) = \frac{f_x(x_1)}{|g'(x_1)|} + \dots + \frac{f_x(x_n)}{|g'(x_n)|} \quad (5.8)$$

### 5.1.2 Functions of Two Random Variables

Assume  $\mathbf{x}$  and  $\mathbf{y}$  have two random variables and we are given a function  $g(x, y)$ . Based on that we form the random variable  $\mathbf{z}$

$$\mathbf{z} = g(\mathbf{x}, \mathbf{y}). \quad (5.9)$$

Further we assume that we know the joint pdf  $f_{xy}(x, y)$ . If  $\mathbf{x}$  and  $\mathbf{y}$  are independent random variables,  $f_{xy}(x, y) = f_x(x)f_y(y)$ .

The cdf of  $z$  is defined by

$$F_z(z) = P\{g(\mathbf{x}, \mathbf{y}) \leq z\} = P\{(\mathbf{x}, \mathbf{y}) \in D_z\} \quad (5.10)$$

where  $D_z$  represents the area in the  $xy$ -plane which satisfies  $g(x, y) \leq z$ . The pdf of  $z$  is deducted by

$$f_z(z) = F'_z(z) \quad (5.11)$$

The most difficult part is to determine the area of  $D_z$ . Unfortunately there is no general way, because  $D_z$  depends on the function  $g(x, y)$ . In the simple case of  $\mathbf{z} = \mathbf{x} + \mathbf{y}$  the cdf is

$$F_z(z) = P\{\mathbf{x} + \mathbf{y} \leq z\} = \int_{y=-\infty}^{\infty} \int_{x=-\infty}^{z-y} f_{xz}(x, y) dx dy. \quad (5.12)$$

The pdf is calculated by using the differentiation rule due to Leibnitz

$$f_z(z) = F'_z(z) = \int_{x=-\infty}^{\infty} f_{xz}(x, z - x) dx. \quad (5.13)$$

## 5.2 Probability Distributions of Signal Strengths

This section deducts the probability of a successful transmission of a single link between any two nodes within transmission range. Analytical results are given as far as possible and meaningful. At some point the expressions get far too complex to express and from this point on, the calculations are carried out numerically. Important intermediate results are checked against simulation results. All simulation results used in this section are taken from the simplified simulation tool (Section 4.9).

All numerical calculations and simulation results are taken from the same scenario. The chosen scenario is inspired by the wireless network standard 802.11 [38]. The parameters are stated in Table 5.1. Furthermore, all assumptions stated in Chapter 3 are still valid.

Symbol	Value	Annotation
$p_0$	9.894 mW	Signal strength at the reference distance $d_0$
$n$	3	Path loss exponent
$\theta$	20	SINR threshold
$r$	46 m	Radius of the transmission range
$R$	250 m	Radius of the interference range
$n_0$	-90 dBm	Thermal noise at the receiver
$\tau$	1 ms	Duration of one transmission
$T$	2000 ms	Simulation duration
	$4,85 \frac{\text{trans}}{\text{km}^2 \text{ms}}$	Transmission density
	$9 \text{ km}^2$	Simulation area
	$500 \frac{\text{node}}{\text{km}^2}$	Node density
a	21825	Transmissions in the interference range

Table 5.1: Used parameters for the simulation and numerical calculations

The pdf of signal strengths is deducted step by step. First, the pdf of the distance between nodes is derived. Next, the pdf of signal strength without and with multipath fading is examined. Finally, a closer look at some examples summing up multiple signals is taken.

### 5.2.1 PDF of Node Distances

The nodes are placed by a poisson process in a two dimensional area. In the one dimensional space the distance to any node within a certain distance is uniformly distributed. In the two dimensional space the probability is higher for greater distances. The probability increases proportionally to the perimeter of the circle. The normalization is given in Equation 5.14 where  $R$  is the maximum distance [44].

$$f_{dist}(r) = \begin{cases} \frac{2r}{R^2} & r \in [0, R] \\ 0 & \text{otherwise} \end{cases} \quad (5.14)$$

### 5.2.2 PDF of Signals Without Multipath Fading

Without multipath fading the signal strength is defined by the log-distance path loss model (Section 2.2.3). Equation 2.5 with the given parameters in

5.1 can be simplified to 5.15.

$$s(d) = ad^{-n} \quad \text{where } a = p_0 d_0^n \quad (5.15)$$

When replacing the distance  $d$  with the pdf  $f_{dist}$  this results in a pdf of the signal strength (Equation 5.16). Figure 5.2 pictures the analytical result and simulation data. The function has a lower bound because the pdf  $f_{dist}$  is bounded as well. The lowest signals strength comes from nodes at the outer edge of the distance interval.

$$f_s(d) = \begin{cases} \frac{2\left(\frac{d}{a}\right)^{-\frac{2+n}{n}}}{anR^2} & d \in [aR^n, \infty] \\ 0 & \text{otherwise} \end{cases} \quad (5.16)$$

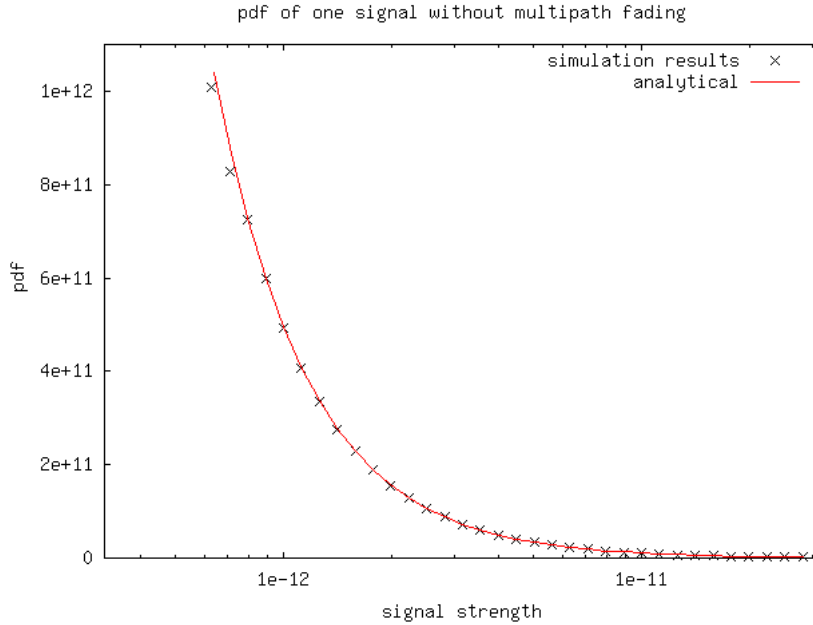


Figure 5.2: Pdf of signal strength of one signal without multipath fading

### 5.2.3 PDF of Signals With Multipath Fading

Signal power variations due to multipath fading are approximated by the exponential distribution  $f_{mp}(x) = e^{-x}$  (Section 2.2.4). The multipath fading coefficient either amplifies or attenuates the signal. The resulting signal is  $sm = s \cdot mp$  where  $s$  is the signal strength without multipath fading and  $mp$

the multipath fading coefficient. All variables are random variables and the result has to be calculated as described in Section 5.1.2.

$$\begin{aligned}
 f_{sm}(z) &= \int_{aR^{-n}}^{z/x} f_s(x) f_{mp}\left(\frac{z}{x}\right) dx \\
 &= -\frac{2 R^{-2+n} \left(\frac{a}{z}\right)^{\frac{2}{n}} \left(\frac{R^n z}{a}\right)^{-1+\frac{2}{n}}}{a e^{\frac{R^n z}{a}} n} + \\
 &\quad + \frac{4 a \left(\frac{a}{z}\right)^{-1+\frac{2}{n}} \left(\Gamma\left(\frac{2}{n}\right) - \Gamma\left(\frac{2}{n}, \frac{R^n z}{a}\right)\right)}{n^2 R^2 z^2}
 \end{aligned} \tag{5.17}$$

where  $\Gamma(a, z)$  is the incomplete gamma functions defined as

$$\Gamma(a, z) = \int_y^\infty t^{a-1} e^{-t} dt \tag{5.18}$$

Figure 5.3 show the analytical and simulation data.

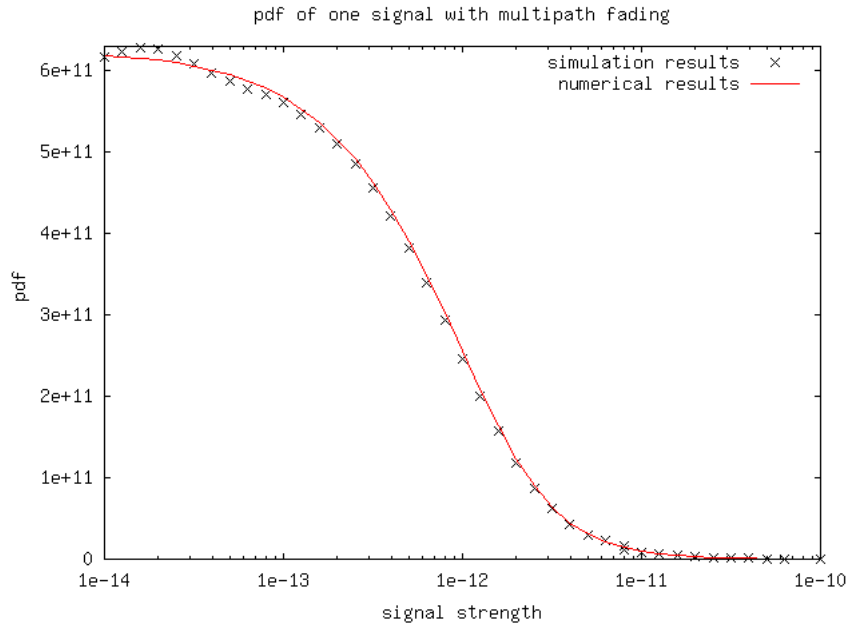


Figure 5.3: pdf of signal strength of one signal with multipath fading

#### 5.2.4 PDF of Multiple Signals

Most of the time more than one interfering node transmits a data packet. The resulting interference at the destination is the sum of all signals. Unfor-



Unfortunately even the analytical addition of two interfering signals is mathematically too complex to express analytically, since it would take over a page to show and does not give any further insights. Therefore, it is useless to derive any further analytical results. The only way to get results is a numerical computation.

The easiest way to sum up the interferences is by adding interfering node by interfering node. In the first step two interfering nodes are summed up. In the next steps always another interfering node is added. To reduced calculation complexity and to gain a calculation speed up, it makes sense to approximate the functions. In general the summation is an addition of two random variables as described in Section 5.1.2.

$$f_2(z) = \int_0^z f_1(x)f_1(z-x)dx \quad (5.19)$$

$$f_3(z) = \int_0^z f_2(x)f_1(z-x)dx \quad (5.20)$$

$$f_4(y) = \dots$$

It is assumed that  $f_1$  is the pdf of the signal strength of the interfering node and that both interfering nodes have the same pdf. Furthermore it is assumed that  $f_1$  is defined on the interval  $[0, \infty]$  and so is the resulting pdf  $f_2$ . The same principles apply to Equation 5.20.

The rest of the section explains how the approximations and calculations are carried out numerically. In the following the application *Wolfram Mathematica 5.0 Student Edition* [45] is used.

Mathematica provides the function `Interpolation[ ]` for interpolation [46]. The function is called with an arbitrary number of data points of the original function. The more data points are supplied, the more accurate is the interpolation. The result is an `InterpolationFunction` which can be called as any other function. If the argument matches one data point, the corresponding value is returned. In all other cases the results are interpolated. Interpolation works by fitting polynomial curves between successive data points. The degree of the fitting curves can be specified by the option `InterpolationOrder`. Throughout the following calculations the default value of 3 is used.

---

<sup>1</sup> data = X = Table[10<sup>x</sup>, {x, -15, -5, 0.05}];

```

2 Do [data[[i]] = {X[[i]], fsm[X[[i]]}], {i, 1, Length[data]}
3 f1 = Interpolation[data];

```

---

Listing 5.1: Approximation of function **fsm**

Listing 5.1 generates an interpolation function of the function **fsm** which is defined in Equation 5.17. The function  $f_{sm}$  is defined on the interval  $(0, \infty)$  and the integral of the function within the bounds is 1 and this should also hold for **fi**. Since interpolated function cannot be defined on infinite length, the boundaries are trimmed. The function **fi** is defined on the range  $(10^{-15}, 10^{-5})$  and all other values outside are assumed to be zero. Therefore the integral of **fi** is smaller than 1. The integral is calculated with `NIntegrate[fdi[x], {x, 10^-15, 10^-5}, MaxRecursion->15]` and equals 0.999295. This result shows a small error which can be neglected.

---

```

1 F[z_] := NIntegrate[fdi[x] fdi[z - x], {x, 0, z}, MaxRecursion -> 15]
2 data =X =Union[Table[10^x, {x, -14, -10, 0.01}], Table[10^x, {x, -9.95, -5, 0.05}]];
3 Do [data[[i]] = {X[[i]], F[X[[i]]}], {i, 1, Length[data]}
4 f2 = Interpolation[data];

```

---

Listing 5.2: Adding two interfering signals

In Listing 5.2 two interfering signals are added. The function **F[z]** represents Equation 5.19. The next three statements generate the interpolation function **f2** like in Listing 5.1. Integration over the approximation function **f2** results due to the same arguments as above in a value smaller 1, namely 0.999821. Again the resulting error is very small and can be neglected.

The further summation of additional interfering signals works analogue to Listing 5.2. The function **F[z]** has to be built on the appropriate functions. Figure 5.4 shows numerical and simulations results of 1, 2, 3 and 4 interfering signals. In Plot 5.4a the analytical and numerically approximated results are plotted. The corresponding comparison between analytical and simulation results is shown in Figure 5.3. The other three plots in Figure 5.4 show the comparison between numerical computation and simulation.

## 5.3 SINR Calculation

The SINR is the ratio of the signal strength from the source and interferences plus noise. Knowing the amount of interference a constant noise level can be added. Adding a constant noise level results in a shift of the probability

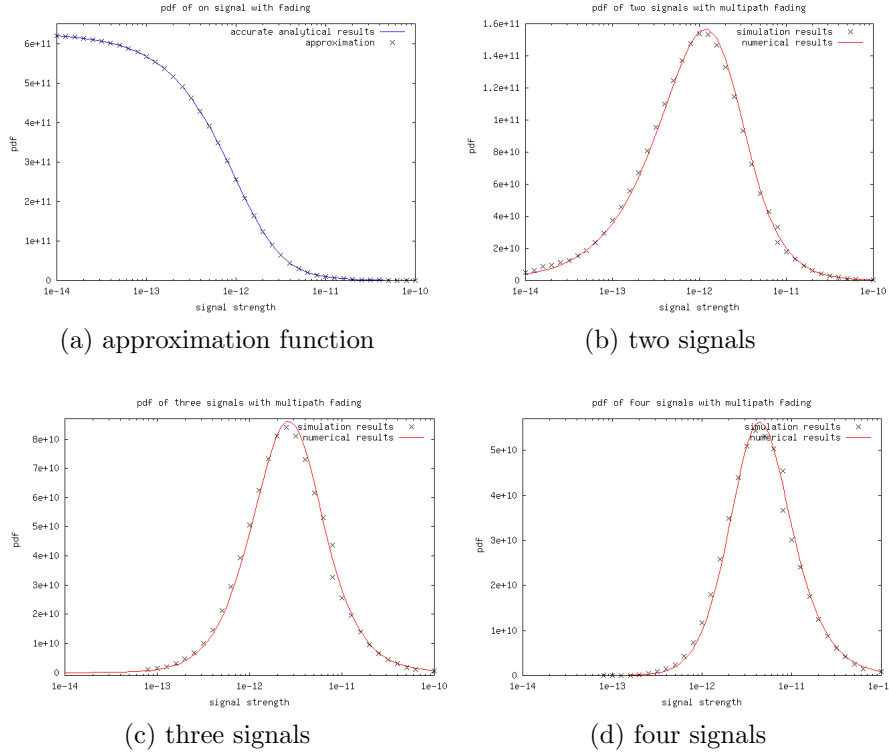


Figure 5.4: Pdfs of interference strengths

density function to the right (to a higher signal strength). Let  $f_{int}$  be the amount of interferences from neighboring nodes and  $n_0$  the noise level. The amount of interferences plus noise level  $f_{int+n}$  is

$$f_{int+n}(z) = f_{int}(z - n_0). \quad (5.21)$$

Figure 5.5 shows the numerical result of two signals with and without noise. The additional noise level results into a shift of the probability function to the right. Note that the plot has logarithmic x-axis which results in an unfamiliar layout of the shift.

Since the signal strength of the transmission as well as the interferences are random variables, the SINR is also a random variable. If  $f_{sig}$  and  $f_{int+n}$  are only defined for all positive values,  $f_{sinr}$  is

$$f_{sinr}(z) = \int_0^\infty x f_{sig}(xz) f_{int+n}(z) dx. \quad (5.22)$$

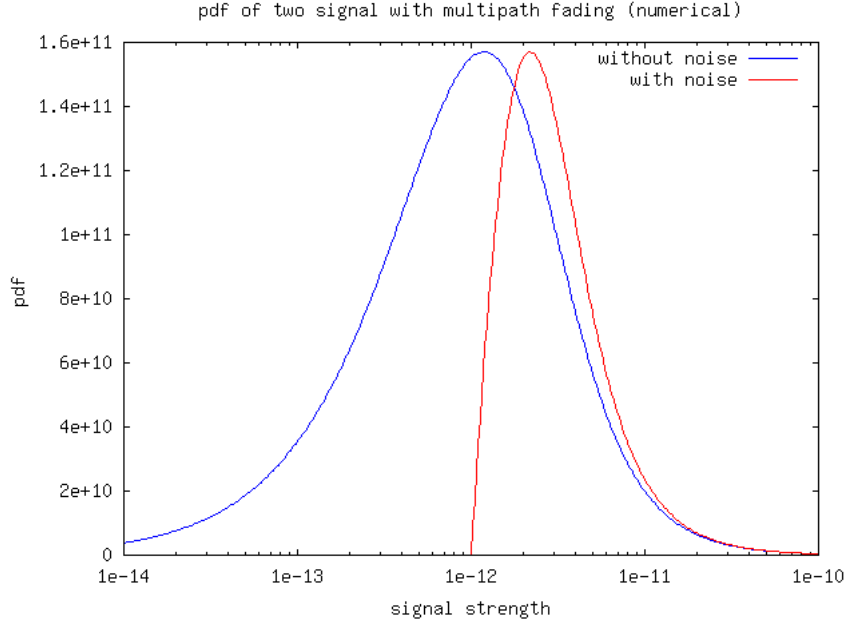


Figure 5.5: Two interfering signals with and without noise

Since both functions are approximated and have lower and upper bounds, the bounds of the integral is not 0 and  $\infty$ . Let  $\underline{f}_{sig}$  be the lower bound of  $f_{sig}$  and  $\bar{f}_{sig}$  the upper bound. Likewise  $\underline{f}_{int+n}$  and  $\bar{f}_{int+n}$  for  $f_{int+n}$ .

$$f_{sinr}(z) = \int_{\max(\underline{f}_{sig}/z, \underline{f}_{int+n})}^{\min(\bar{f}_{sig}/z, \bar{f}_{int+n})} x f_{sig}(xz) f_{int+n}(z) dx \quad (5.23)$$

Based on the pdf  $f_{sinr}$  the success probability of a transmission can be determined. Let  $\theta$  be the SINR threshold which defines whether the transmission was successful or not. The success probability  $p_s$  is

$$p_s = \int_{\theta}^{\infty} f_{sinr}(x) dx. \quad (5.24)$$

Table 5.2 shows SINR values for different numbers of interfering nodes. All interfering nodes are considered to be **DIRECT** transmissions, so no cancellation is active.

interfering nodes	numerical	simulation
0	92.62%	92.70%
1	74.16%	74.20%
2	60.42%	60.40%
3	50.01%	50.03%
4	42.03%	41.98%

Table 5.2: Transmission success probability  $p_s$ 

## 5.4 Distribution of Interfering Transmissions

The previous section describes how to calculate the success probability  $p_s$  for a certain number of interfering nodes. This section deals with the question of how many interference nodes there are.

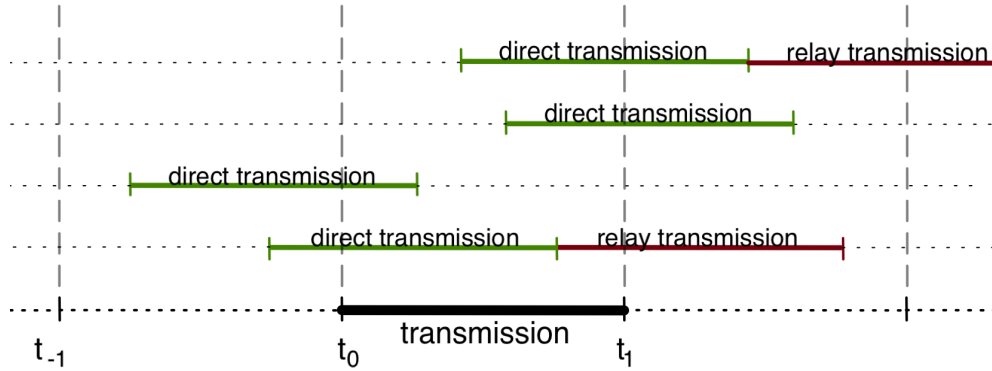


Figure 5.6: Example of a transmission

Figure 5.6 shows an example of a transmission. It is important to distinguish between **DIRECT** and **RELAY** transmissions. Due to the cancellation factor **RELAY** transmissions result in lower signal strengths than **DIRECT** transmissions. Therefore, they have to be treated separately. Each **RELAY** transmission has a preceding **DIRECT** transmission. It may be the case that the **DIRECT** transmission is outside the interference range, but the **RELAY** is not. In this case the **RELAY** is seen without the preceding **DIRECT** transmission. This case can be neglected, because the produced interference by a **RELAY** transmission near the border of the interference range is almost zero and disappears in the noise level. Therefore, it can be safely assumed that all **RELAY** transmissions have a preceding **DIRECT** transmission. Note that the inverse is not true, because there is only a **RELAY** transmission if the **DIRECT**

transmission was successfully received at the relay node.

The number of interfering nodes changes during the transmission. First, the question arises how many interfering nodes there are at the beginning and how they change.

There are two different interfering nodes, those which send **DIRECT** transmissions and those which send **RELAY** transmissions. The probability distribution of both is calculated in a similar way. A interfering transmission takes place at time  $t_0$ , if it started within the time interval  $(t_{-1}, t_0)$ . This applies in the same way for **DIRECT** and **RELAY** transmissions.

It is assumed that the interfering transmissions are uniformly distributed in time. If the simulation time is finite, then the amount of interfering transmission in the time interval  $(t_{-1}, t_0)$  is binomially distributed. For the cases where the simulation time gets large enough, the distribution can also be approximated with a Poisson distribution. In case of infinite time the distribution is Poisson distributed.

Let  $\alpha$  be the distribution of **DIRECT** transmissions at time  $t_0$ ,  $a$  the total number of transmissions within transmission range during the whole simulation,  $\tau$  the length of the time interval  $(t_{-1}, t_0)$  and  $T$  the simulation time.

$$\alpha(k) \sim \text{Bin}(k; n = a; p = \frac{\tau}{T}) \quad (5.25)$$

If each **DIRECT** transmission has a successive **RELAY** transmission, the number of **RELAY** transmissions at time  $t_0$  has the same distribution. In general, this is not the case because the total number of **RELAY** transmissions is lower than of **DIRECT** transmissions. Let  $b$  denote the total amount of **RELAY** transmissions in transmission range.  $b = ap_s$  where  $p_s$  is the probability of successfully receiving a **DIRECT** transmission at the relay node and retransmitting it.

Let  $\beta$  denote the probability distribution of the number of **RELAY** nodes at time  $t_0$ .

$$\beta(k) \sim \text{Bin}(k; n = b; p = \frac{\tau}{T}) \quad (5.26)$$

All **RELAY** transmissions at  $t_0$  clearly start before  $t_0$  and therefore end within the time interval  $(t_0, t_1)$ . Each **DIRECT** transmission either ends or is followed by a successive **RELAY** transmission. The probability of a **RELAY** transmission is  $p_s$ , the probability of ending is  $1 - p_s$ .

Since each **RELAY** transmission has a preceding **DIRECT** transmission, no new **RELAY** transmission starts within the time interval  $(t_0, t_1)$ . It is however, possible that new **DIRECT** transmissions start. The number of new **DIRECT** transmissions is the same as in the interval  $(t_{-1}, t_0)$ . Its probability distribution is the same as for  $\alpha$  as defined in Equation 5.25.

The order of new and ending transmissions as well as changes from **DIRECT** to **RELAY** in the time interval  $(t_0, t_1)$  is uniformly distributed.

Taking all that into account, this results in a huge variety of combinations, each occurring with a certain probability. Each combination has to be examined and the success probability calculated. The final success probability is the weighted sum of all individual success probabilities.

The following section gives two scenarios to further clarify the way of calculation.

## 5.5 Successful Transmission probability $p_s$

In section 5.3 the SINR calculation for a certain number of interfering nodes is discussed. The previous section explains the distribution of number of interfering nodes. This section puts both together. Since a general calculation is not possible, the calculation is shown in two scenarios in full detail. The results are compared with the simulation.

For both scenarios the parameters of Table 5.1 are used. The scenarios differ by the cancelation factor. In the first scenario full cancelation is performed. The second scenario performs no cancelation at all.

### 5.5.1 Scenario One

In this scenario full relay cancelation is performed. All **RELAY** transmissions illustrated in Figure 5.6 are canceled out and produce no interferences. Hence only **DIRECT** transmissions have to be considered. The distribution of **DIRECT** transmissions is calculated as stated in Equation 5.25. Figure 5.7 shows analytical and simulation result.

This distribution describes the number of interfering nodes at the start of the transmission as well as the number of interfering nodes starting during the transmission. Table 5.3 shows the probability for having a certain number of interfering nodes at the beginning (rows) and a certain number of

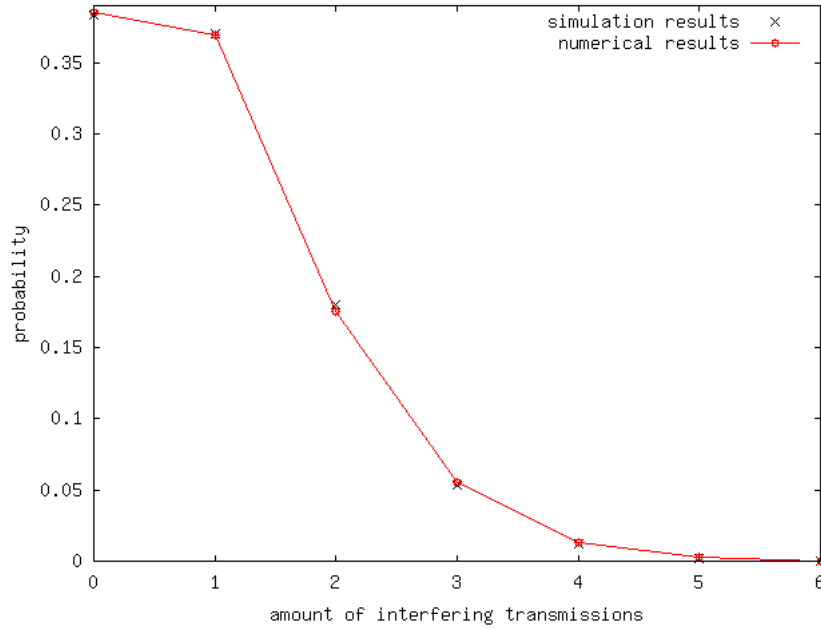


Figure 5.7: Pdf of interfering DIRECT transmissions

interfering nodes starting during the transmission (columns). As can be seen in Figure 5.7, more than three interfering transmissions are possible. However, the probability of having that so is negligibly small. The percentage of having more than 3 interfering nodes at the beginning is shown in the table in the bottommost row, the probability of starting nodes in the rightmost column.

		starts during the transmission				
		0	1	2	3	off cut
at beginning	0	14,81%	14,22%	6,75%	2,11%	0,60%
	1	14,22%	13,64%	6,48%	2,03%	0,57%
	2	6,75%	6,48%	3,07%	0,96%	0,27%
	3	2,11%	2,03%	0,96%	0,30%	0,09%
	offcut	0,60%	0,57%	0,27%	0,09%	0,02%

Table 5.3: Probabilities

Each cell of the table is analyzed to determine the success probability. The events of ending and starting interfering transmissions has to be analyzed in all combinations. The SINR value changes at each event. However, we are only interested in the worst case scenario. The SINR value is worst (smallest)



if a maximum number of interfering nodes are transmitting. Table 5.5 shows the calculation.

The abbreviation *pre* indicates for the number of interfering nodes at the begin of the transmission and *plus* the number of interfering nodes starting during the transmission. The column *changes* shows the order of the events. A + denotes one new transmission starting, - the end of one single transmission. The number of - is defined by the *pre* column, whereas the number of + is defined by the *plus* column. The column *worst case* is the maximum number of interfering nodes which can be found in this combination. Based on the worst case, the success probability is calculated.

All combinations for a certain number of *pre* and *plus* are uniformly distributed. The average is therefore the mean value, these values are shown in Table 5.4.

		plus			
		0	1	2	3
pre	0	92,30%	74,14%	60,42%	50,01%
	1	74,14%	67,28%	56,95%	48,02%
	2	60,42%	56,95%	52,15%	46,82%
	3	50,01%	48,02%	45,40%	43,27%

Table 5.4: Success probabilities

The final transmission success probability  $p_s$  is the weighted sum of the probabilities in Table 5.4.

	numerical	simulation
transmission success probability $p_s$	67,33%	67,77%

### 5.5.2 Scenario Two

In contrast to the previous scenario, in this scenario no cancelation is performed. The **RELAY** transmissions produce the same interferences as **DIRECT** transmissions. The calculation is in principle the same as before. First, a distribution of interfering nodes is deducted. Here not only the **DIRECT** but also **RELAY** transmissions have to be taken into account. Afterwards each case is examined.

<i>pre</i>	<i>plus</i>	<i>changes</i>	<i>worst case</i>	<i>probability</i>	<i>pre</i>	<i>plus</i>	<i>changes</i>	<i>worst case</i>	<i>probability</i>
0	0		0	92,30%	3	0	---	3	50,01%
0	1	+	1	74,14%	3	1	----+	3	50,01%
0	2	++	2	60,42%	3	1	---+-	3	50,01%
0	3	+++	3	50,01%	3	1	-+---	3	50,01%
1	0	-	1	74,14%	3	1	+----	4	42,03%
1	1	-+	1	74,14%	3	2	-----+	3	50,01%
1	1	+-	2	60,42%	3	2	---++-	3	50,01%
1	2	---+	2	60,42%	3	2	-+----+	3	50,01%
1	2	++-	2	60,42%	3	2	+----+	4	42,03%
1	2	++-	3	50,01%	3	2	---+-	3	50,01%
1	3	----+	3	50,01%	3	2	-+--+-	3	50,01%
1	3	++--	3	50,01%	3	2	+---+-	4	42,03%
1	3	++--	3	50,01%	3	2	-++---	4	42,03%
1	3	++--	3	50,01%	3	2	++---	4	42,03%
1	3	++--	4	42,03%	3	2	++---	5	35,83%
2	0	--	2	60,42%	3	3	-----+	3	50,01%
2	1	---+	2	60,42%	3	3	---++-	3	50,01%
2	1	+-	2	60,42%	3	3	-+----+	3	50,01%
2	1	++-	3	50,01%	3	3	+----+	3	50,01%
2	2	---+	2	60,42%	3	3	---++-	3	50,01%
2	2	++-	2	60,42%	3	3	-+--+-	3	50,01%
2	2	++-	3	50,01%	3	3	+---++	3	50,01%
2	2	++-	3	50,01%	3	3	-+--+-	4	42,03%
2	2	++-	3	50,01%	3	3	++---+	4	42,03%
2	2	++-	4	42,03%	3	3	++---	4	42,03%
2	3	---++	3	50,01%	3	3	---++-	4	42,03%
2	3	++--	3	50,01%	3	3	-+--+-	4	42,03%
2	3	++--	3	50,01%	3	3	++---+	4	42,03%
2	3	++--	3	50,01%	3	3	-+--+-	4	42,03%
2	3	++--	3	50,01%	3	3	++---+	4	42,03%
2	3	++--	3	50,01%	3	3	++---+	5	35,83%
2	3	++--	4	42,03%	3	3	++++-	5	35,83%
2	3	++--	4	42,03%	3	3	++---+	5	35,83%
2	3	++--	4	42,03%	3	3	++---+	5	35,83%
2	3	++--	4	42,03%	3	3	++---+	5	35,83%

Table 5.5: Success probabilities

The number of interfering nodes at time  $t_0$  is not as easy to determine as in the previous scenario. Moreover, different cases have to be distinguished. Figure 5.8 shows the different cases of interfering transmissions. In case A and B the **DIRECT** transmission starts before  $t_0$ . The difference between case A and B is that in case A there is no following **RELAY** transmission, whereas in case B there is. The probability of having such a **RELAY** transmission is  $p_s$ . In case C only the **RELAY** transmission lies inside the interval  $(t_0, t_1)$ . In case D a new interfering transmission starts in the interval  $(t_0, t_1)$ . Whether this transmission has a proceeding **RELAY** transmission or not does not matter.

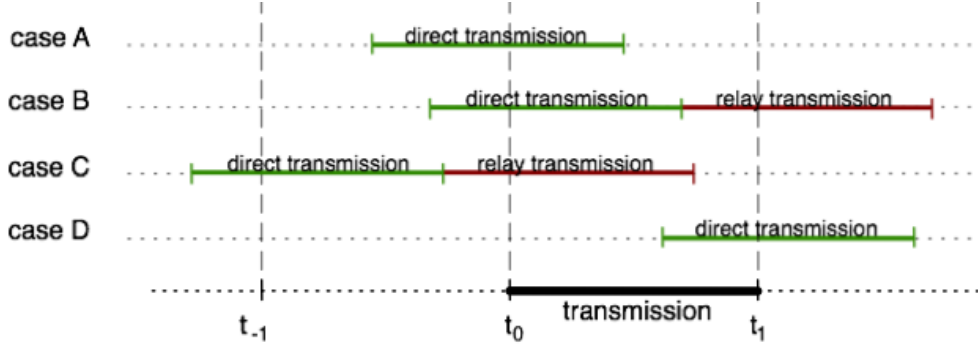


Figure 5.8: Different cases of interfering transmissions

The probability distribution for each case is derived in a similar way as described above. It is binomially distributed with different  $n$ . The probability distributions are labeled by the capital Fraktur letter of the case. As before,

$a$  denotes the total number of **DIRECT** transmission within interfering range,

$p_s$  denotes the probability having a proceeding **RELAY** transmission,

$\tau$  denotes the duration of the interval  $(t_0, t_1)$ , and

$T$  denotes the simulation time.

$$\text{case A: } \mathfrak{A} \sim \text{Bin}(k; n = a(1 - p_s); p = \frac{\tau}{T}) \quad (5.27)$$

$$\text{case B: } \mathfrak{B} \sim \text{Bin}(k; n = ap_s; p = \frac{\tau}{T}) \quad (5.28)$$

$$\text{case C: } \mathfrak{C} \sim \text{Bin}(k; n = ap_s; p = \frac{\tau}{T}) \quad (5.29)$$

$$\text{case D: } \mathfrak{D} \sim \text{Bin}(k; n = a; p = \frac{\tau}{T}) \quad (5.30)$$



## Results and Conclusion

"Forty-two!" yelled Loonquawl. "Is that all you've got to show for seven and a half million years' work?" "I checked it very thoroughly," said the computer, "and that quite definitely is the answer. I think the problem, to be quite honest with you, is that you've never actually known what the question is."

---

Douglas Adams, *The Hitchhiker's Guide to the Galaxy*

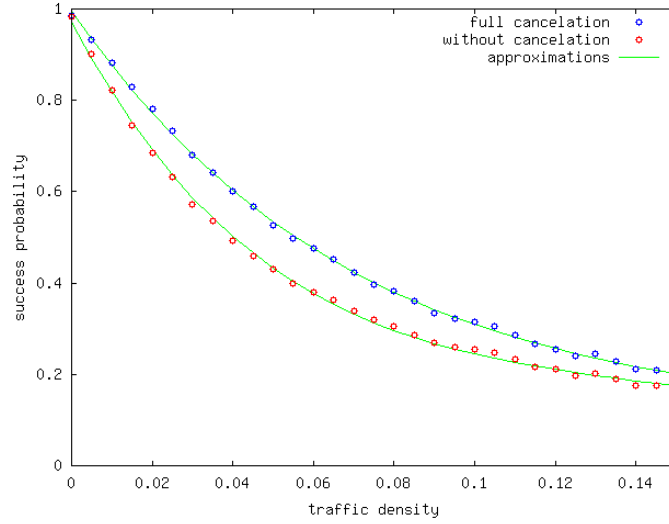
### 6.1 Results

In the following, no cancelation and full cancelation in different scenarios are compared. In practical implementations it will never be possible to cancel out 100% of the interferences. Therefore, the results should be seen as an upper bound. The observed scenarios and their parameters are described in Section 3.3.

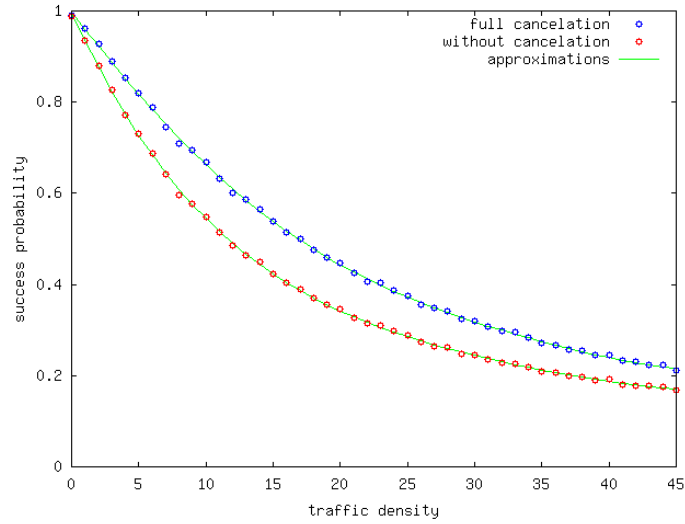
All simulations in this chapter are run multiple times. For each scenario, a confidence interval of 99% is achieved. The results in the plots are mean values and the confidence interval bounds are skipped for better readability.

Figure 6.1 shows different scenarios in which the path loss exponent changes. Via the path loss exponent the signal attenuation is defined over distance. A higher exponent results in smaller transmission and interference ranges. To keep the amount of interfering nodes comparable, the traffic density is adjusted to each case. A low path loss exponent causes interferences further away. The node density has to be kept small, otherwise it would

result in networks with bigger neighborhoods. This could easily result in a massive simulation effort. For higher exponents the opposite applies. In all cases, the traffic density starts at zero and goes up until the transmission success probability drops below 20%. Thereafter, the success probability only decreases slowly and the network is overloaded.

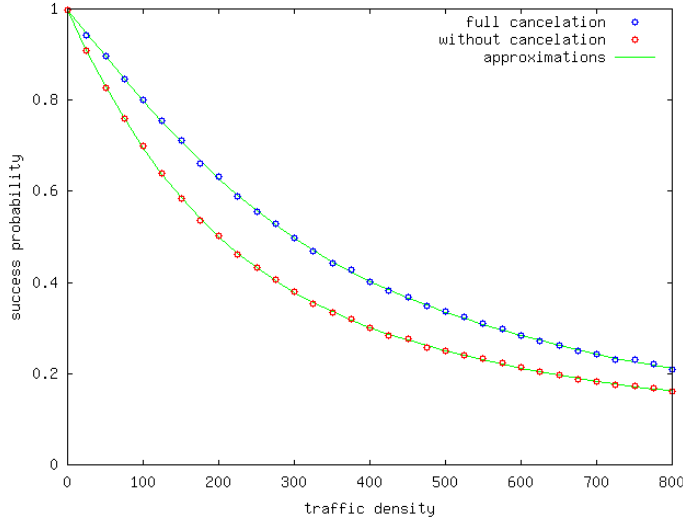


(a) path loss exponent  $n = 2$ , SINR=20, node density=6 nodes/km<sup>2</sup>, simulation time 300ms, simulation area 35 × 35km<sup>2</sup>.



(b) path loss exponent  $n = 3$ , SINR=20, node density=1.000 nodes/km<sup>2</sup>, simulation time 300ms, simulation area 2 × 2km<sup>2</sup>.

Figure 6.1: Receive probability in different path loss models.



(c) path loss exponent  $n = 4$ , SINR=20, node density=15.000 nodes/km<sup>2</sup>, simulation time 300ms, simulation area  $500 \times 500\text{m}^2$ .

Figure 6.1: Receive probability in different path loss models.

Best performance of the signal cancelation is observed when the network is moderately loaded. The performance gain is defined by the percentage points the successful transmission probability gains. In low loaded networks cancelation is not necessary. The amount of interference (from both, **DIRECT** and **RELAY** transmissions) is small. The probability that **RELAY** transmissions interfere and disturb the ongoing transmission is low. With additional load the amount of interfering transmissions increases. The performance gain reaches a maximum and then decreases again. At the peak a high amount of interfering **RELAY** transmissions are canceled and the remaining interferences do not disturb the transmission. After the peak, the amount of **DIRECT** transmissions increases so far that these transmissions alone make a successful transmission impossible. Signal cancelation shows almost no improvement anymore.

The performance gain is explicitly shown in Figure 6.2 and compared to the gain achieved by cooperative diversity. This figure shows only the scenario with a path loss exponent  $n = 3$ . For the other exponents the outcome is similar.

As seen in Figure 6.1, the plots for different path loss exponents look similar. The performance gain is therefore uncorrelated to the path loss exponent.

It depends only on how much the network is loaded. The fluctuations in the graph result from difficulties in the comparison.

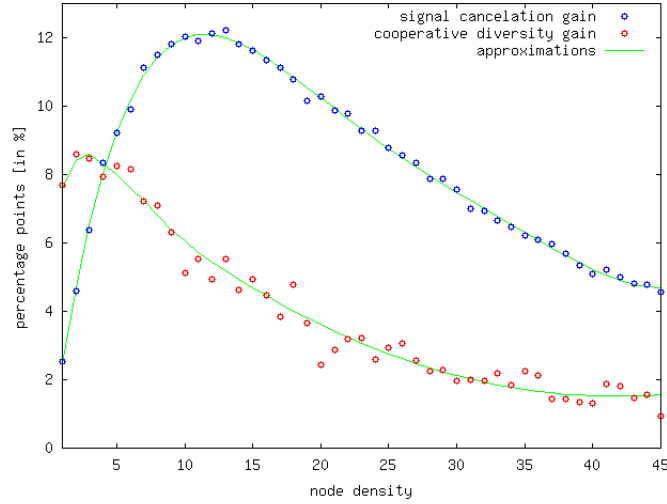


Figure 6.2: Performance gain of signal cancellation and cooperative diversity.

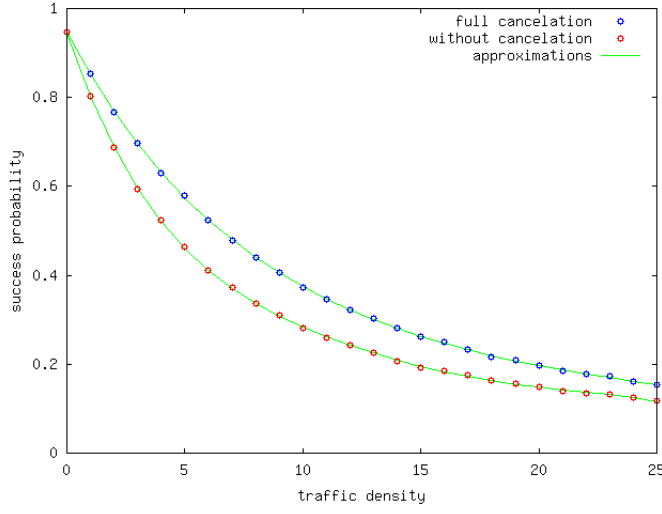
Figure 6.3 shows scenarios with different SINR threshold value. The path loss exponent  $n$  is set to 3, which is a typical value for outdoor scenarios, and sometimes it is even used for indoor. The figure shows plots for a SINR threshold of 10 and 30. The plot for a SINR threshold of 20 is depicted in 6.1b. The behavior shows similarities to the scenarios with different path loss exponents. Most performance gain is achieved in moderately loaded networks. The same arguments as stated before also apply in these scenarios.

## 6.2 Conclusion

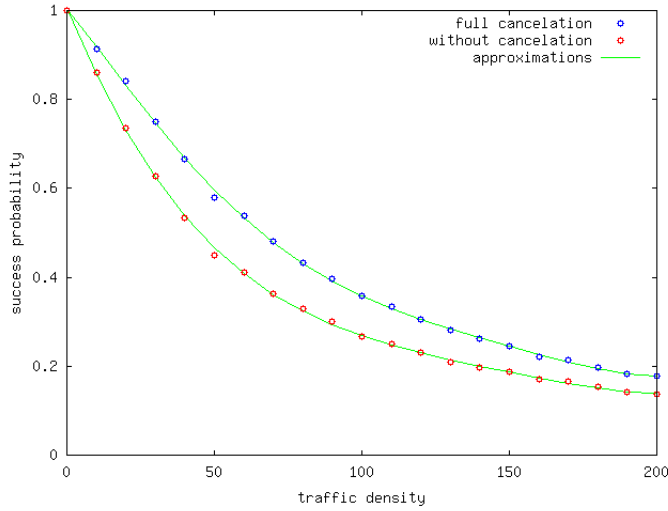
In Chapter 2 the current state of the art is presented. A short introduction to wireless networks is given at the beginning. Different effects that are encountered in radio wave propagation are explained in detail. Based on that knowledge, path loss models predicting average signals at certain distances are presented. Fluctuations in the signal strength mainly result due to multipath fading. This phenomenon is described in great detail and statistical models are given.

Furthermore, cooperative diversity methods are presented. The basic ideas of these techniques are explained. Cooperative relaying is split up into





(a) SINR=10, path loss exponent  $n = 3$ , node density=800 nodes/km<sup>2</sup>, simulation time 300ms, simulation area  $2 \times 2$ km<sup>2</sup>



(b) SINR=30, path loss exponent  $n = 3$ , node density=7.000 nodes/km<sup>2</sup>, simulation time 300ms, simulation area  $500 \times 500$ m<sup>2</sup>

Figure 6.3: Receive probability for different SINR thresholds

three major parts, namely relay selection, relay mode and combining techniques. Relay selection is one of the key issues to achieve improvements with cooperative diversity. Relay selection algorithms are split up into two main categories, preselection and on-demand selection. The selected relay then operates either in amplify-and-forward, decode-and-forward or in a hy-

brid mode. Finally, the two received data packets have to be combined at the destination node. Different combination techniques show different gains and implementation complexities. The most complex technique is called maximum-ratio combining which gives the best results.

At the end of the chapter, a range of simulation tools are presented. Each tool is shortly introduced, and pros and cons are discussed. Each simulation tool has its own focus and sometimes they are difficult to compare. A general discussion about simulation tools and their pitfalls concludes the chapter.

Chapter 3 gives an introduction to the project and describes its ideas. A detailed presentation of the scope and aim of the project is given. All assumptions, which are taken in the project, are summarized and argued for. Moreover, the range of scenarios is presented. All necessary parameters are presented and discussed.

Chapter 4 presents a new self developed simulation tool. Since most simulation tools lack scalability and necessary simulation speed, a new simulation tool was developed. Like most other tools, it is based on the ideals of discrete event driven simulations. The main aspect is scalability and simulation speed. Therefore, it is kept as small and simple as possible. Only the required modules are implemented to keep the overhead minimal. All configuration parameters are defined before the simulation run and are written into a configuration file. Most random decisions are stated in this file allowing to rerun the same simulation several times. This is a major necessity during the development phase. For hunting down bugs and analyzing special behavior, there is a need to rerun certain scenarios with exactly the same setup. The development is assisted by a graphical user interface. It displays the state of the network as well as the next pending events. The user can go through the simulation run step by step and observe the behavior. For final simulation results, the user interface is deactivated to avoid a slowdown of the simulation. In this mode, simulations up to a few thousands nodes and transmissions are possible in a few minutes real time.

Chapter 5 validates the results of the simulation tool. Major drawbacks of self developed simulation tools are comparability and believability. Therefore, this chapter validates the results of the simulation tool. The validation starts by a simple distribution of nodes in the two-dimensional simulation space. Next, the pdf of signal strengths based on the simple path loss model is derived. Multipath fading is statistically approximated by the Rayleigh distribution and incorporated into the pdf of signal strengths. Since most

of the time there are multiple interfering transmissions, these transmissions are summed up. A transmission success probability for each case (with a certain number of interfering transmissions) is given. Based on the derived distribution of these different cases, a final transmission success probability can be computed. Finally, these results are compared to the simulation. The comparison of two different scenarios between the simulation and numerical results shows a high consistency (less than half a percentage point).

In the last chapter the simulation results are presented. A wide range of scenarios are observed to capture all variations of parameters. Plots for different path loss factors and SINR thresholds are given. In each plot the success probabilities of a transmission with and without cancelation are compared. Signal cancelation shows similar behavior for different parameters. The influence mainly depends on the network load. Best results are achieved in moderately loaded networks. As the network load increases, the performance gain of signal cancelation slowly decreases. The overall performance gain is in the same range as for cooperative relaying.

This work gives a basic idea about signal cancelation in cooperative networks. The basic assumptions of signal cancelation and relay selection are rather optimal in the sense of a high performance gain. These abstract assumptions may not be achievable in practical implementations. They will most likely result in worse performance gain. The results must therefore be seen as upper bounds. Nevertheless, the gain is pretty high, also in comparison to cooperative relaying. If only a fraction of the gain can be achieved in practical implementations, the idea seems to perform well.

The performance gain of cooperative diversity starts to decrease when the network load increases. Exactly at this point, signal cancelation shows its best performance. Together, cooperative diversity and signal cancelation result in best performance gains in low and moderately loaded networks. Due to this results it is definitely worth the effort putting further effort into additional research.

## 6.3 Future Work

Since the results are promising, it is worth investing further research work in this area. It is most important to develop more appropriate cancelation methods. In the work at hand, only a basic assumption is analyzed. A more

sophisticated assumption would be to select the cancelation factor based on signal strength of the previously received **DIRECT** transmission.

Additionally the question of how much the underlying relay selection mechanism influences the performance gain arises. The most interesting variation of relay selection is if the relay is only used when needed. In this case the amount of interfering **RELAY** transmissions drops. With a lower amount of **RELAY** transmissions the gain by signal cancelation clearly decreases. This may reduce the gain until it is not worth the effort of implementing the signal cancelation.

Another issue, which is not considered is the influence of multi hop traffic. In case of multi hop traffic the idea of signal cancellation can be further extended. If a data packet travels over multiple hops from the source to the destination it only differs in the header part. The payload of the data packet stays the same all the time. Here, the question arises if it is possible to apply similar signal cancelation.

## List of Figures

2.1	The three basic propagation mechanisms, [1] page 31. . . . .	6
2.2	The three basic propagation mechanisms, [2] page 38f . . . . .	8
2.3	Superimposed signals . . . . .	11
2.4	Rayleigh distribution for different $\sigma$ . . . . .	13
2.5	Multipath fading . . . . .	14
2.6	Difference of SISO and MIMO . . . . .	15
2.7	Cooperative Diversity . . . . .	15
2.8	SNR gain of different diversity techniques . . . . .	20
2.9	Used simulations tools [36] . . . . .	22
3.1	Basic example . . . . .	24
4.1	Network layers . . . . .	39
4.2	Screenshot of the graphical interface . . . . .	46
4.3	Thick red circle indicating the simulation running in real time	47
4.4	Pending events . . . . .	47
5.1	pdf $f_x(x)$ and $g(x)$ [43] page 130 . . . . .	55
5.2	Pdf of signal strength of one signal without multipath fading .	59
5.3	pdf of signal strength of one signal with multipath fading . . .	60

## List of Figures

---

5.4	Pdfs of interference strengths . . . . .	63
5.5	Two interfering signals with and without noise . . . . .	64
5.6	Example of a transmission . . . . .	65
5.7	Pdf of interfering <b>DIRECT</b> transmissions . . . . .	68
5.8	Different cases of interfering transmissions . . . . .	71
6.1	Receive probability in different path loss models. . . . .	74
6.1	Receive probability in different path loss models. . . . .	75
6.2	Performance gain of signal cancelation and cooperative diversity. . . . .	76
6.3	Receive probability for different SINR thresholds . . . . .	77

## Bibliography

- [1] T. S. Rappaport. *Wireless Communications: Principles and Practice (2nd Edition)*. Prentice Hall PTR, 2002.
- [2] J. Schiller. *Mobile Communications*. Addison Wesley, second edition, May 2003.
- [3] A. Neskovic, N. Neskovic, and G. Paunovic. Modern Approaches In Modeling of Mobile Radio Systems Propagation Environment. *IEEE Communications Surveys*, 2000.
- [4] V. Erceg, S. Ghassemzadeh, M. Taylor, D. Li, and D. L. Schilling. Urban/suburban out-of-sight propagation modeling. *Communications Magazine, IEEE*, Jun 1992.
- [5] B. Goldberg and H.S. Bennett. *Communications channels: characterization and behavior*. IEEE Press, 1976.
- [6] J. B. Andersen, T. S. Rappaport, and S. Yoshida. Propagation measurements and models for wireless communications channels. *Communications Magazine, IEEE*, 1995.
- [7] S. Y. Seidel, T. S. Rappaport, S. Jain, M. L. Lord, and R. Singh. Path loss, scattering and multipath delay statistics in four european cities for digital cellular and microcellular radiotelephone. *Vehicular Technology, IEEE Transactions on*, Nov 1991.

- [8] S. O. Rice. Mathematical Analysis of Random Noise. *Bell Systems Tech.*, 1944.
- [9] D. Sexton, M. Mahony, M. Lapinski, and J. Werb. Radio Channel Quality in Industrial Wireless Sensor Networks. *Sensors for Industry Conference, 2005*, Feb. 2005.
- [10] M. A. Beach, D. P. McNamara, and P. Karlsson. Development of a channel measurement system for multiple-input multiple-output (MIMO) applications. *IST Mobile Communications Summit*, 2000.
- [11] J. N. Laneman, D. N. C. Tse, and G. W. Wornell. Cooperative diversity in wireless networks: efficient protocols and outage behavior. *Information Theory, IEEE Transactions on*, 2004.
- [12] A. Sendonaris, E. Erkip, and B. Aazhang. User cooperation diversity. Part I. System description. *Communications, IEEE Transactions on*, 2003.
- [13] A. Bletsas, A. Khisti, Student Member, and D. P. Reed. A simple cooperative diversity method based on network path selection. *IEEE J. Select. Areas Commun*, 2006.
- [14] Aggelos Bletsas, Ashish Khisti, David P. Reed, and Andrew Lippman. A Simple Cooperative Diversity Method Based on Network Path Selection, 2005.
- [15] Yi Shi, Sushant Sharma, Y. Thomas Hou, and Sastry Kompella. Optimal relay assignment for cooperative communications. In *MobiHoc '08: Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*, pages 3–12. ACM, 2008.
- [16] H. D. Ferriere, D. Estrin, and M. Vetterli. Packet combining in sensor networks. In *SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems*, pages 102–115, New York, NY, USA, 2005. ACM Press.
- [17] B. Z., Student Member, and M. C. Valenti. Practical relay networks: A generalization of hybrid-ARQ. *IEEE J. Select. Areas Commun*, 2005.
- [18] Y. Zhao, R. Adve, and T. J. Lim. Improving amplify-and-forward relay networks: optimal power allocation versus selection. *Wireless Communications, IEEE Transactions on*, 2007.



- [19] J. N. Laneman, G. W. Wornell, and D. N. C. Tse. An efficient protocol for realizing cooperative diversity in wireless networks. In *Information Theory. Proceedings. IEEE International Symposium on*, 2001.
- [20] M. Yu and J. t. Li. Is amplify-and-forward practically better than decode-and-forward or vice versa. In *Proc. IEEE ICASSP*, pages 365–368, 2005.
- [21] D. G. Brennan. Linear Diversity Combining Techniques. In *Proc. IRE*, pages 1075–1102, 1959.
- [22] J. Proakis. *Digital Communications*. McGraw-Hill Science/Engineering/Math, August 2000.
- [23] A. Nasipuri, J. Zhuang, and S. Das. A multichannel CSMAMAC protocol for multihop wireless networks, 1999.
- [24] NS-2 [online, cited 13.01.2009].  
<http://www.isi.edu/nsnam/ns/>.
- [25] The REAL network simulator [online, cited 13.02.2009].  
<http://www.cs.cornell.edu/skeshav/real/overview.html>.
- [26] The Rice University Monarch Project: Mobile Networking Architectures [online, cited 13.02.2009].  
<http://www.monarch.cs.rice.edu/>.
- [27] D. Cavin, Y. Sasson, and A. Schiper. On the accuracy of MANET simulators. In *POMC '02: Proceedings of the second ACM international workshop on Principles of mobile computing*, pages 38–43, New York, NY, USA, 2002. ACM Press.
- [28] OMNeT++ Community Site [online, cited 13.01.2009].  
<http://www.omnetpp.org/>.
- [29] Mobility Framework for OMNeT++ [online, cited 13.01.2009].  
<http://mobility-fw.sourceforge.net/>.
- [30] A. Varga and R. Hornig. An overview of the OMNeT++ simulation environment. In *Simutools '08: Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks*

- and systems & workshops*, pages 1–10, ICST, Brussels, Belgium, Belgium, 2008. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).
- [31] GloMoSim [online, cited 13.02.2009].  
<http://pcl.cs.ucla.edu/projects/glomosim/>.
- [32] QualNet [online, cited 13.02.2009].  
<http://www.qualnet.com/products/QualNet/>.
- [33] jProwler [online, cited 09.03.2009].  
<http://w3.isis.vanderbilt.edu/projects/nest/jprowler/>.
- [34] Xu Su and Rajendra V. Boppana. On the impact of noise on mobile ad hoc networks. In *IWCMC '07: Proceedings of the 2007 international conference on Wireless communications and mobile computing*, pages 208–213, New York, NY, USA, 2007. ACM.
- [35] J. Mullen and H. Huang. Impact of multipath fading in wireless ad hoc networks. In *PE-WASUN '05: Proceedings of the 2nd ACM international workshop on Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks*, pages 181–188, New York, NY, USA, 2005. ACM Press.
- [36] S. Kurkowski, T. Camp, and M. Colagrosso. MANET simulation studies: the incredibles. *SIGMOBILE Mob. Comput. Commun. Rev.*, October 2005.
- [37] Crossbow. *Mica2 Wireless Measurement System*.
- [38] M. Gast. *802.11 Wireless Networks: The Definitive Guide, Second Edition (Definitive Guide)*. O'Reilly Media, Inc., April 2005.
- [39] The GNU Standard C++ Library [online, cited 25.01.2009].  
<http://gcc.gnu.org/libstdc++/index.html>.
- [40] J. Elsing. *Das OSI-Schichtenmodell: Grundlagen und Anwendung der X. 200*. IWT-Verlag, 1991.
- [41] N. Abramson. Development of the ALOHANET. *Information Theory, IEEE Transactions on*, 1985.

- [42] M. Schwartz. *Mobile Wireless Communications*. Cambridge University Press, New York, NY, USA, 2004.
- [43] A. Papoulis. *Probability, Random Variables, and Stochastic Processes*. Mc-Graw Hill, 2002.
- [44] P. Omiyi, H. Haas, and G. Auer. Analysis of TDD Cellular Interference Mitigation Using Busy-Bursts. *Wireless Communications , IEEE Transactions on*, 2007.
- [45] Wolfram Mathematica [online, cited 17.02.2009].  
<http://www.wolfram.com/products/mathematica/>.
- [46] S. Wolfram. *The Mathematica Book, Fifth Edition*. Wolfram Media, August 2003.