Vorlesung: Collaborative transmission in wireless sensor networks

Wintersemester 2009/10

Version: February 9, 2010 ($v0.0.1 + \varepsilon$)

Veranstalter: Stephan Sigg

Technische Universität Braunschweig
Institut für Betriebssysteme und Rechnerverbund
Verteilte und Ubiquitäre Systeme

D-38106 Braunschweig

Acknowledgements

I would like to thank the students of my lecture 'Collaborative transmission in wireless sensor networks' at the Technische Universität Braunschweig in the winter term 2009/2010 for their patience, thoughtful questions and constructive feedback and discussions on the topic. In particular, Timo Schulz has spotted many spelling errors in prior versions of this document.

Some findings presented in this document are the result of student thesis at the Technische Universität Braunschweig in 2009. Section 8.3.3 was predominantly influenced by the work of Rayan Merched El Masri. The results in section 8.4.2 have been derived by Julian Ristau.

Contents

Abbreviations and Notation

The following figures and notations are utilised throughout this document. It has been attempted to keep the standard notation from the literature whenever possible. However, since diverse scientific areas are covered, the notation had to be adapted in order to provide an unambiguous notation. The page number given in the table refers to the first occurrence of the mentioned construct.

Notation	Explanation	Page
A	Region where nodes of a sensor network are placed	40
B	Bandwidth	53
c	Speed of light ($3 \cdot 10^8 \frac{m}{s}$)	47
d	Distance	49
$E[x]$	The expectation of a random variable x	66
f	Frequency	47
$\mathcal{F}$	Fitness function	82
G_{RX}	Gain of the receive antenna	49
G_{TX}	Gain of the transmit antenna	49
GSM	Global System for Mobile communications	54
$\Im(s)$	Imaginary part of a complex signal s	
IAC	inquiry access codes	55
ISM	Industrial, Scientific, Medical band	55
J	Joule	53
K	Kalvin	53
κ	Boltzmann constant	53
N	Network size	40
P_N	Thermal noise power	53
P_{RX}	Received signal power	49
P_{TX}	Transmission power	49
$P(x)$	Probability of an event x	64
$P(\chi_1 \vert \chi_2)$	The conditional probability of two events χ_1 and χ_2 with $P(\chi_2) > 0$	66
$\mathcal{P}$	An optimisation problem	82

Notation	Explanation	Page
$\Re(s)$	Real part of a complex signal s	48
Π	Sample space	64
$R_k(t)$	The reliability or fault tolerance of a sensor node	39
R	Transmission range of a sensor network	40
RSS	Received Signal Strength	49
T	Temperature in Kalvin	53
α	Path-loss exponent	50
γ	Phase offset of a transmit signal	47
λ	Wavelength of a transmit signal	47
$\mu(R)$	Density of a sensor network with transmission range R	
μ	Population size of an evolutionary algorithm	
s^*	Complex conjugate of s	
ν	Offspring population size of an evolutionary algorithm. Note: In the literature, typically λ denotes the offspring population size	72
υ	Failure rate	40
η	Density of a sensor network	40
GPS	Global Positioning System	15
GSM	Global System for Mobile Communications	15
$var[\chi]$	The variance of a random variable χ: $E[(\chi - E[\chi])^2]$	67
x	A sample point for a random experiment	62
ID	Identification	25
MIT	Massachusetts Institute of Technology	23
RMSE	Root of the Mean Squared Error	109
S	A search space	71
UbiComp	Ubiquitous Computing	16
UMTS	Universal Mobile Telecommunications System	15
$\overrightarrow{v}$	A vector $v = (v_1, \ldots, v_\kappa)$	
WLAN	Wireless Local Area Network	15

1 Motivation

> *In the long history of humankind (and animal kind, too) those who learned to collaborate and improvise most effectively have prevailed*
>
> (C. Darwin)

In recent years, sensor nodes of extreme tiny size are envisioned [1, 2, 3]. In [4], for example, applications for square-millimetre sized nodes that seamlessly integrate into an environment are detailed. At these small form-factors transmission power of wireless nodes is restricted to several microwatts. Communication between a single node and a remote receiver is then only feasible at short distances. It is possible, however, to increase the maximum transmission range by cooperatively transmitting information from distinct nodes of a network [5, 6]. The basic idea is to superimpose identical RF carrier signal components from various transmitters that function as a distributed beamformer. When the relative phase offset of these carrier signal components at a remote receiver is small, the signal strength of the received sum signal is improved. Cooperation can improve the capacity and robustness of a network of transmitters [7, 8] and decreases the average energy consumption per node [9, 10, 11].

Related research branches are cooperative transmission [12], collaborative transmission [13, 14], distributed adaptive beamforming [15, 16, 17, 18], collaborative beamforming [19] or cooperative/virtual MIMO for wireless sensor networks [20, 21, 22, 23]. One approach is to utilise neighbouring nodes as relays [24, 25, 26] as proposed by Cover and El Gamal in [27]. Cooperative transmission is then achieved by Multi-hop [28, 29, 30] or data flooding [31, 32, 33, 34] approaches. The general idea of multi-hop relaying based on the physical channel is to retransmit received messages by a relay node so that the destination will receive not only the message from the source destination but also from the relay. In data flooding approaches, a node will retransmit a received message at its reception. It has been shown that the approach outperforms non-cooperative multi-hop schemes significantly. It was derived that the average energy consumption of nodes is decreased [9, 10] and the transmission time is reduced compared to traditional transmission protocols in wireless sensor networks [35].

In these approaches, nodes are not tightly synchronised and transmission may be asynchronous. This, however, is achieved by virtual MIMO techniques. In virtual MIMO for wireless sensor networks, single antenna nodes are cooperating to establish a multiple an-

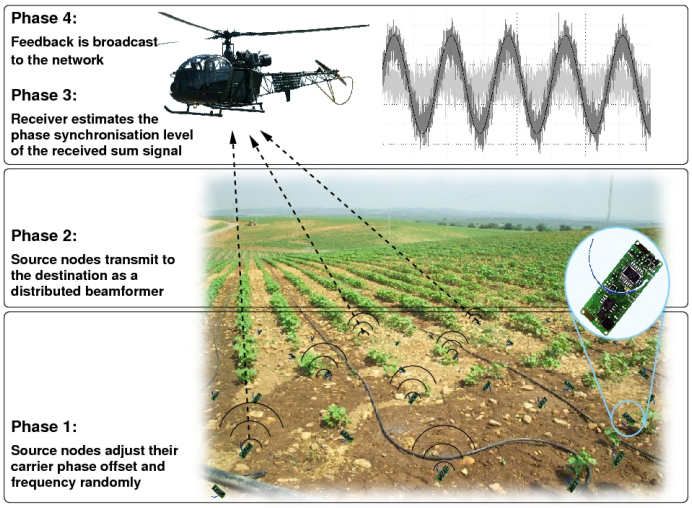


Figure 1.1: Schematic illustration of feedback based distributed adaptive beamforming in wireless sensor networks

tenna wireless sensor network [21, 20, 22]. Virtual MIMO has capabilities to adjust to different frequencies and is highly energy efficient [23, 11]. However, the implementation of MIMO capabilities in WSNs requires accurate time synchronisation, complex transceiver circuits and signal processing that might surcharge the power consumption and processing capabilities of simple sensor nodes.

Other solutions proposed are open-loop synchronisation methods as round-trip synchronisation based [36, 37, 38]. In this scheme, the destination sends beacons in opposed directions along a multi-hop circle in which each of the nodes appends its part of the overall message to the beacons. Beamforming is achieved when the processing time along the multi-hop chain is identical in both directions. This approach, however, does not scale well with the size of a network.

Closed loop feedback approaches include full-feedback techniques, in which carrier synchronisation is achieved in a master-slave manner. The phase-offset between the destination and a source node is corrected by the receiver node. Diversity between RF-transmit signal components is achieved over CDMA channels [39]. This approach is applicable only to small network sizes and requires sophisticated processing capabilities at the source nodes.

A more simple and less resource demanding implementation is the one-bit feedback based closed-loop synchronisation considered in [39, 40]. The authors describe an iterative process in which n source nodes $i \in [1, \ldots, n]$ randomly adapt the phases γ_i of their carrier signal $\Re\left(m(t)e^{j(2\pi(f_c+f_i)t+\gamma_i)}\right)$. Here, f_i denotes the frequency offset of node i to a common carrier frequency f_c. Initially, i.i.d. phase offsets γ_i of carrier signals are assumed. When a receiver requests a transmission from the network, carrier phases are synchronised in an iterative process (cf. figure 7.10).

1. Each source node i adjusts its carrier phase offset γ_i and frequency offset f_i randomly.

2. The source nodes transmit to the destination simultaneously as a distributed beamformer.

3. The receiver estimates the level of phase synchronisation of the received sum signal (e.g. by the SNR).

4. This value is broadcast as a feedback to the network. Nodes interpret this feedback and adapt their phase adjustments accordingly.

These four steps are iterated repeatedly until a stop criteria is met (e.g. maximum iteration count or sufficient synchronisation). The process has been studied by various authors [41, 42, 43, 13] where the approaches proposed differ in the implementation of the first and the fourth step specified above. The authors of [43] show that it is possible to reduce the number of transmitting nodes in a random process and still achieve synchronisation among all nodes.

In [41, 42, 43] a process is described in which each node alters its carrier phase offset γ_i according to a normal distribution with small variance in step one. In [13] a uniform distribution is utilised but the probability for one node to mutate is low. Only in [42] not only the phase but also frequency is adapted.

This lecture is focused on cooperative transmission schemes in wireless sensor networks. Distinct nodes in a network of nodes cooperate in their transmission of data. As detailed above, this cooperation may differ in its exact implementation for various approaches. Some approaches require inter-node communication while others don't. For some approaches the aim is to reduce the failure probability through multiple transmissions while others aim to improve the signal strength. Figure 1.2 depicts for the generic scenario introduced above the organisation of the lecture.

First, the theoretical background required to understand cooperative transmission scenarios is provided in chapters 2 through 4. In Chapters 5 and 6, concepts required for the application and analysis of the algorithms for cooperative transmission in wireless sensor networks in chapters 7 and 8 are introduced. The focus of the algorithms presented is on algorithms for distributed adaptive transmit beamforming.

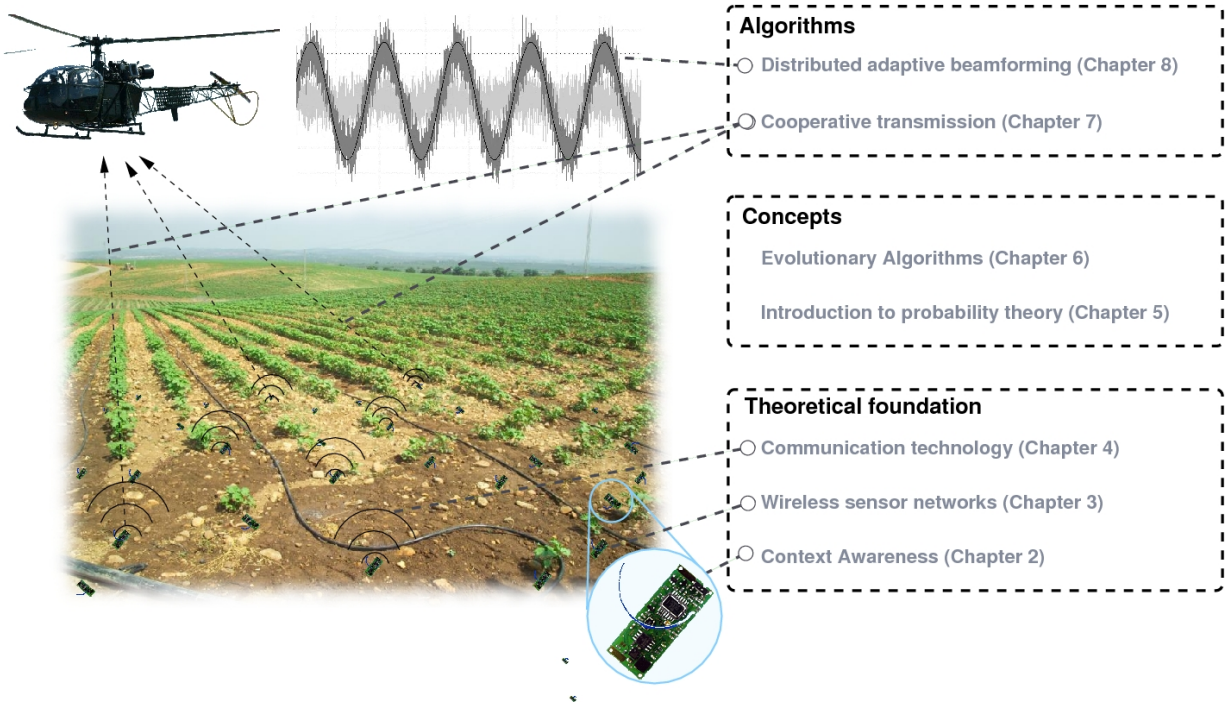


Figure 1.2: Possible scenario for distributed adaptive transmit beamforming

2 Context-awareness

> *Increasingly, the bottleneck in computing is not its disk capacity, processor speed or communication bandwidth, but rather the limited resource of human attention*
>
> (A. Garlan, Toward Distraction-Free Pervasive Computing [44])

The vision of context-awareness is that applications become sensitive to environmental stimuli and adapt their behaviour to the current situation. This vision was far ahead of the technology of the time when it was first studied in research laboratories and the details necessary to implement the vision were seldom provided. With improved technology we have seen prototype applications of isolated ideas from the Context-aware vision become implemented. The first of these are probably the Xerox PARCTAB [45] and the media cup [46].

In recent years, but to a limited degree, we have already seen context-aware features in consumer products. Mobile devices that adjust their screen brightness to the environmental light, devices that automatically rotate the screen when the device is turned, watches that automatically adjust to local time and messages that alert users when their screen work time exceeds a certain limit, are just some examples.

While these applications are quite limited and stand alone, we see more advanced and better integrated context-aware features in multifarious new products. The most versatile and widely used device type for context-aware applications are recent mobile phones. The capabilities of these devices quickly increase as new interfaces to the environment are constantly added. Apart from technologies as basic as microphones, speakers and GSM, we now expect also infrared, bluetooth and a camera in mobile devices. New air interfaces as WLAN or UMTS are added, as well as light sensors, accelerators, touch screens and to an increasing degree GPS receivers. Most of these technologies remain unused for a great part of the time. This multitude of sensors, however, provides a rich environment in which context-aware applications can be taken to the next evolutionary stage. Context-awareness, nowadays, still holds great potential before the development comes anywhere close to the vision of a ubiquitous world that is saturated with context-aware devices.

In recent years, applications and devices have undergone serious changes that move them away from static, reactive entities towards a more environment responsive design. We see applications act in an increasingly adaptive and situation-dependent way. Applications are

able to infer the needs and requirements in a given situation. It is commonly agreed that the general setting a user is in also influences her needs at that point in time. Lucy Suchman [47] states that every course of action is highly dependent upon its material and social circumstances regarding interactions between actors and the environment. To become able to react to the general setting an application is executed in, the design paradigm for applications is shifting from an application-centric approach to an environment-centric approach. Applications become integrated into the environment and react to environmental stimuli. In order to improve the application and device behaviour in this direction, further and in most cases novel sources of information are investigated.

The input provided to an application or device is no longer restricted to explicit instructions on a common user interface. Instead, the interface utilised for the acquisition of input information is extended and coupled by an interface to the environment. The behaviour of applications evolves from a mere passive, input dependent way to an active, environment and situation guided operation.

Information about the environment and situation is extracted and interpreted to trigger situation dependent actions that shall for example provide the user with a richer experience that is adapted to her personal needs. Due to this additional information, the required explicit interaction with an application can be minimised or at least reduced. The computing experience hereby gets increasingly unobtrusive and becomes ubiquitous.

In general, this computing paradigm is referred to as context-awareness or context computing but is described by various further titles. People have been quite creative in finding descriptive names for scenarios similar to the one described above. A (most certainly not exhaustive) set of terms associated with ideas related to context computing is depicted in figure 2.1. A similar list can also be found in [48]

While these catchwords have partly redundant but not identical meanings, a common vision of future computing is captured by all these descriptions. Probably the first study on context-aware computing was the Olivetti Active Badge [49]. Following this pioneering work, numerous further concepts and ideas have been discussed by various research groups.

Recently, the focus for context awareness has shifted from a single device sinsing its environment to an environment capable of sensing and possibly interpreting and reacting on the sensed information. Sensor nodes i.e. tiny computing devices with communication and sensing capabilities become integrated in the environment.

2.1 Context-aware computing

The vision of a world where computing devices seamlessly integrate into the real world was first introduced by Mark Weiser in 1988. He illustrates and describes his vision of future computing in [50]. Computing in his vision is no longer restricted to a single machine but may move off one machine and onto another one at execution time. Ubiquitous computing also incorporates an awareness of the environment the computer is situated in. Furthermore, following the vision of ubiquitous computing, computing becomes invisible and omnipresent simultaneously. Smallest scale computing devices that enrich the envi-

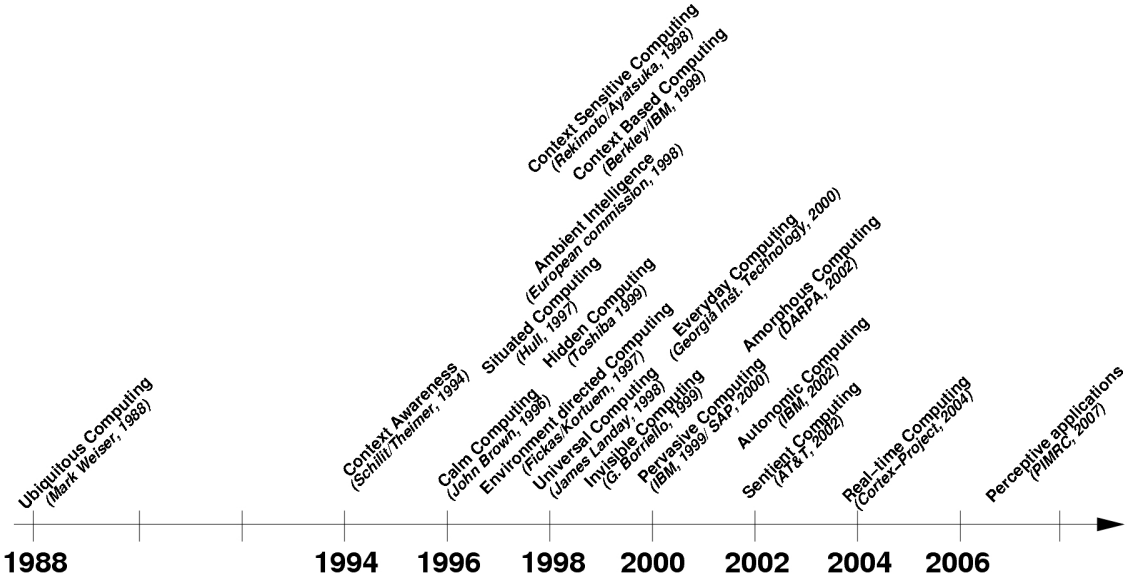


Figure 2.1: Concepts related to Ubiquitous computing

ronment communicate with each other and assist a user unnoticed. Weiser argues that a computer might adapt its behaviour in a significant way if it knows where it is located. As Weiser states, this reaction to the environment does not require artificial intelligence.

Weiser observes the paradox that computing devices are becoming cheaper, smaller and more powerful at the same time. Tiny computing devices become cheap enough to be bought in raw amounts and small enough to be integrated in virtually every real world object.

Weiser envisions that these devices, equipped with sensing technology and communication interfaces are able to communicate with each other and to acquire and spread information on devices, persons and objects in their proximity. This information can then be utilised to enhance the computing experience of a user.

The first experiments with computers aware of their environment have been conducted in the early 1990's. The active badge location system by Olivetti Research [49] and the Xerox PARCTAB location system by Xerox laboratories [45] demonstrated how small mobile devices operate together.

Although the sources of information utilised in these experiments were restricted to location sensors, the basic new concept and possibility inspired numerous people to focus their research on this field.

2.1.1 Definitions of context

Definitions of context are numerous and diverse even when the focus is restricted to computer sciences. In his comprehensive discussion "What we talk about when we talk about context"[51] Paul Dourish attempts to exhaustively discuss several aspects of context and also reviews various definitions of context.

The concept of context in conjunction with context-aware computing was first formulated by Schilit and Theimer in 1994 [52]. Following their definition, a software that "adapts according to its location of use, the collection of nearby people and objects as well as changes to those objects over time" is considered to be context-aware. Later on, Schilit refined this definition by defining context categories in [53]. These categories are 'user context', 'physical context' and 'computing context'. As further categories, Brown added information about the time [54], while Pascoe also considered the blood pressure of users [55]. Dey took the latter proposal to a broader scope by considering emotions and the focus of attention [56].

At about the same time, Albrecht Schmidt, Michael Beigl and Hans W. Gellersen recognised that most so-called context-aware applications are in fact location-aware [57]. Hence, they are considering only location as an aspect of the context. The assertion of the authors is that applications implemented on mobile devices might significantly benefit from a wider understanding of context. Furthermore, they introduce a working model for context and discuss mechanisms to acquire other aspects of context beside location.

In their working model for context, they propose that a context describes a situation and the environment a device or user is located in. They state that a context shall have a set of relevant aspects to which they refer as features.

These features are ordered hierarchically. At the top level a distinction between human factors and physical environment is made. Further, finer grained sub-division of these top-level categories are also proposed. Finally, an overview of available sensor types and contexts obtained from these sensors is given.

As a prerequisite to a definition of context-awareness, Anind K. Dey formulated a definition of context, that is most commonly used today [58].

Definition 2.1.1 : User context

> *Context is any information that can be used to characterise the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and application themselves.*

This definition, while useful, is quite abstract and gives no hint on the actual representation of context in a computing system. For this reason, several authors express criticism considering this definition. As Jani Mäntyjärvi has already stated in [59], this context definition does not result in a more exact definition of context since the abstraction is shifted from context to information.

Karen Henricksen follows the same line of argumentation by remarking that the definition remains too imprecise, since a clear separation of the concepts of context, context modelling and context information is not provided. Henricksen refines the definition of context given by Dey as the set of circumstances surrounding a task that are potentially relevant for its completion [60]. Furthermore, in the model of Henricksen, a context model identifies a subset of the context that is realistically attainable from sensors, applications and users. Following her discussion, context information describes a set of data that was gathered from sensors and users and that conforms to a context model.

However, the discussion about a most suitable definition is not settled yet. In 2000, Lieberman and Selker defined context to be any input other than the explicit input and output [61]. Other projects refine the definition of context to their individual needs. In [62] for example, the definition of Dey is refined by adding the concept of a sentient object.

2.1.2 Context-awareness

Intuitively, applications that utilise context data are context-aware. However, similar to the lively discussion on a definition of context, several definitions for context-awareness have been given in the literature. This section briefly reviews this ongoing discussion.

In [52] Schilit and Theimer formulated a first definition of context-awareness. Following this definition, “Applications are context-aware when they adapt themselves to context”.

In 1998 Pascoe argues that context-aware computing is the ability of devices to detect, sense, interpret and respond to changes in the user’s environment and computing devices themselves [63]. The authors of [64] define context-awareness as the automation of a software system based on knowledge of the user’s context. Several other similar definitions

treat it as applications' ability to adapt or change their operation dynamically according to the state of the application and the user [52, 54, 65].

Later, Dey argued that the existing definitions did not fit to various applications developed at that time that were intended to be context-aware and consequently stated a more general definition of context-aware systems in [58].

Definition 2.1.2 : Context-awareness

> *A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task.*

This discussion is not closed yet as several research groups refine the definition so that it best suits their needs (cf. [62]).

2.1.3 Context processing

Context is an abstract concept to describe a major input of ubiquitous computing applications. However, we cannot build applications with this theoretical construct. The questions are how context can be obtained from the available information sources, in which way context is represented in applications and how context can be further processed. This section discusses popular approaches to these questions.

Various authors propose to pre-process sensor output in order to prepare the data for further computation. Anind K. Dey argues that one of the main reasons why context is not used in applications is because no common way to acquire and handle context is specified [58]. He proposes to separate the context acquisition from the context utilisation process. Dey distinguishes between two basic forms of context. Raw or low-level context data that is directly acquired by sensors and richer or higher-level forms of information. A similar distinction is also made by Guanling Chen [66]. However, no concrete specification of these notions is given.

Albrecht Schmidt on the other hand argues that it is simpler to implement context-aware systems using contexts on entity level [67]. With the notion 'entity level', Schmidt refers to context data that is not further processed or aggregated after it has been obtained from context sources. Furthermore, intrinsic properties of sensors are utilised in the context modelling process. Schmidt refers to this approach as the concept of bottom-up context-awareness. The main research focus of Schmidt is related to context acquisition from a variety of simple sensors. He defines simple sensors as low-end, low-price computing and communication technology.

These ideas are utilised by Johan Himberg. Himberg studies data mining and visualisation for context-awareness and personalisation [68]. He especially focuses on sensor data captured by on-board sensors of mobile phones. He investigates how to infer context from features derived from the sensor signals. Johan Himberg especially only utilises simple statistical methods in order to reach his aim.

An approach focused on the whole process of context inference is proposed by Jani Mäntyjärvi. Mäntyjärvi considers the problem, how low-level contexts can be obtained

from raw sensor data [59]. This problem is basically related to the extraction of features from information sources. For each context a set of features is relevant that determines the context. After the feature inference process, Mäntyjärvi composes the sampled features to obtain a more expressive description of a context. This operation is considered as the processing of low-level contexts to obtain high-level contexts.

Mäntyjärvi presents a procedure for sensor-based context recognition. This approach is referred to by him as bottom-up approach, in contrast to a top-down approach that starts from the high-level context as it had been proposed by Dey in [58]. Included in this procedure is also a method to extract information on contexts and to convert it into a context representation. Following his definition, raw sensor data is sensor data like $24°C$, or 70% humidity. Low-level contexts are defined as pre-processed raw sensor data where the pre-processing may be constituted, for example, from noise removal, data calibration and reforming of data distributions. Generally, low-level contexts are conditions like 'warm' or 'normal humidity'. Higher level contexts are then created by an additional processing of low-level contexts that results in an action like 'having lunch'.

Main assumptions prior to his work are that sensors attached to computing devices have to be carefully chosen in order to be useful and that context actually can be recognised by sensor data.

The term context atom was introduced in [69] and has been used by Jani Mäntyjärvi, Johan Himberg and Pertti Huuskonen to describe basic context dimensions which are derived from low-level sensor data by pre-processing [70].

2.1.4 Frameworks and architectures for context-awareness

In order to facilitate the development of context-aware applications, several authors have proposed frameworks and architectures for this task.

In his PhD thesis in 1994 [71], Schilit concludes that traditional software approaches are not well-suited to building distributed mobile systems. The main reason for this dilemma is that applications are seldom designed to adapt their behaviour to the ever-changing mobile environment of the user in which they are executed. By designing an architecture that communicates context changes to the application, Schilit proposes a solution to this problem.

Additionally, Schilit identifies the problem that the user context may not be shared by distinct applications, although they are actually executed in the same user context. Schilit proposes the use of a user agent that administers the user context in order to provide a persistent dynamic context for all applications of the user.

Furthermore, he presents a system structure for use with context-aware systems. He recommends a distribution of system functions and designs protocols for communication between the entities.

These thoughts are further developed in the context toolkit that was introduced in 2000 [58]. It was proposed and developed by Anind K. Dey at the Georgia Institute of Technology. The context toolkit constitutes a conceptual framework that was designed to support the development of context-aware applications. It is widely accepted as a major reference

for context-aware computing. An important contribution of this framework is that it distinguishes between context sensing and context computing. Context sensing describes the process of acquiring information on contexts from sensors while context computing refers to the utilisation of acquired contexts. Basic components in this architecture are context widgets (encapsulated sensors), aggregators and interpreters. However, the Context Toolkit is not generally applicable for arbitrary context-aware applications since it exclusively features discrete contexts and does not consider unreliable or unavailable sensor information [72].

Later on, Albrecht Schmidt presented a "working model for context-aware mobile computing" which is basically an extensible tree structure [67]. The proposed hierarchy of features starts with distinguishing human factors and the physical environment and expands from there. One of the major contributions of his PhD thesis is a framework supporting design, simulation, implementation and maintenance of context acquisition systems in a distributed ubiquitous computing environment.

In 2003, Karen Henricksen introduced a novel characterisation of context data in ubiquitous computing environments [60]. Her introductory study of the ubiquitous computing environment especially focuses on challenges in providing computing applications in ubiquitous computing environments. These issues can be summarised as the autonomy of computing applications, dynamic computing environments, dynamic user requirements, scalability and resource limitations. Henricksen concludes that this set of challenges necessitates a new application design approach. Henricksen proposes a conceptual framework and a corresponding software architecture for context-aware application development.

This framework consists of programming models to be used for context-aware systems. Furthermore, Henricksen proposes the use of the Context Modelling Language (CML), a graphical notation of context that supports the specification of application requirements by the application designer.

In 2004 the Solar framework was presented by Chen [66]. It provides means to derive higher-level context from lower level sensor data.

The framework basically represents a network of nodes that interact with each other. It is scalable, supports mobility of nodes and is self managed.

Solar is designed as a service-oriented middleware in order to support the distribution of its components. The middleware supports sensors, as well as applications. Components and functions can be shared between applications. The data flow between sensors and applications may be composed as a multi-layered acyclic directed graph both at design time or at runtime.

Together with Solar, Chen provides a graph-based programming model, that can be utilised for the design of context-aware architectures.

2.1.5 Applications utilising context

Several applications that utilise context have been developed in recent years. In this section we introduce a set of applications that illustrate the uses and application fields of context-aware computing applications. The number of context-aware applications has reached

an immense quantity. It is beyond the scope of this document to present an exhaustive overview of these applications. The examples presented are chosen in order to illustrate the broad spectrum of approaches and to show the possibilities for context-aware applications.

With the MediaCup [46], Hans W. Gellersen, Michael Beigl and Holger Krall have presented a context-aware device that demonstrates one part of Mark Weiser's vision of ubiquitous computing. The MediaCup is a coffee cup that is enriched with sensing, processing and communication capabilities. The cup was developed to demonstrate how ordinary, everyday objects can be integrated into a ubiquitous computing environment. The context data obtained by the cup is related to the location of the cup, the temperature and some movement characteristics. This information is obtained by a temperature sensor and an acceleration sensor. Context information can be broadcast with the help of an infrared diode. The MediaCup has been utilised in research projects in order to provide a sense of a remote presence and in order to log user activity.

Another application proposed by Gellersen et al. is context acquisition based on load sensing [73]. With the help of pressure sensors in the floor of a room, the presence and location of objects and individuals can be tracked. Furthermore, it is shown that it is possible to distinguish between objects and that even movement of objects can be traced. The authors consider the use of load sensing in everyday environments as an approach to acquisition of contextual information in ubiquitous computing systems. It is demonstrated that load sensing is a practical source of contexts. It exemplifies how the position of objects and interaction events on a given surface can be sensed.

Various implemented context-aware applications have been developed by the Context-Aware Computing Group at the MIT[1]. An illustrative example is the 'Augmented Reality Kitchen' that monitors the state of objects in a kitchen in order to help the kitchen-worker to keep track of all simultaneous events. The kitchen displays the location of tools and the state of cooking processes. In the related project 'KitchenSense', a sensor-rich networked kitchen is considered that attempts to interpret peoples' intentions and reacts accordingly.

Additionally, the SenseBoard has been proposed in [74]. The SenseBoard approach is to combine the benefits of the digital world with those of the real world. The SenseBoard is a hardware board with a schedule projected onto it. Discrete information pieces that are stored in a computer can be manipulated by arranging small items on the board. These items are entries of the schedule. The naming of each item is computer-controlled and projected onto the item. Like in a digital schedule, items can be easily arranged, grouped together or expanded. Operations and the status of the schedule are projected to the physical schedule on the board. Like with real-world objects, people can manually arrange the items on the hardware board. This makes the operation more intuitive and enables the participation of larger groups in the process of finding an optimal schedule for a given task. Detailed information on each item can be made available and a schedule can be digitally exported, stored or loaded and also printed.

[1] http://context.media.mit.edu/press/index.php/projects/

2.2 Concepts and definitions

As mentioned in section 2.1, the concepts and ideas related to context-awareness that have not yet been commonly adopted among researchers even include the notion of context and context awareness itself. Since context-awareness is a comparably young research field, we find concepts and notions for which a variety of only partially redundant definitions have been given. On the other hand, several supplementing concepts are only vaguely described as, for example, the notion of high-level contexts, low-level contexts and raw data. In order to provide a stringent view on our research topics, we have to agree on non-ambiguous definitions for the concepts we utilise.

In this section we discuss those notions we adopt from recent work and further find comprehensive definitions for insufficiently defined concepts where necessary.

2.2.1 Ubiquitous computing

In our view of ubiquitous computing we agree on the vision introduced by Mark Weiser in [50]. As a prerequisite to our study, we assume a world in which computation has both infiltrated everyday life and vanished from people's perception. We believe that both developments are not only possible but predefined, since computing devices continuously decrease in size and power consumption while increasing in computing power at the same time. In the vision of ubiquitous computing, everyday objects are equipped with computing power and communication interfaces in order to compute and spread information. In our study we assume that computing is done in a ubiquitous environment, where multiple applications on stationary and mobile devices interact with one another.

Several authors have observed challenges of ubiquitous computing environments. The authors of [60] for example, state increased autonomy, a dynamic computing environment, dynamic user requirements, scalability issues and resource limitations as most serious issues in UbiComp environments. Depending on the application type, further issues may be named.

2.2.2 Sensors, context sources and features

In context-aware computing domains, the input data for applications is captured by sensors. Basically, a sensor is a piece of hardware or software that provides information on the environment. Humans or animals are not considered sensors but might trigger and influence sensor outputs. We distinguish between hardware sensors and software sensors. Hardware sensors are physical entities that react to stimuli from the physical environment and provide a software interface to publish notification describing these stimuli. Hardware sensors might, for example, measure the temperature, the light intensity or the humidity. Further hardware sensors are, for instance, a fingerprint reader or also a computer keyboard or a mouse that monitor user input.

Software sensors are applications that react to software generated stimuli and that output a software generated notification describing these stimuli. Example software sensors are a

calendar, an address book or an application a user is interacting with.

A sensor might provide various distinct aspects of a given context. Consider, for example, an audio sensor that provides the loudness as well as the number of zero crossings. These distinct aspects of context are often referred to as context features [57, 67]. Since we take a computation-centric approach, we are especially interested in the entity that provides information about a context feature.

We refer to this entity as a context source and consider context sources as atomic information sources for context-aware architectures. Context sources are not synonymous to sensors that produce context data. One sensor might incorporate several context sources. A context source basically produces output values that are related to one specific feature of a sensor.

2.2.3 Context and context types

As we have discussed in section 2.1.1 various definitions of context have been given in the literature that are only partly redundant. We adopt the definition given by Anind K. Dey in [58] since it is most general and can be applied to all application areas relevant to our research. However, Dey explicitly intertwines context with the interaction of applications and humans or, as he states it, with users. We have a slightly wider understanding of context that is not restricted to the user-application interaction but that covers contexts of arbitrary entities.

Definition 2.2.3 : Context

> *Context is any information that can be used to characterise the situation of an entity. An entity is a person, place, or object.*

Other definitions of context are too restricted to special cases to be applied in our general, computation-centric, consideration. Considering the revised definition given by Karen Henricksen, after which context is the set of circumstances relevant for the completion of a task [60], we disagree.

This revised definition differs from our understanding of context. First of all, we do not agree with the restriction of context to the set of circumstances that are of potential relevance for the completion of a task. The context driving, for example, could be partly sensed through the presence of the bluetooth ID of the car radio. However, the car radio is of no relevance considering the completion of the context driving.

In addition to the general understanding of the concept of context, a more concrete frame is required in order to be able to actually apply computations on context. We introduce the notion of a context element that utilises the definition of Dey and enhances the description to suit our needs in the processing of contexts.

Definition 2.2.4 : Context element

> *Let $i \in \mathbb{N}$ and t_i describe any interval in time. A context element c_i is a non-empty set of values that describe a context at one interval t_i in time.*

An example for a context element that is constituted from the temperature, the light intensity and an IP address is then $c = \{24°C, 20000lx, 141.51.114.33\}$. Observe that this definition refers to an interval in time rather than to a point in time. This accounts for the fact that the information describing a context is obtained by measurements of the real world that typically require a time-span rather than a time instant in which the measurement is performed. However, the shorter the time span the more accurate a context element describes a context at one point in time. Since the values are obtained by measurements, we may assume that the count of context elements is finite.

In [75] it was suggested that the context types location, identity, activity and time are more important than other types in order to describe a context. Undoubtedly, studies that utilise these context types for context-aware applications dominate studies on other context types. One reason for this is that implications obtained from these mentioned context types seem to be intuitive to most people. However, we argue that the type of context useful for an application is inherently dependent on the application type and that this context might be ignorant of the location, identity, activity or time.

Consider, for example, an arbitrary person sitting in her room and reading a book. While this scenario appears to be tranquil when only the four context types location, identity, activity and time are taken into account, the general assessment might change with the utilisation of further context sources. If, for example, the room temperature instantly rises or the amount of methane in the air increases, the same situation then appears in a different light. Danger might be at hand and a swift reaction is required.

We therefore assume that the application defines the relevance of distinct context types. The relevance could be modified by any kind of weighting or duplicating of contexts. Since we propose an architecture that utilises contexts for arbitrary applications, we do not prefer any context type above any other. For the remainder of this document we do not bother about the correct and application specific weighting, but assume that the contexts utilised have been filtered and weighted according to the application needs in advance. Several aspects of context have been introduced in [76, 77]. A further structured and extended distinction of context types is depicted in figure 2.2. This figure should be understood as a working model of context aspects. Context specifications for the context classes depicted in the figure are examples for the context classes and can be carried on by other examples that logically fit into the corresponding context class. Further aspects of context not depicted in the figure might well be found.

2.2.4 Context abstraction levels

Context does not necessarily equal context. Two contexts of the same type that describe the same time interval might nonetheless differ from each other in value. Context has several levels of abstraction depending on the amount of pre-processing applied. A temperature context might, for example, hold the value $24°C$ as well as the value 'warm '. These context values might originate from identical measurements of context sources. However, the data abstraction level differs. The value 'warm' is at a higher abstraction level than the value $24°C$.

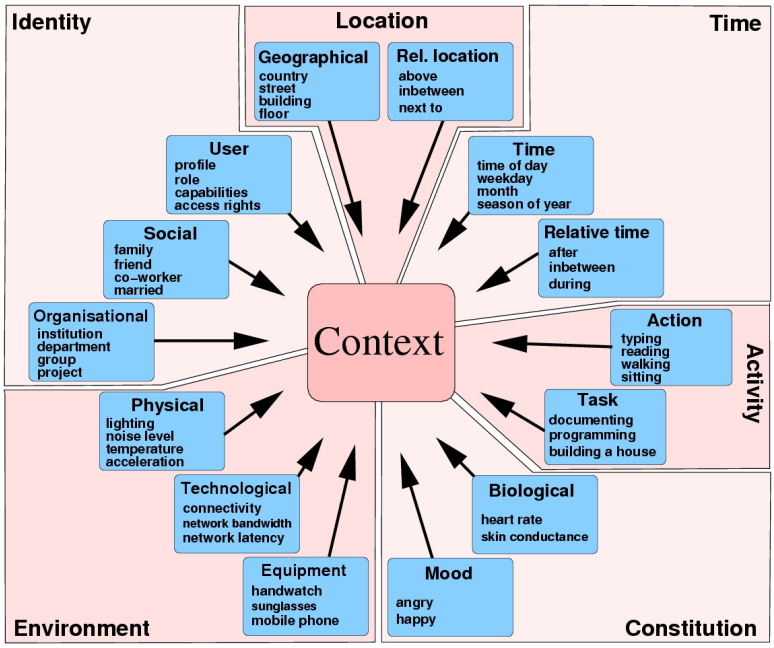


Figure 2.2: Aspects of context

Although several authors use the notions high-level context, low-level context and raw data, in order to describe various context abstraction levels, no exact definition of these notions is given in the literature. These notions are therefore often used with different meanings. Some authors, for example, use the term low-level context in the same sense as other authors use the term raw data. Typically, higher context representations tend to be symbolic while lower representations are more often numeric. Generally, the definition of several data abstraction levels is reasonable since the kind of representation used for operations on context elements may affect the accuracy of the operation [78].

A rough distinction between low-level and higher level contexts is made by Anind K. Dey, Bill Schilit and Marvin Theimer [58, 52]. Following this discussion, low-level context is used synonymously for data directly output from sensors, while high-level contexts are further processed. This processing can, for example, be an aggregation, an interpretation, a data calibration, noise removal or reforming of data distributions.

Jani Mäntyjärvi further distinguishes between processed contexts that describe an action or a condition [59]. Following his notion, raw data can be, for example, $24°C$ or 70% humidity. While for low-level contexts these are further processed to conditions like 'warm' or 'high humidity'. Finally, a high-level context is an activity as, for instance, 'having lunch'.

Actually, these distinctions between high-level and low-level contexts are only required (and properly understood) by humans. From a computational viewpoint, actions and conditions are both string values obtained by further processing of raw data. From a computation-centric standpoint, both constructs are consequently on the same level of data abstraction.

High-level context	Low-level context	Raw data	Context source
walking	14°C	001001111	thermometer
walking	57.2°F	001001111	thermometer
watching movie	64dB	109	microphone
listening music	64dB	109	microphone
at the beach	47° 25.5634'N; 007° 39.3538'E	GPRMC[3]	GPS sensor
swimming	47° 25.5634'N; 007° 39.3538'E	GPGGA[4]	GPS sensor
writing	z	0x79	keyboard [en]
writing	ы	0x79	keyboard [ru]
writing	z	0x7a	keyboard [de]
office occupied	z	0x7a	keyboard [de]

Table 2.1: High-level contexts, low-level contexts and raw context data for exemplary context sources.

[8] GPRMC Example:
$GPRMC,191410,A,4725.5634,N,00739.3538,E,0.0,0.0,181102,0.4,E,A*19
[9] GPGGA Example:
$GPGGA,191410,4725.5634,N,00739.3538,E,1,04,4.4,351.5,M,48.0,M,,*45

A computation-centric approach

We therefore take an alternative, computation-centric, approach and classify the level of abstraction of contexts by the amount of pre-processing applied to the data. Throughout our work we distinguish between high-level context information, low-level context information and raw context data[2] (cf. table 2.1).

In table 2.1, exemplary raw context data, low-level contexts and high-level contexts are depicted. Note that in all data abstraction levels different context representations are possible even if the measurement is identical. An example well-suited to illustrate this is the keyboard sensor. The same key pressed on an English and a Russian keyboard (raw context data identical) might result in different low-level contexts due to an alternative language setting (acquisition procedure). In the Cyrillic layout the letter 'ы' is obtained while it is the letter 'z' for the English layout.

However, for German keyboards the letters 'y' and 'z' are exchanged compared to the English layout, hence leading to the same low-level context even though the raw context data is different. Furthermore, different context interpretation procedures may lead to

[2]For ease of presentation, we utilise the notions 'raw data' and 'raw context data' synonymously.

distinct high-level contexts (office occupied or writing).

A discussion of the three data abstraction levels 'raw context data', 'low-level context' and 'high-level context' is given in the following.

The output of any context source is considered as raw data since it most probably needs further interpretation. Already at the very first abstraction level of raw context data, basic operations on the measured samples might be suggestive. Computations that might be applied on this data include mechanisms to correct possible measurement or sensor errors, filters that might abstract from irrelevant measurements or also processes that weight the measurements.

Different manufacturers produce sensors with varying output even though the sensors might belong to the same class. This is because of possibly different encoding of the sensed information or due to a different representation or accuracy. Two temperature sensors may for instance differ in the unit (Celsius or Fahrenheit), in the measurement accuracy or in the measurement range. A pre-processing of raw context data is necessary so that further processing is not influenced by special properties of the context source itself. We refer to this pre-processing as the context acquisition step. Low-level contexts are acquired from raw context data in this pre-processing step.

The data has become low-level context elements after the context acquisition. The low-level contexts of two arbitrary context sources of the same class measured at the same time in the same place is identical with the exception of a possibly differing measurement accuracy, provided that both context sources are in good order. The output of all context sources for temperature may, for example, be represented in degree Celsius.

In order to obtain high-level context elements, further processing operations are applied. Possible operations are aggregation, interpretation, semantic reasoning, data calibration, noise removal or reforming of data distributions. We refer to this pre-processing as the context interpretation step.

From low-level contexts describing the temperature, light intensity and the humidity it might be possible to infer the high-level context outdoors/indoors. There is no limit to the level of context interpretation. Several high-level contexts may be aggregated to again receive high-level context elements. For our discussion, however, we do not distinguish between high-level contexts of various context abstraction levels. For these three context abstraction levels, the distinguishing factor is the amount of pre-processing applied. Note, however, that we do not exactly define the amount of pre-processing for all three context abstraction levels since it may vary between distinct application scenarios. For our discussion it suffices that this construct of context abstraction levels is hierarchical. The amount of pre-processing applied to high-level contexts always exceeds the amount of pre-processing applied to low-level contexts in the same application scenario.

Observe that it is possible that two contexts of the same context type are differing in their context abstraction level when the amount of pre-processing to derive these contexts differs. While this might intuitively appear inconsistent, it is inherently logical from a computation-centric viewpoint. The amount of computation or pre-processing applied to contexts of distinct context abstraction levels differs. In addition, the information certitude of contexts in distinct abstraction levels might differ. Various context processing steps and

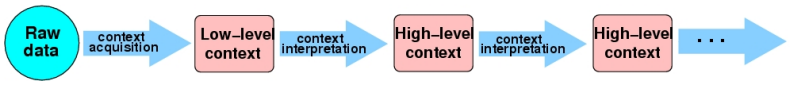


Figure 2.3: Context pre-processing steps.

corresponding input and output data are depicted in figure 2.3.

2.2.5 Context data types

Since context is acquired from a set of heterogeneous context sources and is computed at various levels of abstraction, context processing operations applicable to one subset of contexts might be inapplicable to another subset.

As an example, consider IP addresses as context type on the one hand and temperature as another context type. Temperature contexts contain an implicit order regarding their magnitude while for IP addresses, an order cannot be provided in the same manner.

In [76] four data types have been introduced that group contexts applicable to the same mathematical operations together. Following this discussion, we distinguish context data types between nominal, ordinal and numerical categories. We omit the fourth category interval that was proposed in [76] since the boundaries of any context type (the only use for the interval category described in [76]) are provided for ordinal and numerical contexts in our case anyway.

The only operation applicable to nominal context data is the equals operation. Contexts of nominal context data type are, for example, arbitrary binary contexts, whereas symbolic context representations like, for instance, activities (walking, talking) or tasks (cleaning) are of nominal context data type.

Ordinal context data types further allow the test for an order between these contexts. Examples for contexts of ordinal context data type are physical contexts like lighting or acceleration when represented in symbolic notation like 'dark' and 'bright' or 'fast' and 'slow'.

Contexts of numerical context data type allow arbitrary mathematical operations to be applied on them. A good example for these context data types is the time. By subtraction, the time difference between two contexts of this type can be calculated.

We further consider hierarchical contexts, that are applicable to the 'subset'-operation. Similar to ordinal context data types, for hierarchical context data types an ordering of the contexts is possible. However, the order might be any kind of hierarchy as a directed tree or graph structure. Examples for a context type of this class are geographical contexts in a symbolic representation as 'in office building' or 'in town'.

The operators applicable to one context type limit the number of appropriate context processing methods. A context processing method usually requires a minimum set of operations on contexts. In order to be processed by a processing method, all processed contexts therefore have to share this minimum set of operations. An easy solution to equalise all contexts is to abstract from all operators not applicable to the whole set of

Context type	nominal	ordinal	hierarchical	numerical
Organisational	+		+	
Social	+		+	
User	+			
Geographical	+		+	
Relative location	+		+	
Task	+		+	
Action	+			
Time	+	+	+	+
Relative time	+	+		
Biological	+	+		
Mood	+			
Physical	+	+		+
Technological	+	+		+
Equipment	+		+	

Table 2.2: Operators applicable to various context types

available contexts. Clearly, this reduces the already sparse information we have about the data and artificially restricts us to a smaller number of context processing methods.

Table 2.2 depicts the context data types of the context types introduced in figure 2.2[5].

Observe that the context data type is not related to the data abstraction level of contexts. Low-level and high-level contexts alike might be of ordinal, nominal, numeric or hierarchical context data type.

From one context abstraction level to the next higher one, the context data type may swap to an arbitrary other context data type. While, for instance, in the aggregation of contexts, the resulting context might likely support less operations than the operations applicable to the set of contexts before the aggregation, it is also feasible to add further operations by a mapping of contexts to elements that support these further operations.

2.2.6 Representation and illustration of contexts

We have now introduced the concept of context and have discussed context types, context abstraction levels and context data types at a rather abstract, theoretical level. For any problem domain, a good perception of the contexts and relations in this domain is at least

[5]The classification of context types to context data types represents one example classification that is considered reasonable by the authors. However, a specific scenario might introduce context type classifications that differ from the values depicted in the table. The important point here is that in a given scenario the observed context data types might not be computed by arbitrary context prediction algorithms

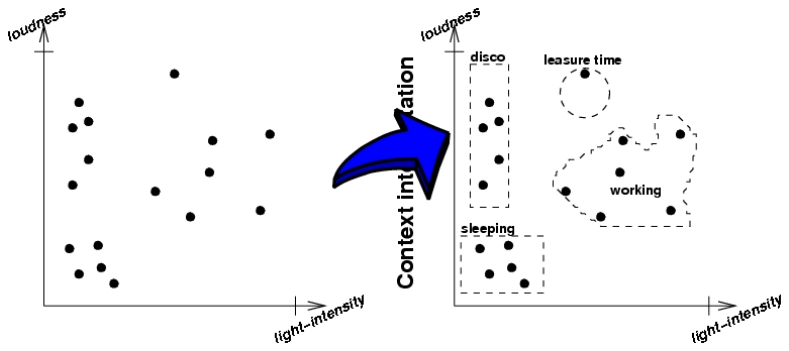


Figure 2.4: Illustration of the context interpretation step.

helpful for the next step, the approach to solve the problem at hand.

A straightforward way to illustrate low-level contexts is to map them into a multi-dimensional coordinate system. This representation has first been considered by Padovitz et al [79, 80, 81]. Although another distinction between low-level contexts and high-level contexts has been applied, the same principle can also be applied in our case. The general idea is to represent for every time interval a low-level context element by a vector in a multi-dimensional coordinate system. Each coordinate axis represents a normalised aspect of a low-level context element.

High-level contexts are then sets of low-level contexts that are assigned a label. Figure 2.4 illustrates the context interpretation step[6].

Low-level contexts are represented on the left hand side by dots in a coordinate system. On the right hand side, these low-level contexts are transformed to high-level contexts which is basically a grouping of several low-level contexts together into a set of low-level contexts.

This geometrical context representation is trivially extended to context sequences in time by simply considering one further axis in the coordinate system that represents the time. This more concrete, geometrical context representation assists us in the discussion of several properties later on.

In our discussion we do not consider overlapping high-level definitions. A discussion of this topic can be found in [81].

[6]The figure connotes that the high-level contexts 'sleeping', 'working', 'leasure time' and 'disco' can be distinguished by the light intensity and the loudness. This labelling of high-level contexts is only for an easier understanding of the described context interpretation step. Note that the necessary context sources to accurately distinguish the mentioned high-level contexts is currently an unsolved research problem.

3 Wireless Sensor networks

> *Adding [...] sensors to everyday objects will create an Internet of Things, and lay the foundations of a new age of machine perception.*
>
> (K. Ashton, co-founder and executive director of the Massachusetts Institute of Technology's Auto-ID Center in a presentation in spring 1998)

With the term wireless sensor network commonly a network of distributed devices is associated that consists of sensing and communication entities and that collaborate to achieve a common objective. A wireless sensor network typically has one or more sinks to collect data from all sensing devices. These sinks might, for instance, constitute the interface of the sensor network to the outside world. Sensor networks can be applied to monitor environmental stimuli and provide feedback on a monitored area. In particular, sensor networks can be implemented in areas that are seldom or even hard to access. Examples are the monitoring of underwater scenarios, animal monitoring by sensor networks, the application in emergency situations like avalanche warning or the monitoring of earthquake regions. Further applications include smart sensors to monitor and control manufacturing, bio-sensors for health monitoring, weather or habitat monitoring and smart environments for home entertainment [82]. Popular features of sensor networks are that these networks are easy to deploy, cheap, and in the optimum case require low maintenance effort. The following sections detail aspects of sensor networks that are related to cooperative and collaborative transmission strategies in these networks.

These sections represent a comprised introduction to wireless sensor networks. An in-depth discussion of these topics is elaborated in [8, 83].

3.1 The sensor node

Sensor networks are constituted of sensor nodes that are densely deployed either very close or directly inside a phenomenon to be observed. Therefore, sensor nodes are routinely required to work unattended in remote geographic areas.

These nodes typically consist of sensing, data processing, storage and communication components. Figure 3.1 schematically depicts typical components of a sensor node. Due to advances in integrated cirquit design the size, weight and cost of nodes is constantly decreasing while accuracy and resolution are improved [82]

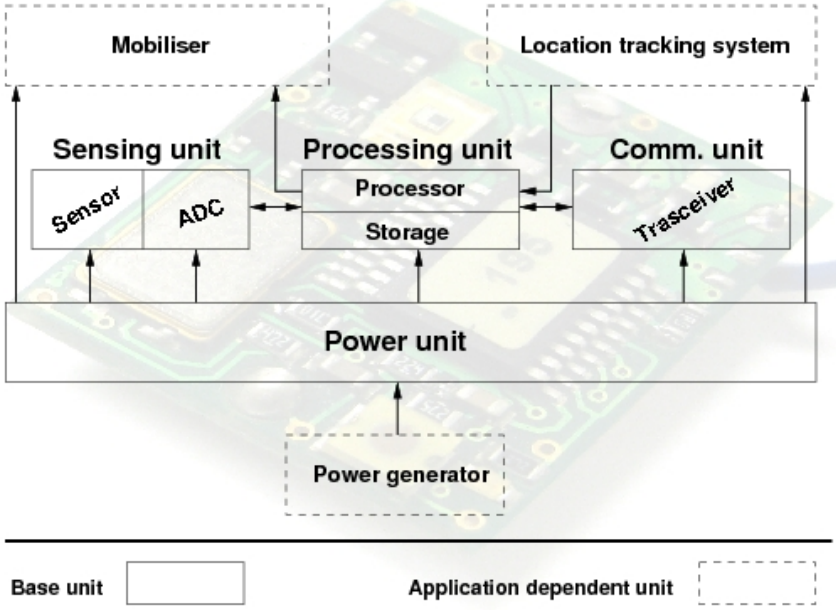


Figure 3.1: Components that typically constitute a sensor node [8]

A sensor node consists of some base units and application dependent units as a mobiliser to apply some spatial movement to the node, a location tracking system as e.g. GPS or relative positioning techniques (e.g. ultrasound) or a power generator (e.g. solar cells). The following section details the base units depicted in the figure.

3.1.1 Power unit

Among the most important design aspects for sensor networks are related to the power unit. As sensor networks are often designed to operate unattended over long time spans and disconnected from an external power source, the energy consumption and capacity of the power unit is a critical aspect.

Since the wireless sensor node is a microelectronic device, it can only be equipped with a very limited power source in the order of less than $0.5Ah, 1.2V$ (Another estimation from [83] is 2.2 to 2.5Ah at 1.5V). In a multi-hop ad-hoc sensor network, each node plays the dual role of a data originator and a data router. The malfunctioning of few nodes can cause significant topological changes and might require rerouting of packets and reorganisation of the network. Therefore, power conservation and power management are of additional importance.

Power consumption in a sensor network is divided into the domains sensing, communication and data processing. The most energy consuming task among these is the communication (reception and transmission of data).

Power units can also be accompanied by power harvesting units such as e.g. solar cells.

3.1.2 Sensing unit

Sensing units are usually composed of two sub-units, sensors and analog-to-digital converters (ADCs). A sensor is a device that transforms physical parameters as temperature, pressure or light intensity into analog electrical signals. The ADC transforms the analog signals produced by the sensors to digital signals that are then further processed in the processing unit and possibly transmitted by the transceiver in the communication unit. A digital signal is a voltage signal that is either on or off but that does not allow for any intermediate value.

It is possible that these digital signals are utilised to control different components or parameters of the sensor nodes via actuators. An actuator is an electromechanical device such as a relay that can control other parts of an electromechanical system.

Typical sensors are, for example, audio, light or humidity sensors as well as ball switches or acceleration sensors.

3.1.3 Processing unit

The processing unit contains a processor and typically a small storage. It is responsible for pre-processing of the information obtained by the sensing unit and for implementing the communication related logic regarding the applications, the transport layer (e.g. TCP) and the network layer (e.g. routing).

Typically, the sensor node will not directly transmit each data item obtained from the sensing unit but convert the raw data to a common representation or possibly discard redundant or highly error prone data in order to save energy for communication. The processor decides when and which data to transmit via the communication unit.

The task of the processing unit can be summarised as enabling cooperation between nodes in a network. Typically, microcontrollers are utilised in the processing unit of sensor nodes due to their low power consumption.

3.1.4 Communication unit

The communication unit consists of a transceiver to connect the node to the sensor network. A transceiver combines a transmitter and a receiver in a single entity. The communication unit might implement radio, infrared or optical technologies. Typical implementations of sensor nodes implement RF technology. However, for low-lying antennas the RF signal propagation is influenced by surface roughness, reflecting and obstructing objects and antenna elevation. For low-lying antennas, the signal intensity drops as the fourth power of distance due to partial cancellation by a ground-reflected ray [84, 85]. Table 3.1 details some path-loss measurements obtained in various environments that have been observed in [85].

It is therefore more energy efficient to implement a multi-hop topology with short hops instead of utilising high-power transmission nodes to cover greater distances. An alternative communication paradigm is introduced in chapter 7. When the distance to a remote

Location	Mean Path loss exponent	Shadowing variance σ^2 (dB)
Apartment Hallway	2.0	8.0
Parking structure	3.0	7.9
One-sided corridor	1.9	8.0
One-sided patio	3.2	3.7
Concrete Canyon	2.7	10.2
Plant fence	4.9	9.4
Small boulders	3.5	12.8
Sandy flat beach	4.2	4.0
Dense bamboo	5.0	11.6
Dry tall underbrush	3.6	8.4

Table 3.1: Mean power loss and shadowing variance obtained over the range 800-1000 MHz and with an antenna height of about 15cm [85].

receiver exceeds the transmission range of a single node, nodes may cooperate during transmission to combine their transmit signal components and improve the transmission range.

Another possible communication mechanism is infrared. Infrared communication is license free and robust to interference from electrical devices which is one major concern in RF-based devices. RF-based transceivers are therefore cheap and easy to build solutions. However, IR is only feasible in direct line of sight between nodes.

An alternative approach is detailed in [86]. The authors utilise the optical medium for transmission. Finally, wired communication is of course possible and in fact often implemented as a communication medium in sensor networks.

In the following sections we will assume sensor networks that utilise RF-transmission technology.

Transceiver design

A transceiver is an entity that combines transmission and reception capabilities. A typical transceiver consists of a Radio Frequency (RF) frontend and a baseband processor. The RF frontend processes the analog signal as it is actually transmitted or received over the wireless channel. The signals in the radio frequency band are typically of high frequency as, for instance, in the 2.4 GHz ISM (Industrial, Scientific and Medical) band.

Processing at these high frequencies would surcharge the limited processing capabilities of sensor nodes. It is, in fact, even uncommon to employ, for example, oscilloscopes that are capable of visualising frequencies higher than about 1 GHz. Therefore, the radio frequency signals are transformed in an intermediate frequency (IF) section to digital signals by analog-digital converters (ADC) and are downconverted by digital downconverters (DDC) to lower frequencies of several kHz (cf. figure 3.2). These lower frequency signals are referred to as the baseband signals which are then processed by the sensor node's processor.

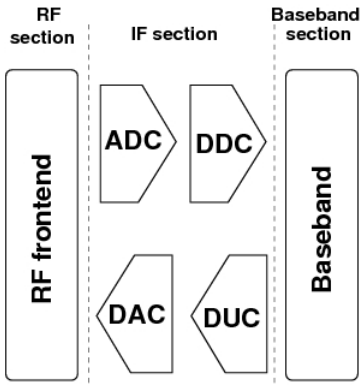


Figure 3.2: Schematic of a transceiver design.

Transceiver states

Normally, a transceiver is in one of four states:

- Transmit
- Receive
- Idle
- Sleep

In the transmit and receive states, the transceiver transmits or receives RF signals. These two states are typically the most energy consuming tasks of a sensor node. Consequently, the time in which the transceiver is in transmit or receive states should be minimised to reduce the energy consumption of a node.

In the idle state, the transceiver is not actually receiving or transmitting but ready to receive an RF signal. In this state, some parts of the receive circuitry are switched off to reduce the energy consumption.

In the sleep state, major parts of the transceiver are switched off. Many architectures distinguish several sleep states depending on the amount of parts that are switched off. For these sleep states, the recovery times and energy required to wake up differ.

3.2 Sensor networks

Sensor networks are constituted by sets of sensor nodes that build up a wireless network and accomplish one or more tasks collaboratively. For sensor networks to be dispatchable in inaccessible terrains and to work unattended over a long time span, it is desired to incorporate self-organising capabilities. These enable the network, for instance, to establish a wireless connection among nodes as well as routing paths along the network.

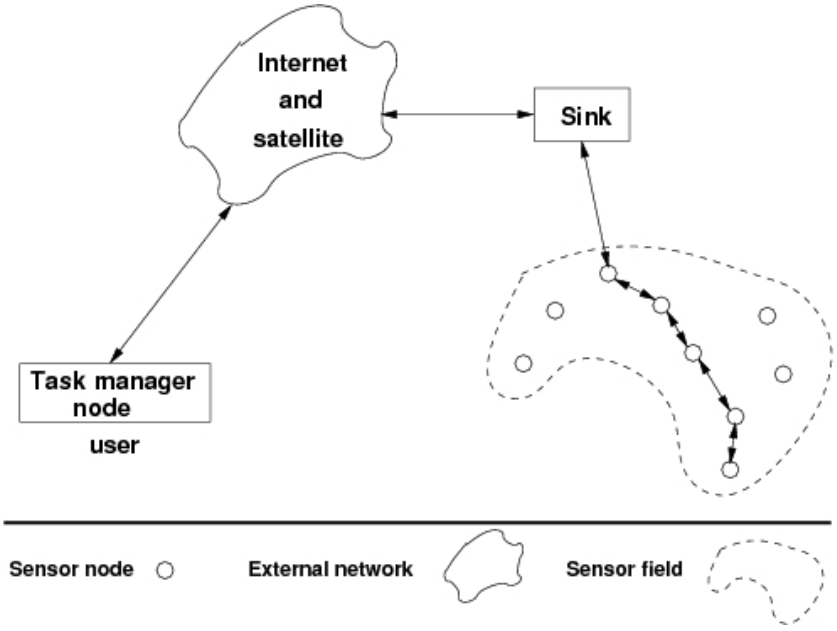


Figure 3.3: Schematic illustration of a sensor network implementation [8]

Applications for sensor networks range from health care over military or disaster monitoring to home appliances.

Figure 3.3 illustrates a possible implementation of a wireless sensor network. The sensor nodes might be scattered in a field. In sensor networks we speak of sources and sinks. For any communication between nodes in a network, the source is the sender node of the information and the sink is the recipient. Each node has the capability to collect data and to route data from a source to a sink either directly over a single hop or over a multi-hop protocol. The sink may communicate over an external communication network with a task manager node that is controlled and/or read out by a user.

In order to accomplish these tasks, wireless sensor networks require ad-hoc networking techniques. However, common ad-hoc networking protocols are usually not well suited for wireless sensor networks since they typically differ in various aspects from wireless ad-hoc networks. Some of the differences are

- The number of nodes in a sensor network might be several orders of magnitude higher than in ad-hoc networks
- Sensor nodes are densely deployed
- Sensor nodes are prone to failures
- Due to node failures or environmental impacts, the topology of a sensor network changes frequently
- Sensor nodes mainly use a broadcast communication paradigm, whereas most ad-hoc networks are based on point-to-point communications

- Sensor nodes are limited in power, computational complexity and memory
- Sensor nodes typically not utilise global identification because of the large amount of overhead and the large number of nodes

As we will see in section 4 the transmission range of a node is restricted due to growing signal decay with increasing transmission distance. Consequently, the maximum distance for a single hop is restricted. The classical approach to reach a distant receiver is to apply a multi-hop protocol for transmission. In such a protocol, intermediate nodes are utilised as relay nodes that forward received packages along the network.

It is sometimes claimed, that multi-hop transmission schemes are more energy efficient than direct transmission schemes since the radiated energy for a direct connection is higher than the sum of the radiated energy of relay nodes along the transmission path. The intuition behind this assumption is that the decay in signal strength is more than linear with increasing transmission distance (we will detail this path-loss property in section 4.1.2).

However, in general, this assertion is not correct. We can easily reduce this assumption to absurdity by shortening the distance between relay nodes to a small ε so that nodes are placed next to each other. The overall energy consumed by the extra nodes is then much higher than the gain in transmission energy. For further details on this effect we refer to [86].

Multi-hop transmission is feasible only when a chain of intermediate nodes exists so that for each two neighbouring nodes in this chain the distance is smaller than the transmission range. If this is not the case, multi-hop transmission is not possible. In section 7 we introduce cooperative strategies to reach a distant receiver for which no intermediate nodes are required. alternatively, relay stations with increased transmission range can be utilised to bridge longer distances. Since the latter approach contradicts the WSN principle of utilising standard, low-power and low cost nodes, we will not consider this idea in the following.

3.2.1 Metrics to measure the quality of a WSN

Different wireless sensor network installations are utilised for distinct applications. Each application has other requirements on a wireless sensor network. In order to quantify the suitability of a concrete wireless sensor network instrumentation for a given application, special quality metrics are developed. In the following section we discuss some typical metrics to derive the quality of a given wireless sensor network instrumentation.

Fault tolerance

As nodes of a sensor network are cheap and low power devices, nodes might fail in the course of the operation of a sensor network due to, for instance, power shortage, physical damage and environmental interference. A network is therefore required to provide some means of fault tolerance as, for example, an ability to sustain network functionality in the presence of node failures. The reliability $R_k(t)$ or fault tolerance of a sensor node is

modelled in [87] by using the Poisson distribution. The probability of not having a failure within time interval $(0, t)$ is then

$$R_k(t) = e^{-\upsilon_k t} \tag{3.1}$$

where υ_k is the failure rate of sensor node k and t is the time period.

Scalability

Another relevant aspect for sensor networks is the scalability of the network. As sensor networks consist of densely deployed nodes, the number of nodes in a network may easily reach extreme values. The density might range to hundreds of nodes in a region of no more than 10 m^2. In order to be able to support such huge and dense networks, new protocols are developed.

In [88] the density of a sensor network is calculated as

$$\eta(R) = \frac{N \cdot \pi k^2}{A} \tag{3.2}$$

where N is the number of nodes, A is the region and R is the transmission range.

3.2.2 Mobility in wireless sensor networks

In practical environments, for WSNs, often mobility of some kind is involved. Generally, three types of mobility may apply. These are mobility of nodes, mobility of sinks or mobility of objects that are monitored by the network.

When the wireless nodes themselves are mobile we speak of node mobility. This might be the case, when sensor nodes are attached to moving vehicles, or also cattle, for example. In the presence of node mobility, the network is required to frequently reorganise, for example, connection tables and routing paths.

Generally, the trade-off for this reorganisation is to achieve sufficient organisation of the network at reasonable overhead in communication and energy consumption.

When sinks are mobile, especially, when this concerns since nodes external to the network as, for example, a user with a PDA walking alongside the network, the same challenges occur as in the scenario of mobile nodes, but the effects are restricted to nodes near the mobile sinks [89].

A common case in wireless sensor networks is that an object that is monitored is mobile. In this case, mobility mainly affects the activity of nodes. While nodes that are far away from the moving object can remain in a sleep mode, nodes near the object are required to wake up from sleep. Especially interesting in this context are algorithms to derive the nodes that wake up next, with regard to the movement of the object. In [90] this has been introduces as the Frisbee-model along with algorithms to handle the wake-up front of nodes.

3.3 MAC protocols

MAC protocols coordinate the communication among nodes or, more precisely, they coordinate the times when a number of nodes access the shared medium.

Since energy consumption is one of the key problems in wireless sensor networks and communication is typically the most energy consuming task, an important aspect of a MAC protocol is its ability to reduce the energy consumption of a node. These protocols therefore might switch the transceiver into a sleep mode.

Medium Access Control (MAC) protocols are the first protocol layer above the physical layer and are therefore impacted by the physical layer properties.

3.3.1 Requirements and design constraints

Requirements for MAC protocols are a high throughput, stability, fairness, low access delay (time between packet arrival and first attempt to transmit it), low transmission delay (delay between reception of a packet and first approach to further forward the packet over the wireless channel) and a low overhead.

The overhead may result from collisions, from the exchange of extra control packets but also from protocol overhead, such as large header or checksum fields.

Typically, as nodes communicate over a wireless channel, also a high robustness against bit errors is required. Furthermore, a constant problem is the energy consumption due to protocol design. Popular protocol related energy problems result from collisions, overhearing and idle listening.

Collisions: Whenever a collision occurs the transmission energy spent for all packets involved in the collision is wasted. Collisions can be avoided by design, for instance, through fixed assignment schemes such as TDMA or by demand assignment protocols. Another way to reduce the probability of a collision is, naturally, to reduce the number of packets transmitted in the network. Therefore, when the expected load in a sensor network is sufficiently low, collision avoidance schemes can be omitted as they also require channel resources.

Overhearing: Due to the broadcast nature of the wireless channel a transmitted packet is received by all neighbouring nodes and not only by the intended recipient. Although the non-destination nodes might discard the packages received, they had spent energy on their reception in the first hand. Especially in dense sensor networks, schemes that reduce the frequency of overhearing, the gain in energy resources can be significant [**?**, **?**]. However, observe that overhearing is sometimes wanted. Multi-hop-routing, for instance, is only possible when nodes overhear and forward packets that are not addressed to them.

Idle listening: A node in its idle state is ready to receive a packet but not actually transmitting or receiving information. Although the energy consumption in this state is less resource demanding than for the transmit or receive states, the energy

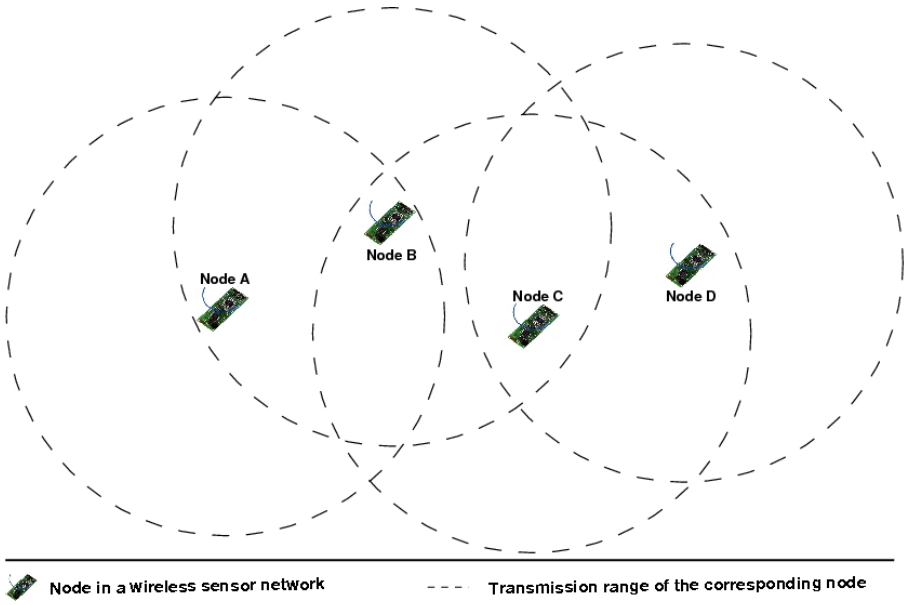


Figure 3.4: Schematic illustration of the hidden node problem and the exposed node scenario

consumption might be significant for longer idle periods. It is therefore desired not only to reduce the time a node remains in transmit and receive states but also to reduce the time a node resides in its idle state. A solution is to switch to sleep mode more often. However, since mode-changes also cost energy, the frequency of these also pose a tradeoff to consider in the design of protocols.

A prominent problem in wireless sensor networks is the hidden node problem. It occurs, when the nodes transmit in a CSMA protocol. Consider the sensor placement in figure 3.4 In the figure, dashed lines represent transmission ranges. When node A is transmitting information to node B and C is transmitting information to node B at the same time, both nodes, A and C are not capable of sensing the transmission of the respective other node and thus are not capable of preventing the collision occurring at node B when both, node A and node B transmit simultaneously.

Another aspect is the exposed node scenario. In this scenario, node B transmits a packet to node A and a short time later, node C is ready to transmit a packet to node D. While both transmissions are possible simultaneously since node D is out of reach of node B and node A can not be reached by node C, node C would not transmit its data, since it senses that the channel is already used and concludes that a transmission would lead to a collision. In this way, the transmission is delayed unnecessarily.

3.3.2 A standard protocol for wireless sensor networks

Other wireless protocols can, of course, also be utilised for wireless sensor networks, however, the special requirements of the wireless communication in a dense network of resource

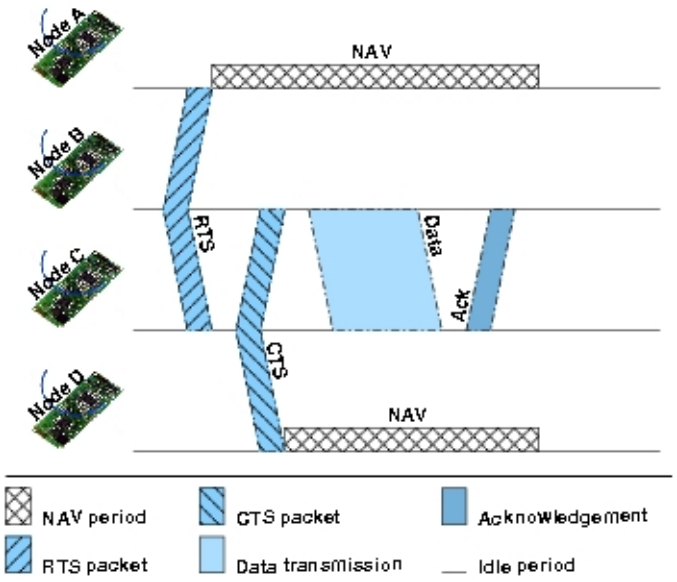


Figure 3.5: Schematic illustration of IEEE 802.11 RTS/CTS handshake

restricted nodes may lead to a low efficiency of these protocols. As an example we will consider the application of the IEEE 802.11 protocol [91] with its RTS/CTS handshake. RTS and CTS packets are special control packets. The abbreviations stand for Ready to Send and Clear to Send. Figure 3.5 illustrates the RTS/CTS handshake in IEEE 802.11 protocols.

In this example, node B wants to transmit data to node C. In order to announce its intention, node B sends an RTS packet. This packet contains the addressee of the communication and a probable transmission time for the transmission. Both neighbouring nodes A and C receive the RTS message and therefore the estimated transmission time. Based on this information, node A allocates an internal timer called Network Allocation Vector. (NAV) The NAV indicates a busy medium so that node A will not try to access the medium. Node C, the addressee of the RTS packet answers with a CTS packet that again contains an estimation of the time required to transmit the data. Apart from node B all other nodes neighbouring C receive this CTS packet and can again calculate a NAV. Since the NAV was calculated by nodes receiving both, the CTS and the RTS packets, all nodes neighbouring both communicating nodes B and C calculate a NAV and free the medium in order to avoid collisions (we will see later, that collisions can occur anyway). After receiving the CTS packet, node B then transmits the data and awaits an acknowledgement from node C.

Problems with RTC/CTS handshake

While this RTS CTS handshake scheme is successfully applied in the IEEE 802.11 protocol, in dense wireless sensor network, the frequent transmission of control packets (RTS, CTS,

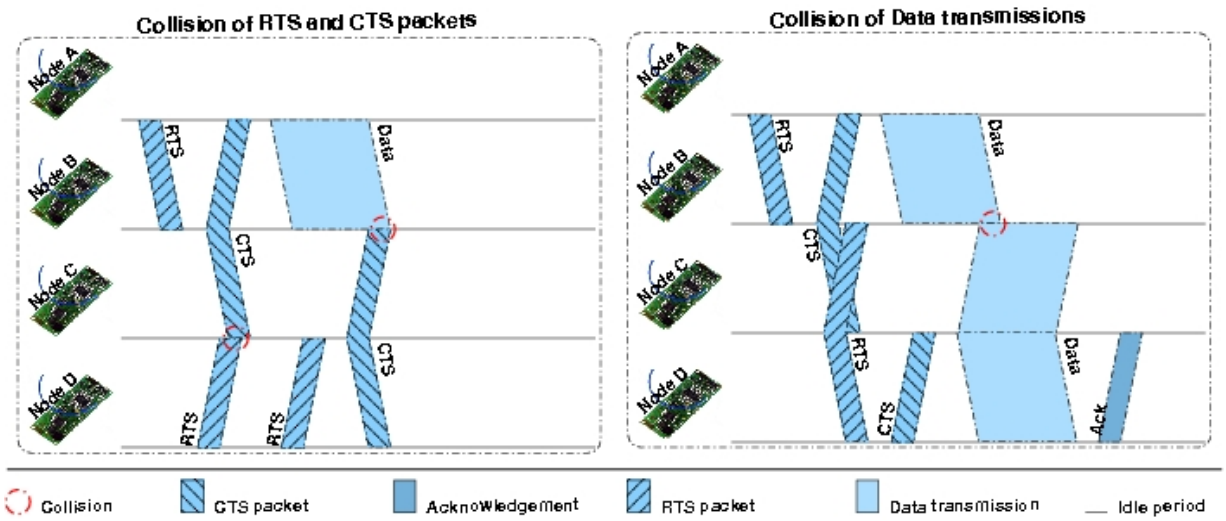


Figure 3.6: Schematic illustration of the hidden node problem and the exposed node scenario

Ack) may lead to congestion and collision. Figure 3.6 illustrates two problems with the RTS/CTS handshake procedure in dense wireless networks.

One problem may occur when one node is not able to receive a CTS packet due to a collision with an RTS packet. In this case (see figure 3.6, left), a node A sends an RTS packet to a node B that in turn answers with a CTS packet. However, this CTS packet is not received by all neighbouring nodes of B. In particular, a node D transmits an RTS packet to node C so that the RTS and CTS packets are colliding at node C. Consequently, node C neither answers the RTS request from D nor does it know of the CTS transmission by node B. Node D, after not receiving a CTS response retransmits its RTS request. This time it is acknowledged by node C. unfortunately, the CTS transmitted by node C now collides at node B with the data transmitted by node A.

Another problem is the collision of data transmissions as illustrated in figure 3.6, right. Here again node A requests a transmission to node B. This time, the CTS transmitted by node B overlaps with an RTS request sent by node C to node D. As both, node A and node D did not know of this problem, the protocol is continued as normal until a collision in the data transmission from node C and node A occur.

In both cases, the transmission was not successful and has to be repeated at the cost of further energy for the transmission of control messages and data between nodes. Is is argued if this high control overhead of the RTS/CTS handshake is worthwile when only small data packets are transmitted. For long data packets this overhead can be neglected. However, in wireless sensor networks, the data to be transmitted is typically small. Also, longer data packets are more likely to suffer form bit errors and must be retransmitted more often so that in wireless sensor networks long data packets are typically fragmented into smaller ones.

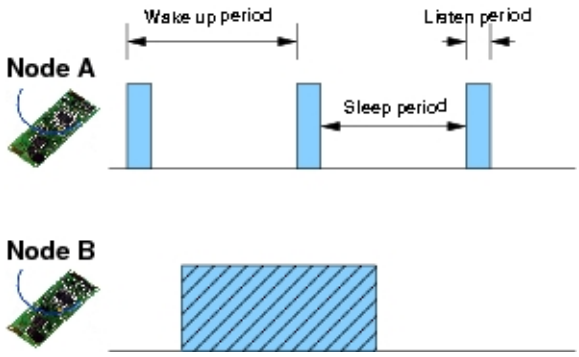


Figure 3.7: A generic Wake Up Radio scheme

3.3.3 Wake up radio

Since transmission and receive states are most energy consuming tasks and also the idle task is quite resource consuming, one might think of building a radio that stays in the sleep mode for a maximum amount of time. In an optimum case it would be able to wake up the radio of a node on demand. The main idea of a wake up radio is to implement a very low power receiver that has no other task than to wake up the regular transceiver of a node. In this manner, a node can be in the receiving state only when it is requested to listen to the channel. Unfortunately, such a radio has not yet been designed, although various running projects are claiming to build it as part of their project aim[1].

A popular implementation is therefore for the radio to periodically wake up from the sleep mode to listen on the channel for potential communication requests. Figure 3.7 illustrates this concept.

Here, the wake up period denotes the time between two consecutive listen periods of one node. In a listen period, the node listens to the channel for communication requests. Finally, in the sleep period, the node is not listening. We denote the duty cycle of a node by the fraction of listen period to wake up periods:

$$\text{Duty Cycle} = \frac{\text{Length of the listen period}}{\text{Length of the wake up period}} \tag{3.3}$$

A second node, willing to communicate has to transmit a preamble that lasts not more than one wake up period so that the listen period of the receiving node is covered. Two often cited protocols that implement this basic idea are the STEM protocol and the S-MAC protocol. For the STEM protocol, an extra channel is utilised for data transmission and notification of sleeping nodes. In case of the S-MAC protocol, the same channel is utilised.

Another solution that deviates from the generic approach depicted in figure 3.7 is the mediation device protocol. In this protocol a mediation device is utilised ad an intermediate node to coordinate the communication between other nodes. All but the mediation node have their own, not synchronised sleep schedules. The mediation node is never in the sleep

[1]One example is the FP7-ICT-2007-2 project CHOSeN (http://www.chosen.eu)

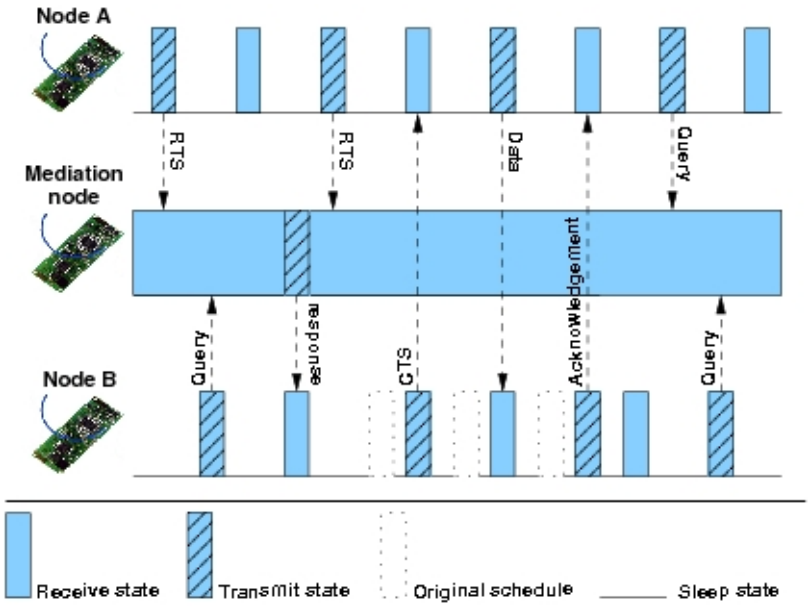


Figure 3.8: A schematic illustration of the mediation protocol

mode. When a node wakes up and wants to send data, it informs the mediation node of the address of the receiver repeatedly until an answer is received. Figure 3.8 schematically illustrates the idea of this communication protocol. A Node that has data to send transmits a Ready To Send (RTS) packet to the mediation node that also contains the address of the addressee. A node that wakes up and has no data to transmit, requests data from the mediation node via a Query packet. If not packet is received in reply to this Query request, the node returns to the sleep mode.

When the Query of the node is, however, answered by the mediation node in a Query response , it sends a Clear To Send (CTS) packet directly to the node willing to transmit. Synchronisation of transmit and receive periods is achieved since the mediation node obverses the frequency of RTS packets from the transmitting node. This transmission timing is included in the CTS packet sent to the receiving node together with the address of the transmitter node. The receiver then transmits the data and awaits an Acknowledgement from the receiving node. Afterwards the timing of the receiving node is changed back to the original schedule and both nodes again query the mediation node for transmission requests.

4 Wireless communications

> *Communication technologies are necessary, but not sufficient, for us humans to get along with each other. [...]Technology tools help us to gather and disseminate information, but we also need qualities like tolerance and compassion to achieve greater understanding between peoples and nations.*
>
> (Sir Arthur Charles Clarke – British author, inventor and futurist.[92])

Similar to wired communication, wireless communication is about transmitting information over a medium – in this case the air. An electromagnetic waveform is transmitted between communication partners and information is modulated on top of this signal wave. Basically, some phenomena observed in a wireless channel as, for instance, path-loss are similar in a wired channel. However, wireless communication is more challenging than wired communication as the medium utilised is shared between multiple communication partners. Therefore, new effects are observed as slow fading and fast fading as well as interference between transmitted signal components. We will introduce basic properties of the wireless radio channel in the following sections.

4.1 Aspects of the mobile radio channel

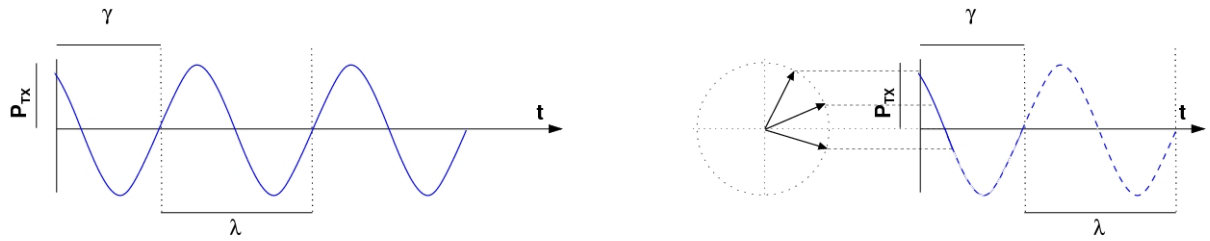
Electromagnetic signals, utilised in wireless communication are radiated in wave-form. These waves propagate from a transmitter omnidirectionally (in all directions/dimensions) at speed

$$c = 3 \cdot 10^8 \frac{m}{s}. \tag{4.1}$$

c is the speed of light and approximates the propagation speed of electromagnetic waves. An intuitive analogue (in two dimensions) for the propagation of electromagnetic waves are the waves on the water surface created by a single drop into the water.

The signal is characterised by the transmission power P_{TX}, by the frequency f and by a phase offset γ. The wavelength

$$\lambda = \frac{c}{f} \tag{4.2}$$



(a) Signal wave with phase offset γ and wavelength λ

(b) Analogy between the wave form of an analog signal and a rotating vector

Figure 4.1: Schematic illustration of a signal wave

of a signal describes the length (in meters) of one signal period. Figure 4.1(a) schematically depicts an electromagnetic wave with frequency $f = \frac{c}{\lambda}$, phase offset γ and transmit power P_{TX}.

We can imagine that this wave-form is created by the real part of a rotating vector $\zeta = \Re\left(e^{j(ft+\gamma)}\right)$ as depicted in figure 4.1(b). The height $\cos(\zeta)$ of the vector $\zeta = \Re\left(e^{j(ft+\gamma)}\right)$ equals the signal strength P_{TX} and the rotation speed determines the frequency of the signal.

4.1.1 Superimposition of electromagnetic signals

Since electromagnetic signals are transmitted generally undirected over a broadcast channel, several signal components originating from the same signal source may arrive the receiver on multiple signal paths. Signals are reflected from obstacles or also might change their propagation direction by Diffraction at edges of objects. Since the propagation speed of all signal components is identical, signals propagated along indirect signal paths take longer than a signal component along the direct line of sight (LOS).

At a receiver, all incoming electromagnetic waves add up to one superimposed sum signal. In general, this is true for signals of arbitrary frequency and from arbitrary signal source. However, in practise, a receiver is capable of filtering a restricted frequency spectrum so that only signal components that fall in this spectrum are considered by the receiver. Figure 4.2 illustrates how a single sum signal is composed of individual signal components. The interference of other signal components can therefore be constructive or destructive. constructive interference occurs, when identical signal components arrive at a remote receiver in phase so that the sum signal reflects an amplification of the individual signal components. Destructive interference occurs when signal components arrive out of phase. Naturally, destructive interference is the rule and might lead to a heavily distorted sum signal that is not distinguishable by a receiver.

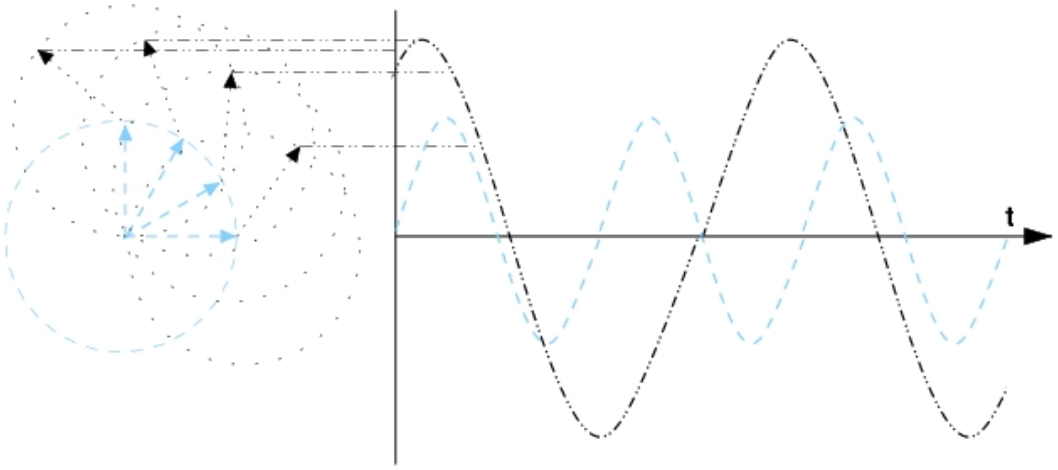


Figure 4.2: Composition of a superimposed sum signal from two individual signal components

4.1.2 Path-loss

Electromagnetic waves decrease in signal strength when propagating over the wireless channel. The order of this decay varies in different environments. With higher frequencies the effect is greater but it can also be reduced with improved (as, for instance, directed) antennas. The actual path-loss in a given environment is heavily dependent on the environmental characteristics. For analytic consideration, the path-loss in a given scenario can therefore be approximated at most. For various scenario classes, different formulae are proposed to approximate this path-loss.

A general formula to calculate the received signal strength (RSS) in free space is the Friis free-space equation [84]:

$$P_{RX} = P_{TX} \cdot \left(\frac{\lambda}{2\pi d}\right)^2 \cdot G_{TX} \cdot G_{RX} \tag{4.3}$$

In this formula, P_{TX} and P_{RX} denote the transmit and receive power (i.e. the signal strength), d is the distance in meters between sender and receiver and G_{TX}, G_{RX} are the gain of the transmit and receive antennas. Observe that the distance d impacts the received signal strength quadratically. This is due to the decrease of the electric field, which decreases linear in the distance d so that the power per square meter decreases as d^{-2}.

The free-space equation can be utilised for outdoor scenarios in which no reflections of the signal components are considered and in which therefore only a single direct signal component from each transmitter is assumed. This assumption holds approximately in wireless networks, when the receiver is sufficiently far away from the transmitter in the so-called far-field distance. For cellular systems like GSM or UMTS this distance is in the range of 1 km while for short range systems like WLAN it is in the range of 1 m [84].

The path-loss is defined as the fraction of the transmitted signal strength P_{TX} to the received signal strength P_{RX}:

$$PL^{FS}(\zeta_i) = \frac{P_{TX}(\zeta_i)}{P_{RX}(\zeta_i)} \tag{4.4}$$

A path-loss model suited in buildings or densely populated areas is the log-distance path loss model [93]:

$$PL^{LD}(\zeta_i) = \frac{P_{TX}(\zeta_i)}{P_{RX}(\zeta_i)} = 10^{\frac{L_0}{10}} \cdot d^{\alpha} \cdot 10^{\frac{x_g}{10}} \tag{4.5}$$

or in dB:

$$PL^{LD}(\zeta_i) = P_{TX}(\zeta_i) - P_{RX}(\zeta_i) = L_0 + 10 \cdot \alpha \cdot \log_{10}\left(\frac{d}{d_0}\right) + X_g[dB] \tag{4.6}$$

In these formulae, L_0 represents the path-loss at a reference distance d_0 in dB, while d represents the length of the path and α is the path-loss exponent. X_g is a random variable with zero mean that represents the attenuation due to fading in dB. When only slow fading occurs, this variable may follow a Gaussian distribution with standard deviation σ in dB (or a log-normal distribution of the received power in Watt). When only fast fading occurs, this gain may be modelled as a variable following a Rician or Rayleigh distribution. Both these distributions are introduced in section 4.1.3.

In Urban scenarios, where often no direct signal component exists, reflection of signals and multipath propagation are common and have to be considered. Effectively, the received signal strength is decreased at a higher pace in such scenarios. A diverse set of distinct models to account for this increased path-loss are proposed in the literature. For the purpose of our lecture we will not detail these models since generality is not achieved by any of these. It is, however, important to note that the path-loss differs in various scenarios and that this impact can be expressed by the path-loss exponent α. In the Friis equation, α was set to 2. In other environments than in free space, it may also take other values up to 6 (for instance shadowed areas and obstructed in-building scenarios) [84] but also $\alpha < 2$ is possible. Table 3.1 depicts various path-loss exponents for distinct wireless scenarios.

4.1.3 Fading

As we have learned in the last section, the signal strength generally deteriorates with greater transmission distance. Furthermore, the signal quality is fluctuating with location and with time even when the distance to receiver is kept identical. There are two effects that account for this problem and which are referred to as slow fading and fast fading.

Slow fading occurs when the coherence time of the channel is large relative to the delay constraint of the channel. It is the result of environmental changes so that signal paths are blocked or changed. Examples are Trees moving in the wind, moving vehicle, open or closed doors or also pedestrians. The amplitude change caused by these effects can be modeled by a log-normal distribution with a standard deviation according to the log-distance path loss model.

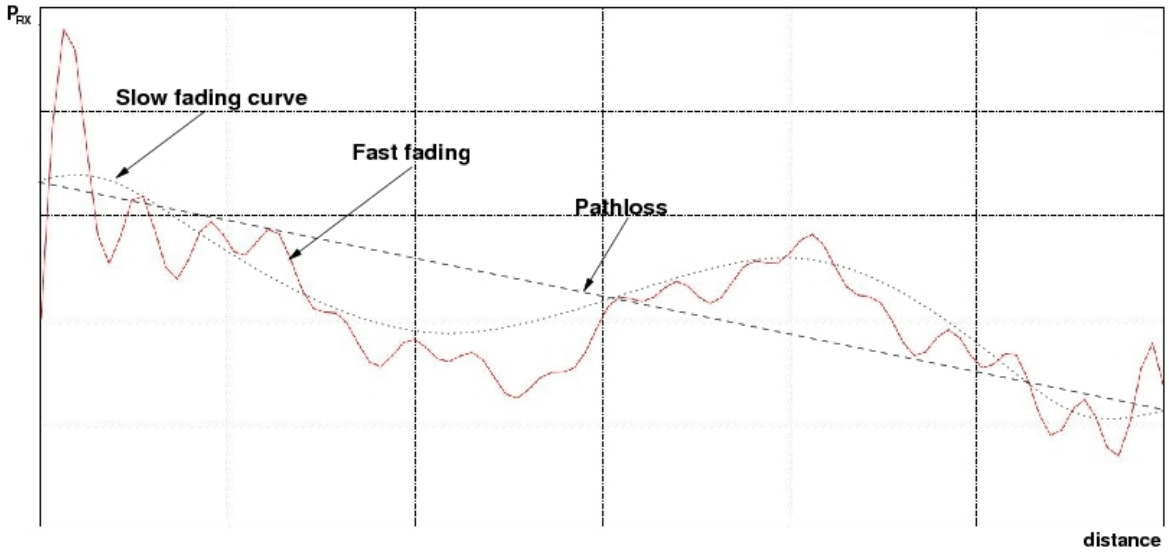


Figure 4.3: Schematic illustration of fading effects on the RF signal

In contrast, fast fading describes a phenomenon that several signal components on multiple paths cancel out each other on the RF-level or also add up to a stronger signal. Figure 4.3 depicts these effects schematically. Fading incursions of a fast fading profile are to be expected in the distance of $\frac{\lambda}{2}$ so that the channel quality changes drastically even when the receiver is moved for only a short distance. For simulation studies and to model the channel in a given environment, typically statistical measures are utilised to model the fading profile. Popular probability distributions to estimate the probability of fading incursions are, for example, the Rice and Rayleigh distributions.

When a direct line of sight is experienced between receiver and transmitter, a dominating signal component exists. We can then assume that the fast fading is weakened because of the dominance of the main signal. This case can be modelled by the Rice distribution which is defined as

$$f(A) = \frac{A}{\sigma^2} e^{-\frac{A^2+s^2}{2\sigma^2}} I_0\left(\frac{As}{\sigma^2}\right) \tag{4.7}$$

In this formula, $I_0(x)$ is the modified Bessel function with order 0. It is

$$I_0(x) = \frac{1}{2\pi} \int_0^{2\pi} e^{x\cos(\Psi)} d\Psi \tag{4.8}$$

The value s in equation (4.7) describes the dominant component of the received signal and σ is the standard deviation. We define the Rice factor as

$$K = \frac{s^2}{2\sigma^2}. \tag{4.9}$$

This factor impacts the probability density function of equation (4.7) or impacts, intuitively, the most probable outcome for the probability function.

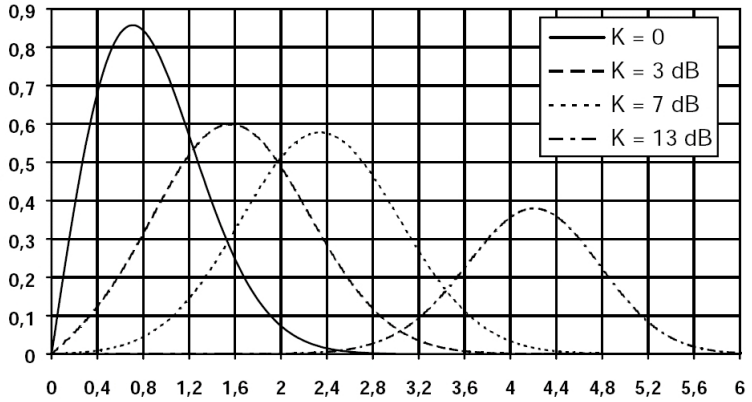


Figure 4.4: Probability density functions for various values of K

For $K = 0$, the Rice distribution migrates to the Rayleigh distribution:

$$\begin{aligned}
\lim_{K\to 0} f(A) &= \lim_{K\to 0} \frac{A}{\sigma^2} e^{-\frac{A^2}{2\sigma^2}-K} I_0\left(\frac{A\sqrt{2K}}{\sigma}\right) \\
&= \lim_{K\to 0} \frac{A}{\sigma^2} e^{-\frac{A^2}{2\sigma^2}-K} \frac{1}{2\pi} \int_0^{2\pi} e^{\frac{A\sqrt{2K}}{\sigma}\cos(\Psi)} d\Psi \\
&= \frac{A}{\sigma^2} e^{-\frac{A^2}{2\sigma^2}-0} \frac{1}{2\pi} \int_0^{2\pi} e^{\frac{A\sqrt{2\cdot 0}}{\sigma}\cos(\Psi)} d\Psi \\
&= \frac{A}{\sigma^2} e^{-\frac{A^2}{2\sigma^2}} \qquad (4.10)
\end{aligned}$$

Figure 4.4 illustrates probability density functions for various values of K.

The Rayleigh distribution

$$\frac{A}{\sigma^2} e^{-\frac{A^2}{2\sigma^2}} \qquad (4.11)$$

describes the probability density function of the amplitude of the received signal if we consider $n \gg 1$ signals of equal strength. It is utilised when all components of a received signal are indirect and approximately equal in power as, for example, in urban scenarios with dense house blocks.

With large K (very small dispersion component) the rice distribution evolves to the

Gauss distribution:

$$
\begin{aligned}
I_0(x) \quad &\rightarrow_{x\gg 1}\rightarrow \quad \frac{e^x}{\sqrt{2\pi}} \\
\Rightarrow f(A) \quad &\rightarrow_{x\gg 1}\rightarrow \quad \frac{A}{\sigma^2} e^{-\frac{A^2}{2\sigma^2}-K} \frac{e^{\frac{A\sqrt{2K}}{\sigma}}}{\sqrt{2\pi\frac{A\sqrt{2K}}{\sigma}}} \\
f(A) \quad &= \quad \frac{A}{\sigma^2\sqrt{\frac{2\pi}{\sigma}}\sqrt{A\sqrt{2K}}} e^{-\frac{A^2}{2\sigma^2}-\frac{s^2}{2\sigma^2}} e^{\frac{A\sqrt{2K}}{\sigma}} \\
&= \quad \frac{A}{\sigma^2\sqrt{\frac{2\pi}{\sigma}}\sqrt{A\sqrt{2K}}} e^{-\frac{A^2+s^2-2As}{2\sigma^2}} \\
&= \quad \sqrt{\frac{A}{s}} \frac{1}{\sigma 2\pi} e^{-\frac{1}{2}\left(\frac{A-s}{\sigma}\right)^2}
\end{aligned} \tag{4.12}
$$

This expression differs only in the term $\sqrt{\frac{A}{s}}$ from the Gauss distribution:

$$
f_{Gauss}(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{A-s}{\sigma}\right)^2} \tag{4.13}
$$

With $\sqrt{\frac{A}{s}} \approx 1$ the Rice distribution can be approximated by the Gauss distribution.

4.1.4 Noise, interference and spread spectrum systems

Electromagnetic waves are transmitted from multiple sources at various frequencies. Signal components are reflected at buildings and other obstacles and are scattered in arbitrary directions. Therefore, a certain signal power is always present at a given frequency range. It is referred to as thermal noise. The exact noise figure is heavily fluctuating and dependent on the environment encountered. In measurements taken in [94] a typical noise power of

$$
P_N = -103dBm \tag{4.14}
$$

was observed.

The thermal noise power can also be estimated analytically as

$$
P_N = \kappa \cdot T \cdot B, \tag{4.15}
$$

where $\kappa = 1.3807 \cdot 10^{-23} \frac{J}{K}$ is the Boltzmann constant, T is the temperature in Kalvin and B is the bandwith of the signal. For example, in a GSM system with $200kHz$ bands and an average temperature of $300K$, the noise power is estimated to be $P_N = -120.82dBm$.

In networks in which various parties communicate over the same wireless channel neighbouring nodes cause interference as discussed in section 4.1.1. Electromagnetic waves that are received simultaneously are combined and add up. Figure 4.5 illustrates this property schematically for three distinct sinusoid signal components.

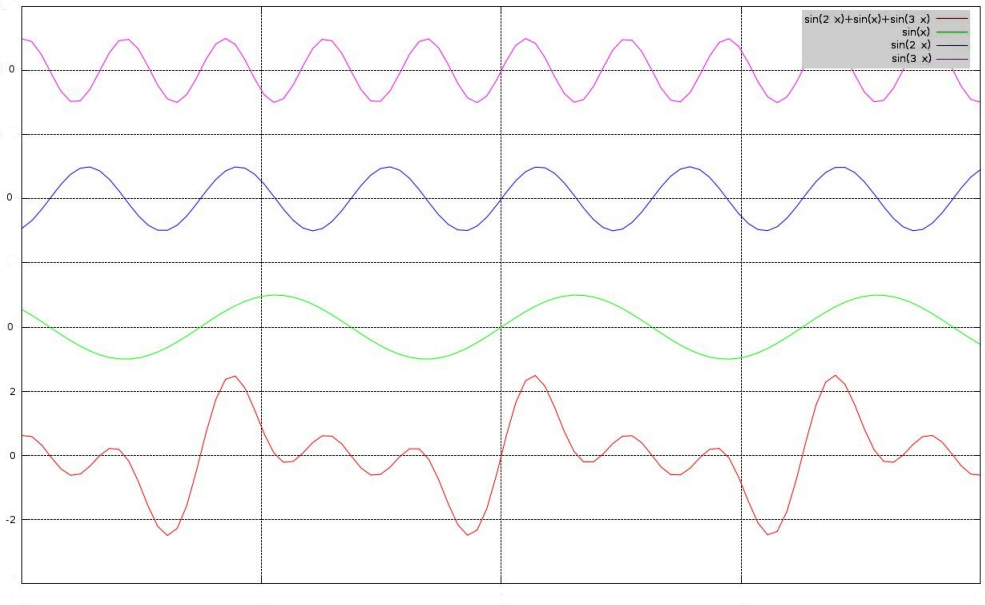


Figure 4.5: Superimposition of signal waves

For ι signal components $\Re\left(e^{j(f_i t+\gamma_i)}\right)$ that are received simultaneously, the sum signal is

$$\zeta_{\text{sum}} = \sum_{i=1}^{\iota} \Re\left(e^{j(f_i t+\gamma_i)}\right). \tag{4.16}$$

To cope with interference and noise, concepts like clustering or CDMA are introduced. A radio system typically requires a specific minimum signal power over the interference and noise level. This is referred to as the signal to interference and noise ratio (SINR) which is defined as

$$SINR = \frac{P_{\text{signal}}}{P_{\text{noise}} + P_{\text{interference}}} \tag{4.17}$$

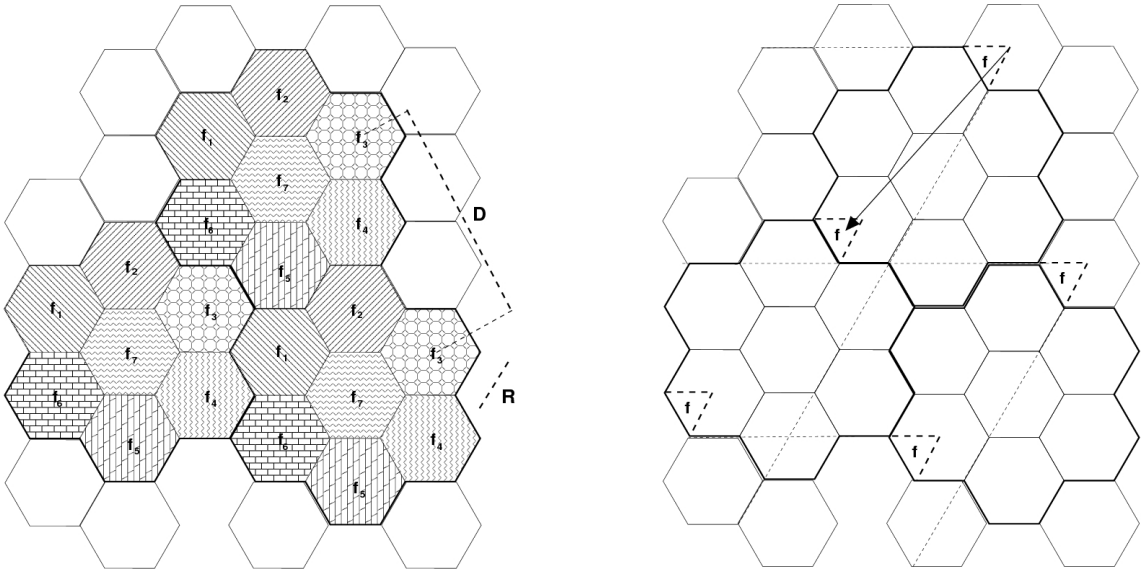
In cellular networks, to reduce the inter-cell interference from cells that utilise the same frequency range, cells can be clustered to groups of neighbouring cells that operate at different sub-frequencies in the whole available frequency band.

Figure 4.6(a) illustrates this procedure schematically with cells depicted as octagons.

Since cells with identical frequencies are separated a considerable distance, the interference level is reduced as signal strength fades with transmission distance. The concept of clustering is, for example, utilised in GSM instrumentations. The number of interfering cells can be further reduced by utilising sectored antennas as depicted in figure 4.6(b)

For sensor networks, clustering is typically not implemented since relative locations of sensors are seldom known so that the organisation of a cluster structure is not straightforward.

Instead, spread-spectrum systems are utilised in some sensor network instrumentations. The general idea of these techniques is to spread the spectrum utilised for transmission to



(a) schematic illustration of a cellular network (b) Interfering cells with sectored antennas

Figure 4.6: Cellular network structures

reduce the impact and/or probability of interference for a transmission.

One possibility is to utilise frequency hopping techniques as, for example, implemented in current Bluetooth networks. Bluetooth devices work in the ISM band between 2.402 and 2.480 GHz. The utilised frequency band is then divided into 79 sub-frequency bands that are spaced by 1 MHz. A Bluetooth device changes the transmission frequency at a high pace up to 1600 times per second.

The hop sequence is defined in a pseudo-random manner from the hardware-address of a master node. When a new node wants to enter the network, it listens to the master and waits for it to send inquiry access codes (IAC). On hearing such a message, the device sends a package containing its device address and timing information. The master node then controls the hopping of the device. Figure 4.7 illustrates this communication paradigm.

Due to this fast hopping frequency, interferences that affect only small frequency bands are reduced. However, the processing required for frequency hopping schemes would typically surcharge the processing capabilities of sensor nodes.

A spread spectrum technique utilised in WSNs is code division (for instance, CDMA). In these approaches, carrier signals are transmitted over a very wide spectrum (in contrast to the frequency hopping approach) so that interference in limited frequency bands will not have great impact on the overall transmitted information. Additionally, various devices share the same frequency and and transmit simultaneously. The overlayed received signal at a receiver is then again extracted and allocated to the distinct transmitters by utilisation of code division.

Basically, the transmit sequence is combined by a pseudo-noise sequence at the transmitter and then transmitted over the channel. Each transmitter has a unique pseudo

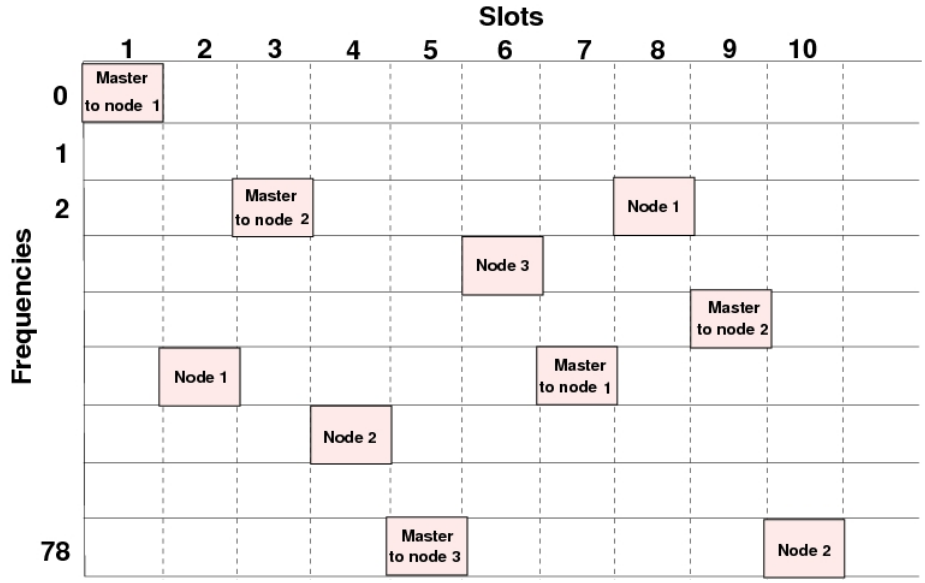


Figure 4.7: Frequency hopping communication in Bluetooth

noise-sequence. At the receiver, the original signal is obtained by decoding the transmitted signal with the correct pseudo-noise sequence for a given transmitter.

Figure 4.8 illustrates the generation of a modified signal from the original signal through XOR combination with a code sequence.

For CDMA systems code sequences for distinct nodes are required to be orthogonal to each other. This means that it shall not be possible that one code sequence occurs in any other code sequence. Therefore, the number of code sequences of a given length is sharply limited. Such orthogonal codes are created by various algorithms. One example is Orthogonal Variable Spreading Factor (OVSF).

OVSF guarantees that spreading codes of different length remain orthogonal. Starting with a root spreading code $c_{i,j} \in \{0,1\}^i; i,j \in \mathbb{N}$, new spreading codes are created as $c_{2i,2j-1} = (c_{i,j}c_{i,j})$ and $c_{2i,2j} = (c_{i,j}\overline{c_{i,j}})$. In this creation rule, $\overline{x}$ is the binary complement of x. Figure 4.9 illustrates all OVSF codes of length 16, starting with $c_{1,1} = (1)$.

4.2 MIMO

For wireless communication we traditionally assume one transmitter and one receiver each with a single antenna. For such systems, capacity is increased by utilising diversity schemes as, for instance, time diversity (several nodes are scheduled in transmit and receive time slots), frequency diversity (several communication channels are implemented on various frequencies) or diversity by modulation with different random sequences as, for instance, CDMA.

Another possibility is to utilise spatial diversity. In section 4.1.4 we have already learned of a frequency and spatial diversity scheme: Clustering. In this scheme, the same frequency

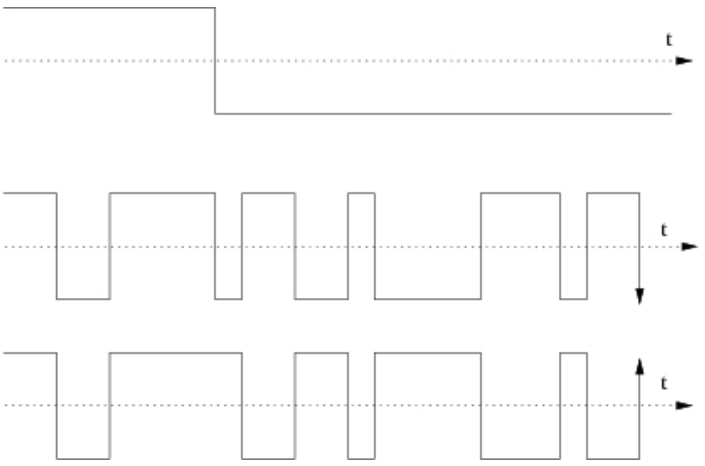


Figure 4.8: DS-CDMA scheme

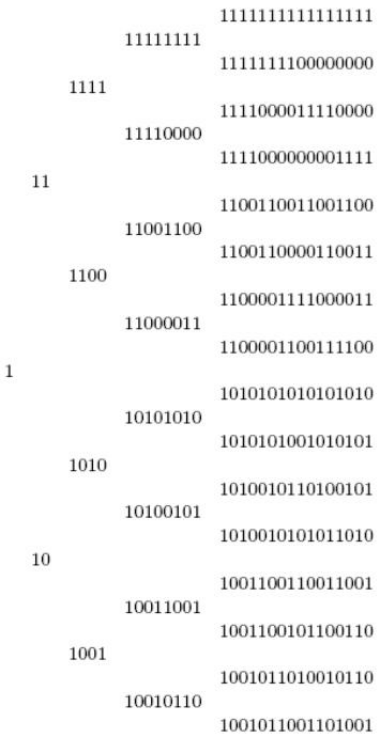


Figure 4.9: Orthogonal OVSF spreading codes of length 16 with $c_{1,1} = (1)$

is utilised in distinct cells that have low interference to each other so that an area of the communication system is served by several base stations in these cells. This is more energy efficient than the alternative of a single base station since energy radiated degrades quadratically at least (cf. section 4.1.2)

Another approach to utilise spatial diversity is by multiple transmit antennas for a single communication link that are spatially separated. When the distance between antennas is sufficiently large (a typical estimation is $\frac{\lambda}{2}$), the communication channels established by these antennas are considered independent. This means that for two distinct communication channels the channel quality fluctuates independently from each other so that with sufficient amount of distinct channels the probability of all channels to experience inferior channel quality is low.

We distinguish communication channels by the number of transmit and receive antennas. The traditional and straight forward implementation is a system with one transmit and one receive antenna, i.e. a single input, single output system (SISO). In a SISO system, diversity is utilised only in the form of several independent parallel channels.

Other channel models are SIMO or MISO channels, where multiple transmitters but only one receiver or vice versa are utilised.

When on both ends of the communication channels multiple transmitters/receivers reside, we speak of MIMO systems.

The transmission behaviour of a MIMO-System is characterised by the following Vector-Matrix:

$$\overrightarrow{\zeta^{RX}} = \begin{bmatrix} \zeta_1^{RX} \\ \zeta_2^{RX} \\ \vdots \\ \zeta_M^{RX} \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1L} \\ h_{21} & \ddots & & h_{2L} \\ \vdots & & \ddots & \vdots \\ h_{M1} & h_{M2} & \cdots & h_{ML} \end{bmatrix} \begin{bmatrix} \zeta_1^{TX} \\ \zeta_2^{TX} \\ \vdots \\ \zeta_L^{TX} \end{bmatrix} + \begin{bmatrix} \zeta_1^{\text{noise}} \\ \zeta_2^{\text{noise}} \\ \vdots \\ \zeta_M^{\text{noise}} \end{bmatrix} \tag{4.18}$$

In this formula,

$$\overrightarrow{\zeta^{RX}} = (\zeta_1^{RX}, \zeta_2^{RX}, \ldots, \zeta_M^{RX})^T \tag{4.19}$$

is the vector of received signal components ζ_i^{RX}. The receiver has M inputs and the transmitter L outputs. The channel matrix H defines how each input is connected to each output. The vector

$$\overrightarrow{\zeta^{\text{noise}}} = (\zeta_1^{\text{noise}}, \zeta_2^{\text{noise}}, \ldots, \zeta_M^{\text{noise}})^T \tag{4.20}$$

is the vector of noise signals that is added onto the individual transmit channels for each one receiver input.

The channel matrix can typically be altered only in particular boundaries e.g. by reordering of the antennas. It is, however, mainly dependent on the environmental situation of the wireless system.

These individual signal components can be utilised to improve the overall quality of the communication channel. We can, for instance, improve the communication speed when

symbols are transmitted over these spatially separated channels in parallel instead of sequential transmission in a SISO-system. Alternatively, identical symbols can be transmitted on all channels synchronised to improve the robustness against interference and signal fading [95].

4.3 Beamforming

The general idea of (centralised) beamforming in wireless communication systems is to create an antenna beam that is focused on a restricted area. Intuitively, signal components form various transmit antennas are coherently overlaid to create constructive interference in this area while outside, signal components constructively and destructively interfere to create a general low noise level. For a fixed antenna array in which the exact relative location of all antennas is known, the signals sent from each antenna element are suitably weighted to focus and control the beam on a target location.

From each antenna element, identical symbols are transmitted. When these symbols from n such antenna elements are perfectly aligned at a receiver location, the SINR is amplified by factor n in the best case.

We can see this easily: When $y_i(t) = \Re(e^{j(2\pi f_i t + \gamma_i)})$ is the received signal component of transmitter $i, i \in [1...n]$,

$$\sum_{i=1}^{n} \Re(e^{j(2\pi f_i t + \gamma_i)}) \tag{4.21}$$

defines the received sum signal. Consequently, when all signal components are in phase, the signal strength is increased linear to the number of signal components. For centralised beamforming, all antenna elements are tightly synchronised so that the antenna beam can be controlled by the transmitter node.

5 Basics on probability theory

> *The generation of random numbers is too important to be left to chance.*
>
> (Robert R. Coveyou – Oak Ridge National Laboratory.)

The methods discussed in later chapters require some basic knowledge on the notion of probability. We can differentiate between probability spaces with finite or infinite number of events. For this lecture it suffices to consider only probability spaces with finite events. This section is designated to provide these basics on probability theory. Some of the examples and illustrative explanations are lend from [96, 97].

5.1 Discussion

Historically, probability theory was designed to describe phenomena observed in games of chance, that could not be described by classical mathematical theory. A simple example is the description of the outcome of a coin-tossing.

Today we find probability in many aspects of everyday life. An omnipresent example is the weather forecast. What we can learn from it is typically not that it will definitely rain or not, but that a certain probability of rain was derived from distinct measurements at various places all over the world and over a considerable amount of time. Although this information provides no guarantee that it will actually rain or not, we have developed an intuitive understanding of probability so that the information is useful to us although it inherits the chance to be incorrect. Other examples are insurance, where probability is used to calculate the probability of ruin or also lottery. The latter example is fascinating as people regularly ignore the infinitesimal probability to win a considerable amount of money.

A further instance where we regularly stumble across probability are quiz shows on TV. Consider, for instance, the following setting (see figure 5.1). A quiz master confronts a candidate with three doors A, B and C and explains that behind one of these the candidate will find (and win) a treasure while the candidate will win nothing if he opens one of the other doors. The candidate then decides for one of these doors, but before it is opened, the quiz master opens one of the remaining two doors and proves that this door does not cover the treasure. The candidate is now given the opportunity to rethink his decision and vote for the closed door he didn't vote for initially.

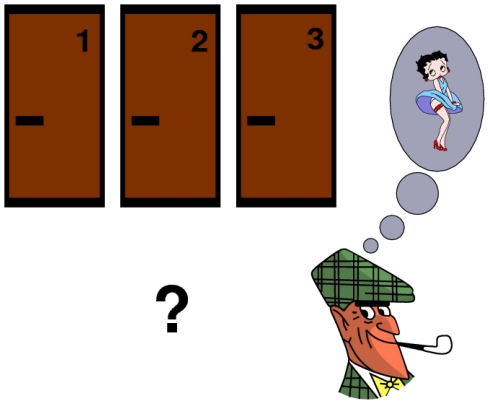


Figure 5.1: Which door hides the treasure?

What should the candidate do in order to maximise the probability to win the treasure? We can show that it is better to alter the initial choice since the remaining closed door contains the treasure with probability $\frac{2}{3}$.

Exercise 5.1.1 :

> *Explain why the probability that the treasure is covered behind the remaining closed door is* $\frac{2}{3}$.

5.2 Preliminaries

In order to calculate with probabilities we have to define an idealised model. We generally assume that probabilities can be verified by conceptual experiments. This means that the probability for an event represents the quotient of the number of occurrences of this event to the number of repetitions of the experiment when the experiment is repeated very often. An event with probability 0.4 should be expected to occur forty times out of one hundred in the long run. A typical, often quoted example is the tossing of a coin. We expect as an outcome of the experiment one of the two events head or tail with equal probability $\frac{1}{2}$. Note that this assumption is idealised in that we assume a 'fair' coin and disregard some possible but unlikely events like the case that the coin will fall on neither side but, for example, might roll away or stand on its edge.

We have already introduced some typical notation. The results of experiments or observations are called events. Events are sets of sample points. We denote events by capital letters. The fact that a sample point x is contained in event E is denoted by $x \in E$. We write $E_1 = E_2$ when no point $x \in E_1 \wedge x \notin E_2$ or $x \in E_2 \wedge x \notin E_1$ exists. The sample space of an experiment is the set of all possible events. The sample space for the experiment of tossing a coin two times is *head-head, head-tail, tail-head, tail-tail.*

The following examples shall illustrate these concepts

Example 5.2.1 :

Consider the following experiment: Three distinct balls (a,b,c) are to be placed in three distinct bins. The following illustration depicts all possible outcomes of this experiment that together form the sample space

Event / Bin	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	abc			ab	ab	c		c		ac	ac	b		b
2		abc		c		ab	ab		c	b		ac	ac	
3			abc		c		c	ab	ab		b		b	ac

Event / Bin	15	16	17	18	19	20	21	22	23	24	25	26	27	
1		bc	bc	a		a		a	a	b	b	c	c	
2	b	a		bc	bc		a	b	c	a	c	a	b	
3	ac		a		a	bc	bc	c	b	c	a	b	a	

However, the sample space is altered when the conditions of the experiment are altered. Suppose that the three balls are not distinguishable. Consequently the following sample space belongs to this experiment.

Event / Bin	1	2	3	4	5	6	7	8	9	10
1	***			**	**	*		*		*
2		***		*		**	**		*	*
3			***		*		*	**	**	*

When we also consider indistinguishable bins, the sample space becomes

Event / Bin	1	2	3
1	***	**	*
2		*	*
3			*

5.3 Relation between events

We assume an arbitrary but fixed sample space for the remainder of this lecture. The following definitions are essential for our notion of events.

Definition 5.3.1 : Impossible event

With $\chi = \{\}$ we denote the fact that event χ contains no sample points. It is impossible to observe event χ as an outcome of the experiment.

For every event χ there is an event $\neg\chi$ that is defined as 'χ does not occur'.

Definition 5.3.2 : Negation of events

The event consisting of all sample points x with $x \notin \chi$ is the complementary event (or negation) of χ and is denoted by $\neg\chi$.

For two events χ_1 and χ_2 we can also denote new events by the conditions that 'both χ_1 and χ_2 occur' or that 'either χ_1 or χ_2 occur'. These events are denoted by $\chi_1 \cap \chi_2$ and $\chi_1 \cup \chi_2$ respectively. These events are defined by

$$\chi_1 \cap \chi_2 = \{x | x \in \chi_1 \wedge x \in \chi_2\} \tag{5.1}$$

and

$$\chi_1 \cup \chi_2 = \{x | x \in \chi_1 \vee x \in \chi_2\} \tag{5.2}$$

and can be generalised to arbitrary many events. When the events χ_1 and χ_2 have no sample point x in common, the event $\chi_1 \cap \chi_2$ is impossible: $\chi_1 \cap \chi_2 = \{\}$. We say that the events χ_1 and χ_2 are mutually exclusive.

5.4 Basic definitions and rules

All events χ that might possibly occur are summarised in a sample space.

Definition 5.4.3 : Sample space

A sample space Π is a set of elementary events χ_i

These events, together with an occurrence probability constitute the probability space.

Definition 5.4.4 : Probability space

A probability space (Π, P) consists of a sample space Π and a probability measure $P : \Pi \to \mathbb{R}$. This function satisfies the following conditions

- *For each subset $\Pi' \subseteq \Pi, 0 \leq P(\Pi') \leq 1$*
- $P(\Pi) = 1$
- *For each $\Pi' \subseteq \Pi, P(\Pi') = \sum_{\chi \in \Pi'} P(\chi)$*

Given a sample space Π and sample points $x_i \in \Pi$, we denote the probability that x_i is observed by $P(x_i)$ with $P : \Pi \to [0, 1]$ and $P(x_1) + Px_2 + \cdots = 1$.

Definition 5.4.5 : Probability of events

Given a sample space Π *and an event* $\chi \in \Pi$*, the occurrence probability* $P(\chi)$ *of event* χ *is the sum probability of all sample points from* χ*:*

$$P(\chi) = \sum_{x \in \chi} P(x). \tag{5.3}$$

Since all probabilities of the sample space sum up to 1 it follows that

$$0 \leq P(\chi) \leq 1 \tag{5.4}$$

for any event χ.

Consider two arbitrary events χ_1 and χ_2. In order to compute the probability $P(\chi_1 \cup \chi_2)$ that either χ_1 or χ_2 or both occur we add the occurrence probabilities that a sample point either in χ_1 or in χ_2 is observed.

$$P(\chi_1 \cup \chi_2) \leq P(\chi_1) + P(\chi_2) \tag{5.5}$$

The '$\leq$'-relation is correct since sample points might be contained in both events. We therefore obtain the exact probability by

$$P(\chi_1 \cup \chi_2) = P(\chi_1) + P(\chi_2) - P(\chi_1 \cap \chi_2). \tag{5.6}$$

Example 5.4.2 :

We toss a coin twice so that the sample space contains the four sample points head-head, head-tail, tail-head and tail-tail that are associated with probability $\frac{1}{4}$ *each. Consider the two events* χ_1 *– head occurs first –* and χ_2 *– tail occurs second.* χ_1 *then contains head-head and head-tail while* χ_2 *contains head-tail and tail-tail. Consequently,* $\chi_1 \cup \chi_2$ *consists of the sample points head-head, head-tail, tail-tail while* $\chi_1 \cap \chi_2$ *consists of the single sample point head-tail. We therefore obtain the probability that either* χ_1 *or* χ_2 *occurs as*

$$P(\chi_1 \cup \chi_2) = \frac{1}{2} + \frac{1}{2} - \frac{1}{4}. \tag{5.7}$$

This can be generalised to higher event counts. For arbitrary events $\chi_1, \chi_2, \ldots$ this is expressed by the inequality

$$P(\chi_1 \cup \chi_2 \cup \ldots) \leq P(\chi_1) + P(\chi_2) + \cdots - P(\chi_1 \cap \chi_2) - P(\chi_1 \cap \chi_3) \ldots \tag{5.8}$$

In the special case that all events $\chi_1, \chi_2, \ldots$ are mutually exclusive, we obtain

$$P(\chi_1 \cup \chi_2 \cup \ldots) = P(\chi_1) + P(\chi_1) + \ldots \tag{5.9}$$

since $P(\chi_i) \cap P(\chi_j) = \{\}$ for any two mutually exclusive events χ_i and χ_j.

In some cases we are interested in the probability that a specific event occurs in the presence of another event. This can be expressed by the conditional probability.

Definition 5.4.6 : Conditional probability

The conditional probability of two events χ_1 and χ_2 with $P(\chi_2) > 0$ is denoted by $P(\chi_1|\chi_2)$ and is calculated by

$$\frac{P(\chi_1 \cap \chi_2)}{P(\chi_2)} \tag{5.10}$$

$P(\chi_1|\chi_2)$ describes the probability that event χ_2 occurs in the presence of event χ_2 (read: The probability of χ_1 given χ_2). With rewriting and some simple algebra we obtain the Bayes rule that states

$$P(\chi_1|\chi_2) = \frac{P(\chi_2|\chi_1) \cdot P(\chi_1)}{\sum_i P(\chi_2|\chi_i) \cdot P(\chi_i)}. \tag{5.11}$$

This equation is useful in many statistical applications. Note that the denominator is summed over the probability for all possible events. This means that everything on the right hand side of the equation is conditioned on the events χ_i. When we say that χ_i is the important variable, the shape of the distribution $P(\chi_1|\chi_2)$ depends on the numerator $P(\chi_2|\chi_1) \cdot P(\chi_1)$ with the denominator as a normalising factor that ensures that the $P(\chi_1|\chi_2)$ sum to 1. The Bayes rule is therefore interpreted that it inverts $P(\chi_1|\chi_2)$ to $P(\chi_2|\chi_1)$. This is useful when it is easy to calculate $P(\chi_2|\chi_1)$ but not to calculate $P(\chi_1|\chi_2)$. With Bayes rule it is then easy to calculate $P(\chi_1|\chi_2)$ provided that we know $P(\chi_2|\chi_1)$ and $P(\chi_1)$.

Definition 5.4.7 : Independence

A collection of events χ_i that form the sample space Π is independent if for all subsets $S \subseteq \Pi$

$$P\left(\bigcap_{\chi_i \in S} \chi_i\right) = \prod_{\chi_i \in S} P(\chi_i). \tag{5.12}$$

Statistical independence is required for many useful results in probability theory. This means, on the other hand, that we have to be careful to apply such results not in cases where independence between sample points is not provided.

Definition 5.4.8 : Expectation

The expectation of an event χ is defined as

$$E[\chi] = \sum_{x \in \mathbb{R}} x \cdot P(\chi = x) \tag{5.13}$$

Although intuitively the expectation of an event represents the expected outcome of the event, the expectation is not necessary equal to one of the possible sample points.

Consider, for instance, the event χ of throwing a dice. The Sample space is given by $S_\chi = \{1, 2, 3, 4, 5, 6\}$. However, the expectation of this event is

$$E[\chi] = \frac{1}{6} \cdot (1 + 2 + 3 + 4 + 5 + 6) = 3.5. \tag{5.14}$$

It is also possible to perform calculations with expectations of events.

Definition 5.4.9 : Linearity of expectation

For any two random variables χ_1 and χ_2,

$$E[\chi_1 + \chi_2] = E[\chi_1] + E[\chi_2]. \tag{5.15}$$

For an independent random variables χ_1 and χ_2,

$$E[\chi_1 \cdot \chi_2] = E[\chi_1] \cdot E[\chi_2]. \tag{5.16}$$

Definition 5.4.10 : Variance

The variance of a random variable χ is defined as

$$var[\chi] = E[(\chi - E[\chi])^2]. \tag{5.17}$$

For any random variable χ

$$var[\chi] = E[\chi^2] - E[\chi]^2 \tag{5.18}$$

For any independent random variables χ_1 and χ_2

$$var[\chi_1 + \chi_2] = var[\chi_1] + var[\chi_2]. \tag{5.19}$$

For any random variable χ and any $c \in \mathbb{R}$,

$$var[c\chi] = c^2 var[\chi]. \tag{5.20}$$

5.4.1 The Markov inequality

To estimate the deviation of a random variable from its expectation, the following bound is useful.

Definition 5.4.11 : Markov inequality

Let (Π, P) be a probability space and $x : \Pi \to \mathbb{R}^+$ a non-negative random variable. For $t \in \mathbb{R}^*$ the following inequality holds:

$$P(x \geq t \cdot E[x]) \leq \frac{1}{t} \tag{5.21}$$

Proof. For an arbitrary $s \in \mathbb{R}^+$ we define an indicator variable $Y_s : \Omega \to \{0,1\}$ with

$$Y_s := \begin{cases} 1 & \text{if } x \geq s \\ 0 & \text{else} \end{cases} \tag{5.22}$$

We have $x \geq s \cdot Y_s$ for all s. Therefore we obtain

$$\begin{aligned} E[x] &\geq E[s \cdot Y_s] \\ &= s \cdot E[Y_s] \\ &= s \cdot 1 \cdot P(Y_s = 1) + 0 \cdot P(Y_s = 0) \\ &= s \cdot P(x \geq s) \end{aligned} \tag{5.23}$$

Observe that we have shown $P(x \geq s) \leq \frac{E[x]}{s}$. When we choose $s := t \cdot E[x]$ the assertion follows. □

5.4.2 The Chernoff bound

A stronger bound useful to estimate the deviation of a random variable from its mean is the Chernoff bound.

Definition 5.4.12 : Chernoff bound

Let (Π, P) be a probability space and $x_1, x_2, \ldots, x_n : \Pi \to \{0,1\}$ independent random variables with $0 < P(x_i = 1) < 1$ for all $i \in \{1, 2, \ldots, n\}$. For $X := \sum_{1 \leq i \leq n} x_i$ and $\delta > 0$ the following inequality holds:

$$P(X < (1+\delta)E[X]) < \left(\frac{e^\delta}{(1+\delta)^{1+\delta}} \right)^{E[X]} \tag{5.24}$$

and for all δ with $0 < \delta < 1$

$$P(X < (1-\delta)E[X]) < e^{-\frac{E[X]\delta^2}{2}} \tag{5.25}$$

Proof. We first prove the first estimation. Basically we apply the Markov inequality. From there we come to a much stronger result by utilising the random variable $e^{\varsigma X}$ for a suitable constant $k\varsigma$ instead of utilising X. Observe that

$$P(X < (1+\delta)E[X]) = P(e^{\varsigma X} < e^{\varsigma(1+\delta)E[X]}) \tag{5.26}$$

holds for all $\varsigma \in \mathbb{R}^+$. We now apply the Markov inequality and obtain

$$P(X > (1+\delta)E[X]) < \frac{E[e^{\varsigma X}]}{e^{\varsigma(1+\delta)E[X]}} \tag{5.27}$$

We utilise the definition of X and can therefore write

$$E[e^{\varsigma X}] = E\left[e^{\varsigma \sum_{i=1}^n x_i}\right] = E\left[\prod_{i=1}^n e^{\varsigma x_i}\right] \tag{5.28}$$

Following the precondition, the random variables x_i are independent and therefore

$$E\left[\prod_{i=1}^{n} e^{\varsigma x_i}\right] = \prod_{i=1}^{n} E\left[e^{\varsigma x_i}\right] \tag{5.29}$$

Altogether we have now

$$P(X > (1+\delta)E[X]) < \frac{\prod_{i=1}^{n} E\left[e^{\varsigma x_i}\right]}{e^{\varsigma(1+\delta)E[X]}} \tag{5.30}$$

Since X are binary random variables we obtain

$$\begin{aligned} E\left[e^{\varsigma x_i}\right] &= P(x_i = 1)e^{\varsigma \cdot 1} + (1 - P(x_i = 1))e^{\varsigma \cdot 0} \\ &= 1 + P(x_i = 1)(e^{\varsigma} - 1) \end{aligned} \tag{5.31}$$

We estimate $e^x \geq 1 + x$ with $x := P(x_i = 1)(e^{\varsigma} - 1)$ to obtain

$$\begin{aligned} P(X > (1+\delta)E[X]) &< \frac{\prod_{i=1}^{n} e^{P(x_i=1)(e^{\varsigma}-1)}}{e^{\varsigma(1+\delta)E[X]}} \\ &= \frac{e^{(e^{\varsigma}-1)\sum_{i=1}^{n} P(x_i=1)}}{e^{\varsigma(1+\delta)E[X]}} \\ &= \frac{e^{(e^{\varsigma}-1)E[X]}}{e^{\varsigma(1+\delta)E[X]}} \end{aligned} \tag{5.32}$$

Since $\varsigma \in \mathbb{R}^+$ and $\delta > 0$ we choose $\varsigma := \ln(1+\delta)$ and obtain

$$P(X > (1+\delta)E[X]) < \left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}}\right)^{E[X]} \tag{5.33}$$

as claimed.

To prove the second estimation we proceed similarly. It is for all $\varsigma \in \mathbb{R}^+$

$$\begin{aligned} P(X < (1-\delta)E[X]) &= P(-X > -(1-\delta)E[X]) \\ &= P(e^{-\varsigma X} > e^{-(1-\delta)E[X]}) \end{aligned} \tag{5.34}$$

With the Markov inequality we obtain

$$P(X < (1-\delta)E[X]) < \frac{\prod_{i=1}^{n} E\left[e^{-\varsigma x_i}\right]}{e^{-\varsigma(1-\delta)E[X]}} \tag{5.35}$$

With $E\left[e^{-\varsigma x_i}\right] = P(x_i = 1)e^{-\varsigma} + 1 - P(x_i = 1)$ we obtain

$$P(X < (1-\delta)E[X]) < \frac{e^{E[X]\left(e^{-\varsigma}-1\right)}}{e^{-\varsigma(1-\delta)E[X]}} \tag{5.36}$$

for all $\varsigma \in \mathbb{R}+$. We choose $\varsigma = \ln\left(\frac{1}{1-\delta}\right)$ and obtain

$$P(X < (1-\delta)E[X]) < \left(\frac{e^{-\delta}}{(1-\delta)^{1-\delta}}\right)^{E[X]} \tag{5.37}$$

We can additionally assume $0 < \delta \leq 1$ and obtain $(1-\delta)^{1-\delta} > e^{-\delta+\frac{\delta^2}{2}}$ and obtain

$$P(X < (1-\delta)E[X]) < e^{-E[X]\frac{\delta^2}{2}} \tag{5.38}$$

as claimed. □

6 Evolutionary algorithms

> *The theory of evolution by cumulative natural selection is the only theory we know of that is in principle capable of explaining the existence of organised complexity.*
>
> (Richard Dawkins)

Evolutionary algorithms have been developed in the 1960s. The primary aim at that time was to model phenomena observed in the evolution and to use these models to solve algorithmic problems [98]. The two main operators to guide the optimisation process are mutation and crossover. While crossover combines distinct search points to generate a new one, mutation is designed to apply small changes to a single search point. We can distinguish four main branches.

Genetic algorithm (GA): John Holland developed genetic algorithms [99] that operate on the search space $\mathbb{B}^n$ and mainly utilise k-point crossover.

Evolution strategy (ES): Evolution strategies have been developed by Hans-Paul Schwefel and Ingo Rechenberg [100]. These strategies are first applied in the search space $\mathbb{R}^n$ and mainly utilised mutation as recombination operator.

Evolutionary programming (EP): Evolutionary programming has been developed by Larry Fogel [101]. Fogel considers the space of finite automata as search space and utilises mainly mutation as search operator.

Genetic programming (GP): Finally, John Koza developed genetic programming [102]. For this approach, the search space is mostly represented by a set of graphs that represent logical expressions or programs.

For several years now these approaches are less clear to separate from each other so that we speak of evolutionary approaches or evolutionary algorithms. Evolutionary algorithms operate on a search space S. All points in this search space are associated with a fitness value that is provided by a fitness function f. Crucial for the optimisation and application in practical settings is usually the time until a search point is found that reaches or exceeds a given fitness threshold or at least that could not been improved for some while. An evolutionary algorithm maintains in a population of search points a set of μ search points

(a) The classical approach to developing an algorithm

(b) The development and application of evolutionary algorithms

Figure 6.1: Approaches to algorithmic development: A comparison

(individuals) from the search space S. It iterates an ever identical sequence of operations (generations) until a stop criteria is reached. In every generation initially all members of the populations are evaluated with a non-monotonously-decreasing fitness function $f : S \to \mathbb{R}$. In a random process, individuals are elected for mutation or crossover. This leads to a set of ν individuals of an offspring population. Depending on the strategy implemented, either μ individuals with the highest fitness score from all $\mu + \nu$ individuals ($(\mu + \nu)$-strategy) or the μ individuals with highest fitness value are picked from the offspring population of size ν ((μ, ν)-strategy).

The research field of evolutionary algorithms has been studied also in the scope of theoretical computer science in recent years [103, 104, 105, 106]. Most relevant in our context are the results on $(1 + 1)$-strategies, [107, 108, 109, 110, 111]. We will see in chapter 8 that this is the natural scenario for distributed adaptive transmit beamforming in wireless sensor networks.

6.1 Basic principle and notations

Evolutionary algorithms differ from classical algorithms at first in the way they are developed. In the classical approach a problem is the origin of the development or optimisation of an algorithm. The problem is analysed and, based on this analysis an algorithm for this very problem is developed. After the algorithm has been designed a theoretical analysis might help to improve the algorithmic design. Also, the algorithm might be implemented and applied to experimental data or in problem scenarios. In both cases, the experience gained during the implementation or application of the algorithm might affect the design of the algorithm (cf. figure 6.1(a)).

For evolutionary algorithms, the procedure is different. The design and optimisation of the algorithm is then independent from the concrete problem. When the algorithm is developed, it is applied on one or many distinct problems. These problems might have been existent beforehand and it is not said that the evolutionary algorithm is especially well suited to solve these problems. So, why is this a reasonable approach? Normally,

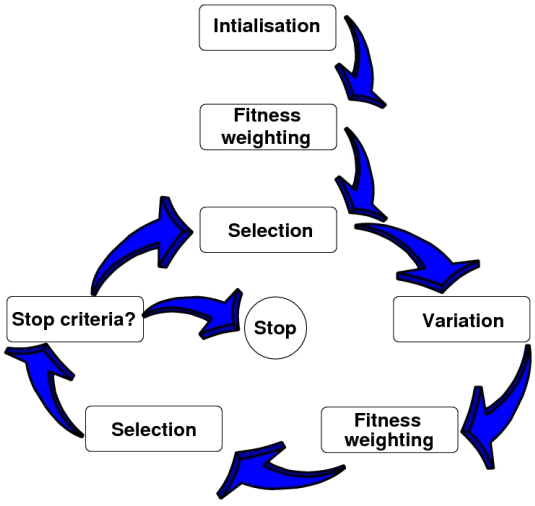


Figure 6.2: Modules of an evolutionary algorithms

the problems to which evolutionary algorithms are applied are not well understood or too complex to describe them analytically. Therefore it might be very complex or even impossible to state a deterministic algorithm that is able to achieve sufficient results on the given problem. Evolutionary algorithms are designed for specific search spaces and might inherit properties that enable them to cope with these search spaces. A theoretical analysis of evolutionary algorithms is therefore commonly applied for a given algorithm on a specific problem. These results then might impact the design and optimisation of the algorithm, but this regression is not always applied so that the algorithm might be taken as an as-is implementation that is applied on arbitrary problems (cf. figure 6.1(b)).

Evolutionary algorithms are random search heuristics that operate in ever identical iterations. Figure 6.2 depicts the various modules/phases that constitute an evolutionary algorithm.

6.1.1 Initialisation

Before the start of the optimisation the population of the algorithm is to be initialised. Typically, the μ individuals from the search space S that constitute the initial population are drawn uniformly at random. Typical search spaces are $S = \mathbb{R}^n$ or $S = \mathbb{B}^n$. In order to achieve sufficient coverage of the search space, we can also define a distance measure d and populate the initial population with points that are at least of distance d to each other. Also, in order to improve the optimisation time and the quality of the solution, we can utilise fast heuristics to find the individual population.

6.1.2 Fitness function – Weighting of the population

At the start of the optimisation, the individuals of the population are weighted for their fitness value. For each individual in the population, a fitness function $f : S \to \mathbb{R}$ is applied.

The fitness function is a monotonous function that maps individuals from the search space onto $\mathbb{R}$.

6.1.3 Selection for reproduction

Dependent on the fitness values reached by the individuals in the population, individuals are chosen to produce the offspring population. The intuition is that individuals with good fitness value have a higher probability to produce individuals for the offspring population that also inherit a higher fitness value than individuals with lower fitness values. Typical selection mechanisms are

Uniform selection: An individual is chosen uniformly at random from all individuals in the population.

Fitnessproportional selection: An individual $\mathcal{I}$ is chosen from the population $\mathcal{P}$ with probability $\frac{f(\mathcal{I})}{\sum_{x \in \mathcal{P}} f(x)}$. A problem with this scheme is that the selection behaviour is considerably different for two fitness functions f and $f + c$. Furthermore, when fitness values are sufficiently separated, the selection is nearly deterministic while it is very similar to the uniform selection when the differences between fitness values are small relative to their absolute values.

Tournament selection: The intuition for this selection method is to find the best individuals from a randomly drawn subset of the whole population. This selection method also allows individuals with non-optimal fitness values to be chosen for the variation.

Apart from these approaches it is, of course, also possible to deterministically choose the k highest rated individuals for the selection.

6.1.4 Variation

For the selected individuals, the offspring population is created by mutation and/or crossover. While mutation is typically a local search operator, crossover allows to find search points in regions of the search space that are currently not populated. Dependent on the stage/progress of the optimisation, some implementations therefore adapt the probability to apply crossover and/or mutation.

Mutation

Mutation alters the representation of an individual, e.g. by flipping of bits for individuals from $\mathbb{B}^n$. The intuition for the mutation is that it produces individuals for the offspring population that differ only slightly from the parent-individuals. For the mutation operator, one parent individual produces one offspring individual. It is clear that possible mutation operators differ between search spaces.

Mutation operators for individuals from $\mathbb{B}^n$:

Standard bit mutation: The offspring individual is created bit-wise from the parent individual. Every bit is 'flipped' with probability p_m and sustained with probability $(1 - p_m)$. A common choice is $p_m = \frac{1}{n}$ as this represents the case that one bit is flipped on average in one mutation.

1 bit mutation: The offspring individual is identical to the parent individual in all but one bit. This bit is chosen uniformly at random from all n bits in the individual representation.

Mutation operators for $\mathbb{R}^n$: An offspring individual is, in general, generated by adding a vector $m \in \mathbb{R}^n$ to the parent individual. For this operation we can distinguish between restricted or unrestricted mutation. For restricted mutation the vector is chosen form a restricted interval (e.g. all vector entries are chosen from $[-a, a]$).

Crossover

Crossover is an alternative technique that produces one or more offspring individuals from two or more parent individuals.

Crossover operators for $\mathbb{B}^n$:

k-point crossover: From the possible n positions in one individual representation, $k \leq n$ positions are chosen uniformly at random. The offspring individuals are created such that the representation is concatenated from sub-sequences of the parent individuals when the origin of the sub-sequence is altered at every one of the k chosen positions. For two parent individuals x_1, x_2 and 2-point crossover with k_1, k_2, an offspring individual y can be created as follows:

$$\begin{array}{rl} x_1 = & x_{11}, x_{1,2}, \ldots, x_{1,k_1} | x_{1k_1+1}, \ldots, x_{1k_2} | x_{1k_2+1}, \ldots, x_{1n} \\ x_2 = & x_{21}, x_{2,2}, \ldots, x_{2,k_1} | x_{2k_1+1}, \ldots, x_{2k_2} | x_{2k_2+1}, \ldots, x_{2n} \\ \hline y_1 = & x_{11}, x_{1,2}, \ldots, x_{1,k_1} | x_{2k_1+1}, \ldots, x_{2k_2} | x_{1k_2+1}, \ldots, x_{1n} \\ y_2 = & x_{21}, x_{2,2}, \ldots, x_{2,k_1} | x_{1k_1+1}, \ldots, x_{1k_2} | x_{2k_2+1}, \ldots, x_{2n} \end{array}$$

Uniform crossover: Each bit in the individual representation is chosen with uniform probability from one of the parent individuals.

Crossover operators for $\mathbb{R}^n$:

k-point crossover: Analogous to k-point crossover in $\mathbb{B}^n$

Uniform crossover: Analogous to uniform crossover in $\mathbb{B}^n$

Arithmetic crossover: An individual $\mathcal{I} \in \mathbb{R}^n$ for the offspring population is created as weighted sum from k parents $x_1, \ldots, x_k$:

$$\mathcal{I} = \sum_{i=1}^{k} \alpha_i x_i; \text{ with } \sum_{i=1}^{k} \alpha_i = 1 \tag{6.1}$$

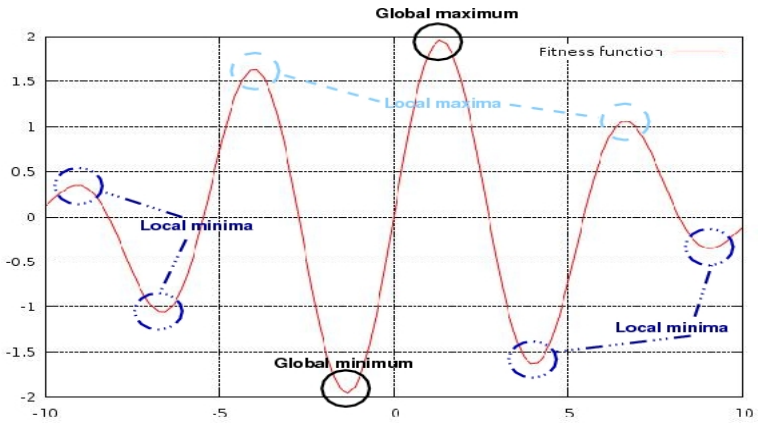


Figure 6.3: Illustration of an exemplary function that is weak multimodal

6.1.5 Fitness function – Weighting of the offspring population

After the variation, all newly generated offspring individuals are weighted by a fitness function f.

The structure of this fitness function also impact the performance of random search approaches applied to the problem. A fitness function with many local optima, for example, may require that the search approach considers (also) search points in greater distance to the current best rated search point in order to avoid that the algorithm converges in a local optimum while the global optimum may still be far away.

Definition 6.1.1 : Optima

Let $f : G \to P$ be a real valued fitness function. $x^* \in G$ is an optimum point of for $\varepsilon > 0$ with $|x - x^*| < \varepsilon$ the inequality $f(x^*) \geq f(x)$ $(f(x^*) \leq f(x))$ holds.

Global optimum *An optimum point x^* is called global optimum, if $f(x^*) \geq f(x)$ $(f(x^*) \leq f(x))$ for all $x \in G$.*

Local optimum *An optimum point which is not globally optimal is called local optimum.*

We distinguish between multimodal and unimodal functions dependent on the count of optima.

Definition 6.1.2 : Multimodality and Unimodality

A function f is called unimodal when only one global optimum exists. Otherwise it is called multimodal.
An unimodal or multimodal function f with no local optima is called strong multimodal (unimodal). Otherwise it is called weak multimodal (unimodal).

Figure 6.3 depicts an exemplary weak multimodal function.

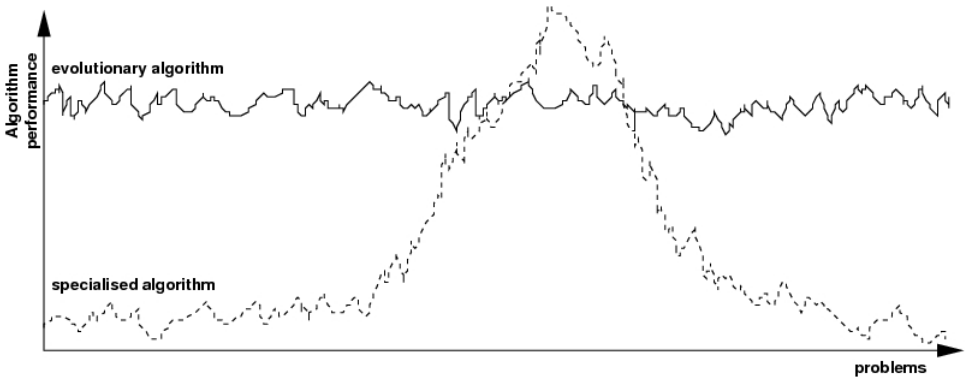


Figure 6.4: Illustration of the assumption that evolutionary algorithms have a better average performance than classical algorithms

6.1.6 Selection for substitution

Since the number of search points has increased due to the variation, we have to reduce the population size to μ again for the next iteration. Commonly, individuals with higher fitness values have a higher probability to stay in the population. Sometimes, also the age of individuals is considered so that older individuals have a higher probability to be replaced. The selection operators are basically the same as for the selection for reproduction.

Regarding the pool from which the individuals are drawn, we distinguish between plus and comma strategies. In $(\mu + \nu)$ strategies, the offspring population is chosen from the μ individuals of the old population 'plus' the ν offspring individuals created in the variation phase.

In (μ, ν) strategies, the μ individuals for the next population are drawn from the ν individuals created in the variation phase while the μ individuals of the old population are discarded.

6.2 Restrictions of evolutionary algorithms

In the early days of evolutionary algorithms, assumptions have been formulated that these algorithms are superior to classical algorithms in the sense that the performance of evolutionary algorithms is higher on average over all possible problems although a specialised algorithm may outperform an evolutionary algorithm on a small set of problems. Figure 6.4 illustrates this intuition.

After some thoughts about this assumption we can easily see that it is a bit absurd or at least not sufficiently concrete. In particular, how can any algorithm be suited for all problems? Clearly, the input format might differ among problems. Can an algorithm suited to solve the travelling salesman-problem also be applied to the task of recognising patterns in image databases or to the task of computing prime factors of an integer?

We see that we have to be more concrete in order to qualify the assumption illustrated in figure 6.4.

We consider algorithms that operate on the search space S which is mapped onto an ordered set W by a fitness function $f : S \to W$ from the set F of all possible fitness functions. We assume that both, S and W are finite. This is a reasonable assumption, as digital computers, which are utilised to host the algorithms can only store finite sets and address a finite number of points.

We also assume that no search point is requested twice. This is a simplifying assumption for technical reasons in favour of randomised search approaches as especially random approaches might request a search point more than once. This means that this assumption does not negatively impact the performance of evolutionary algorithms.

For $f \in F$ and an algorithm A we denote the number of search points requested until an optimum is found by this algorithm with $A(f)$.

Theorem 6.2.3 : No free lunch theorem (NFL)

For two algorithms A and A' the relation $A_{S,W} = A'_{S,W}$ holds.

Proof. We show this assertion by induction over the size of the search space: $s := |S|$. We consider all functions $F_{s,i,N}$ on an search space S with $|S| = s$ and an fitness domain of $\{1, \ldots, N\}$ with at least one $x \in S$ with $f(x) > i$. Initially, we have the situation $F_{s,0,N}$. Observe that, since we are talking about all possible functions, i.e. all possible mappings from S to W , it is not required to consider all search spaces of size $|S| = s$. Since another search space is just another permutation of S on W this would only reorder the functions f. For each function f and each permutation π on S also $f_\pi(s) = f(\pi(s))$ is a member of $F_{s,i,N}$.

We denote the average performance of a search strategy A in $F_{s,i,N}$ with $A_{s,i,N}$ and claim $A_{s,i,N} = A'_{s,i,N}$ for arbitrary search strategies A and A' and $s \in \mathbb{N}, N \in \mathbb{N}, 0 \leq i < N$

Induction start: $s = 1$**.** Each algorithm has to request the only search point and finds the optimum with the first request.

Induction step: $s - 1 \to s$**.** Since for every f and permutations π also f_π belongs to $F_{s,i,N}$, it exists a function $a : S \to \mathbb{N}$ so that the share of functions with $f(x) = j$ is $a(j)$ and thus $a(j)$ is the probability to choose a search point with fitness value j. With probability $a(j)$, every algorithm observes a search point $x \in S$ with $f(x) = j$.

If $j \leq i$, x is not optimal and the new scenario is $F_{s-1,i,N}$. For each $f' \in F_{s-1,i,N}$ with $S' = \{1, \ldots, s-1\}$ exists exactly one function $f \in F_{s,i,N}$ with $f(x)$ in $f'(x)$ for $x \in S'$ and $f(s) = j$. Therefore, all functions from $F_{s-1,i,N}$ are equally probable.

Let $j > i$. Since with f also f_π belongs to $F_{s,i,N}$, the number of functions with $f(x) = j$ and x optimal is equal to the number of functions f' with $f'(x) = j$. Consequently, the probability to find an optimal point is not dependent on the algorithm in this scenario.

□

6.3 Design aspects

In the following sections we discuss some general design aspects for the implementation of evolutionary algorithms.

6.3.1 Search space

For problems that have their origin not in the computer science, the problem has to be represented as a search problem and, in particular, the representation of individuals and the search space are to be designed.

It is often sufficient to represent search points as vectors of constant length for a given set (e.g. $\mathbb{R}, \mathbb{Z}, \mathbb{N}, \mathbb{B}$). The decision taken on the modelling of the search space impacts also the mutation and crossover operators applicable and can therefore impact the optimisation performance.

Generally, it is favourable to choose an individual representation that reflects the problem characteristics well. Otherwise the representation might change the actual problem and possibly increase its complexity. Assume, for example, that we want to encode numbers in their binary representation. The hamming distance between 2^n and $2^n + 1$ is 1 while the hamming distance between 2^n and $2^n - 1$ is $n + 1$! A solution to this problem might be the utilisation of Gray codes. A Gray code is recursively defined. The numbers 0 and 1 are encoded as 0 and 1. When the numbers $0, \ldots, 2^n - 1$ are encoded by $a_0, \ldots, a_{2^n-1}$ we encode $0, \ldots, 2^{n+1} - 1$ as $0a_0, \ldots, 0a_{2^n-1}, 1a_{2^n-1}, \ldots, 1a_0$. Obviously, the distance of neighbouring numbers is 1. However, also greater numbers have a smaller distance: $\text{hammingDist}(0a_0, 1a_0) = 1$.

6.3.2 Selection principles and population structure

Selection is an important method for evolutionary algorithms. Without selection, the choice of search points were unguided and thus merely random. Of course, we intend to keep individuals with good fitness scores but this also contains the danger that the population might degenerate to an isolated region in the search space that possibly constitutes only a local optimum.

To circumvent this problem we can consider further measures in the calculation of the selection probability as, for instance, the distance of search points or the count of points in a defined neighbourhood.

In order to sustain a diverse population that is spread widely across the search space, we thrive to keep isolated individuals with respectable fitness score. On the other hand, we are not so keen to keep many similar individuals with similar fitness values.

In order to achieve this aim we can replace the fitness score $f(x)$ of an individual x by $f'(x) = \frac{f(x)}{d(x,\mathcal{P})}$ where $d(x, \mathcal{P})$ measures the singularity of x in a population $\mathcal{P}$.

In the case that the problem structure is such that multiple global optima exist (multimodal fitness function), a crossover of search points from distinct global optima may lead to a point somewhere in the middle which has a weak fitness value. In such a case it may

be beneficial to restrict the crossover parents to individuals from a singular region in the search space.

6.3.3 Comments on the implementation of evolutionary algorithms

Evolutionary search approaches are very simple schemes that are straightforward to implement. However, evolutionary algorithms are time consuming optimisation methods when compared to specialised optimisation approaches since the optimisation progress is guided by a random process.

Most of this time is spent for the calculation of fitness values. A straightforward approach is therefore to prevent that a fitness value is calculated multiple times by storing already visited search points.

Another aspect is the calculation of random bits. As the calculation of good random bits is expensive and weak random generators can create dependencies between search points, strategies to reduce the number of random bits are required. This is, for instance, possible by calculating the next flipping bit in an individual representation instead of taking a random decision for each bit separately.

6.4 Asymptotic bounds and techniques

To foster the theoretical analysis of evolutionary algorithms, some methods have been derived that are especially well suited for the analysis of evolutionary approaches. Generally, as dimensions and structure of problems evolutionary approaches are applied to are typically too complex for a complete analytical consideration, most analytical methods consider evolutionary algorithms with very restricted population size and offspring population size.

6.4.1 A simple upper bound

In this section we describe the method of the fitness based partition. It is a comparably simple method that is suitable to provide respectable upper bounds on the expected optimisation time of evolutionary algorithms with 'plus'-selection. We describe the method for the $(1+1)$-EA. However, it can be easily adapted for other random search schemes with 'plus'-selection.

Preparatory, we require the notion of a fitness-based partition.

Definition 6.4.4 : Fitness-based partition

Let $f : \mathbb{B}^n \to \mathbb{R}$ be a fitness function. A partition $L_0, L_1, \ldots, L_k \subseteq \mathbb{B}^n$ with $\mathbb{B}^n = L_0 \cup L_1 \cup \cdots \cup L_k$ is a fitness based partition of f when

1. $\forall i, j \in \{0, \ldots, k\}, \forall x \in L_i, y \in L_j : (i < j \Rightarrow f(x) < f(y))$ *and*
2. $L_k = \{x \in \mathbb{B}^n | f(x) = \max\{f(y) | y \in \mathbb{B}^n\}\}$

hold.

Since the considered algorithm utilises plus-selection, the population of the algorithm can follow these partitions only in ascending order until the optimum is reached with L_k. The question is then, how long it takes to leave a partition L_i.

Definition 6.4.5 : Vacation probability

Let $f : \mathbb{B}^n \to \mathbb{R}$ be a fitness function and $L_0, \ldots, L_k$ be a fitness based partition of f. For a standard bit mutation probability of p and $i \in \{0, 1, \ldots, k-1\}$

$$s_i := \min_{x \in L_i} \sum_{j=i+1}^{k} \sum_{y=L_j} p^{H(x,y)}(1-p)^{n-H(x,y)} \tag{6.2}$$

defines the vacation probability of L_i. In this formula, $H(x, y)$ describes the hamming distance from x to y.

We can understand this definition as follows. The term $p^{H(x,y)}(1-p)^{n-H(x,y)}$ describes the probability to mutate from x to y. When we fix x for several y and sum up these probabilities we obtain the probability to mutate from x to one of these y. Since we have, for an $x \in L_i$ summed up the y of all L_j with $i < j$, we obtain the probability to leave L_i. This probability is minimised over all possible x so that we achieve the worst case probability. s_i is a lower bound for the probability to leave L_i with one mutation. Consequently, the expected count of mutations until this happens is bounded from above by s_i^{-1}. This leads to the following result.

Assertion 6.4.1 : A simple Upper bound

Let $f : \mathbb{B}^n \to \mathbb{R}$ be a fitness function and $L_0, \ldots, L_k$ a fitness based partition of f. The expected optimisation time of an $(1+1)$-EA is then bounded from above by

$$E[T_{\mathcal{P}}] \leq \sum_{i=0}^{k-1} s_i^{-1}. \tag{6.3}$$

6.4.2 A simple lower bound

We show a simple general lower bound for evolutionary algorithms that utilise only mutation as variation operator with standard bit mutation and mutation probability $p = \frac{1}{n}$ on functions $f : \mathbb{B}^n \to \mathbb{R}$ with exactly one global optimum.

Assertion 6.4.2 : A simple lower bound

Let $f : \mathbb{B}^n \to \mathbb{R}$ be a function with exactly one global optimum x^* and A an evolutionary algorithm that initialises its population uniformly at random and utilises only standard bit mutation with mutation probability $p = \frac{1}{n}$. The expected optimisation time of this algorithm is then

$$E[T_{\mathcal{P}}] = \Omega(n \log(n)) \tag{6.4}$$

Proof. Let μ be the population size of A. For $\mu = \Omega(n \log(n))$ the algorithm requires already $\Omega(n \log(n))$ evaluations of fitness values for search points prior to finding x^* for the random initialisation of the population with probability $1-2^{-\Omega(n)}$. When $\mu = \mathcal{O}(n \log(n))$, we can see by application of Chernoff bounds that the probability that the hamming distance of a search point x to the optimum x^* is smaller than $\frac{n}{3}$ is

$$P(H(x, x^*) < \frac{n}{3}) = 2^{-\Omega(n)}. \tag{6.5}$$

We can therefore assume that at least $\frac{n}{3}$ bits have to be flipped in order to reach the optimum. The mutation to flip one bit is $p = \frac{1}{n}$. The probability to not flip the bit in t mutations is $(1 - \frac{1}{n})^t \geq e^{-\frac{t}{n-1}}$. With $t = (n-1)\ln(n)$ we obtain $e^{-t(n-1)} = \frac{1}{n}$. The probability that from $\frac{n}{3}$ bits in t mutations at least one not mutates is therefore at least $1 - (1 - \frac{1}{n})^{\frac{n}{3}} \geq 1 - e^{-\frac{1}{3}}$. This leads to

$$E_{T_\mathcal{P}} = (1 - 2^{-\Omega(n)}) \cdot (1 - e^{-\frac{1}{3}}) \cdot (m-1)\ln(n) = \Omega(n \log(n)). \tag{6.6}$$

□

6.4.3 The method of the expected progress

For some problems the optimisation process is similar over the whole optimisation run. In such a case we can utilise the idea that an algorithm does not deviate much from the expectation in most cases.

This can be utilised to derive a lower bound on the optimisation time. The idea is to identify steps that are required for the optimisation and which are to be applied often in order to reach a global optimum. When we then bound the probability to achieve such a step from above, a lower bound can be derived.

When the expected count of these steps can be determined, we can typically bound the probability to obtain many of these steps fast from above. If the steps are also independent from each other, we know the expected number of these steps in a fixed number of generations.

Through Chernoff bounds we show that the probability to deviate from this expected number is very small.

We describe the method for the $(1+1)$-EA. We denote the optimisation problem with $\mathcal{P}$ and assume a progress measure $\mathcal{F} : \mathbb{B}^m \to \mathbb{R}_0^+$ such that $\mathcal{F}(s_t) < \Delta$ represents the case that a global optimum was not found in the first t iterations. Let $T_\mathcal{P}$ denote the count of iterations required by the optimisation algorithm to reach one of the optimum values for the problem $\mathcal{P}$.

For every $t \in \mathbb{N}$ we have

$$\begin{aligned} E[T_\mathcal{P}] &\geq t \cdot P[T_\mathcal{P} > t] \\ &= t \cdot P[\mathcal{F}(s_t) \\ &< \Delta] \\ &= t \cdot (1 - P[\mathcal{F}(s_t) \geq \Delta]) \end{aligned} \tag{6.7}$$

With the help of the Markov-inequality we obtain $P[\mathcal{F}(s_t) \geq \Delta] \leq \frac{E[\mathcal{F}(s_t)]}{\Delta}$ and therefore $E[T_{\mathcal{P}}] \geq t \cdot \left(1 - \frac{E[\mathcal{F}(s_t)]}{\Delta}\right)$ This means that we can obtain a lower bound on the optimisation time by providing the expected progress after t iterations.

7 Cooperative transmission schemes

> *Cooperative transmission is a cross-layer approach bridging the network and physical layers with the objective of forwarding information that is available at multiple terminals more reliably.*
>
> (Birsen Sirkeci-Mergen and Anna Scaglione: Randomised cooperative Transmission in Large-Scale Sensor Networks [112])

Cooperation is one of the major challenges of research on sensor networks. It covers many aspects in sensor networks such as energy consumption, sharing of resources for computation or finding of routing paths. Cooperation can also be utilised to improve aspects of data transmission in wireless networks. These approaches generally make use of spatial diversity of noise or achieve redundancy of transmission.

The utilisation of cooperative diversity in sensor networks has been studied by various research institutes during the past years [113, 114, 7, 115]. These approaches have in common that nodes in a network are utilised as relays [24, 25, 26], as it was first proposed by Cover and El Gamal in [27]. Neighbouring nodes repeat a signal sequence once it was received and achieve – sufficient temporal synchronisation presumed – constructive interference between superimposed transmit signals. The major complexity of these approaches lies in the temporal synchronisation of transmit signals that is commonly preconditioned. In these schemes, the capacity and robustness of a network of sensor nodes [7, 8] as well as the maximum transmission range [6] can be increased. Furthermore, the average energy consumption per node [9] and the time to propagate messages in sensor networks can be reduced [35] by these approaches. Superimposing signals was studied in [24] with regard to the probability of packet loss and in [116] with regard to the Shannon-capacity.

Another approach to obtain a transmission gain is to utilise the spatial diversity of distributed nodes and to combine the RF-transmit signal components of distinct nodes on the channel.

This problem is addressed by the approach of virtual MIMO in wireless sensor networks [21, 20, 22]. In virtual MIMO, single antenna nodes are cooperating to establish a multiple antenna wireless sensor network. Virtual MIMO has capabilities to adjust to different frequencies and is highly energy efficient [23, 11]. However, the implementation of MIMO capabilities in WSNs requires accurate time synchronisation, complex transceiver circuity and signal processing that might surcharge the power consumption and processing capabilities of simple sensor nodes.

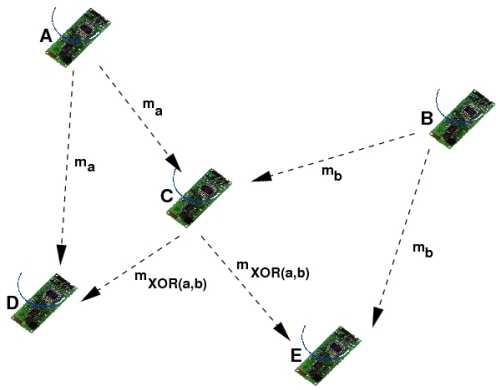


Figure 7.1: An example for network coding

Other solutions proposed are open-loop synchronisation methods as round-trip synchronisation based [36, 37, 38]. In this scheme, the destination sends beacon signals in opposed directions along a multi-hop circle. Beamforming is achieved when the processing time along the multi-hop chain is identical in both directions. This approach, however, does not scale with the size of a network.

Closed-loop feedback approaches include full-feedback techniques, in which carrier synchronisation is achieved in a master-slave manner. The phase-offset between the destination and a source node is corrected by the receiver node. Diversity between transmit signals is achieved over CDMA channels [39]. This approach is applicable only to small network sizes and requires sophisticated processing capabilities at the source nodes.

7.1 Cooperative transmission

In cooperative networks, nodes relay each others messages to provide spatial diversity and to increase the spectral efficiency by combining a received signal at the physical layer.

7.1.1 Network coding

In traditional approaches, relay nodes forward messages they receive unmodified to reach a next hop. Another approach, however, is to allow relay nodes to modify incoming messages before further transmission. The essential idea of network coding is that nodes combine several incoming messages so that the cost for transmission is reduced [117, 118]. Figure 7.1 illustrates a network coding scenario.

Assume that nodes A and B both transmit messages m_a and m_b. Clearly, nodes D and E receive the messages directly from these nodes. Since wireless transmission is inherently a broadcast scheme, also node C overhears both messages. In traditional broadcast schemes, node C would transmit messages m_a and m_b separately so that both nodes D and E received both messages m_a and m_b in the end (one message is even received twice by each of the nodes). In network coding, node C could combine both messages m_a and m_b by, for

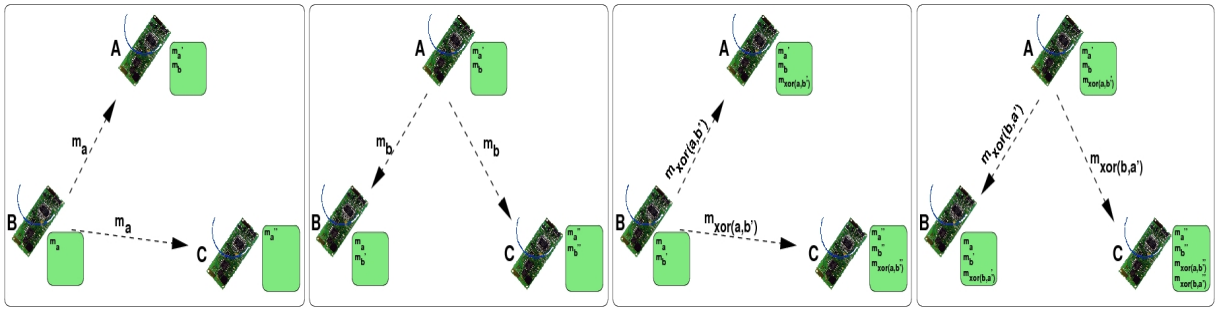


Figure 7.2: Error reduction by network coding

instance, XOR to receive a combined message $m_{XOR(a,b)}$. After receiving $m_{XOR(a,b)}$, both nodes D and E obtain the messing message (m_a or m_b) by computing the logical XOR between the messages they received.

Regarding the overall energy consumption, one transmission is omitted in this network coding example, so that this scheme is more energy efficient compared to the traditional broadcast scheme.

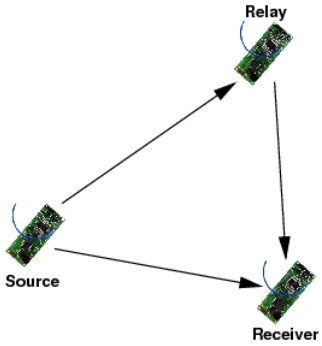
A similar situation is depicted in figure 7.2.

Both nodes A and B transmit a message (m_a, m_b) to a remote node C. Also, after receiving the message from the respective other source node (A or B), both nodes transmit $m_{XOR(a,b')}$ and $m_{XOR(b,a')}$. The notion a' and b' indicates that the message received over the wireless channel might be subject to transmission errors. Now, node C holds $m''_a, m''_b, m''_{XOR(a,b')}, m''_{XOR(b,a')}$. It can consequently obtain three copies of m_a and m_b by combining the received messages. When the probability for bit errors is low, this scheme can be utilised to detect and correct errors. Consider, for example one bit position i for three copies of message m_a:

- $m_a(i)'$
- $XOR(m'_b, m_{XOR(b,a')}(i)'$
- $XOR(m'_b, m_{XOR(a,b')}(i)'$

Assume that one bit is erroneous with probability $\frac{1}{m}$ for all transmissions. Then, $m_a(i)'$, $m_b(i)'$ are incorrect with probability $\frac{1}{m}$ each and $m_{XOR(b,a')}(i)'$, $m_{XOR(a,b')}(i)'$ are incorrect with probability $1 - \left(1 - \frac{1}{m}\right)^2$ each. Consequently, the probability that more than one of the received messages is incorrect at position i is at most $\frac{1}{m} \cdot \left(1 - \left(1 - \frac{1}{m}\right)^2\right)$, which is smaller than $\frac{1}{m}$.

A more detailed discussion that also considers the coverage area, pathloss and fading is found in [119].



	Block 1	Block 2	Block 3	Block 4
Source	$c_1(1, w_1)$	$c_1(w_1, w_2)$	$c_1(w_2, w_3)$	$c_1(w_3, 1)$
Relay	$c_2(1)$	$c_2(w_1)$	$c_2(w_2)$	$c_2(w_3)$

Figure 7.3: A two-hop strategy for multi-hop relaying

7.1.2 Multi-Hop approaches

In the case of multi-hop-relaying based on the physical channel, multi-hop is interpreted as a multi dimensional relay channel. Communication is allowed between all nodes [28]. It was shown that this approach optimally divides the network ressources regarding information theoretic metrics [12]. In larger multi-hop-scenarios it is, however, not well suited since the count of successfully transmitted bits per square meter decreases quadratically with the size of the network [29, 30].

The general idea of this approach is to retransmit received messages by a relay node so that the destination will receive not only the message from the source destination but also from the relay. Figure 7.3 illustrates this scheme for a two-hop strategy.

A message w is divided into B blocks $w_1, \ldots, w_B$. The transmission is performed in $B+1$ blocks using the code words $c_1(w_i, w_j)$ and $c_2(w_i)$. The relay node will always transmit the word w_i recently observed from the source node while the source still transmits both, w_i and w_{i+1}. Although omitting further details, the general idea of this approach is to encode redundancy in $c_1(w_i, w_j)$ and $c_2(w_i)$ that both depend on w_i so that the destination node can reliably decode w_i.

For the destination it can be shown that by sufficiently combining the blocks received from source and relay the data rate is improved [28].

7.1.3 Data flooding

An alternative approach attempts to 'flood' the network with a message to be transmitted [31, 32]. At Cornell University 'Opportunistic Large Arrays' (OLAs) have been proposed for cooperative transmission [120, 121, 112]. A node will retransmit a received message at its reception. It has been shown that the approach outperforms non-cooperative multi-

hop schemes significantly. This approach floods the network with nodes that retransmit a desired transmit signal. Basically, in this scheme, an avalanche of signals is proceeded through the network. When the network is sufficiently dense, the distinct transmit signals are superimposed with special OLA modulations it is then even possible to encode information into this signal wave. This transmission scheme is robust to environmental noise but not capable of coping with moving receivers due to the inherent randomness of the protocol.

It was derived that the average energy consumption of nodes is decreased [9, 10] and the transmission time is reduced compared to traditional transmission protocols in wireless sensor networks [35].

7.2 Multiple antenna techniques for networks of single antenna nodes

Multi antenna systems have been studied for their potential to increase the channel capacity in fading channels [122]. MIMO systems can achieve higher data rates under identical transmit power and bit-error-rate as SISO systems. However, a key requirement for these systems is that the distinct antennas are separated by at least $\frac{\lambda}{2}$. As sensor nodes are typically of very restricted dimensions, this requirement might easily lead to the conclusion that single sensor nodes with multiple antennas are absurd. When, however, single nodes are allowed to cooperate for their transmission so that a set of nodes is treated as a multi-antenna transmitter or receiver, a cooperative MIMO system can be constructed. This approach to cooperative transmission was proposed in [123]. The idea is to create clusters of collaboratively transmitting nodes in a network [12]. Figure 7.4 illustrates this transmission scheme. It is denoted as virtual MIMO or cooperative MIMO.

In [123] the general idea to apply MIMO transmission schemes to a scenario of distributed transmitters and receivers was proposed by using Alamouti diversity codes [122]. The Alamouti code with two transmit antennas proposed in [124]. The extension of the Alamouti code to more than two antennas is discussed in [122]. We will discuss Alamouti codes with two transmit antennas in the following.

The Alamouti diversity scheme for two receivers is depicted in figure 7.5

In the figure, two transmit nodes transmit to a single receive node. At the receiver, the system is constituted by a channel estimator, a combiner and a maximum likelihood detector.

Both transmit nodes will, at a given symbol period t simultaneously transmit two signals. The signal transmitted from node A is denoted as s_0 and the signal transmitted from node B is denoted s_1. In the following symbol period $t+T$, node A transmits signal $(-s_1^*)$ and node B transmits s_0^*.[1] Here, s^* denotes the complex conjugate of s.

[1] In this description, distinct signals are separated in time. Therefore, this scheme is referred to space-time coding. It is, however, also possible to separate signals in frequency by utilising distinct transmit channels. In this case, the coding scheme is denoted as space-frequency coding

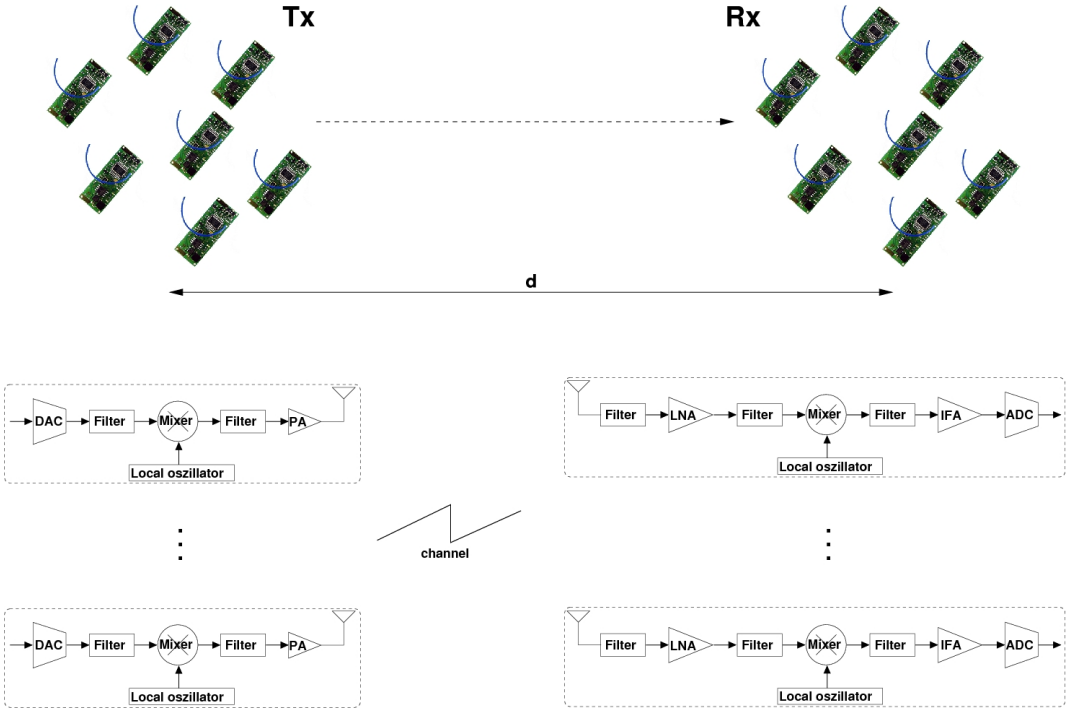


Figure 7.4: Illustration of virtual MIMO in wireless sensor networks

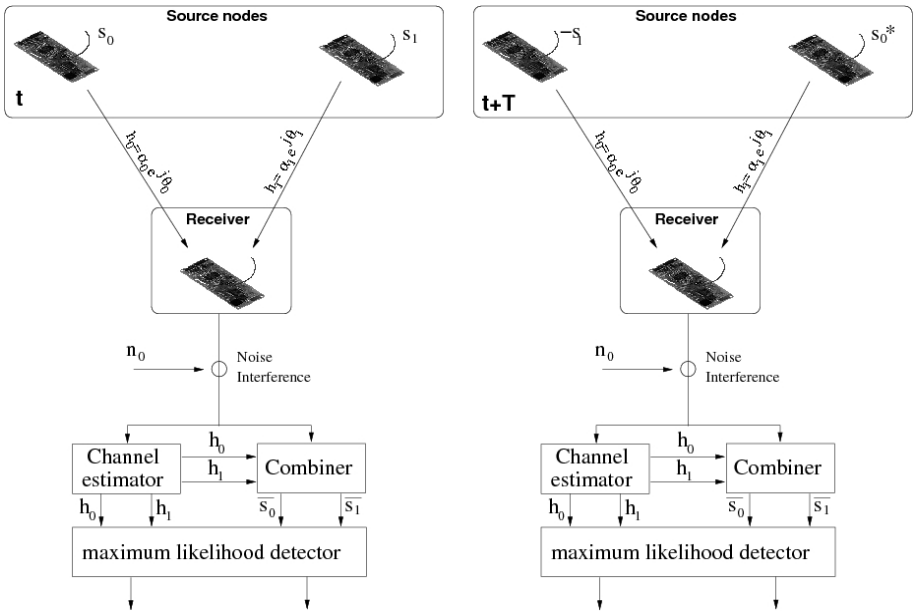


Figure 7.5: Illustration of the Alamouti transmit diversity scheme with two receivers

Assume that the channel at time t is modelled by a complex multiplicative distortion $H_A(t)$ for node A and $H_B(t)$ for node B. when we further assume that fading is constant over one symbol period, we can write

$$\begin{aligned} H_A(t) &= H_A(t+T) = H_A = \alpha_A e^{j\Theta_A} \\ H_B(t) &= H_B(t+T) = H_B = \alpha_B e^{j\Theta_B} \end{aligned} \tag{7.1}$$

The received signals are then expressed as

$$\begin{aligned} r_0 &= r(t) = H_A s_0 + H_B s_1 + n_0 \\ r_1 &= r(t+T) = -H_A s_1^* + H_B s_0^* + n_1 \end{aligned} \tag{7.2}$$

with r_0 and r_1 representing the received signals at t and $t+T$. n_0 and n_1 represent receiver noise and interference.

The combiner module depicted in figure 7.5 creates the signals

$$\begin{aligned} \overline{s_0} &= H_A^* r_0 + H_B r_1^* = (\alpha_A^2 + \alpha_B^2) s_0 + H_A^* n_0 + H_B n_1^* \\ \overline{s_1} &= H_B^* r_0 - H_A r_1^* = (\alpha_A^2 + \alpha_B^2) s_1 - H_A n_1^* + H_B^* n_0 \end{aligned} \tag{7.3}$$

and forward these to the maximum likelihood detector. The maximum likelihood detector uses for s_0 and s_1 the decision rules

- Choose s_i iff

$$\begin{aligned} & (\alpha_0^2 + \alpha_1^2 - 1)|s_i|^2 + d^2(\overline{s_0}, s_i) \\ \leq\ & (\alpha_0^2 + \alpha_1^2 - 1|s_k|^2 + d^2(\overline{s_0}, s_k), \\ & \forall i \neq k \end{aligned} \tag{7.4}$$

- Choose s_i iff

$$d^2(\overline{s_0}, s_i) \leq d^2(\overline{s_0}, s_k), \forall i \neq k \tag{7.5}$$

where $d^2(s_i, s_j)$ is the squared Euclidean distance between signals s_i and s_j:

$$d^2(s_i, s_j) = (s_i - s_j)(s_i^* - s_j^*). \tag{7.6}$$

In virtual MIMO schemes, it is assumed that each node has a preassigned index i and will transmit the transmission sequence that the i-th antenna would transmit in an Alamouti MIMO system. At the receiver side, the receiver nodes join the received sum signals cooperatively.

All nodes in a cluster cooperate when receiving or transmitting data from and to neighbouring nodes. Basically, a cluster is seen as a single multiple antenna device so that MIMO, SIMO, or MISO transmission is possible in wireless sensor networks. By grouping of nodes, the complexity of synchronising nodes in a cluster is reduced. [12, 5] derive results

regarding the energy efficiency and the optimum cluster design. It was shown that this scheme can be more energy efficient than traditional SISO transmission between nodes of a sensor network. The approach allows the utilisation of existing routing algorithms and multi-hop theory when a cluster is understood as minimum entity. The capacity of a sensor network that is organised by this approach is, however, decreased compared to other approaches of cooperative transmission [30, 125].

It was presumed that the local oscillators of all transmit and receive nodes are synchronised. However, this synchronisation is typically not easy to establish among distributed wireless nodes.

Several solutions to this requirement are proposed by distributed transmit beamforming. A key distinguishing feature of distributed transmit beamforming with respect to centralised beamforming is that each source node in a distributed beamformer has an independent local oscillator. These typically multiply the frequency of a crystal oscillator up to a fixed nominal frequency. Carrier frequencies generated in this manner, however, typically exhibit variations in the order of 10-100 parts per million (ppm) with respect to the nominal. If uncorrected, these frequency variations among sources are catastrophic for transmit beamforming since the phases of the signals may drift out of alignment over the duration of the transmission and may even result in destructive combining at the destination. The first goal, therefore, is to synchronise the carrier frequencies for the different sources to minimise or eliminate frequency offset.

One approach to frequency synchronisation is to employ a master-slave architecture [18] where slave source nodes use phase-locked loops (PLLs) to lock their phase to a reference carrier signal broadcast by a master node.

A simple form of a PLL consists of three components:

- A phase detector that compares the phase offset of an input signal $Y(s)$ with the phase offset of a controlled oscillator. It creates an output signal $E(s)$ – the error signal – that is proportional to the phase offset
- A filter that computes with the function $F(s)$ from the error signal $E(s)$ the control signal $C(s)$.
- A variable electronic oscillator. This module might, for instance be implemented by a Voltage Controlled Oscillator (VCO) that might be altered in its frequency by, for instance, a capacity diode. Other possible implementations are Numerically Controlled Oscillators (NCO) that takes numerical inputs to alter the oscillation frequency.

Alternatively, the destination node could broadcast a reference carrier to facilitate frequency synchronisation among the source nodes [36, 37, 39]. A source node that estimates its frequency offset to be Δf can multiply its complex baseband transmitted signal by $e^{-j2\pi\Delta ft}$. When frequency synchronisation is achieved, the phase of the transmission from distinct transmitters must be synchronised to achieve reasonable phase coherence at a receiver.

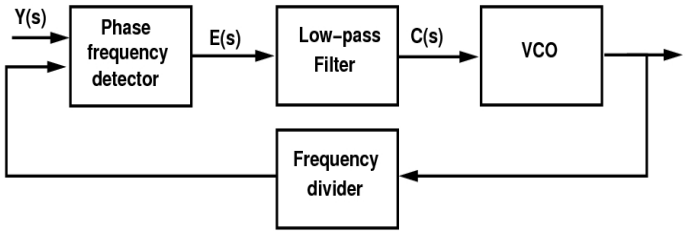


Figure 7.6: Schematic illustration of a phase locked loop

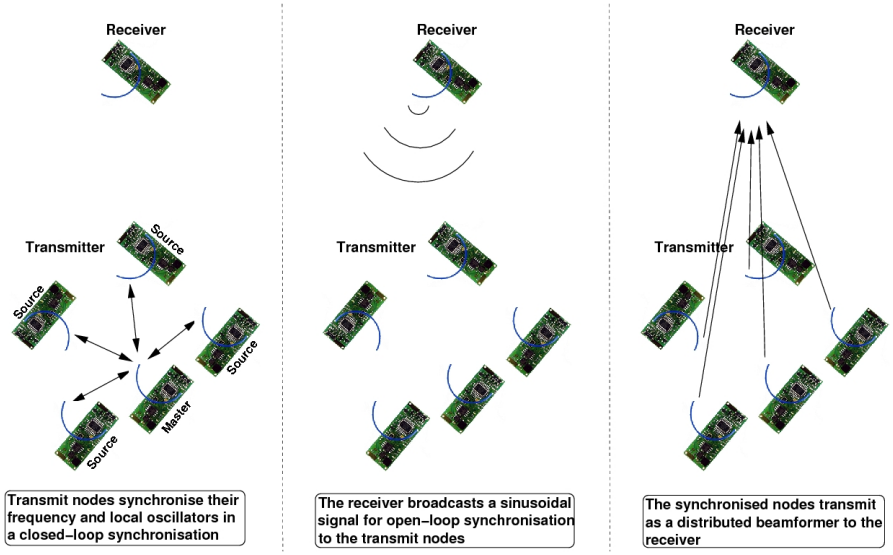


Figure 7.7: Illustration of the master-slave open-loop distributed adaptive carrier synchronisation scheme

In the following sections we detail open-loop and closed-loop distributed carrier synchronisation.

7.2.1 Open-loop distributed carrier synchronisation

In open-loop phase synchronisation schemes, transmit nodes communicate to cooperatively achieve phase synchronisation at a remote receiver. Interaction with the receiver is minimised in these approaches. The receiver may, for instance, transmit a sinusoidal signal. Transmitters utilise this signal together with the signals from other transmitters to achieve sufficient phase synchronisation.

Master-slave open-loop distributed carrier synchronisation

The master-slave open-loop carrier synchronisation approach was described in [18]. The general principle is illustrated in figure 7.7

Initially, among the transmitters one is identified as master node, while the others are

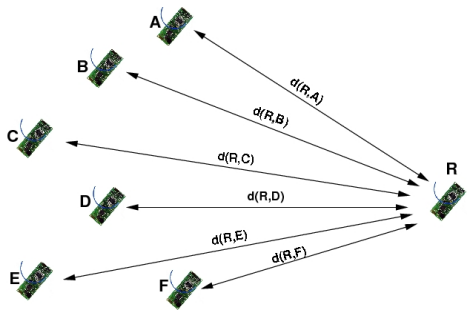


Figure 7.8: Illustration of the open-loop distributed beamforming scenario with known relative node locations

considered slaves. These master and slave nodes in the sensor network communicate to synchronise their frequency and local oscillators. For frequency synchronisation, the master node broadcasts a sinusoidal signal to the slave nodes that in turn estimate and correct their relative frequency offset to this signal. Phase synchronisation is then achieved in a closed-loop method.

In order to achieve beamforming to a receiver node, all transmitters then estimate their channel response to the destination. This can be achieved by the destination broadcasting a sinusoidal signal on the carrier frequency. As transmitters are already synchronised, they can estimate their individual complex channel gain to the destination using their phase and frequency synchronised local oscillators. The transmitters then transmit as a distributed beamformer by applying the complex conjugate of the gains to their transmitted signals.

Carrier synchronisation when locations of distributed nodes are known

An approach to synchronise the carrier phase offsets of a set of distributed nodes was described in [17]. The assumption in this approach is that the distance between the receiver node and the transmit nodes is known and that a Line-of-sight (LOS) scenario is considered. The receiver node serves as a master node broadcasting both, carrier and timing signals. Assume that a slave node i is located at distance $d(i) = d_0(i) + d_e(i)$ from a receiver node. In this formula, $d_0(i)$ denotes the distance between the receiver and node i and $d_e(i)$ is an additional placement error. Figure 7.8 illustrates this scenario.

The receiver node broadcasts a carrier signal $\Re\left(m(t)e^{j(2\pi f_0 t)}\right)$. A transmit node i receives a noisy signal

$$\Re\left(n_i(t)m(t)e^{j(2\pi f_0 t+\gamma_0(i)+\gamma_e(i))}\right) \tag{7.7}$$

where

$$\gamma_0(i) = \frac{2\pi f_0 d_0}{c} = \frac{2\pi d_0}{\lambda_0} \tag{7.8}$$

is the normal phase offset from the transmitted carrier and

$$\gamma_e(i) = \frac{2\pi f_0 d_i(i)}{c} = \frac{2\pi d_i(i)}{\lambda_0} \tag{7.9}$$

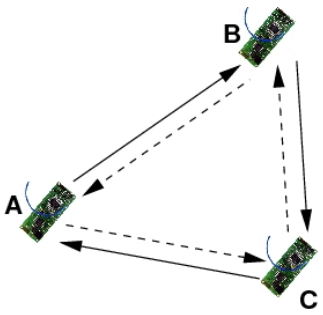


Figure 7.9: Illustration of the Round-trip open-loop distributed adaptive carrier synchronisation scheme

represents the phase error resulting from the placement error $d_e(i)$. $n_i(t)$ is the noise signal. Each transmit node applies a PLL to lock on to the carrier with the result

$$\Re\left(n_i(t)m(t)e^{j(2\pi f_0 t+\gamma_\Delta(i))}\right) \tag{7.10}$$

In this equation, $\gamma_\Delta(i) = \gamma_0(i) + \gamma_e(i) - \gamma_{pll}(i)$. The phase variations across different slaves originate from placement errors and PLL errors. The PLL error can be reduced by increasing the SNR. When the locations of nodes are sufficiently well known and fixed, phase synchronisation among distributed carrier signals is therefore possible with this approach.

Carrier synchronisation when locations of distributed nodes are not known

In a scenario where distances among nodes are not known, the same approach as above can be applied when relative distances among nodes are calculated and the slave clocks are updated accordingly. The authors of [39] describe this approach. In advance of synchronising carrier phase offsets, clock-offsets are estimated by a standard approach [126].

When nodes are moving, this estimation of clock offsets is a constant procedure. However, when the velocity is too high, it is not feasible to synchronise a sufficiently large number of nodes by this approach. Furthermore, the overhead for the time synchronisation is energy demanding.

Round-trip open-loop distributed carrier synchronisation

An approach to open-loop phase synchronisation between two source nodes and one receiver node that allows for high mobility of source and destination nodes was presented in [36, 37, 38]. Figure 7.9 illustrates this scenario. The destination node C will in this system transmit a sinusoidal beacon signal $\Re\left(m(t)e^{j(2\pi f_0 t+\gamma_0)}\right)$ to both source nodes. At the nodes the signals $\Re\left(m(t)e^{j(2\pi f_0 t+\gamma_0^A)}\right)$ and $\Re\left(m(t)e^{j(2\pi f_0 t+\gamma_0^B)}\right)$ are received. Both source nodes employ a primary frequency synthesis PLL [127] tuned onto the beacon frequency f_0 and generate a low-power secondary sinusoidal beacon that is phase locked to the received beacon but of frequency $f_1^A = \frac{N_1^A}{M_1^A} f_0^A$ and $f_1^B = \frac{N_1^B}{M_1^B} f_0^B$. N_1 and M_1 are integers. The

signals $\Re\left(m(t)e^{j(2\pi f_1^A t+\gamma_1^A)}\right)$ and $\Re\left(m(t)e^{j(2\pi f_1^B t+\gamma_1^B)}\right)$ are then transmitted among the source nodes. At receiving a secondary beacon signal, a source node again generates a carrier signal at frequency $f_c^A = \frac{N_1^A}{M_1^A} \cdot f_1^B = f_c^B = \frac{N_1^B}{M_1^B} \cdot f_1^A$ that is phase locked to the secondary beacon signal. These carrier signals are then utilised to transmit to the destination node. The received signal at the destination can be written as

$$\Re\left(m(t)e^{j(2\pi f_c^A t+\gamma_2^A)} + m(t)e^{j(2\pi f_c^B t+\gamma_2^B)}\right). \tag{7.11}$$

Observe that frequencies from both source nodes are identical. Also, when the round trip time along $C \to A \to B \to C$ and $C \to B \to A \to C$ are similar, the phase offset $\gamma_\Delta = \gamma_2^A - \gamma_2^B$ is small at the destination node.

Since the round-trip signal propagation has low delays, this method can be applied also at high velocities of nodes. It is, however, only feasible for exactly two source nodes. The expected maximum gain is therefore limited.

7.2.2 Closed-loop distributed carrier synchronisation

In closed-loop phase synchronisation schemes, the destination directly controls the relative phase offset between RF transmit signal components. This is achieved by measuring a function of the received phases of the source transmissions and then transmitting feedback signals that guide the transmitter's phase adjustment process. Interaction between transmit nodes is minimal in these approaches.

Full feedback closed-loop synchronisation

A full feedback based closed-loop carrier synchronisation technique suitable for distributed beamforming is described in [39]. Carrier frequency synchronisation is achieved over a master-slave approach. The Receiver node acts as the master node. The phase offset between the master and the i-th transmitter is corrected over a closed-loop protocol:

- The master broadcasts a common beacon to all source nodes.
- Source nodes 'bounce' this beacon back to the master utilising a different frequency. Transmission from source nodes to the master nodes is achieved in a direct-sequence code division multiple access (DS-CDMA) scheme in which all source nodes utilise distinct codes. By this approach, the master is then able to distinguish the received signals.
- After receiving the bounced beacons, the master estimates the phase of each source relative to its originally transmitted beacon. These estimates are divided by two, quantised and then transmitted to the source nodes via DS-CDMA. This message is utilised by the source nodes for phase compensation and may also contain clock correction information to facilitate symbol timing synchronisation.

- Source nodes extract their phase compensation estimate, and adjusts their carrier phase.

When phase offsets did not change significantly between synchronisation and beamforming intervals, the bandpass transmissions from all source nodes will then combine coherently. The authors of [39] also considered the impact of energy allocation between synchronisation and information transmission on the error probability of signals transmitted in this distributed beamforming manner. It was derived that an optimal energy tradeoff exists and that allocating too much or too little energy to carrier synchronisation is inefficient.

1-bit feedback based closed loop synchronisation

In [15] an iterative process to synchronise signal phases of transmit signals at a remote receiver without inter-node communication is described. It is assumed that, for a network of size n, initially the carrier phase offsets γ_i of transmit signals $e^{j(2\pi(f+f_i)t+\gamma_i)};\ i \in \{1..n\}$ are arbitrarily distributed. When a receiver requests a transmission from the network, carrier phases are synchronised in an iterative process.

1. Each source node i adjusts its carrier phase offset γ_i and frequency offset f_i randomly
2. The source nodes transmit to the destination simultaneously as a distributed beamformer.
3. The receiver estimates the level of phase synchronisation of the received sum signal (e.g. by SNR or comparison to an expected signal).
4. A feedback on the level of synchronisation is broadcast to the network. Nodes sustain their phase adjustments if the feedback has improved or else reverse them.

These four steps are iterated repeatedly until a stop criteria is met (e.g. maximum iteration count or sufficient synchronisation). Figure 7.10 illustrates this procedure. The computational complexity for transmit nodes in this approach is low as only the phase and frequency of the transmit signal is adapted according to a random process. Further processing is not required for synchronisation.

This process was successfully applied in [13, 14, 40, 42].

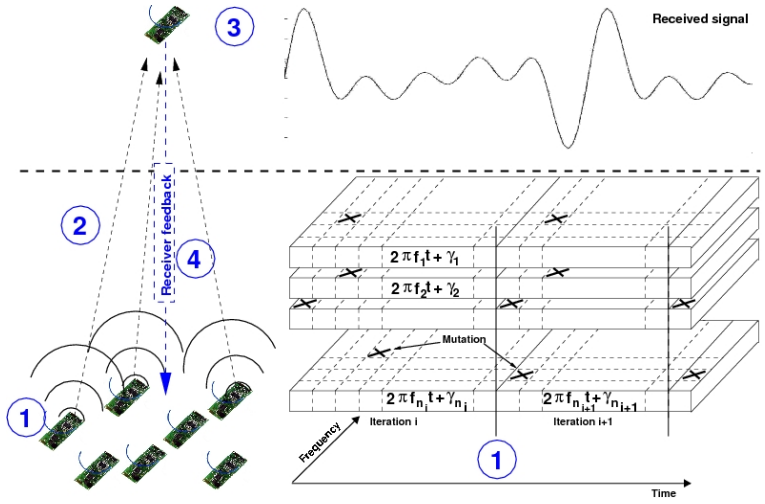


Figure 7.10: An iterative approach to closed-loop distributed adaptive transmit beamforming

8 Analysis of a simple closed loop synchronisation approach

> *[...] distributed adaptive beamforming is on the cusp of feasibility. The prototypes reported in the literature thus far have focused on demonstrating that the critical task of aligning carrier phases at the intended destination is feasible. The next step is to investigate and demonstrate distributed Beamforming in a networked context.*
>
> (Raghuraman Mudumbai et al: Distributed Transmit Beamforming: Challenges and Recent Progress [15])

As detailed in section 7.2.2, the 1-bit feedback based closed loop synchronisation is probably the slowest approach to synchronise carrier phases from the methods presented. However, the computational load for a single node is also by far the smallest among all methods. Beside applying a random decision on the carrier phase offset of single nodes, in each iteration only an additional binary decision is taken whether to keep the carrier phase based on the feedback received from a master node.

The process has been studied by different authors [41, 42, 43, 13]. The distinct approaches proposed differ in the implementation of the first and the fourth step specified above. The authors of [43] show that it is possible to reduce the number of transmitting nodes in a random process and still achieve sufficient synchronisation among all nodes.

In [41, 42, 43] a process is described in which each node alters its carrier phase offset γ_i according to a normal distribution with small variance in each iteration. In [13] a uniform distribution is utilised instead but the probability for one node to alter the phase offest of it's carrier signal is low. Only in [42] not only the phase but also frequency is adapted.

Significant differences among these approaches also apply to the feedback and the reactions of nodes in the iterative process. In [15, 42, 43] a one-bit feedback is utilised. Nodes sustain their phase modifications when the feedback has improved and otherwise reverse them. In [43] it was shown that the optimisation time is improved by factor two when a node as response to a negative feedback from the receiver applies a complementary phase offset instead of simply reversing its modification. In [13], authors suppose to utilise more than one bit as feedback so that parameters of the optimisation can be adapted with regard to the optimisation progress.

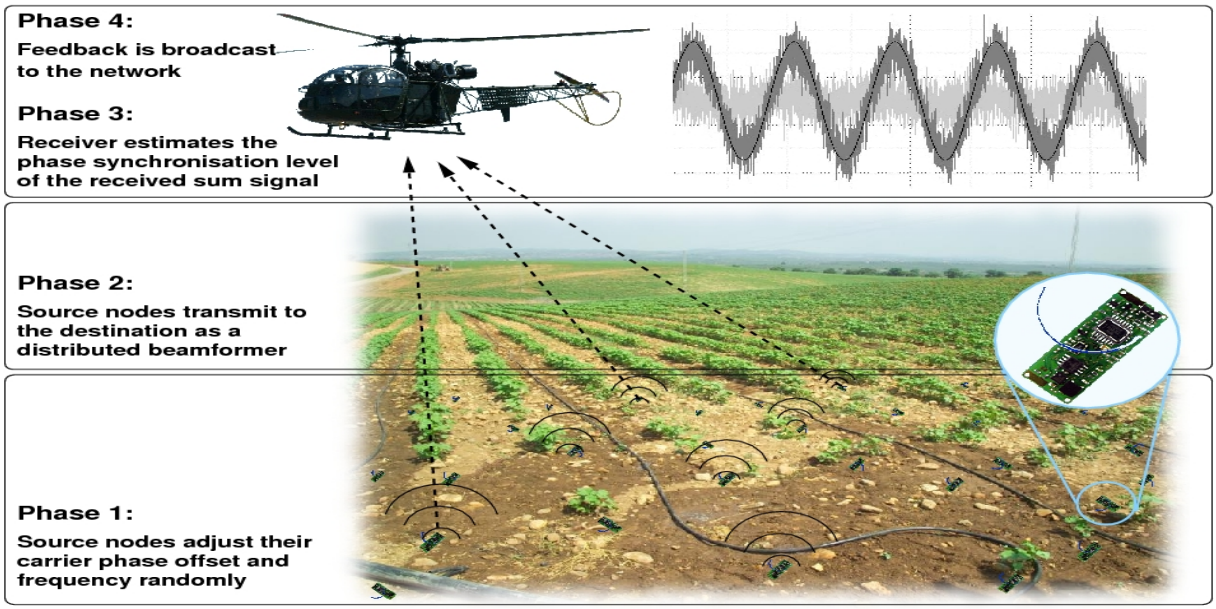


Figure 8.1: A schematic overview on feedback based closed-loop distributed adaptive transmit beamforming in wireless sensor networks

The strength of feedback based closed-loop distributed adaptive beamforming in wireless sensor networks is its simplicity and low processing requirements that make it feasible for the application in networks of tiny sized, low power and computationally restricted sensor nodes.

The first step to problem analysis is always a very precise understanding of the problem itself. A reasonable general understanding of the optimisation process was already gained by the definition of the iterated process detailed in section 7.2.2. We will in this chapter describe this process algorithmically exact to become able to provide some quantitative results on the optimisation performance as well es impacts of parameter and problem settings. This discussion will detail design decisions as, for example, the random phase alteration method or the feedback function. After describing these aspects in a mathematical sense, we are able to provide and possibly even prove sound results on the optimisation scenario.

8.1 Analysis of the problem scenario

In this section we discuss algorithmic aspects of the 1-bit feedback based closed loop phase synchronisation approach for distributed adaptive transmit beamforming in wireless sensor networks. Figure 8.1 illustrates a general scenario for this approach.

As a first observation, note that the iterative optimisation approach basically represents an evolutionary random search method. By altering the phase offset of carrier signals, new search points are requested. When the fitness value feedback by the remote receiver is decreasing, the modifications of one iteration are discarded.

Generally, this is the behaviour of a $(1 + 1)$-evolutionary algorithm. Naturally, the population size and the offspring population size are 1, since at one time instant, each node can only transmit a carrier signal at one distinct carrier phase offset (and not multiple phase offsets at once). Although also greater population sizes may be modelled by subsuming several iterations, the standard optimisation approach proposed can be modelled by a standard $(1 + 1)$-EA. A suitable algorithmic representation of the optimisation problem is discussed in the following.

8.1.1 Representation of individuals

The optimisation process described consists of several iterations. During each iteration (e.g. after step 1) the phases and frequencies of all transmit signals are fixed while they might change from one iteration to another. We therefore model the iteration-wide fixed phase and frequency configurations of all distinct transmit signals as one individual in the sense of the optimisation process. In other words, an individual in the optimisation process is the set of all phase and frequency modifications (γ_i, f_i) of the base band signal for all source nodes $i \in \{1, \ldots, n\}$.

How shall we represent an individual for the analytic consideration in the following sections? Possible representations are

- As ordered set of phase and frequency pairs (γ_i, f_i). A drawback of this solution is that similarity measures between individuals are not straightforward so that an intuitive understanding of a neighbourhood of individuals is not provided.

- A vector $V = v_1, \ldots, v_{2n}$ of phases and/or frequencies (e.g. $v_{2i-1} = \gamma_i; v_{2i} = f_i$). This representation has the favourable property that we can then represent configurations as points in multi dimensional search spaces so that simple distance (similarity) measures between configurations are straightforward (e.g. Euclidean distance). Therefore, a neighbourhood measure for the configuration is implicitly given.

- Binary representations in which possible phase and/or frequency shifts are encoded for each source node and source node sub-configurations are concatenated to one global binary representation of individuals (Figure 8.2 illustrates this representation scheme exemplary). Basically, all possible phase shift values for a single source node are enumerated and represented by a binary value. Although the number of possible phase shifts is infinite, any practical sensor node implementation can only realise a finite number of possible phase shifts. We therefore assume that this abstraction from an infinite number of possibilities does represent the problem scenario well.

  This solution is favourable since various results to binary search spaces are known and can be applied. In order to respect neighbourhood relationships, similar phase offsets are to be mapped onto similar binary representations. One problem with a binary configuration is the neighbourhood function. The hamming distance between neighbouring points (with similar phase shift for a single source node) might be

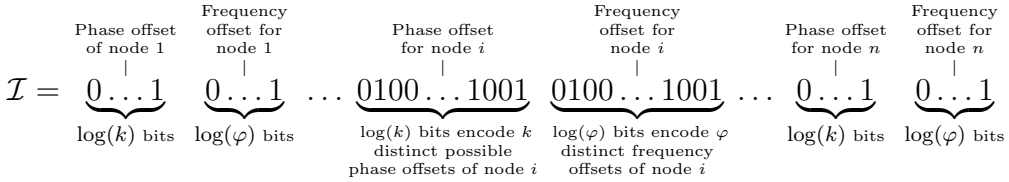


Figure 8.2: Possible binary representation of individuals

very large when possible phase offsets are simply enumerated. We can overcome his problem by using Gray codes (cf. section 8.1.3). A drawback of Gray Codes is that numbers that are very far apart also have hamming distance 1.

For our analytic consideration, Grey Codes will be sufficient. We therefore assume in the following that individuals are encoded by Grey Codes as described above.

8.1.2 Feedback function

At receiving the superimposed sum signal

$$\zeta_{\text{sum}} = \Re\left(m(t)e^{j2\pi f_c t}\sum_{i=1}^{n}\text{RSS}_i e^{j(\gamma_i+\phi_i+\psi_i)}\right) \tag{8.1}$$

the receiver estimates the quality of the synchronisation. A possible, simple implementation for this calculation is to utilise the signal power or SNR of the superimposed signal ζ_{sum} at the receiver. Figure 8.3 illustrates the superimposition of received signals components $\Re\left(m(t)e^{j(2\pi f_i t+\gamma_i+\phi_i+\psi_i)}\right)$ and $\Re\left(m(t)e^{j(2\pi f_{\bar{i}} t+\gamma_{\bar{i}}+\phi_{\bar{i}}+\psi_{\bar{i}})}\right)$ from two nodes i and $\bar{i}$. For ease of presentation, the phase offset due to the signal propagation, ϕ_i and the phase offset due to distinct phases of the local oscillator ψ_i are omitted in the figure. At the receiver, both signals combine to a new, overlayed signal

$$\zeta_{i,\bar{i}} = \Re\left(m(t)e^{j2\pi f_c t}\sum_{\varsigma\in[i,\bar{i}]}\text{RSS}_i e^{j(\gamma_\varsigma+\phi_\varsigma+\psi_\varsigma)}\right) \tag{8.2}$$

After the quality of the synchronisation is calculated at the receiver, a feedback is broadcast to the sensor nodes to guide the optimisation process. The authors of [40, 41] propose to utilise a one bit feedback. A '0' in this feedback scheme means that the synchronisation quality has deteriorated while a feedback of '1' means that the quality is at least as good as in the last iteration. The advantage of this approach is that the amount of data transmitted between network and receiver is minimised. This efficiency can be 'invested' into higher redundancy schemes for the transmission in order to guard the information from transmission errors. A drawback of this solution is, however, that the expected progress of the optimisation is not known at the source nodes, although it might be known at the receiver node. The receiver might, for instance, require a minimum SNR so that the distance

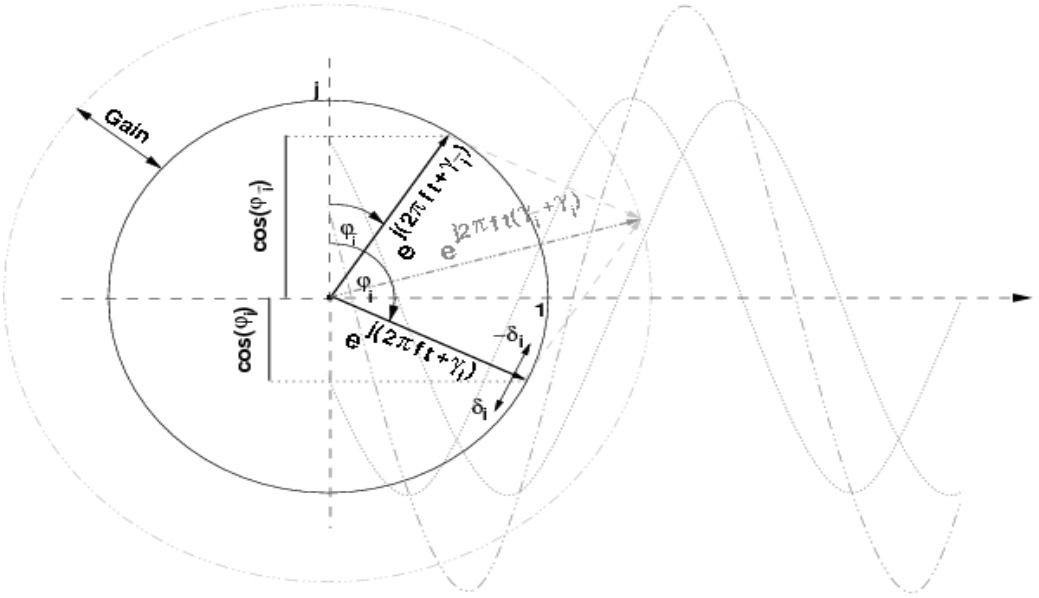


Figure 8.3: Superimposition of received signal components from nodes i and $\bar{i}$.

to the currently reached SNR represents the progress of the approach. When only a binary feedback is provided, the transmit nodes have restricted means to estimate the progress of the optimisation process. When the progress is not known, only less advanced optimisation schemes are possible. In section 8.3.3 we discuss a synchronisation scheme that greatly improves the synchronisation performance but requires more than binary feedback information.

Another approach not feasible with binary feedback values is an adaptive random search operator. The idea is that early in the optimisation process the reached search point is not well synchronised with high probability. In this case, many search points have a higher fitness value than the current one. In particular, we expect the search points in the neighbourhood of the current one to differ only slightly so that it is probable to reach a search point with improved fitness value by applying greater changes to the current search point. When, however, later in the optimisation process the optimum is nearly reached, most other search points have a fitness value worse than the current one and one global optimum is expected to be near, it is more beneficial to apply only slight modifications to the search points (i.e. to search in its direct proximity).

In [40, 41, 42] this was achieved by changing the standard deviation of the Gaussian process to alter the signal phases over the course of the optimisation. Since a progress was not reported, in these implementations the modification scheme was purely time based. With progress information, more ambiguous optimisation schemes are possible.

One solution to this problem is to transmit the (possibly normalised) output of the quality function to the network e.g. as integer value. When maximum and minimum values are known by the sensor nodes, an estimation on the optimisation progress and

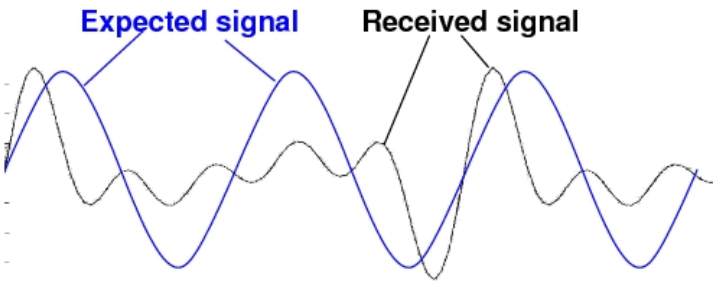


Figure 8.4: Schematic illustration of a calculated expected signal and a received superimposed sum signal.

therefore a more exact control of the optimisation process is possible.

In order to calculate the synchronisation quality of the received signals at a receiver, various approaches are feasible. Apart from very ambiguous (and computationally expensive) methods, less accurate and computationally cheaper methods are detailed in the following.

SNR

One possible approach is to calculate the SNR of the received signal. The SNR specifies the strength of the received sum signal above thermal noise. When the SNR increases, the transmit signals are probably better synchronised than before so that the amount of constructive vs. destructive interference is increased.

A higher SNR is therefore interpreted as an improved synchronisation of transmit signals.

Simple distance between received signals

Another possible solution is to estimate an expected sum signal and to calculate the difference (basically the surface) between the received superimposed sum signal and the (possibly phase shifted) expected signal (see figure 8.4).

When the surface becomes smaller, this is interpreted as an improved phase synchronisation. In order to estimate the expected signal when all signals are perfectly synchronised we would like to specify the signal strength of each single signal component, the frequency and the transmit sequence.

Regarding the transmit sequence utilised for synchronisation purposes we assume that this is predefined among all nodes and known by the receiver. The same is assumed for the transmit power of each single node. Although the receiver is not able to communicate with distinct nodes separately, it is possible to estimate a median distance to the sensor network over the round trip time between the network and the receiver.

Together with the transmit power of distinct nodes, this directly leads to an estimated signal strength of a received and optimally synchronised superimposed sum signal. The missing value, is now the count of transmitting nodes. In [128] an approach was detailed by which the count of simultaneously transmitting sensor nodes can be estimated from signal

statistics of the received signal. Basically, the absolute energy received from simultaneously transmitting unsynchronised nodes is related to the number of nodes transmitting.

8.1.3 Search space

The performance of the optimisation algorithm is heavily dependent on the underlying search space. In particular, we would like to know if the search space is unimodal or multimodal, i.e. if local optima exist. In this case, the optimisation approach could converge in a local optimum. We easily see that the feedback function is not unimodal. The reason is that, given the search point corresponding to an optimum sum signal ζ_{opt} we can state another optimum by adding the same phase offset γ' to all carrier signals. In particular, the fitness function is weak multimodal which means that no local optimum exists. We detail this observation in the following.

Identical transmit frequencies

When carrier frequencies among nodes are identical, i.e.

$$e^{j(2\pi ft+\gamma_i)}; \forall i \in \{1, \ldots, n\} \tag{8.3}$$

a local optimum exists if we can identify at least one search point $s_{\overline{\zeta}}$ for which all small phase modifications decrease the fitness value, while some larger modifications increase it. The smallest possible modification is realised when the transmit phase is altered for exactly one carrier signal ζ_i. Figure 8.5 illustrates that the fitness of a signal is given by the distance between the rotation angles φ_{opt} and φ_i of an optimal configuration $s_{\zeta_{\text{opt}}}$ and s_{ζ_i} as

$$\left|\cos(\varphi_{\text{opt}}) - \cos(\varphi_i)\right|. \tag{8.4}$$

A phase shift of $\delta_i \neq 0$ alters the fitness value. For some t the fitness increases while for others it decreases. W.l.o.g. assume $(\varphi_i + \delta_i) - \varphi_{\text{opt}} < 180°$ and $\varphi_i > \varphi_{\text{opt}}$. As depicted in figure 8.5, for $\varphi_i > 180° \wedge \varphi_{\text{opt}} < 180°$ (or $\varphi_i > 360° \wedge \varphi_{\text{opt}} < 360°$, respectively) the contribution to the fitness function $\mathcal{F}$ is zero overall, while in all other cases a phase offset of δ_i will have either always positive or always negative impact on the fitness value.

Compared to s_{opt} no configuration short of the optimum configuration $s_i = s_{\text{opt}}$ exists for which the distance between signal components is increased for phase offset δ_i regardless of the sign of δ_i (cf. figure 8.5).

Distinct transmit frequencies

Although no local optima exist when all transmit frequencies are identical, this might differ when frequencies are also subject to change. For this configuration it is sufficient to consider the phase offset between two signals only: An individual carrier signal ζ_i that is to be modified in order to improve the amplitude of the sum signal ζ_{sum} and the nearest global optimum ζ_{opt} to the superimposed sum signal.

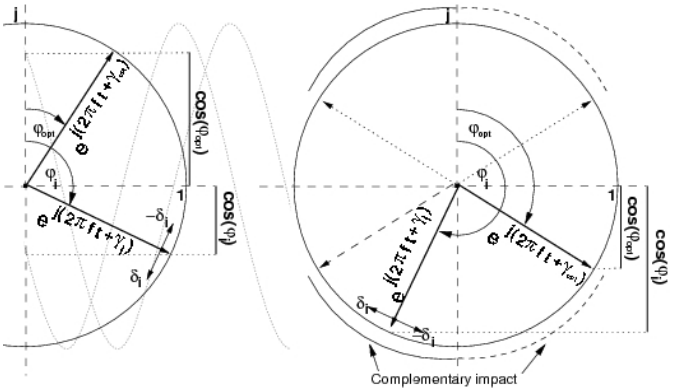


Figure 8.5: Fitness calculation of signal components. The fitness of the superimposed sum signal is impacted by the relative phase offset of an optimally aligned signal and a carrier signal i.

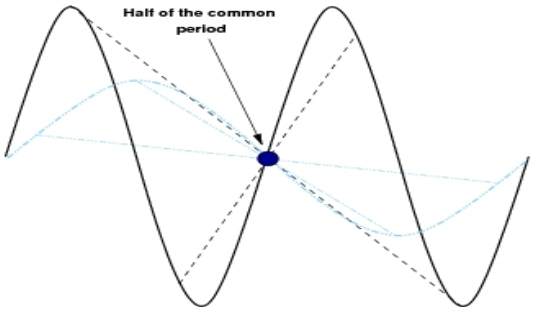


Figure 8.6: Pattern of periodic signals.

When signal frequencies differ, the feedback function is not affected by phase modifications only. The reason for this is that we can for every positive contribution to the fitness function also find a negative contribution of the same amount in the common period of ζ_{opt} and ζ_i. The periodic function of the second half of the common period Φ is a reflection of the first half of the function at the point $(\Phi, 0)$. This property is illustrated in figure 8.6.

Consequently, for any time t relative to the common period Φ, when the relative distance between a point of the individual signal $\zeta_i(t \mod \Phi)$ and the optimum signal $\zeta_{\text{opt}}(t \mod \Phi)$ is decreased by an arbitrary phase offset γ_i' applied to the carrier phase of ζ_i, we can state a value t' for at which the distance is increased by the same amount at $\zeta_i(t' \mod \Phi)$. This means that, in between these two configurations, the paths of the vectors are symmetrical so that for each $t \mod \Phi$ a corresponding $t' \mod \Phi$ exists with

$$\begin{aligned} & e^{j(2\pi(f_1)t \mod \Phi+\gamma_1)} - e^{j(2\pi ft \mod \Phi)} \\ = & -\left(e^{j(2\pi(f_1)t' \mod \Phi+\gamma_1)} - e^{j(2\pi ft' \mod \Phi)}\right) \end{aligned} \tag{8.5}$$

When the phase of one singal is now shifted, this property is not impacted. Also, when

the frequency is changed, the length of the overall period is impacted but within the period the property described remains unchanged. The important points to note here are that

1. signal quality is not affected by phase adaptations when frequencies are unsynchronised

2. without frequency synchronisation, phase synchronisation alone is useless in order to improve the signal quality

In summary, in both cases no local but several global optima exist in the search space so that local random search approaches will always converge a global optimum.

8.1.4 Variation operators

Generally, for revolutionary algorithms, mutation and crossover can be applied in order to proceed to new search points in the search space. We discuss both in the following.

Mutation

Mutation of individuals shall apply small modifications on the individual in general so that a target individual with small distance to the current individual is more probable than individuals that are farther away.

For feedback based phase adaptation of nodes for distributed adaptive beamforming in wireless sensor networks, mutation constitutes phase modifications of one or more carrier signal components. Design parameters are therefore the number of altered carrier signal components and the alteration method. An example for a possible mutation operator is detailed in [41].

Example 8.1.1 :

> *The authors apply normal distributed phase modifications and propose to restrict the standard deviation in order to receive a local search module. The probability that a signal phase is adapted by a source node was chosen as* 1*. This means that each node alters the phase of its carrier signal in each iteration. Later in the optimisation this might lead to a small convergence speed and, when the optimum is near, many transmit signal components already have a nearly optimal phase offset and need not be modified.*
> *It was proposed in [41] to adapt the mutation speed purely by changing the standard deviation of the phase modification and not by the probability to apply phase modifications of single transmit signals.*

Figure 8.18 illustrates the impact of the standard deviation of the normal distributed phase modification on the synchronisation performance. These results have been achieved in a Matlab-based simulation environment in which 100 nodes are randomly distributed on a $30m \times 30m$ square area. The receiver node is located $30m$ above the center of this area

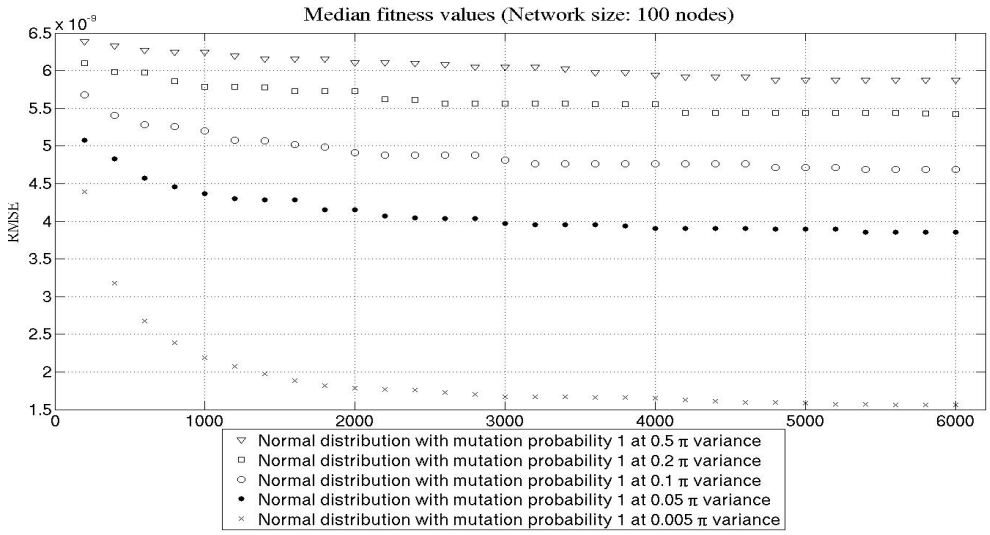
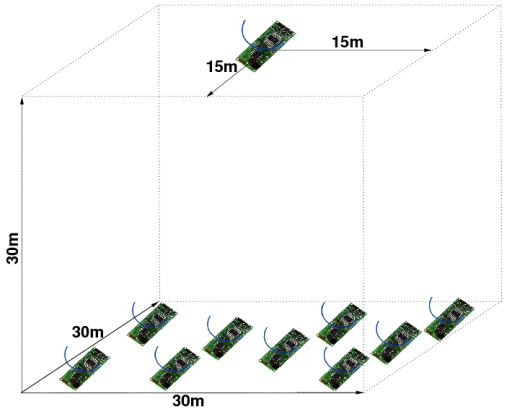


Figure 8.7: Uniform distribution of phase mutations



Property	Value
Node distribution area	$30m \times 30m$
Location of the receiver	$(15m, 15m, 30m)$
Mobility	stationary nodes
Base band frequency	$f_{base} = 2.4$ GHz
Transmission power of nodes	$P_{tx} = 1$ mW
Gain of the transmit antenna	$G_{tx} = 0$ dB
Gain of the receive antenna	$G_{rx} = 0$ dB
Iterations per simulations	6000
Identical simulation runs	10
Random noise power [94]	-103 dBm
Pathloss calculation (P_{rx})	$P_{tx} \left(\frac{\lambda}{2\pi d}\right)^2 G_{tx} G_{rx}$

Figure 8.8: Configuration of the simulation environment. P_{rx} is the the received signal power, d is the distance between transmitter and receiver and λ is the wavelength of the signal

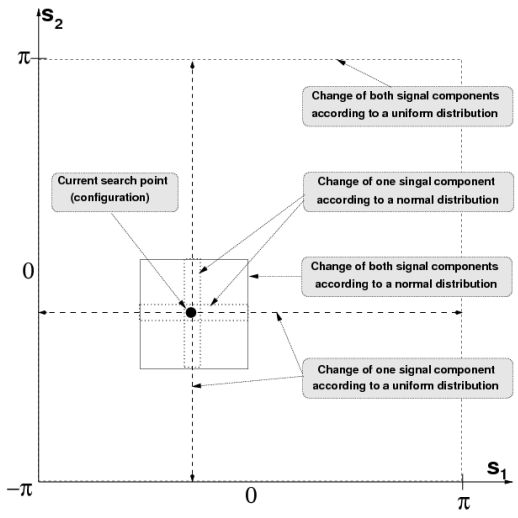


Figure 8.9: A point in the search space (configuration of transmit nodes) spanned by the phase offsets of the carrier signals s_1 and s_2

(see figure 8.1) For each one configuration, 10 simulation runs are completed to obtain the median values depicted in the figure. In the figure, the Root of the mean square error

$$RMSE = \sqrt{\sum_{t=0}^{\tau} \frac{\left(\zeta_{\text{sum}} + \zeta_{\text{noise}} - \zeta_{\text{opt}}\right)^2}{n}}. \tag{8.6}$$

between the optimum signal ζ_{opt} and the received superimposed sum signal ζ_{sum} is illustrated. In this formula, τ is chosen to cover several signal periods.

We can observe that better synchronisation performances are achieved with small variances (i.e. greater probability to proceed to similar individuals) for the normal distributed phase alteration operation. This property is further discussed in section 8.2.4.

Of course, also a uniformly distributed phase offset is possible for the mutation operator. For the uniform distributed operator we can only modify the number of nodes altering the phase offset of their carrier phase in each iteration but not the deviation of the distribution. The uniform distribution leads to a global random search approach since all points in the search space have a positive probability to be elected as next search point.

Since the uniform distribution is better suited for our individual representation, we will assume a uniformly distributed phase alteration procedure in our theoretical analysis in section 8.2. In section 8.2.4 we will observe that the performance of uniform and normal distributed phase modification operators is similar.

For both distributions it is of course possible to restrict the neighbourhood for the search by applying sharp neighbourhood boundaries. This approach leads to a local random search mechanism and is detailed in section 8.3.2. Figure 8.9 illustrates the effect of various distributions for the phase offset on the neighbourhood size.

Crossover

Until now, we considered the basic scenario to be solved by a $(1 + 1)$-EA. This approach is straightforward in that it considered only one individual at a time. As an individual is constituted by one configuration (phase- and frequency changes applied to the carrier signal of transmit nodes) of all signal components, multiple individuals are achieved artificially at most. Also, when only mutation is applied, a multi-start strategy seems to be equally beneficial as an increased population size. With population size 1, however, no recombination (crossover) is possible.

In general, two approaches might enable crossover in collaborative transmission scenarios.

1. Simultaneous transmission of distinct transmit signals by source nodes

2. Time-shifted transmission of transmit signals related to distinct individuals

The first solution, however, would at least require more ambiguous source nodes, and can therefore be disregarded in wireless sensor networks. For the second solution and population size of $\mu > 1$, one iteration of the algorithm is then extended to μ transmission periods each of which is associated to one individual. This scenario has not yet been considered in the literature. We will in section 8.4.3 discuss this approach for optimisation purposes.

8.1.5 Discussion

Does collaborative transmission always converge? In [41] it was proved that this is the case when all transmit signals are of identical frequency. We have discussed the problem scenario thoroughly regarding several aspects so that we can easily come to the same result and even disregard the frequency constraint.

As observed in section 8.1.3 that no local optima exist in the search space. Also, the random uniform and normal distributed search methods are not restricted to any local neighbourhood in the search space (although more similar points have greater probability to be reached). Furthermore, the algorithm does never accept search points with a worse fitness value. These results together are already sufficient to see that one of the global optima is reached with probability 1. Even if the mutation operator were a local search operator, an optimum would be reached with probability 1 since the fitness function is only weak multimodal, i.e. no local optima exist.

In the following section we will derive the expected time until a global optimum is reached for this phase synchronisation approach.

8.2 Analysis of the convergence time of 1-bit feedback based distributed adaptive transmit beamforming in wireless sensor networks

We analyse the process of distributed adaptive beamforming in wireless sensor networks as described above and assume that each one of the n nodes in step 1) decides with probability $\frac{1}{n}$ to change the phase of its carrier signal uniformly at random in the interval $[0, 2\pi]$. On obtaining the feedback of the receiver, nodes that recently updated the phase of their carrier signal either sustain this decision or reverse it, depending on whether the feedback has improved or not. A feedback function $\mathcal{F} : \zeta^*_{\text{sum}} \rightarrow \mathbb{R}$ maps the superimposed received carrier signal

$$\zeta_{\text{sum}} = \Re\left(m(t)e^{j2\pi f_c t}\sum_{i=1}^{n}\text{RSS}_i e^{j(\gamma_i+\phi_i+\psi_i)}\right) \tag{8.7}$$

to a real-valued fitness score. A possible feedback function that is proportional to the distance between an observed superimposed carrier ζ_{sum} and an optimum sum carrier signal

$$\zeta_{\text{opt}} = \Re\left(m(t)\text{RSS}_{\text{opt}}e^{j(2\pi f_c t+\gamma_{\text{opt}}+\phi_{\text{opt}}+\psi_{\text{opt}})}\right) \tag{8.8}$$

is $\mathcal{F}(\zeta_{\text{sum}}) = \int_{t=0}^{2\pi}\left|\zeta_{\text{sum}} - \zeta_{\text{opt}}\right|$. Since this function can be mapped onto other fitness measures as, for instance, the signal to noise ratio (SNR) or the received signal strength (RSS), the following discussion remains valid for these fitness measures. While the multimodality of this fitness function is straightforward, observe that it is also weak multimodal so that no local optima exist (The weak multimodality of the fitness function is derived in section 8.1.3).

Distributed adaptive beamforming in wireless sensor networks is a search problem. The search space $\mathcal{S}$ is given by the set of possible combinations of phase and frequency offsets γ_i and f_i for all n carrier signals. A global optimum is a configuration of individual carrier phases that result in identical phase and frequency offset of all received direct signal components. For the analysis, we assume that the optimisation aim is to achieve for an arbitrary k a maximum relative phase offset of $\frac{2\pi}{k}$ between any two carrier signals. We therefore divide the phase space for a single carrier signal into k intervals of width $\frac{2\pi}{k}$.

For a specific superimposed carrier signal ζ at a receiver we model the corresponding search point $s_\zeta = (\Gamma_t, F_t)_\zeta \in \mathcal{S}$ at iteration t by a specific combination of phase and frequency offsets with $\Gamma_t = (\gamma_{t,1}, \ldots, \gamma_{t,n})$ and $F_t = (f_{t,1}, \ldots, f_{t,n})$. In order to respect neighbourhood similarities we represent search points as Gray encoded binary strings $s_\zeta \in \mathbb{B}^{n\cdot\log(k)}$ so that similar search points have a small hamming distance [129]. A search point is then composed from n sections of $\log(k)$ bits each. Every block of length $\log(k)$ describes one of the k intervals for the phase offset of one carrier signal. For the analysis, we assume that the frequency offset f_i is zero for all carriers. The optimisation problem is denoted as $\mathcal{P}$ and $T_\mathcal{P}$ describes the count of iterations required to reach one of the optimum values for the problem $\mathcal{P}$.

8.2.1 An upper bound on the expected optimisation time

The value of the fitness function increases with the number of carrier signals ζ_i that share the same interval for their phase offset γ_i at the receiver. We can roughly divide the values of the fitness function into $n \cdot \log(k)$ partitions $L_1, \ldots, L_{n \cdot \log(k)}$ depending on the number of bits in the individual representation that are identical with an individual global optimum. For each one transmitter, the probability to adapt its phase to one specific interval is $\frac{1}{k}$. The probability to increase the fitness value so that at least the next partition is reached is then at least

$$\frac{1}{k} \cdot \frac{1}{n \cdot \log(k)} \tag{8.9}$$

since one carrier signal ζ_i is altered with probability $\frac{1}{n \cdot \log(k)}$ and the probability to reach any particular of the partitions that would increase the fitness value is $\frac{1}{k}$. In partition i, a total of

$$\left(\begin{array}{c} n \cdot \log(k) - i \\ 1 \end{array} \right) = n \cdot \log(k) - i \tag{8.10}$$

carrier signals each suffice to improve the fitness value with probability $\frac{1}{n \cdot \log(k)} \cdot \frac{1}{k}$. We therefore require that at least one of the not synchronised carrier signals is correctly altered in phase while all other $n - 1$ signals remain unchanged. This happens with probability

$$\begin{aligned} & \left(\begin{array}{c} n \cdot \log(k) - i \\ 1 \end{array} \right) \cdot \frac{1}{n \cdot \log(k)} \cdot \left(1 - \frac{1}{n \cdot \log(k)}\right)^{n \cdot \log(k) - 1} \\ = \quad & \left(\frac{n-i}{n \cdot k}\right) \cdot \left(1 - \frac{1}{n}\right)^{n-1}. \end{aligned} \tag{8.11}$$

Since

$$\left(1 - \frac{1}{n}\right)^{n} < \frac{1}{e} < \left(1 - \frac{1}{n}\right)^{n-1} \tag{8.12}$$

We obtain the probability $P[L_i]$ that L_i is left and a partition j with $j > i$ is reached as

$$P[L_i] \geq \frac{n \cdot \log(k) - i}{n \cdot \log(k) \cdot e}. \tag{8.13}$$

The expected number of iterations to change the layer is bounded from above by $P[L_i]^{-1}$. We consequently obtain the overall expected optimisation time as

$$\begin{aligned} E[T_{\mathcal{P}}] \quad &\leq \quad \sum_{i=0}^{n \cdot \log(k) - 1} \frac{e \cdot n \cdot \log(k)}{n \cdot \log(k) - i} \\ &= \quad e \cdot n \cdot \log(k) \cdot \sum_{i=1}^{n \cdot \log(k)} \frac{1}{i} \\ &< \quad e \cdot n \cdot \log(k) \cdot (\ln(n \cdot \log(k)) + 1) \\ &= \quad \mathcal{O}\left(n \cdot \log\left(n \cdot \log(k) + k\right)\right). \end{aligned} \tag{8.14}$$

8.2.2 A lower bound on the expected optimisation time

After the initialisation, the phases of the carrier signals are identically and independently distributed. Consequently for a superimposed received sum signal ζ, each bit in the binary string s_ζ that represents the corresponding search point has an equal probability to be 1 or 0. The probability to start from a search point s_ζ with hamming distance $h(s_{\text{opt}}, s_\zeta)$ not larger than $l \in \mathbb{N}$; $l \ll n \cdot \log(k)$ to one of the global optima s_{opt} directly after the random initialisation is at most

$$\begin{aligned} P[h(s_{\text{opt}}, s_\zeta) \leq l] &= \sum_{i=0}^{l} \left(\begin{array}{c} n \cdot \log(k) \\ n \cdot \log(k) - i \end{array} \right) \cdot \frac{k}{2^{n \cdot \log(k) - i}} \\ &\leq \frac{(n \cdot \log(k))^{l+2}}{2^{n \cdot \log(k) - l}} \end{aligned}$$

In this formula,

$$\left(\begin{array}{c} n \cdot \log(k) \\ n \cdot \log(k) - i \end{array} \right) \tag{8.15}$$

is the count of possible configurations with i bit errors to a given global optimum, $\frac{1}{2^{n \cdot \log(k) - i}}$ represents the probability for all these bits to be correct and k is the count of global optima.

This means that with high probability (w.h.p.) the hamming distance to the nearest global optimum is at least l. We will use the method of the expected progress to calculate a lower bound on the optimisation time. The general idea is the following.

Let (s_ζ, t) denote the situation that search point s_ζ was achieved after t iterations of the algorithm. We assume a progress measure $\Lambda : \mathbb{B}^{n \cdot \log(k)} \to \mathbb{R}_0^+$ such that $\Lambda(s_\zeta, t) < \Delta$ represents the case that a global optimum was not found in the first t iterations. For every $t \in \mathbb{N}$ we have

$$\begin{aligned} E[T_\mathcal{P}] &\geq t \cdot P[T_\mathcal{P} > t] \\ &= t \cdot P[\Lambda(s_\zeta, t) < \Delta] \\ &= t \cdot (1 - P[\Lambda(s_\zeta, t) \geq \Delta]). \end{aligned} \tag{8.16}$$

With the help of the Markov-inequality we obtain

$$P[\Lambda(s_\zeta, t) \geq \Delta] \leq \frac{E[\Lambda(s_\zeta, t)]}{\Delta} \tag{8.17}$$

and therefore

$$E[T_\mathcal{P}] \geq t \cdot \left(1 - \frac{E[\Lambda(s_\zeta, t)]}{\Delta}\right). \tag{8.18}$$

This means that we can obtain a lower bound on the optimisation time by providing the expected progress after t iterations. The probability for l bits to correctly flip is at most

$$\begin{aligned} & \left(1 - \frac{1}{n \cdot \log(k)}\right)^{n \cdot \log(k) - l} \cdot \left(\frac{1}{n \cdot \log(k)}\right)^{l} \\ \leq \; & \frac{1}{(n \cdot \log(k))^{l}}. \end{aligned} \tag{8.19}$$

In this formula, $\left(1-\frac{1}{n\cdot\log(k)}\right)^{n\cdot\log(k)-l}$ describes the probability that all 'correct' bits do not flip while the remaining l bits mutate with probability $\left(\frac{1}{n\cdot\log(k)}\right)^{l}$. The expected progress in one iteration is therefore

$$\begin{aligned} E[\Lambda(s_\zeta,t),\Lambda(s_{\zeta'},t+1)] &\leq \sum_{i=1}^{l}\frac{i}{(n\cdot\log(k))^i} \\ &< \frac{2}{n\cdot\log(k)} \end{aligned} \tag{8.20}$$

and the expected progress in t iterations is consequently not greater than $\frac{2t}{n\cdot\log(k)}$.

When we choose $t=\frac{n\cdot\log(k)\cdot\Delta}{4}-1$, the double of the expected progress is still smaller than Δ. With the Markov inequality we can show that this progress is not achieved with probability $\frac{1}{2}$. Altogether we conclude that the expected optimisation time is bounded from below by

$$\begin{aligned} E[T_{\mathcal{P}}] &\geq t\cdot\left(1-\frac{E[\Lambda(s_\zeta,t)]}{\Delta}\right) \\ &\geq \frac{n\cdot\log(k)\cdot\Delta}{4}\cdot\left(1-\frac{\frac{2\cdot n\cdot\log(k)}{4\cdot n\cdot\log(k)}\cdot\Delta}{\Delta}\right) \\ &= \Omega(n\cdot\log(k)\cdot\Delta) \end{aligned} \tag{8.21}$$

With $\Delta=\log(n\cdot\log(k))$ we obtain a lower bound in the same order as the upper bound derived in section 8.2.1 and consequently an asymptotically sharp bound of

$$E[T_{\mathcal{P}}]=\Theta\left(n\cdot\log\left(n\cdot\log\left(k\right)+k\right)\right). \tag{8.22}$$

8.2.3 Simulation and experimental results for the basic scenario

In the analytic consideration in section 8.2 we could derive asymptotic bounds on the expected synchronisation time of a (1+1) evolutionary algorithm with standard bit mutation and mutation probability $\frac{1}{n}$ in the scenario of distributed adaptive transmit beamforming. In the analysis we had to abstract from several physical properties. These are, for instance, the noise of the receiver, multipath propagation as well as distinct transmit and receiver powers of nodes or phase and frequency instability of local oscillators.

The following sections detail the performance of a $(1+1)$ evolutionary algorithm for distributed adaptive beamforming achieved in simulations and experimental settings.

Simulation results

We have implemented the scenario of distributed adaptive beamforming in Matlab to obtain simulation results for greater scale sensor networks. In these simulations, 100 transmit

Property	Value
Node distribution area	$30m \times 30m$
Location of the receiver	$(15m, 15m, 30m)$
Mobility	stationary nodes
Base band frequency	$f_{base} = 2.4$ GHz
Transmission power of nodes	$P_{tx} = 1$ mW
Gain of the transmit antenna	$G_{tx} = 0$ dB
Gain of the receive antenna	$G_{rx} = 0$ dB
Iterations per simulations	6000
Identical simulation runs	10
Random noise power [94]	-103 dBm
Pathloss calculation (P_{rx})	$P_{tx} \left(\frac{\lambda}{2\pi d}\right)^2 G_{tx} G_{rx}$

Table 8.1: Configuration of the simulations. P_{rx} is the the received signal power, d is the distance between transmitter and receiver and λ is the wavelength of the signal

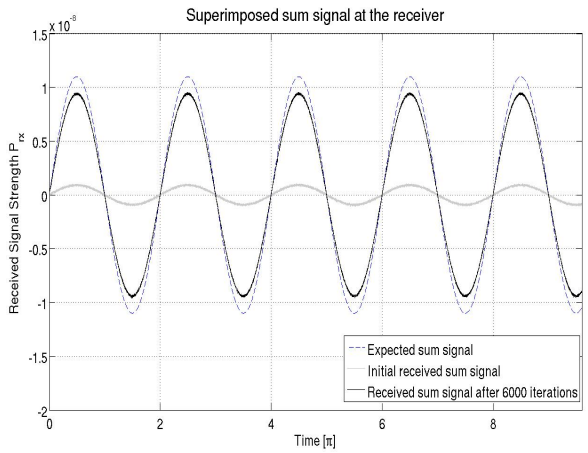
nodes are placed uniformly at random on a $30m \times 30m$ square area. The receiver is located $30m$ above the centre of this area. Receiver and transmit nodes are stationary. Simulation parameters are summarised in Table 8.1.

Frequency and phase stability are considered perfect. We derived the median and standard deviation from 10 simulation runs. One iteration consists of the nodes transmitting, feedback computation, feedback transmission and feedback interpretation at the network side. It is possible to perform these steps within few singal periods so that the time consumed for a synchronisation of 6000 iterations is in the order of milliseconds for a base band signal frequency of 2.4 GHz. Signal quality is measured by the RMSE of the received signal to an expected optimum signal.

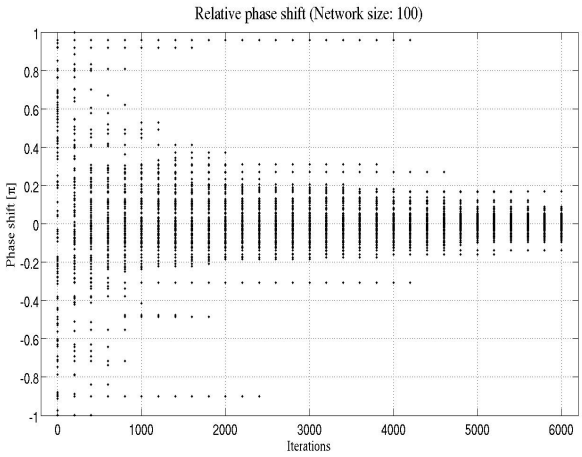
$$RMSE = \sqrt{\sum_{t=0}^{\tau} \frac{\left(\sum_{i=1}^{n} s_i + s_{noise} - s_{opt}\right)^2}{n}} \tag{8.23}$$

In this formula, τ was chosen to cover several signal periods. The terms $s_i = RSS_i \Re\left(m(t) e^{j(2\pi(f+f_i)t+\overline{\gamma_i})}\right)$ and $s_{opt} = nRSS_{sum} \Re\left(m(t) e^{j(2\pi f_{opt} t+\gamma_{opt})}\right)$ represent the i-th signal component of the received sum signal and the expected optimum signal, respectively. The optimum signal is calculated as the perfectly aligned and properly phase shifted received sum signal from all transmit sources. For the optimum signal, noise is disregarded.

Figure 8.10(a) depicts the optimum carrier signal, the initial received sum signal and the synchronised carrier after 6000 iterations when carrier phases are altered with probability $\frac{1}{n}$ in each iteration according to a uniform distribution. In figure 8.10(b), the phase offset of received signal components for an exemplary simulation run with the same parameters are illustrated. We observe that after 6000 iterations about 98% of all carrier signals converge to a relative phase offset of about +/- 0.1π. The median of all variances of the phase offsets for simulation runs with this configuration is 0.2301 after 6000 iterations.



(a) Received sum signal from 100 transmit nodes without synchronisation and after 6000 iterations



(b) Evolution of the phase adaptation process

Figure 8.10: Simulation results for a simulation with 100 transmit over 6000 iterations of the random optimisation approach to distributed adaptive beamforming in wireless sensor networks.

Experimental results

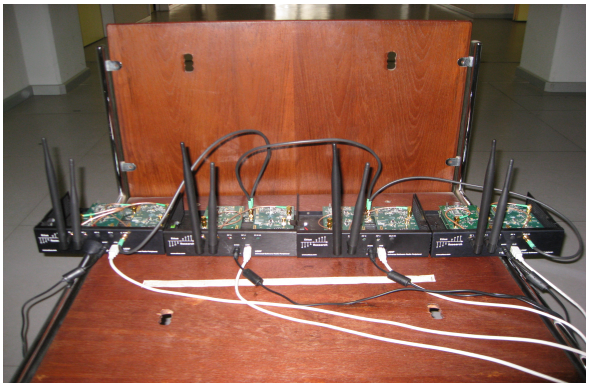
We have utilised USRP software radios (http://www.ettus.com) to model a sensor network capable of distributed adaptive transmit beamforming. Figure 8.11(b) depicts an experimental setting for this scenario. The software radios are controlled via the GNU radio framework (http://gnuradio.org). GnuRadio is a software to control the USRP software radios from a personal computer. It enables processing, analysis and visualisation of RF-signals from standard hardware (cf. figure 8.12(a)) The signal flow graph can be assembled graphically (cf. figure 8.12(b)).

The transmitter and receiver modules implement the feedback based distributed adaptive beamforming. For the superimposed transmit channel and the feedback channel we utilised widely separated frequencies so that the feedback could not impact the synchronisation performance. Two experiments have been conducted with low and high RF transmit frequencies of 27MHz and 2.4GHz, respectively. Table 8.2.3 summarises the configuration and results of both experiments.

After 10 experiments at an RF transmit frequency of 27MHz we achieved a median gain in the received signal strength of 3.72dB for three independent transmit nodes after 200 iterations. In 14 experiments with 4 independent nodes that transmit at 2.4GHz the achieved median gain of the received RF sum signal was 2.19dB after 500 iterations. In figure 8.13(a) the synchronisation performance is depicted over 600 iterations of the synchronisation algorithm.

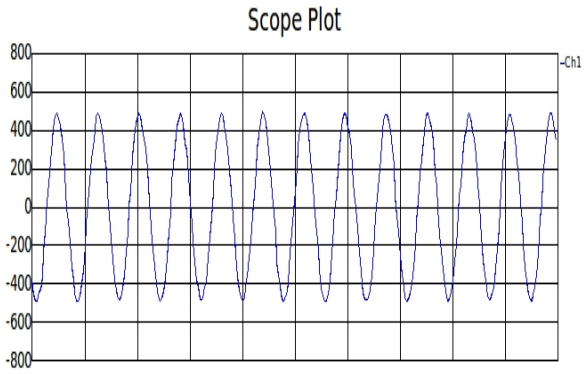
While the amplitude achieved in these synchronisations differs for various positions of the receiver nodes relative to the transmit nodes, figure 8.13(b) shows that the general synchronisation run is similar in all cases.

(a) A single USRP software radio with two transceiver boards

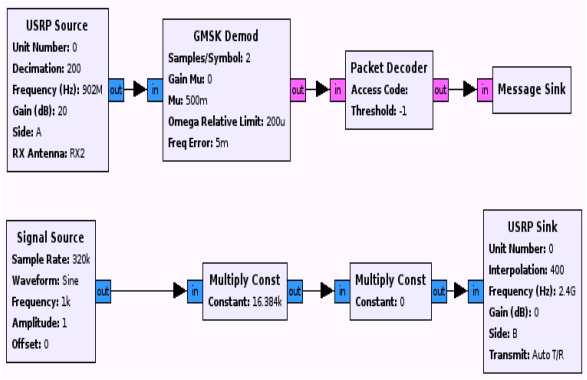


(b) Experimental setting with four clock synchronised USRP software radios

Figure 8.11: An experimental setting for a distributed adaptive beamforming scenario with USRP software radios



(a) Scope plot of Gnuradio while synchronising four USRP software radios at 2.4 GHz

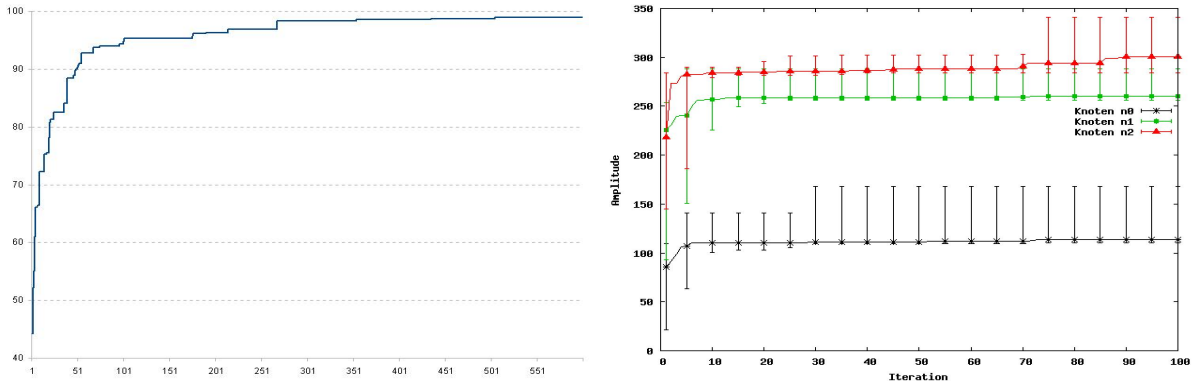


(b) A GnuRadio flow graph

Figure 8.12: GnuRadio is utilised to control the USRP software radios

	Experiment 1	Experiment 2
Sender	4	3
Mobility	stationary	stationary
Distance to receiver [m]	≈ 0.75	≈ 4
Separation of TX antennas [m]	≈ 0.21	≈ 0.3
Transmit RF Frequency [MHz]	$f_{TX} = 2400$	$f_{TX} = 27$
Receive RF Frequency [MHz]	$f_{RX} = 902$	$f_{RX} = 902$
Gain of receive antenna [dBi]	$G_{RX} = 3$	$G_{RX} = 3$
Gain of transmit antenna [dBi]	$G_{TX} = 3$	$G_{TX} = 1.5$
Iterations per experiment	500	200
Identical experiments	14	10
Median gain (P_{RX}) [dB]	2.19	3.72

Table 8.2: Experimental results of software radio instrumentations



(a) Experimental measurement of distributed adaptive transmit beamforming by USRP software radios

(b) Optimisation performance for nodes at distinct relative locations to a receiver

Figure 8.13: Performance of distributed adaptive beamforming in an experimental setting with USRP software radios

The transmit nodes have been synchronised in their clock but no other communication or synchronisation between nodes was allowed. The clock synchronisation is necessary to ensure that the frequency of transmit nodes is identical. Otherwise, as derived in section 8.1.3 a phase synchronisation is not feasible since signals would quickly run out of phase. In future implementations GPS for the clock synchronisation of nodes could be utilised.

8.2.4 Impact of distinct parameter configurations

In distributed adaptive beamforming in wireless sensor networks an individual node i in a network of size n cooperates with other nodes to reach a distant receiver by superimposing its individual carrier signal $\Re\left(m(t)e^{j(2\pi(f+f_i)t+\gamma_i)}\right)$ with carrier signals transmitted from other nodes. The approach constitutes a closed-loop feedback technique that is attained in an iterative process. During each iteration, following a random distribution, each transmit node i adds random phase and frequency offsets γ_i', f_i' to its carrier phase γ_i and frequency f_i. According to the random distribution it is possible that all, some or no node adjusts the phase or frequency of its carrier signal. The nodes then transmit to the destination simultaneously. From this received superimposed sum signal $\Re\left(RSS_{sum}m(t)\sum_{i=1}^{n}e^{j(2\pi(f+f_i)t+\overline{\gamma_i})}\right)$ the receiver estimates the level of phase synchronisation achieved (e.g. by SNR or comparison to an expected signal). At the end of each iteration, the receiver broadcasts this estimation as a channel quality indicator to the network. Nodes then sustain their (adapted) carrier phases $\gamma_i + \gamma_i'$ and frequencies $f_i + f_i'$ when the feedback has improved to the feedback obtained during the last iteration or else reverse them.

These steps are repeated until a stop criteria is met. Possible stop criteria are, for example, the maximum iteration count or sufficient quality of the estimated channel.

The exact implementation of the phase alteration process in each iteration impacts the performance of the optimisation approach. Generally, two parameters are of importance. One is the number of nodes altering their carrier phase and frequency in each iteration and the other is the probability distribution applied on the alteration of phases and frequencies of one carrier signal.

When we illustrate the search space of the algorithm with respect to these parameters we observe that they effectively impact the neighbourhood size of the search approach (see figure 8.14). When the probability to alter the carrier phase of a single node is low, few carrier signals are altered in one iteration. In the figure this might translate to the situation that the phase offset of only one carrier signal is altered. The new search point then differs from the recent one in only one dimension. When a uniform distribution is implemented, all points along this dimension are equally probable while for a normal distribution the most probable points are near to the recent search point as specified by the variance of the probability distribution utilised to alter phase and frequency offsets. When, however, the mutation probability is increased, so that more often several carrier signals (in the figure, for example, both carrier signals s_1 and s_2) are altered, the new search point is drawn from a region in the search space. In the case of the uniform distribution the whole search space might constitute this region. Consequently, a smaller probability for nodes to alter their

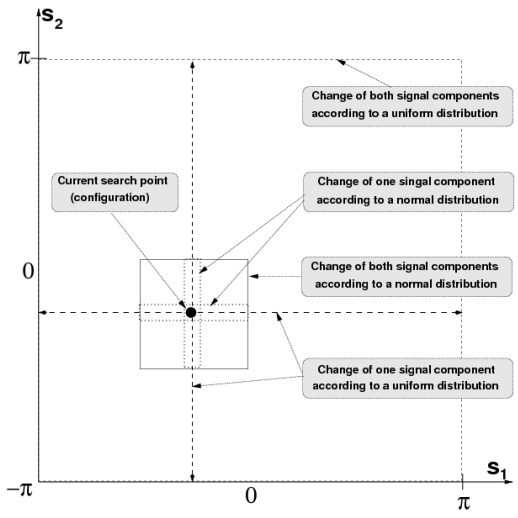


Figure 8.14: A point in the search space (configuration of transmit nodes) spanned by the phase offsets of the carrier signals s_1 and s_2

phase and frequency or a smaller variance for the random process to alter these parameters increases the probability to draw a search point that is near to the recent one.

We expect that a moderate size of the neighbourhood (i.e. probability to alter phase or frequency of one carrier signal and variance for the random process to alter these values) is beneficial at the start of the synchronisation. The initial search point or the initial synchronisation of carrier phases is typically far away from the optimum. Therefore, many search points have a higher fitness value (are better synchronised) than the initial one. Consequently, since the probability to improve the fitness value is high, it might be beneficial to increase the neighbourhood size so that also bigger improvements are possible. Later in the optimisation process, however, when the optimum is near, most search points reduce the fitness value so that the probability to increase it is best for search points drawn from a small neighbourhood.

In this section we consider various different parameter configurations for feedback based distributed adaptive transmit beamforming in wireless sensor networks. These are the probability for individual nodes to alter the phase offset as well as the distribution with which the phase is altered. We distinguish between a uniformly distributed and a normal distributed offset. Both are considered in the following.

Uniformly distributed phase offset

In the first set of simulations we consider the impact of the mutation probability on the synchronisation performance when new transmit phase offsets are drawn according to a uniform distribution.

When the average number of carrier signals that are modified in one iteration is altered, this affects the performance of the optimisation approach, as the stepwidth of the random process is changed. We are interested in the optimum percentage of nodes that alter their

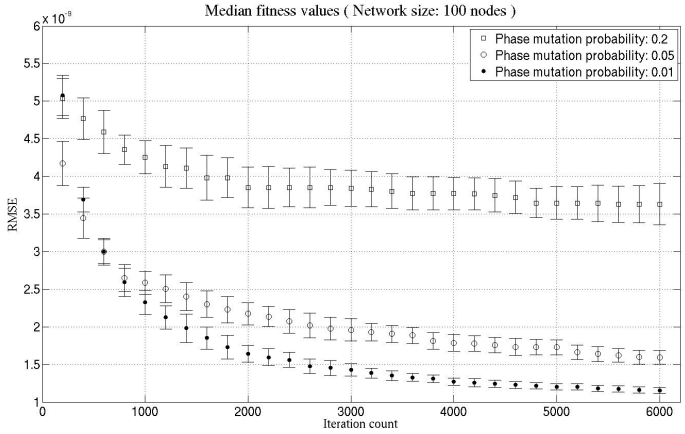


Figure 8.15: Performance of distributed adaptive beamforming in a wireless sensor network of 100 nodes and normal and uniform probability distributions on the phase mutation probability. Uniform distribution of phase mutations.

phase per iteration (mutation probability) when the phase-alteration for each mutating node is drawn according to a uniform distribution. Figure 8.2.4 depicts simulation results for several mutation probabilities.

A mutation probability of κ represents the case that on average $\kappa \cdot n$ out of n transmit signal components mutate (randomly alter their phase offset) in one iteration.

We observe that with a low mutation probability the overall performance of the synchronisation process is improved. At the start of the optimisation, however, higher mutation probabilities are more beneficial. This accounts for the fact that initially many configurations exist that would improve the fitness value as the initial point is not well synchronised with high probability (cf. section 8.2).

Since a mutation probability of 0.01 translates to 1 node to adapt the phase offset of its carrier signal per iteration on average, smaller mutation probabilities are not beneficial.

Normal distributed phase offset

We also considered a normal distribution for the phase alteration as it was applied in [42, 16, 41, 18] and study the impact of the mutation probability and the variance utilised.

Figure 8.16 shows the performance of this approach with several values for the mutation variance applied to the phase mutation process when all nodes mutate in every iteration (mutation probability of 1). This configuration is identical to the simulations conducted in [42, 16, 41, 18]. For ease of presentation, error bars are omitted in this figure.

We observe that a smaller variance is beneficial for the performance of the synchronisation process. The reason for this is again the reduced neighbourhood size of the search method at smaller variances. In particular, by means of the mutation variance the neighbourhood size can be adapted to an optimum for a given mutation probability.

However, when both, the variance and the mutation probability are small, the simulation

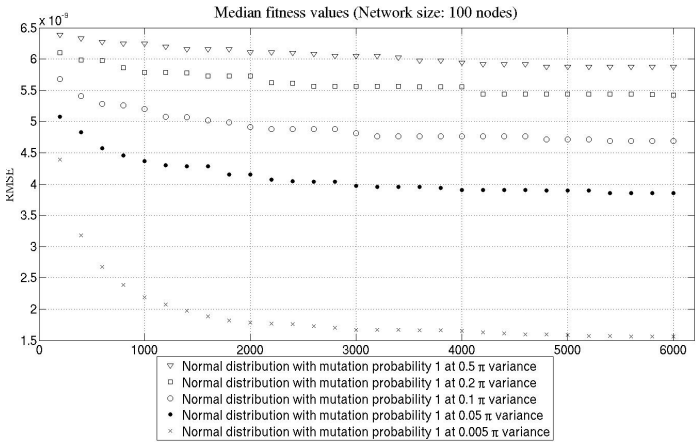


Figure 8.16: Normal distribution of phase mutations with mutation probability 1

performance degrades as the improvements in the fitness space are minor. Figure 8.17 depicts this property for various combinations of p_γ and σ_γ^2.

We observe that the mutation performance can be improved by altering the mutation variance only to a particular extent. When the variance is decreased further, the overall synchronisation performance degrades.

We approximated the optimum variance for several mutation probabilities. Generally, a higher mutation probability necessitates a smaller variance to achieve the optimum performance. Figure 8.18 depicts the performance for mutation probabilities of 0.2, 0.05 and 0.01 and near optimum variance, respectively.

For these configurations the performance is similar to the performance of the uniform distribution with mutation probability 0.01.

Discussion

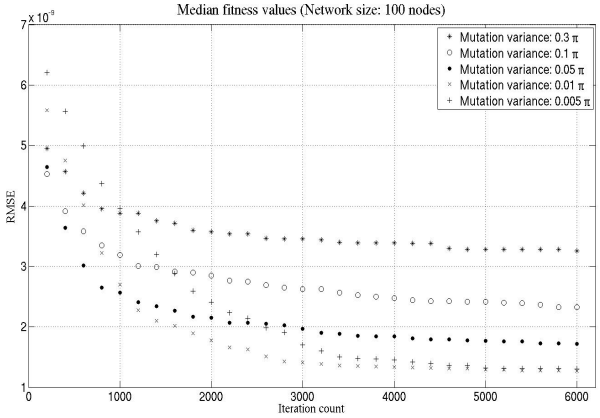
Summarising, we conclude that the best performance was achieved for small modifications of a search point in each iteration. This is achieved for the uniformly distributed phase offset by a small probability to alter the phase offset of an individual node. For the normal distributed process, small steps in the search space can be achieved either by a small variance or by a small probability to alter the phase offset of an individual node.

This optimisation performance can be further improved at the start of the synchronisation process when a moderate mutation probability is implemented that is gradually adapted in the course of the synchronisation process.

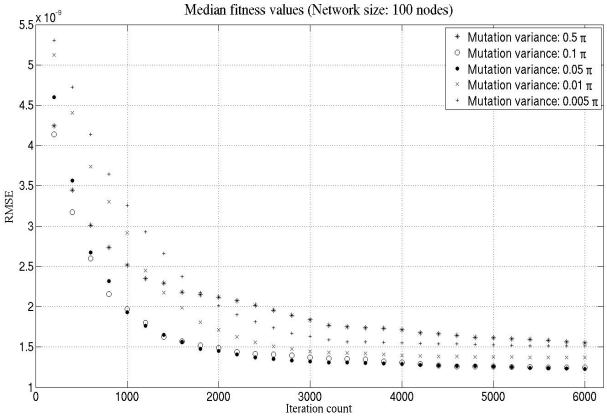
For the normal distribution the performance can be improved by adopting the variance of the process.

Overall, results for near optimum configurations are similar for normal and uniform distributed phase alteration processes. A straightforward implementation would, for instance, constitute an adaptive phase mutation process with uniform distribution.

Since in [130] the search space of distributed adaptive beamforming was classified as



(a) Normal distribution of phase mutations with mutation probability 0.2 and various values for the mutation variance σ_γ^2



(b) Normal distribution of phase mutations with mutation probability 0.05 and various values for the mutation variance σ_γ^2

Figure 8.17: Normal distribution of phase mutations with a fixed mutation probability and various values for the mutation variance σ_γ^2

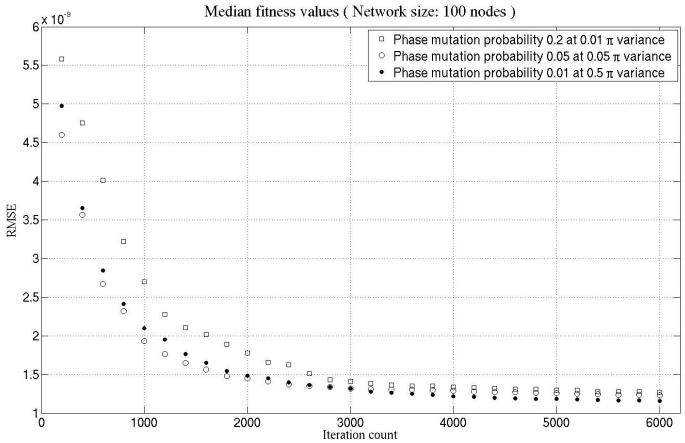


Figure 8.18: Performance of distributed adaptive beamforming in a wireless sensor network of 100 nodes and normal and uniform probability distributions on the phase mutation probability. Normal distribution of phase mutations.

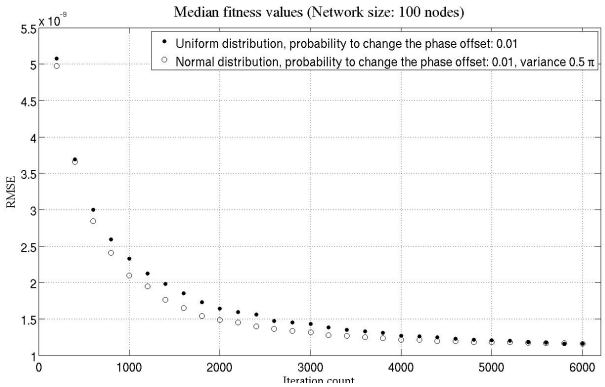


Figure 8.19: Performance of normal and uniform distributions for a network size of 100 nodes and $p_\gamma = 0.01, \sigma_\gamma^2 = 0.5\pi$.

weak unimodal, we expect a nearest neighbour search approach to be best suited to achieve fast synchronisation. We implemented and tested a local random search heuristic for this scenario with various neighbourhood sizes. The implementation differs from the normal distribution mainly since it utilises a fixed size neighbourhood instead of a variance on the phase alteration process. Figure 8.19 shows that the performance is similar to the implementation with uniformly distributed phase alteration probabilities. We therefore conclude that the global random search approaches utilised are already near optimal solutions for distributed adaptive beamforming in wireless sensor networks.

Distributed adaptive beamforming in wireless sensor networks has been studied in the literature according to various random phase alteration processes. The authors in [15, 16, 17] report good results when the probability p_γ to alter the phase of a single carrier signal in one iteration is 1 for all nodes and the phase offset is chosen according to a normal distribution. The variance σ_γ^2 applied is not reported. In [13, 14] p_γ was set to $\frac{1}{n}$ for each one of the n nodes while the phase is altered according to a uniform distribution.

For both, uniform and normal distributed processes, we consider several values for p_γ and σ_γ^2. Generally, we achieved good performance when modifications in one iteration were small. For the uniform distribution this translates to $p_\gamma = \frac{1}{n}$. For the normal distribution, good results are achieved when σ_γ^2 and p_γ are balanced so that the modification to the overall sum signal is small. With increasing p_γ good results are achieved with decreasing σ_γ^2. Figure 8.19 depicts the simulation results for $p_\gamma = \frac{1}{n}$ and $\sigma_\gamma^2 = 0.5\pi$

The figure shows the median RMSE value achieved in 10 simulations by normal and uniform distributed processes over the course of 600 iterations. For ease of presentation, error bars are omitted in this figure. However, the standard deviation is low for both processes (the standard deviation of this normal distributed process is depicted in figure 8.29(b)).

The normal distributed process has a slightly improved synchronisation performance. The optimum fitness value reached is, however, identical to the uniformly distributed process.

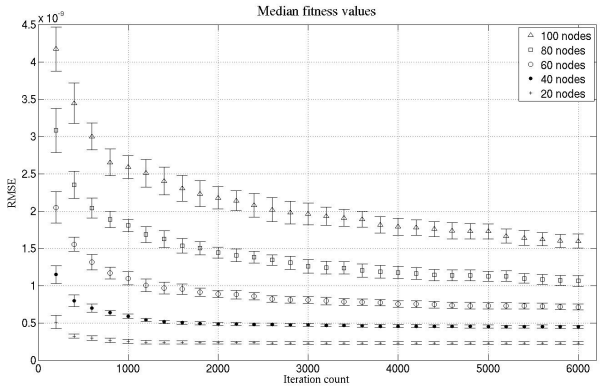


Figure 8.20: The synchronisation performance for various network sizes in a uniformly distributed process with $p_\gamma = 0.05$.

8.2.5 Impact of environmental parameters

Apart from parameters of the optimisation algorithm, also the environmental setting might impact the synchronisation performance of distributed adaptive beamforming in wireless sensor networks. We consider the impact of the network size and the distance between a receiver and the network.

Impact of the network size

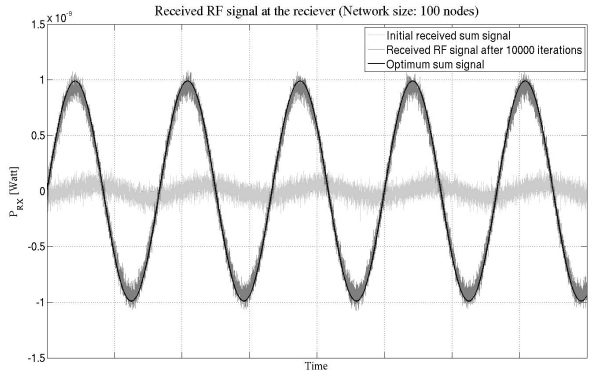
When the count of nodes that participate in the synchronisation is altered, this also impacts the performance of this process (cf. section 8.2). We conducted several simulations with network sizes ranging from 20 to 100 nodes. Figure 8.20 depicts the performance of several synchronisation processes with varying network sizes.

In these simulations, we set $p_\gamma = 0.05$ and utilised a uniformly distributed phase alteration process. We see that the maximum fitness value achieved is lower for smaller network sizes. This is due to the RMSE measure that compares the achieved sum signal to an expected optimum superimposed signal. As the count of participating nodes diminishes, also the amplitude of the optimum signal decreases. As expected, the optimum value is reached earlier for smaller network sizes.

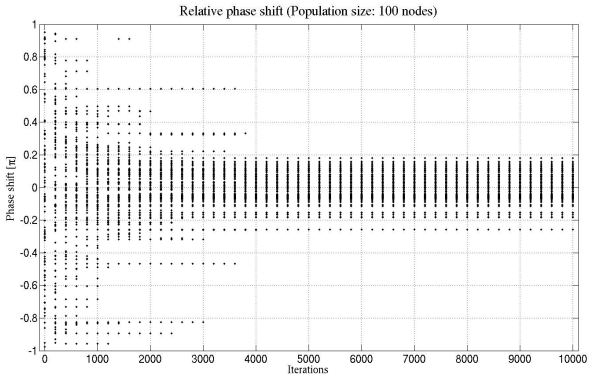
Distance between receiver and network

We are also interested in the performance of distributed adaptive beamforming when the distance between the network and a receiver is increased. For a uniformly distributed phase alteration process with $p_\gamma = \frac{1}{n}$ we increase the transmission distance successively. Figure 8.21 depicts the phase coherency achieved and the received sum signal for various transmission distances.

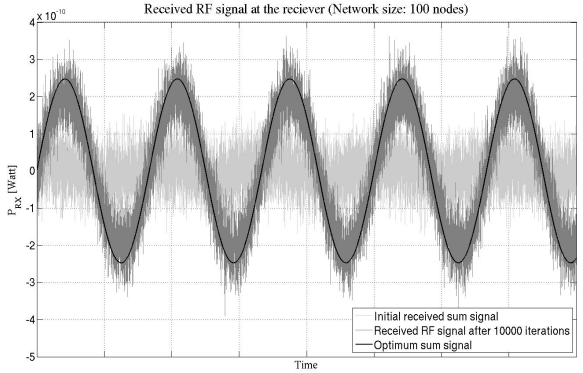
Although the noise power relative to the sum signal increases, synchronisation is possible at about 200 meters distance. Observe that in our model with $P_{tx} = 1mW$ we expect a



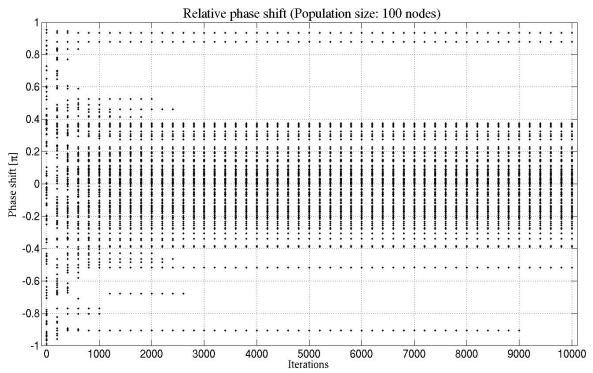
(a) Receiver distance: 100 meters – Received RF signal



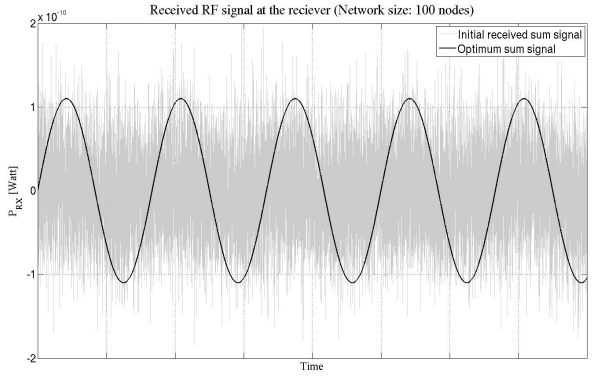
(b) Receiver distance: 100 meters – Relative phase shift of signal components



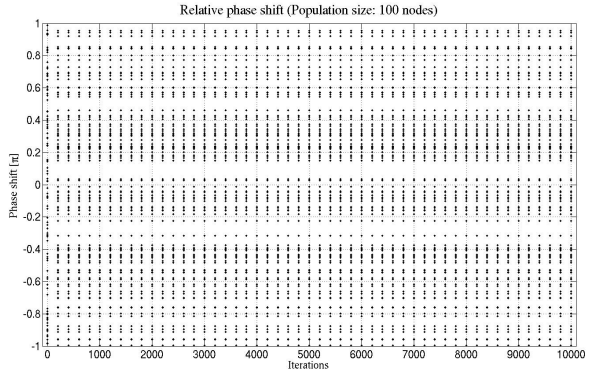
(c) Receiver distance: 200 meters – Received RF signal



(d) Receiver distance: 200 meters – Relative phase shift of signal components

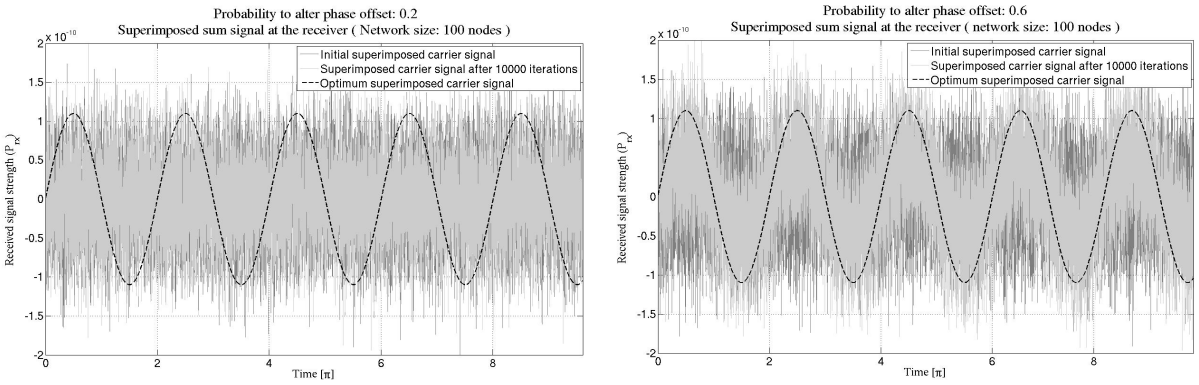


(e) Receiver distance: 300 meters – Received RF signal



(f) Receiver distance: 300 meters – Relative phase shift of signal components

Figure 8.21: RF signal strength and relative phase shift of received signal components for a network size of 100 nodes after 10000 iterations. Nodes are distributed uniformly at random on a $30m \times 30m$ square area and transmit at $P_{TX} = 1mW$ with $p_\gamma = \frac{1}{n}$.



(a) Receiver distance: 300 meters, $p_\gamma = 0.2$

(b) Receiver distance: 300 meters, $p_\gamma = 0.6$

Figure 8.22: RF signal strength and relative phase shift of received signal components for a network size of 100 nodes after 10000 iterations. Nodes are distributed uniformly at random on a $30m \times 30m$ square area and transmit at $P_{TX} = 1mW$.

signal strength at the receiver of $0.1\mu W$ or $-40dBm$ for each single carrier at this distance.

When the distance is further increased to 300 meters, however, synchronisation is not possible with this configuration. This is due to the high impact of the noise fluctuation on the received signal. The fluctuation of the noise figure has a higher impact on the signal than the alteration of single carrier signals.

However, when more carrier signals are altered simultaneously, a weak synchronisation is still possible. Figure 8.22 depicts the received carrier signal after 100 iterations for the uniformly distributed process with $p_\gamma = 0.2$ and $p_\gamma = 0.6$. We see that the synchronisation quality is improved with increasing p_γ. While the superimposed signal is indistinguishable for $p_\gamma = 0.2$, the synchronisation quality increases with $p_\gamma = 0.6$. Although the signal is heavily distorted, the carrier can be extracted.

8.2.6 Impact of algorithmic modifications

The iterative feedback-based distributed adaptive beamforming in wireless sensor networks was analysed from various authors in the literature. Some of these have proposed small algorithmic alterations of the basic approach in order to improve the synchronisation performance. We present three of these approaches in the following sections.

Reelection of unsuccessful nodes

The author of [43] observed that when nodes discard their last alteration to their carrier phases at the end of an iteration, the information available from the receiver feedback is not optimally utilised.

In particular, when a set of nodes alters their individual phase offsets simultaneously, this constitutes a distinct shift in the carrier phase of the received sum signal.

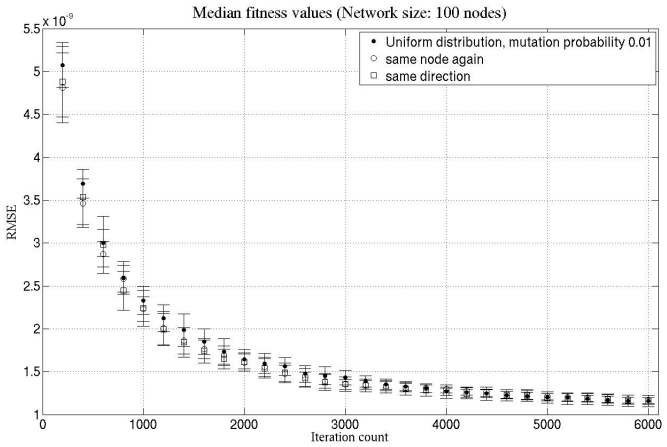


Figure 8.23: Performance of distributed adaptive beamforming in WSNs when successful nodes are re-considered for mutation

When the fitness value reached after this operation is decreased, however, it means that the distance between the phase of an optimal configuration and the current superimposition was increased. Consequently, when all nodes would apply the inverse of their last applied phase offset, this distance would be decreased and the fitness improved.

In [43] it was proposed to apply the inverse of the last applied phase modulation each time a fitness value is decreased. The author could show that the optimisation time is reduced by factor $\frac{1}{2}$ by this approach. The reason is, clearly, that the fitness decrease which is achieved with probability about $\frac{1}{2}$ (an improved fitness value in in every second iteration on average) is now converted into an increase of the fitness value.

Reelection of successful nodes

For distributed adaptive beamforming, an optimum is reached when all received transmit signal components are in phase. In the random synchronisation process, this situation is approached iteratively, which means that several steps to an optimum are required. In particular, as long as the optimum phase offset for one transmit signal component is not reached, the fitness value can be improved by sufficiently altering the phase offset for this transmit signal component again. To exploit this property we modify the implementation of the original approach in such a way that transmit signal components that were altered in an iteration in which the fitness value was increased are altered again in the next iteration. The intuition behind this approach is that for a transmit signal component which was successfully altered we expect that the optimum phase offset is not yet reached so that further benefit is possible. We implemented two distinct approaches. The first approach is to alter the phase offset of the same transmit signal component again uniformly at random. For the second implementation also the direction in which the phase was altered is sustained. Figure 8.23 depicts simulation results for these two approaches when compared to the standard uniformly distributed phase alteration approach. We observe that both

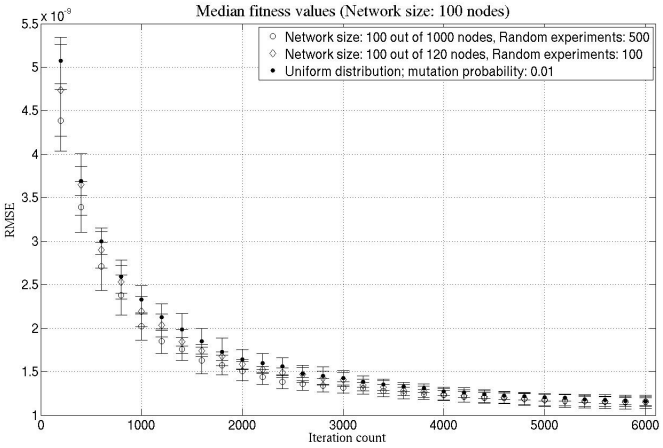


Figure 8.24: Performance of distributed adaptive beamforming in WSNs when participating nodes are chosen based on random experiments

approaches have similar performance and improve the fitness early in the synchronisation process. With increasing iteration count, however, the standard approach catches up.

Preconfigured nodes

In distributed adaptive beamforming, several nodes in a network collaboratively reach a distant receiver. The synchronisation time for these nodes is dependent on the number of nodes that participate in the synchronisation (cf. [14, 15]). Consequently, when not all nodes are required to reach a given distance, it is beneficial to utilise only a subset of nodes. In particular, we would like to pick those nodes for the synchronisation process that are well pre-synchronised so that the initial fitness value is already high. We conducted several simulation runs in which a set of pre-synchronised nodes is identified in random experiments prior to the synchronisation. In these random experiments a set of 100 randomly chosen nodes transmit simultaneously. The fitness value reached in this simultaneous transmission is stored for each experiment. After all random experiments are completed the synchronisation is conducted with the nodeset that scored the best fitness value during the random experiments. Figure 8.24 details the simulation results. For a network size of 1000 and 120 nodes, 100 nodes are picked in 500 and 100 random experiments, respectively. These simulations are compared to simulation results in which a network of 100 nodes is synchronised with uniformly distributed phase alterations and a mutation probability of 0.01 but without pre-synchronisation of nodes. We see that the pre-synchronised node-sets reach better fitness values earlier in the simulation. In particular, an improvement is already visible for 100 nodes chosen in 100 random experiments out of 120 nodes. This shows that we can benefit from the node choice already with few random experiments and with network sizes that deviate from the required network size only slightly. However, observe also that the lead in fitness value is shrinking with increasing iteration count.

8.3 Alternative algorithmic approaches

We have so far discussed the feasibility of a simple iterative approach to achieve distributed adaptive beamforming in wireless sensor networks. Based on these findings we discuss three modifications to this general iterative method that are suited to improve the synchronisation performance.

8.3.1 Hierarchical clustering

Since this bound on the synchronisation time is more than linear in n (cf. section 8.2), it strikes that the $\mathrm{RSS}_{\mathrm{sum}}$ changes linear with the network size n. Therefore, the overall energy consumption and synchronisation time might benefit from a reduced number of nodes transmitting at increased power level.

We propose the following hierarchical clustering scheme that synchronises all transmit nodes iteratively in clusters of reduced size.

1. Determine clusters (e.g. by a random process initialised by the receiver node)

2. Guided by the receiver node, synchronise clusters successively as described above with possibly increased transmit power for nodes. When cluster ι is sufficiently synchronised, all nodes in this cluster sustain their carrier signal and stop transmitting until all other clusters are synchronised.

3. At this stage, carrier signals in each one of the clusters are in phase but carrier phases of distinct clusters might differ. To achieve overall synchronisation, we build an overlay-cluster of representative nodes from all clusters. These nodes are then synchronised.

4. Nodes in all clusters alter the phase of their carrier signal by the phase offset experienced by the corresponding representative node. Let $\zeta_i = \Re\left(m(t)\mathrm{RSS}_i e^{j2\pi f_c t(\gamma_i+\phi_i+\psi_i)}\right)$ and $\zeta_i' = \Re\left(m(t)\mathrm{RSS}_i e^{j2\pi f_c t(\gamma_i'+\phi_i+\psi_i)}\right)$ be the carrier signals of representative node i from cluster ι before and after synchronisation between representative nodes was achieved. A node h from the same cluster ι will then alter its carrier signal $\zeta_h = \Re\left(m(t)\mathrm{RSS}_h e^{j2\pi f_c t(\gamma_h+\phi_h+\psi_h)}\right)$ to $\zeta_h' = \Re\left(m(t)\mathrm{RSS}_h e^{j2\pi f_c t(\gamma_h+\phi_h+\psi_h+\gamma_i-\gamma_i')}\right)$. Under ideal conditions, all nodes should be in phase after this step although an overall synchronisation was not applied.

5. To account for synchronisation errors a final synchronisation phase in which all nodes participate concludes the overall synchronisation process.

Figure 8.25 illustrates this procedure. A potential problem for this approach is phase noise. As only one cluster is synchronised at a time, due to practical properties of oscillators, phases of nodes in the inactive clusters experience phase noise and start drifting

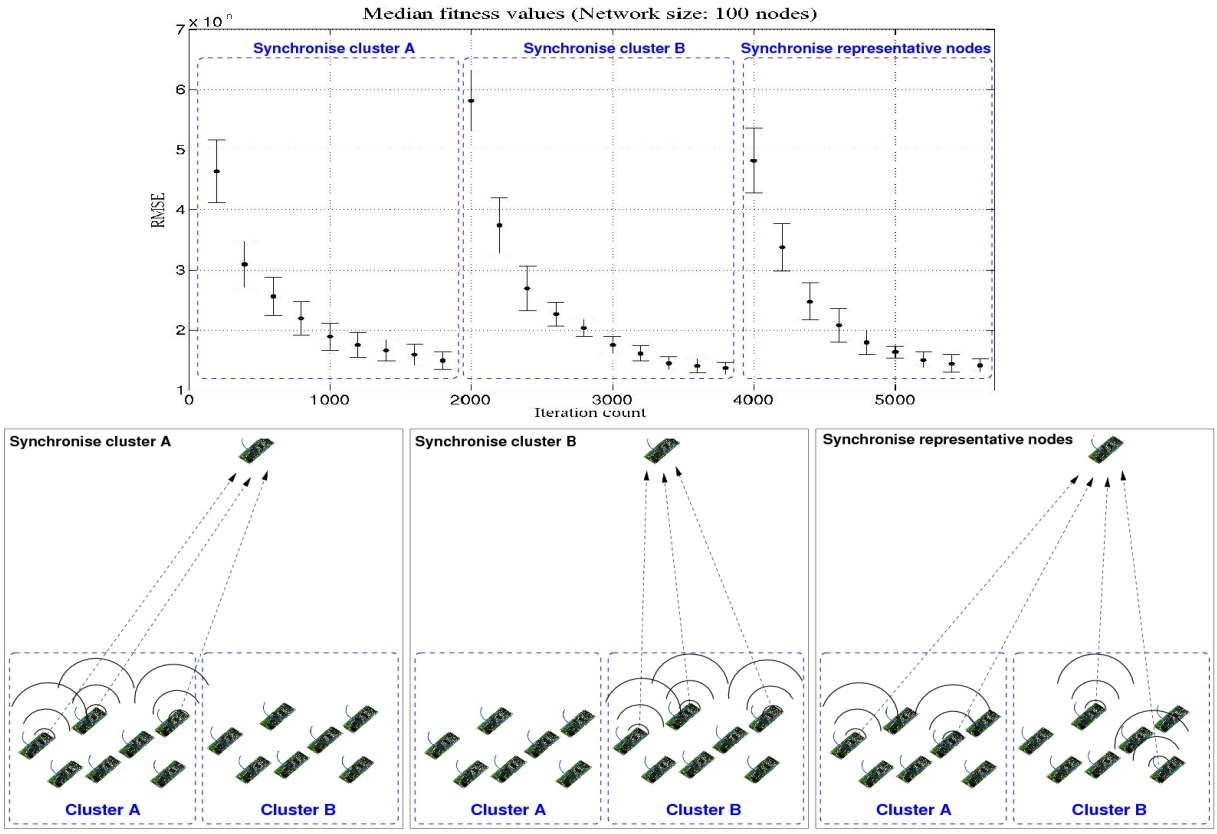


Figure 8.25: Illustration of the approach to cluster the network of nodes in order to improve the synchronisation time of feedback based closed-loop distributed adaptive beamforming.

out of phase. However, sufficient synchronisation is possible in the order of milliseconds. [1] Therefore, we do not consider phase noise an important issue for the hierarchical clustering approach.

Observe that all coordination is initiated by the receiver node so that no inter-node communication is required for coordination. Clusters are formed by a random process and synchronisation between and within clusters is achieved by the distributed adaptive beamforming approach described above.

Depending on the size of the network, more than one hierarchy stage might be optimal for the optimisation time and the energy consumption. In order to estimate the optimal hierarchy depth and the optimum cluster size, the count of nodes participating in the synchronisation process must be computed. We assume that the nodes themselves do not know of the network size. This means that the remote receiver derives the network size, calculates optimal cluster sizes and hierarchy depths and transmits this information to the nodes in the network. In [128] it was demonstrated that the superimposed sum signal from arbitrarily synchronised nodes over some time interval is sufficient to compute an estimation on the number of transmitting nodes. We can derive the optimum hierarchy depth and cluster size by integer programming in time $\mathbb{O}(n^2)$.

We estimate the expected optimisation time for a network of size n by $E[T_{\mathcal{P}n}] = c \cdot k \cdot n \cdot \log(n)$ (cf. section 8.2) for a suitable constant c and the expected energy consumption by $E[\mathcal{E}_{\mathcal{P}n}] = c \cdot k \cdot n \cdot \log(n) \cdot \overline{\mathcal{E}_{\mathcal{P}n}}$ where $\overline{\mathcal{E}_{\mathcal{P}n}}$ is the energy consumption of all n nodes in one iteration [14]. A hierarchy and cluster structure that minimises these formulae when summed over all hierarchy stages is optimal in our sense. We derive the optimum cluster sizes and hierarchy depths by integer programming. Observe that for a cluster size of m the above formulae have the property

$$E[T_{\mathcal{P}n}] = E[T_{\mathcal{P}\frac{n}{m}}] \cdot \frac{n}{m} \cdot E[T_{\mathcal{P}m}]$$
$$E[\mathcal{E}_{\mathcal{P}n}] = E[\mathcal{E}_{\mathcal{P}\frac{n}{m}}] \cdot \frac{n}{m} \cdot E[\mathcal{E}_{\mathcal{P}m}].$$

We define the recursion by

$$E_{\text{opt}}[T_{\mathcal{P}n}] = min_m \left[E_{\text{opt}}[T_{\mathcal{P}\frac{n}{m}}] \cdot \frac{n}{m} \cdot E_{\text{opt}}[T_{\mathcal{P}m}] \right]$$
$$E_{\text{opt}}[\mathcal{E}_{\mathcal{P}n}] = min_m \left[E_{\text{opt}}[\mathcal{E}_{\mathcal{P}\frac{n}{m}}] \cdot \frac{n}{m} \cdot E_{\text{opt}}[\mathcal{E}_{\mathcal{P}m}] \right]$$

and the start of the recursion by $E_{\text{opt}}[T_{\mathcal{P}\eta}]$ and $E_{\text{opt}}[\mathcal{E}_{\mathcal{P}\eta}]$ with η being the minimum feasible cluster size for distributed adaptive beamforming when the maximum transmission power and distance are given. Since η is dependent on the distance to the receiver, it can be calculated over the round trip time between the sensor network and the receiver.

[1] Consider, for instance, that the synchronisation sequence requires several ten signal periods and that the RSS estimation and application of phase shift is also possible in similar order. A simple calculation (e.g. frequency $f = 2.4\text{GHz}$, transmission distance $d = 100m$, $c = 3 \cdot 10^8 \frac{m}{s}$) reveals that a single iteration requires only microseconds.

The time required for the calculation of the optimum hierarchy depth and cluster sizes is quadratic. With a network of n nodes, at most n^2 distinct terms $E_{\text{opt}}[T_{\mathcal{P}i}]$ and $E_{\text{opt}}[\mathcal{E}_{\mathcal{P}i}]$ with $i \in \{1, \ldots, n\}$ are of relevance. We can start by calculating $E_{\text{opt}}[\mathcal{E}_{\mathcal{P}\eta}]$ and $E_{\text{opt}}[T_{\mathcal{P}\eta}]$ and obtain all other values by table look-up according to $E_{\text{opt}}[T_{\mathcal{P}n}]$ and $E_{\text{opt}}[\mathcal{E}_{\mathcal{P}n}]$ in time $\mathcal{O}(n^2)$ since every one of the (at most) n entries has not more than n possible predecessors.

8.3.2 A local random search approach

Recent approaches to distributed adaptive transmit beamforming in wireless sensor networks utilise a random search approach that could in each iteration reach any point in the search space with positive probability. We define a search point as one combination of phase offsets for all RF signal components and a neighbourhood relation over the difference in phase offsets in these configurations. As the search space is multimodal, no local optima exist so that in an arbitrary neighbourhood around a given search point the fitness value is either optimal or search points with an improved fitness value exist.

Therefore, a local random search can not converge in a local minima. Also, the optimisation landscape for a local search approach is then simple. As long as it manages to follow a path with increasing fitness values, it will find an optimum with probability 1.

We can also show that, when the distance to the optimum and to the worst point is greater than the neighbourhood radius, the algorithm has an equal chance to improve or worsen the fitness score. We see this as follows. Assume that the fitness of the sum signal is proportional to the amplitude of the signal. When $n-1$ signal components are received simultaneously at a given frequency f the resulting sum signal is of the same frequency as the individual RF signal components. When the phase offset of the n-th signal component is identical or complementary to the phase offset of the sum signal, the amplitude of the signal created by the superimposition of these two signals is at its maximum or minimum, respectively. In between these two values it degrades symmetrically. Consequently, when the minimum or maximum is not inside the neighbourhood, an equal number of points incorporate better or worse fitness values (cf. figure 8.26) – although the slope of the fitness landscape might differ among search points that improve or decrease the fitness value.

This means that an algorithm with restricted neighbourhood size has a probability of 0.5 to increase the fitness value for a long time during the optimisation until the optimum point is within the neighbourhood. The price for this high probability to improve the fitness value in each iteration is that the chance to reach the optimum in one step or, generally to achieve great progress in one step (as possible with an unrestricted neighbourhood size) is lost. Since this event is intuitively less probable, we are prepared to pay this price. We provide an upper bound on the optimisation time of a local random search approach for distributed adaptive beamforming in wireless sensor networks.

To ease the analysis we model the optimisation problem in a binary representation. We assume that each one of the sensor nodes is able to apply k distinct phase offsets of transmit RF signals. For each of the n nodes, we binary encode the distinct phase offsets by $\log(k)$ bits. Overall, a binary string of length $n \cdot \log(k)$ describes the contribution of phase offsets

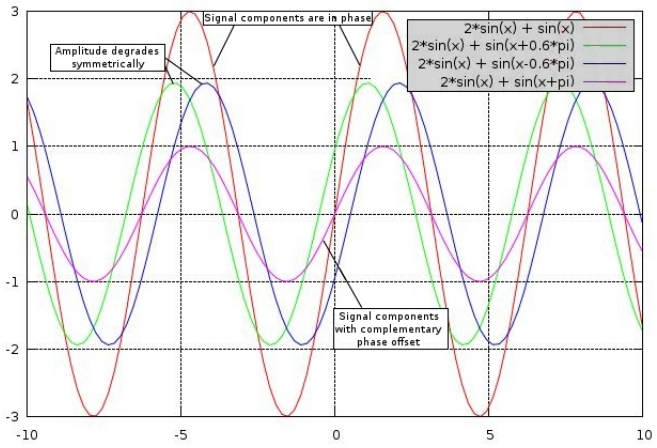


Figure 8.26: Example of sinusoid sum signals. The amplitude of the sum signal degrades symmetrically when the phase offset between the two signal components increases

in one iteration. The neighbourhood of a given configuration is defined by the maximum phase offset applicable to an individual node altering the phase of its carrier signal. An individual node i alters the phase offset γ_i of its carrier signal uniformly at random within a range of $[\gamma_i - \kappa, \gamma_i + \kappa]$ for suitable κ. We assume that configurations are encoded such that the hamming distance between two configurations increases with increasing difference in phase offsets.

We analyse the count of bit mutations of this representative bitstring until a bitstring is found that encodes the phase configurations of a global optimum. We define a global optimum as a superimposition of signal components in which all signal components are within $\frac{2\pi}{k}$ of a superimposition with perfect phase coherency. We choose the mutation probability as $\frac{1}{n}$ for a network of n nodes. In this case, the $\log(k)$ bits that represent the phase offset of the corresponding node are altered uniformly at random inside the neighbourhood boundaries. With Chernoff bounds we can show that w.h.p. the hamming distance to an optimum is not much smaller than $\frac{n \cdot \log(k)}{2}$.

An asymptotic upper bound

We derive an upper bound for the local random search approach similar to the upper bound derived in section 8.2. We define the neighbourhood of size N for each binary representation for a binary representation of an individual carrier signal ζ_i (remember that the individual representation was concatenated from all n binary representations of individual carrier signals of length $\log(k)$ (cf. figure 8.2)) individual string $\mathcal{I}$ as all individual representations with hamming distance at most N to $\mathcal{I}$. As long as the optimum is far away (i.e. outside the neighbourhood boundaries of all carrier signals), the probability to reach a new search point with better fitness value is at least $\frac{1}{2}$ as detailed above. The expected time until all

carrier signals have the optimum inside their neighbourhood boundaries is then

$$E[\mathcal{P}_1] = \mathcal{O}(n \cdot \log(k)). \tag{8.24}$$

Afterwards, the analysis of the optimisation time is similar to the upper bound derived for the global random search approach in section 8.2. The difference for the local random search approach is, that the distance to the optimum is $n \cdot N$ at most. Consequently, the estimation of the upper bound on the synchronisation performance in equation (8.14) is then altered to

$$\begin{aligned} E[T_{\mathcal{P}_2}] &\leq \sum_{i=n\cdot\log(k)-n\cdot N}^{n\cdot\log(k)-1} \frac{e \cdot n \cdot \log(k)}{n \cdot \log(k) - i} \\ &= e \cdot n \cdot \log(k) \cdot \sum_{i=n\cdot\log(k)-n\cdot N+1}^{n\cdot\log(k)} \frac{1}{i} \\ &< e \cdot n \cdot \log(k) \cdot (\ln(n \cdot \log(k)) + 1 - \ln(n \cdot \log(k) - n \cdot N) + 1) \\ &= \mathcal{O}\left(n \cdot \log\left(\frac{n \cdot \log(k)}{n \cdot \log(k) - n \cdot N} + k\right)\right). \end{aligned} \tag{8.25}$$

The upper bound on the expected overall optimisation time is then composed from the upper bound on the optimisation times of both phases as

$$\begin{aligned} E[T_{\mathcal{P}}] &= E[T_{\mathcal{P}_1}] + E[T_{\mathcal{P}_2}] \\ &= \mathcal{O}\left(n \cdot \log\left(\frac{n \cdot \log(k)}{n \cdot \log(k) - n \cdot N} + k\right)\right). \end{aligned} \tag{8.26}$$

An asymptotic lower bound

For the local random search approach we can obtain a lower bound on the synchronisation performance by the method of the expected progress similar to the approximation in section 8.2. With similar arguments as in section 8.2 this leads to a lower bound of

$$\Omega\left(n \cdot \log\left(\frac{n \cdot \log(k)}{n \cdot \log(k) - n \cdot N} + k\right)\right). \tag{8.27}$$

Simulation results

For our implementation of the local random search algorithm we modify the phase alteration mechanism applied by individual nodes. While for the original random search approach the phase offset is chosen uniformly at random from all possible phase offsets in $[-\pi, \pi]$, we restrict the set of possible future phase offsets to a subset of this range: $\gamma \in [\sigma_1, \sigma_2]$ with $-\pi \leq \sigma_1 < \sigma_2 \leq \pi$. Figure 8.27 shows the simulation results of the algorithm. We observe that the hill-climbing approach reaches lower fitness values faster than the original approach. In particular, the near optimal fitness value reached by the original algorithm after 6000 iterations is achieved by the hill climber already after about 3000 iterations.

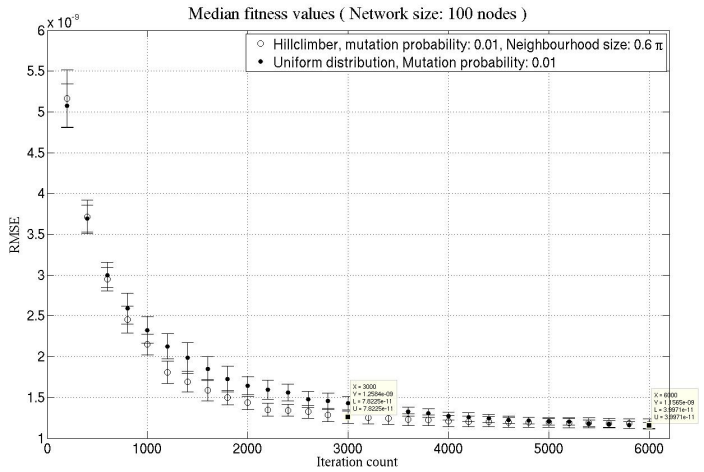


Figure 8.27: Performance of the local random search implementation for distributed adaptive beamforming in wireless sensor networks

8.3.3 Multivariable equations

The recent approaches discussed for distributed adaptive transmit beamforming in wireless sensor networks all heavily rely on random experiments. This was necessary since distinct nodes in the scenario do not communicate with each other. Therefore, each node has access only to very limited information on the current search point. Consequently, as a concise view on the current synchronisation progress is not given, a random search heuristic can be seen as the natural approach for this problem scenario.

The introduction of randomness into the search approach, however, also decreases the synchronisation performance. First approaches to reduce the amount of randomness have been introduced in section 8.3.1 and section 8.3.2. In the former approach, the problem was divided into sub-problems of smaller size. For these smaller problem instances the synchronisation performance improved since the search space is drastically smaller for smaller network sizes. In particular, it decreases exponentially in n as each node corresponds to a further dimension in the search space.

For the latter approach, randomness was reduced by restricting the neighbourhood size of the search approach. As long as the optimum is not inside the neighbourhood, the probability to improve the fitness value was $\frac{1}{2}$.

In the following section we present an approach by which a single node is able to estimate the optimum phase offset of its carrier signal relative to the current phase offsets of all other nodes. Due to the distributed nature of this scenario, we are still bound to a small amount of randomness. However, we can show that the approach achieves an asymptotically optimal synchronisation performance.

The idea to reach this synchronisation performance is to optimally utilise the receiver feedback. In the approaches discussed above, the feedback required is binary. However, when more information is encoded into the feedback value so that information on the actual

feedback function can be derived, transmitters can utilise this additional information to apply improved synchronisation methods.

Consequently, a vital requirement on the design of the algorithm is that a rich feedback is provided by the receiver node. In particular, the approach described is not feasible with binary feedback values.

Algorithmic modelling

Since no local optimum exists in the search space due to its weak multimodality (cf. section 8.1.3), the performance of $\Theta(n \cdot \log(n \cdot \log(k) + k))$ we derived for the random search approach seems weak. In every iteration the receiver provides additional information over a feedback value so that a node i can learn the optimum phase offset of its own carrier

$$\zeta_i = \Re\left(m(t)\mathrm{RSS}_i e^{j2\pi f_c t(\gamma_i+\phi_i+\psi_i)}\right)$$

relative to the superimposed sum signal

$$\zeta_{\mathrm{sum}\setminus i} = \Re\left(m(t)e^{j2\pi f_c t}\sum_{o\in[1,n];o\neq i}\mathrm{RSS}_o e^{j(\gamma_o+\phi_o+\psi_o)}\right)$$

of all other nodes, provided that the latter does not change significantly. $\zeta_{\mathrm{sum}\setminus i}$ is a sinusoid signal. The fitness value is at its maximum when ζ_i and $\zeta_{\mathrm{sum}\setminus i}$ have identical phase offset at a receiver. With increasing phase offset

$$|(\gamma_i + \phi_i + \psi_i) - (\gamma_{\mathrm{sum}\setminus i} + \phi_{\mathrm{sum}\setminus i} + \psi_{\mathrm{sum}\setminus i})|$$

the fitness value decreases symmetrically. Consequently, the fitness function (note that we are now considering the fitness function $\mathcal{F}$ and not the received superimposed sum signal ζ_{sum}) is of the form

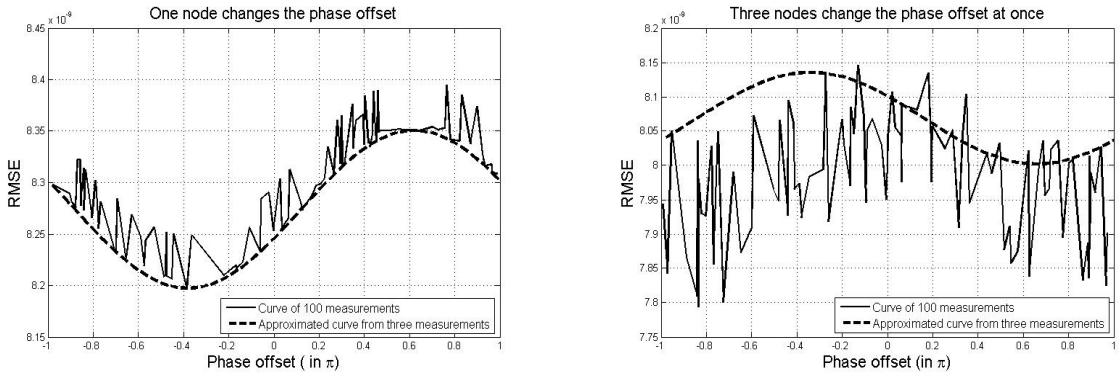
$$\mathcal{F}(\gamma_i) = A\sin(\gamma_i + \Phi) + c. \tag{8.28}$$

This is an equation with the three unknowns A (amplitude), Φ (phase offset of $\mathcal{F}$) and the additive term c so that a node i can calculate it with three distinct measurements.

These three function values for three distinct phase offsets $\gamma_1, \gamma_2, \gamma_3$ of a carrier signal can be calculated when the one node with a non-fixed carrier signal acquires the corresponding feedback values from the remote receiver. In figure 8.28(a), we depict the accuracy at which the RMSE fitness function can be estimated by this procedure. We calculate the root of the mean square error (RMSE) in this figure as

$$RMSE = \sqrt{\sum_{t=0}^{\tau}\frac{\left(\zeta_{\mathrm{sum}} + \zeta_{\mathrm{noise}} - \zeta_{\mathrm{opt}}\right)^2}{n}}. \tag{8.29}$$

In this formula, τ is chosen to cover several signal periods.



(a) RMSE-γ-relationship when only one sender node changes the phase offset

(b) RMSE-γ-relationship when three sender nodes change the phase offset at once

Figure 8.28: Approximation of the RMSE- phase offset- relationship

The dashed line in the figure depicts the fitness function estimated from three distinct feedback values while the solid line is created from 100 feedback calculations in a Matlab-based simulation environment. The 100 nodes are placed randomly in a field with a minimal distance of 30 meters from the receiver. The transmit nodes send with 2.4 GHz and a transmission power of 0.1 mW. From the calculated expected fitness function we can can easily determine the optimum phase offset for this carrier signal.

We experienced a maximum deviation between the the approximated and the measured RMSE values of less than 1% when all but one node keep their phase offset constant. In our measurements, the deviation of the calculated fitness curve did not exceed 0.6% when only one node adapts its phase offset.

However, since inter-node communication is not assumed in the scenario of distributed adaptive transmit beamforming in wireless sensor networks, more than one node might alter its phase offset at once. In this case, we can show that the calculated fitness curve more significantly deviates from the actual fitness function. With two nodes simultaneously adapting their phase offset we already experienced a deviation of approximately 1.5%. Figure 8.28(b) represents the approximation of the fitness function when three nodes change their phase offset simultaneously. The deviation between the approximated and the measured values is greater than that in the previous case and reaches values of more than 3%.

This observation leads to two important conclusions. The first is that a precise calculation of the optimum phase offset for a single node is possible when only one sender changes the phase offset of its carrier signal during a single iteration. The second fact is that verification of the correctness of the results is possible by measuring the significance of the deviation between calculated and actual fitness function.

In the following we describe the steps of the synchronisation procedure. Each four iterations are logically aggregated to one action. During an action, a node my either participate in by calculating it's optimum phase offset γ_i^* (active node) or it may transmit

its carrier signal unmodified (passive node).

With a probability $p_i = \frac{1}{n}$ (for a network of n nodes) a node i becomes an active participant and otherways stays passive.

An active node will :

1. Transmit it's carrier signal with three distinct phase offsets $\gamma_1 \neq \gamma_2 \neq \gamma_3$ and measure the feedback generated by the remote receiver. Feedback value and corresponding phase offset are stored by the node.
2. From these three feedback values and phase offsets, it estimates the feedback function and calculates the optimum phase offset γ_i^*.
3. Transmit a fourth time with $\gamma_4 = \gamma_i^*$.
4. If the deviation is less than 1% save γ_i^* as the optimal phase offset, otherwise discard it.

A passive node will :

1. Transmit the carrier signal four times with identical phase offset γ_i.

As nodes are chosen according to a random process, it is possible that more than one node simultaneously alters its phase offset. In this case, the node's conclusions on the impact of their phase-alteration on the fitness value are biased. Therefore, in the fourth measurement, when the measured value deviates significantly from the expected fitness value a node concludes that it was not the only one to alter its phase and reverses its decision.

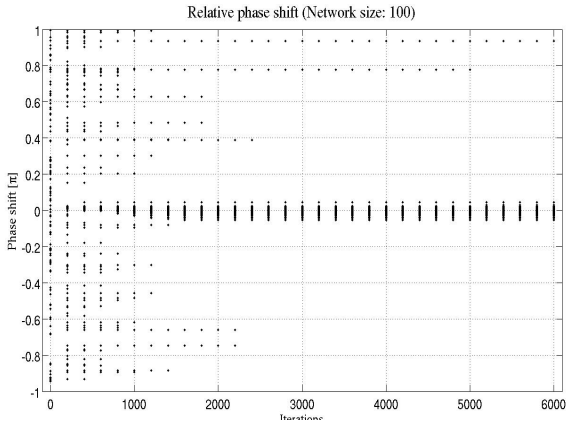
In order to decrease the count of idle actions at which either more than one node or no node actively participate, the nodes may adjust the value of p_i. After receiving the fourth feedback message, an active node i that has calculated γ_i^* successfully, becomes a passive node for a certain number of iterations. The node sets $p_i = 0$ to reduce the interference for other active nodes. All passive nodes that register an improvement of the feedback value after the fourth transmission assume that a node has calculated its γ_i^* successfully and alter their p_i-value to $p_i = \frac{1}{n - \text{successful actions}}$. The probability that during a time slot a node successfully calculates γ_i^* is 0.35.

As this procedure is guided purely by the feedback of the receiver node, inter-node communication is not required as the feedback is broadcast to all nodes in the network.

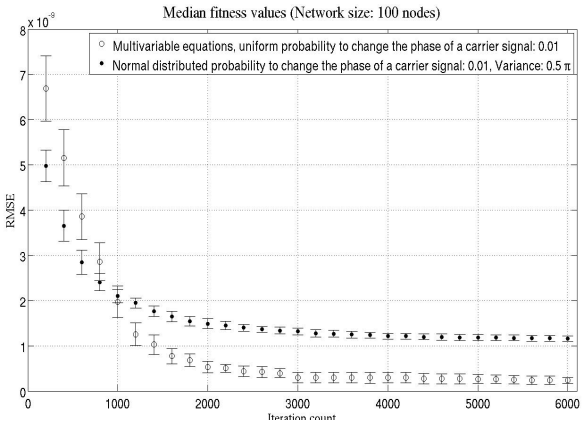
Further performance improvements can be achieved when nodes utilise only three subsequent iterations and acquire the first measurement from the last transmission of the preceding three subsequent iterations.

Analytic consideration

In our current implementation, we require about $12n$ iterations for all nodes to finally find and set the optimum phase offset of their carrier signal. This is asymptotically optimal, since the optimum phase offset of the carrier signal has to be calculated for each single node. Asymptotically, the synchronisation time of this algorithm is $\Theta(n)$ since on average the count of carrier signals that are in phase increases by 1 in each iteration.



(a) Phase offset achieved by the proposed optimisation algorithm for distributed adaptive beamforming in WSNs



(b) Performance of the proposed optimisation algorithm for distributed adaptive beamforming in WSNs

Figure 8.29: Distributed adaptive beamforming with a network size of 100 nodes where phase alterations are drawn uniformly at random. Each node adapts its carrier phase offset with probability 0.01 in one iteration. In this case, multivariable equation are solved to determine the optimum phase offset of the carrier signal.

Simulation results

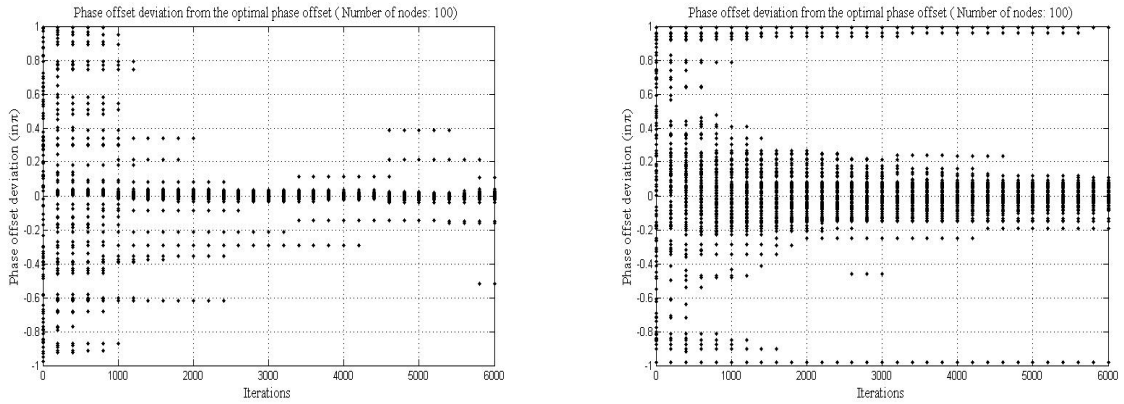
We also conducted simulation runs in which our implementation of the numerical algorithm described in section 8.3.3 is compared to the classical process with normal distributed phase alterations. When optimum phase offsets are calculated by solving multivariable equations at the transmit nodes, the synchronisation performance can be greatly improved as detailed in section 8.3.3. Figure 8.29 depicts the performance improvement achieved by solving multivariable equations to determine the fitness function compared to a global random search approach. We observe that the global random search heuristic is outperformed already after about 1000 iterations and the fitness value reached is greatly improved.

The phase offset of distinct nodes is within $+/-0.05\pi$ for up to 99% of all nodes.

Figure 8.30(a) shows the relative deviation of the phase offsets γ_i among 100 nodes. The results have been obtained in a Matlab-based simulation environment. After about 1500 iterations most of the nodes (about 90 %) have near optimum phase offsets. For comparison, the global random search approach typically utilised for distributed adaptive beamforming in wireless sensor networks has a greatly degraded performance (cf. figure 8.30(b)).

8.4 Environmental changes

For distributed adaptive beamforming in wireless sensor networks not only the algorithm applied to the problem scenario but also the environmental aspects impact the synchronisation performance. In section 8.2.5 we have already observed that a higher noise figure negatively impacts the synchronisation performance. Also, we learned that this effect can



(a) Results using the numeric algorithm

(b) Results using a global random search approach

Figure 8.30: Deviation of the phase offsets from the optimal phase offsets using the numerical and the random method

be partly compensated with an increased mutation probability.

Also, velocity of nodes might impact the synchronisation performance of distributed adaptive transmit beamforming in wireless sensor networks. Since the transmission beam sought is focused on a restricted area, the synchronisation of nodes might be biased if the receiver traverses out of this area. We consider this problem in section 8.4.1.

Another natural scenario regards the consideration of more than one receiver node. Typically, in a sensor network, the nodes in the network are identically and a special exposed node with extended capabilities does not exist. When a sub-network is, however, separated from another part of the network, distributed adaptive transmit beamforming might help to establish a connection between both sub-networks. The central question in this scenario, however, is how communication between both sub-networks is feasible when nodes are not able to overhear messages from nodes of the other sub-network directly. This probelm is considered in section 8.4.2.

So far, we considered a population size of 1 for the evolutionary optimiser since this seemed to be the natural choice. Section 8.4.3 discusses several aspects regarding the utilisation of increased population sizes for distributed adaptive beamforming in wireless sensor networks.

Finally, in section 8.4.4 we consider the problem of receive beamforming. This concept might decrease the communication amount in the network sufficiently.

8.4.1 Velocity of nodes

In current studies on distributed adaptive beamforming in wireless sensor networks, all nodes are considered static. An interesting case to be studied is that of mobility of nodes. The next section presents results from a Matlab-based simulation environment where mobility is applied to transmitter or receiver nodes.

All nodes in the following simulations transmit at a frequency of 2.4 GHz. The sender nodes utilise a transmission power of 0.1 mW. Ambient white Gaussian noise (AWGN) with a noise power of $-103dBm$ is applied as proposed in [94]. 100 nodes are placed in a $30m \times 30m \times 30m$ field. The transmit nodes are distributed randomly at the bottom of the field and the receiver is placed initially at the center of the field's top, so that the minimum possible distance between a sender and receiver is 30 meters.

In the simulation, we considered the Doppler effect due to the mobility of nodes. Individual signal components are summed up at the receiver node to generate the superimposed sum signal ζ_{sum}. Path loss was calculated by the Friis free space equation [84]

$$P_{tx} \left(\frac{\lambda}{2\pi d} \right)^2 G_{tx} G_{rx} \tag{8.30}$$

with $G_{tx} = G_{rx} = 0$. We disregarded shadowing and signal reflection so that only the direct signal component is utilised but not multipath propagation. We implemented a global random search approach and the asymptotically optimal algorithm introduced in section 8.3.3 for this scenario as well as two mobility models. The first mobility model is a random-walk model and the second a linear model.

Nodes in the random-walk model have no specific constant direction. After every iteration the movement direction is altered uniformly at random. The distance travelled between two consecutive iterations is constant and depends on the speed of motion specified. In the linear model, the sender or the receiver move linearly in a constant direction and with a constant speed.

In order to quantify the maximum speed at which a synchronisation is possible, we define a synchronisation as successful when the signal strength achieved is at minimum 75% of the signal strength possible with perfect synchronisation. All simulation results obtained are depicted in figure 8.31. These results are discussed in the following sections.

Performance of the global random search approach

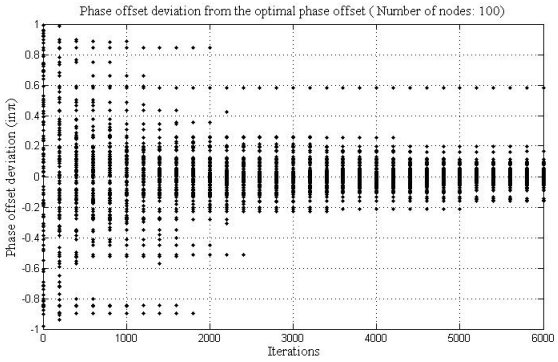
In our first scenario, the receiver node moves in a random walk mode. All transmit nodes remain static. We observed that a successful synchronisation in this scenario is possible with a movement speed of 5m/sec at most. Figure 8.31(a) shows the relative phase offset of individual carrier signals.

The standard deviation σ is about 0.1π for about 95% of all nodes after 6000 iterations. When, however, transmitters follow a random-walk movement while the receiver is not moving, the maximum speed is about 2 m/sec (cf. figure 8.31(b)).

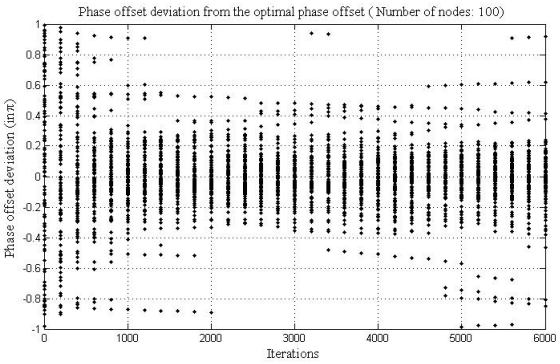
For the linear movement, the maximum relative speed between transmit and receive nodes with the global random search implementation is 30 m/sec regardless of whether the receive or the transmit nodes are moving (cf. figure 8.31(e).

Performance of the numeric algorithm

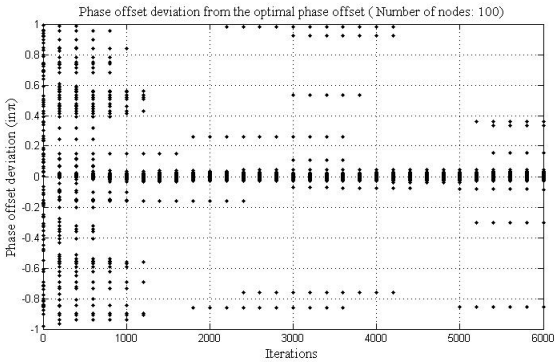
For the proposed numerical algorithm we applied the same settings as in section 8.4.1. when the receiver moves in a random-walk model while transmitters are static, the movement



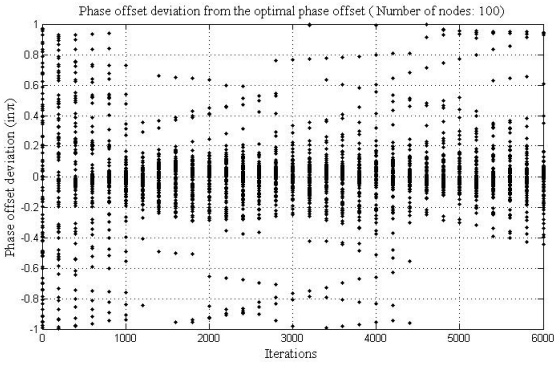
(a) Receiver moving at 5 m/sec following a random-walk model



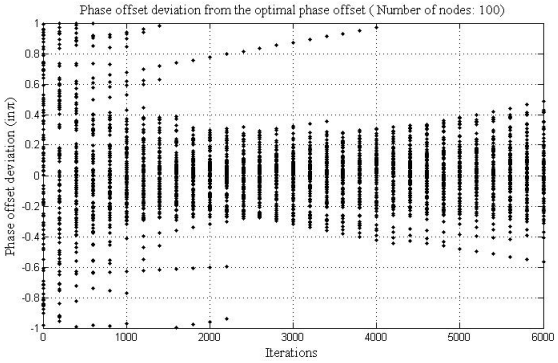
(b) Transmit nodes move at 2 m/sec following in a random-walk model



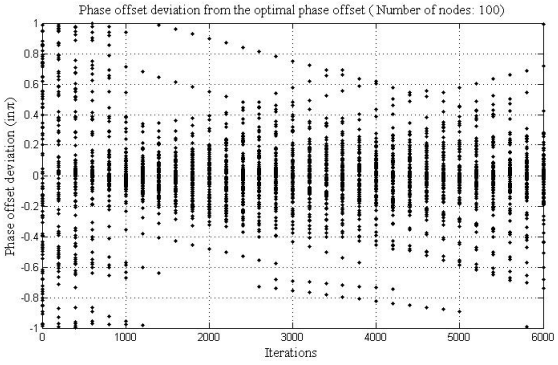
(c) Receiver node moving at 5 m/sec following a random-walk model



(d) Transmit nodes moving at 5 m/sec following a random-walk model



(e) Nodes move with 30 m/s in a linear mode



(f) Nodes move with 60 m/s in a linear mode

Figure 8.31: Performance of two approaches to distributed adaptive transmit beamforming for wireless sensor networks in a Matlab-based simulation environment

speed of 5 m/sec is easily supported (cf. figure 8.31(c)).

In the figure, the standard deviation σ is about 0.03 π. Figure 8.31(d) depicts the phase deviations when the receiver is static and the transmit nodes are moving. In this case, a standard deviation of $\sigma = 0.22\pi$ is achieved after 6000 iterations.

Concluding, we remark that the numeric method enables well higher movement speeds as well as an improved synchronisation performance. The maximum relative movement speed for the linear movement model is about 60 m/sec. Figure 8.31(f) depicts deviations of phase offsets.

The standard deviation in this case is $\sigma = 0.18\pi$

8.4.2 Consideration of multiple receivers

Nicht prüfungsrelevant (WS 09/10).

8.4.3 Increase of Population size – On the use of crossover

Nicht prüfungsrelevant (WS 09/10).

8.4.4 Receive beamforming

Nicht prüfungsrelevant (WS 09/10).

Index

List of Tables

List of Figures

Bibliography

[1] Culler, D., Estrin, D., Srivastava, M.: Overview of sensor networks. IEEE Computer **37**(8) (2004) 41–49

[2] Zhao, F., Guibas, L.: Wireless Sensor Networks: An Information Processing Approach. Morgan Kaufmann, Los Altos, CA (2004)

[3] Norman, D.: The invisible computer. MIT press (1999)

[4] Butera, W.J.: Programming a paintable computer. PhD thesis, Massachusetts Institute of Technology (2002)

[5] Pillutla, L., Krishnamurthy, V.: Joint rate and cluster optimisation in cooperative mimo sensor networks. In: Proceedings of the 6th IEEE Workshop on signal Processing Advances in Wireless Communications. (2005) 265–269

[6] Scaglione, A., Hong, Y.W.: Opportunistic large arrays: Cooperative transmission in wireless multihop ad hoc networks to reach far distances. IEEE Transactions on Signal Processing **51**(8) (2003) 2082–2092

[7] Sendonaris, A., Erkop, E., Aazhang, B.: Increasing uplink capacity via user cooperation diversity. In: IEEE Proceedins of the International Symposium on Information Theory (ISIT). (2001) 156

[8] Laneman, J., Wornell, G., Tse, D.: An efficient protocol for realising cooperative diversity in wireless networks. In: Proceedings of the IEEE International Symposium on Information Theory. (2001) 294

[9] Hong, Y.W., Scaglione, A.: Critical power for connectivity with cooperative transmission in wireless ad hoc sensor networks. In: IEEE Workshop on Statistical Signal Processing. (2003)

[10] Hong, Y.W., Scaglione, A.: Energy-efficient broadcasting with cooperative transmission in wireless sensor networks. IEEE Transactions on Wireless communications (2005)

[11] Jayaweera, S.K.: Energy analysis of mimo techniques in wireless sensor networks. In: 38th conference on information sciences and systems. (2004)

[12] del Coso, A., Sagnolini, U., Ibars, C.: Cooperative distributed mimo channels in wireless sensor networks. IEEE Journal on Selected Areas in Communications **25**(2) (2007) 402–414

[13] Sigg, S., Beigl, M.: Collaborative transmission in wsns by a (1+1)-ea. In: Proceedings of the 8th International Workshop on Applications and Services in Wireless Networks (ASWN'08). (2008)

[14] Sigg, S., Beigl, M.: Randomised collaborative transmission of smart objects. In: 2nd International Workshop on Design and Integration principles for smart objects (DIPSO2008) in conjunction with Ubicomp 2008. (2008)

[15] Mudumbai, R., Brown, D.R., Madhow, U., Poor, H.V.: Distributed transmit beamforming: Challenges and recent progress. IEEE Communications Magazine (2009) 102–110

[16] Mudumbai, R., Wild, B., Madhow, U., Ramchandran, K.: Distributed beamforming using 1 bit feedback: from concept to realization. In: Proceedings of the 44th Allerton conference on communication, control and computation. (2006) 1020–1027

[17] Barriac, G., Mudumbai, R., Madhow, U.: Distributed beamforming for information transfer in sensor networks. In: Proceedings of the third International Workshop on Information Processing in Sensor Networks. (2004)

[18] Mudumbai, R., Barriac, G., Madhow, U.: On the feasibility of distributed beamforming in wireless networks. IEEE Transactions on Wireless communications **6** (2007) 1754–1763

[19] Ochiai, H., Mitran, P., Poor, H.V., Tarokh, V.: Collaborative beamforming for distributed wireless ad hoc sensor networks. IEEE Transactions on Signal Processing **53**(11) (2005) 4110 – 4124

[20] Chen, W., Yuan, Y., Xu, C., Liu, K., Yang, Z.: Virtual mimo protocol based on clustering for wireless sensor networks. In: Proceedings of the 10th IEEE Symposium on Computers and Commmunications. (2005)

[21] Youssef, M., Yousif, A., El-Sheimy, N., Noureldin, A.: A novel earthquake warning system based on virtual mimo wireless sensor netwroks. In: Canadian conference on electrical and computer engineering. (2007) 932–935

[22] del Coso, A., Savazzi, S., Spagnolini, U., Ibars, C.: Virtual mimo channels in cooperative multi-hop wireless sensor networks. In: 40th annual conference on information sciences and systems. (2006) 75–80

[23] Jayaweera, S.K.: Energy efficient virtual mimo based cooperative communications for wireless sensor networks. IEEE Transactions on Wireless communications **5**(5) (2006) 984–989

[24] Laneman, J., Wornell, G.: Distributed space-time coded protocols for exploiting cooperative diversity in wireless networks. IEEE Transactions on Information theory **49**(10) (2003) 2415–2425

[25] Sendonaris, A., Erkip, E., Aazhang, B.: User cooperation diversity – part i: System description. IEEE Transactions on Communications **51**(11) (2003) 1927–1938

[26] Zimmermann, E., Herhold, P., Fettweis, G.: On the performance of cooperative relaying protocols in wireless networks. European Transactions on Telecommunications **16**(1) (2005) 5–16

[27] Cover, T.M., Gamal, A.A.E.: Capacity theorems for the relay channel. IEEE Transactions on Information Theory **525**(5) (1979) 572–584

[28] Kramer, G., Gastpar, M., Gupta, P.: Cooperative strategies and capacity theorems for relay networks. IEEE Transactions on Information Theory **51**(9) (2005) 3037–3063

[29] Scaglione, A., Hong, Y.W.: Cooperative models for synchronization, scheduling and transmission in large scale sensor networks: An overview. In: 1st IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing. (2005) 60–63

[30] Gupta, P., Kumar, R.P.: The capacity of wireless networks. IEEE Transactions on Information Theory **46**(2) (2000) 388–404

[31] Mitran, P., Ochiai, H., Tarokh, V.: Space-time diversity enhancements using collaborative communications. IEEE Transactions on Information Theory **51**(6) (2005) 2041–2057

[32] Simeone, O., Spagnolini, U.: Capacity region of wireless ad hoc networks using opportunistic collaborative communications. In: Proceedings of the International Conference on Communications (ICC). (2006)

[33] Krohn, A., Beigl, M., Decker, C., Varona, D.G.: Increasing connectivity in wireless sensor network using cooperative transmission. In: 3rd International Conference on Networked Sensing Systems (INSS). (2006)

[34] Krohn, A.: Optimal non-coherent m-ary energy shift keying for cooperative transmission in sensor networks. In: 31st IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP). (2006)

[35] Hong, Y.W., Scaglione, A.: Cooperative transmission in wireless multi-hop ad hoc networks using opportunistic large arrays (ola). In: SPAWC. (2003)

[36] Brown, D.R., Prince, G., McNeill, J.: A method for carrier frequency and phase synchronization of two autonomous cooperative transmitters. In: sixth IEEE workshop on signal processing advances in wireless communications. (2005)

[37] Brown, D.R., Poor, H.V.: Time-slotted round-trip carrier synchronisation for distributed beamforming. IEEE Transactions on Signal Processing **56** (2008) 5630–5643

[38] Ozil, I., Brown, D.R.: Time-slotted round-trip carrier synchronisation. In: Proceedings of the 41st Asilomar conference on signals, signals and computers. (2007) 1781–1785

[39] Tu, Y., Pottie, G.: Coherent cooperative transmission from multiple adjacent antennas to a distant stationary antenna through awgn channels. In: Proceedings of the IEEE Vehicular Technology Conference. (2002) 130–134

[40] Mudumbai, R., Hespanha, J., Madhow, U., Barriac, G.: Scalable feedback control for distributed beamforming in sensor networks. In: Proceedings of the IEEE International Symposium on Information Theory. (2005) 137–141

[41] Mudumbai, R., Hespanha, J., Madhow, U., Barriac, G.: Distributed transmit beamforming using feedback control. IEEE Transactions on Information Theory ((In review))

[42] Seo, M., Rodwell, M., Madhow, U.: A feedback-based distributed phased array technique and its application to 60-ghz wireless sensor network. In: IEEE MTT-S International Microwave Symposium Digest. (2008) 683–686

[43] Bucklew, J.A., Sethares, W.A.: Convergence of a class of decentralised beamforming algorithms. IEEE Transactions on Signal Processing **56**(6) (2008) 2280–2288

[44] Garlan, D., Siewiorek, D., Smailagic, A., Steenkiste, P.: Project aura: Toward distraction-free pervasive computing. IEEE Pervasive computing **4** (2002) 22–31

[45] Want, R., Schilit, B., Adams, N., Gold, R., Petersen, K., Goldberg, D., Ellis, J., Weiser, M.: An overview of the parctab ubiquitous computing experiment. In: IEEE personal communications. Volume 2. (1995) 28–43

[46] Gellersen, H.W., Beigl, M., Krull, H.: The mediacup: Awareness technology embedded in an everyday object. In Gellersen, H.W., ed.: 1th International Symposium on Handheld and Ubiquitous Computing (HUC99). Volume 1707 of Lecture notes in computer science., Springer (1999) 308–310

[47] Capra, L., Musolesi, M.: Autonomic trust prediction for pervasive computing. In: Proceedings of IEEE Workshop on Trusted and Autonomic Computing Systems 2006 (TACS'06). (2006)

[48] Ferscha, A., Holzman, C., Leitner, M.: Interfaces everywhere – interacting with the pervasive computer (2006) Half-day tutorial, 10th ACM International Conference on Intelligent User Interfaces (IUI 2006), Sydney, Australia.

[49] Want, R., Hopper, A., Falcao, V., Gibbons, J.: The active badge location system. In: ACM Transactions on Information Systems. Volume 1. (1992) 91–102

[50] Weiser, M.: The computer for the 21st century. In: Scientific American. Volume 3. (1991) 66–75

[51] Dourish, P.: What we talk about when we talk about context. In: Personal and Ubiquitous Computing. Volume 8. (2004)

[52] Schilit, B.N., Theimer, M.M.: Disseminating active map information to mobile hosts. In: IEEE Network. Volume 5. (1994) 22–32

[53] Schilit, B.N., Adams, N., Want, R.: Context-aware computing applications. In: IEEE Workshop on Mobile Computing Systems and Applications. (1994)

[54] Brown, P.J., Bovey, J.D., Chen, X.: Context-aware applications: from the laboratory to the marketplace. In: IEEE personal communications. Volume 4. (1997) 58–64

[55] Pascoe, J.: The stick-e note architecture: Extending the interface beyond the user. In: Proceedings of the 2nd International Conference on Intelligent user Interfaces. (1997) 261–264

[56] Dey, A.K., Abowd, G.D., Wood, A.: Cyberdesk: A framework for providing self-integrating context-aware services. In: Knowledge-Based Systems. Volume 11. (1998) 3–13

[57] Schmidt, A., Beigl, M.: There is more to context than location: Environment sensing technologies for adaptive mobile user interfaces. In: Workshop on Interactive Applications of Mobile Computing (IMC'98). (1998)

[58] Dey, A.K.: Providing architectural support for building context–aware applications. PhD thesis, Georgia Institute of Technology (2000)

[59] Mäntyjärvi, J.: Sensor-based context recognition for mobile applications. PhD thesis, VTT Technical Research Centre of Finland (2003)

[60] Henricksen, K.: A Framework for Cotnext-Aware Pervasive Computing Applications. PhD thesis, School of Information Technology and Electrical Engineering at the University of Queensland (2003)

[61] Lieberman, H., Selker, T.: Out of context: Computer systems that adapt to, and learn from, context. In: IBM Systems Journal. Volume 39. (2000) 617–632

[62] Fitzpatrick, A., Biegel, G., Cahill, S.C.V.: Towards a sentient object model. In: Workshop on Engineering Context-Aware Object Oriented Systems and Environments (ECOOSE). (2002)

[63] Pascoe, J.: Adding generic contextual capabilities to wearable computers. In: Proceedings of the second International Symposium on Wearable Computers. (1998) 92–99

[64] Dey, A.K., Salber, D., Abowd, G.D., Futakawa, M.: The conference assistant: comtining context-awareness with wearable computing. In: Proceedings of the third International Symposium on Wearable Computers. (1999) 21–28

[65] Kortuem, G., Segall, Z., Bauer, M.: Context-aware, adaptive wearable computers as remote interfaces to 'intelligent' environments. In: Proceedings of the second International Symposium on Wearable Computers. (1998) 58–65

[66] Chen, G.: Solar: Building A Context Fusion Network for Pervasive Computing. PhD thesis, Hanover, New Hampshire (2004)

[67] Schmidt, A.: Ubiquitous Computing – Computing in Context. PhD thesis, Lancaster University, UK (2002)

[68] Himberg, J.: From insights to innovations: data mining, visualisation, and user interfaces. PhD thesis, Helsinki University of Technology (2004)

[69] Himberg, J., Korpiaho, K., Mannila, H., Tikanmäki, J., Toivonen, H.: Time series segmentation for context recognition in mobile devices. In: Proceedings of the 2001 IEEE International Conference on Data Mining. (2001) 203–210

[70] Mäntyjärvi, J., Himberg, J., Huuskonen, P.: Collaborative context recognition for handheld devices. In: Proceedings of the first IEEE International Conference on Pervasive Computing and Communications (PerCom'03). (2003) 161–168

[71] Schilit, W.N.: A System Architecture for Context-Aware Mobile Computing. PhD thesis, Columbia University (1995)

[72] Dey, A.K., Abowd, G.D., Salber, D.: A context-based infrastructure for smart environments. In: Proceedings of the first International Workshop on Managing Interactions in Smart Environments (MANSE'99). (1999) 114–128

[73] Schmidt, A., Laerhoven, K.V., Strohbach, M., Friday, A., Gellersen, H.W.: Context acquisition based on load sensing. In: Proceedings of Ubicomp 2002, Lecture Notes in Computer Science. Volume 2498., Springer Verlag (2002) 333 – 351

[74] Jacob, R.J., Ishii, H., Pangaro, G., Patten, J.: A tangible interface for organising information using a grid. In: Conference on Human Factors in Computing Systems (CHI 2002). (2002)

[75] Abowd, G.D., Dey, A.K., Brown, P.J., Davies, N., Smith, M., Steggles, P.: Towards a better understanding of context and context-awareness. In: HUC '99: Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing, London, UK, Springer-Verlag (1999) 304–307

[76] Mayrhofer, R.M.: An Architecture for Context Prediction. PhD thesis, Johannes Kepeler University of Linz, Altenbergstrasse 69, 4040 Linz, Austria (2004)

[77] Mayrhofer, R.M., Radi, H., Ferscha, A.: Recognising and predicting context by learning from user behaviour. In: The International Conference On Advances in Mobile Multimedia (MoMM2003). Volume 171. (2003) 25–35

[78] Brooks, R.A.: Elephants don't play chess. In: Robotics and Autonomous Systems. Volume 6. (1990)

[79] Padovitz, A., Bartolini, C., Zaslavski, A., Loke, S.W.: Extending the context space approach to management by business objectives. In: 12th Workshop of the HP OpenView University Association. (2005)

[80] Padovitz, A., Loke, W.W., Zaslavsky, A.: On uncertainty in context-aware computing: Appealing to high-level and same-level context for low-level context verification. In: Proceedings of the 1st International Workshop on Ubiquitous Computing (IWUC'04). (2004) 62–72

[81] Padovitz, A., Loke, S.W., Zaslavsky, A., Burg, B.: Towards a general approach for reasoning about context, situations and uncertainty in ubiquitous sensing: Putting geometrical intuitions to work. In: 2nd International Symposium on Ubiquitous Computing Systems (UCS'04). (2004)

[82] Li, Y., Thai, M., Wu, W.: Wireless sensor networks and applications. Signals and Communication Technology. Springer (2008)

[83] Karl, H., Willig, A.: Protocols and Architectures for Wireless Sensor Networks. Wiley (2005)

[84] Rappaport, T.: Wireless Communications: Principles and Practice. Prentice Hall (2002)

[85] Sohrabi, K., Manriquez, B., Pottie, G.: Near-ground wideband channel measurements. In: Proceedings of the 49thvehicular technology conference. (1999) 571–574

[86] Kahn, J.M., Katz, R.H., Pister, K.S.J.: Next century challenges: Mobile networking for smart dust. In: Proceedings of the ACM MobiCom. (1999) 271–278

[87] Hoblos, G., Staroswiecki, M., Aitouche, A.: Optimal design of fault tolerant sensor networks. In: Proceedings of the IEEE International Conference on Control Applications. (2000) 467–472

[88] Bulusu, N., Estrin, D., Girod, L., Heidemann, J.: Scalable coordination for wireless sensor netwroks: Self-configuring localisation systems. In: Proceedings of the 6th IEEE International Symposium on Communication Theory and Application. (2001)

[89] Shen, C.C., Srisathapornphat, C., Jaikaeo, C.: Sensor information networking architecture and applications. IEEE Personal Communications **8**(4) (2001) 52–59

[90] Cerpa, A., Elson, J., Estrin, D., Girod, L., Hamilton, M., Zhao, J.: Habitat monitoring: Application driver for wireless communications technology. In: Proceedings of the ACM SIGCOMM Workshop on Data Communications in Latin America and the Caribbean. (2001)

[91] of IEEE 802.11, T.E.: Ieee standard for wireless lan medium access control (mac) and physical layer (phy) specifications (1997)

[92] Jerome, J.K.: Three men in a boat. Collector's Library (2005)

[93] Seybold, J.S.: Introduction to RF propagation. Wiley (2005)

[94] 3GPP: 3rd generation partnership project; technical specification group radio access networks; 3g home nodeb study item technical report (release 8). Technical Report 3GPP TR 25.820 V8.0.0 (2008-03) (March)

[95] Ohm, J.R., Lüke, H.D.: Signalübertragung – Grundlagen der digitalen und analogen Nachrichtenübertragungssysteme. Volume 10. Springer (2007)

[96] Feller, W.: An Introduction to Probability Theory and its Applications. Wiley (1968)

[97] Duda, R., Hart, P., Stork, D.: Pattern Classification. 2nd edn. Wiley Interscience (2001)

[98] Schwefel, H.P.: Direct search for optimal parameters within simulation models. Proceedings of the twelfth annual simulation symposium (1979) 91–102

[99] Holland, J.: Genetic algorithms and the optimal allocation of trials. SIAM, Journal of computing **3**(4) (1974) 88–105

[100] Schwefel, H.P.: Evolution and optimum seeking. Wiley-Interscience (1995)

[101] Fogel, L.J., Fogel, D.B.: A preliminary investigation on extending evolutionary programming to include self-adaptation on finite state. Informatica **18**(4) (1994)

[102] Koza, J.R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press (1992)

[103] Bäck, T.: An overview of parameter control methods by self-adaptation in evolutionary algorithms. Fundamenta Informaticae **35** (1998) 51–66

[104] Droste, S.: Efficient genetic programming for finding good generalizing boolean functions. In Koza, J.R., Deb, K., Dorigo, M., Fogel, D.B., Garzon, M., Iba, H., Riolo, R.L., eds.: Proceedings of the second Genetic Programming conference (GP 97), Morgan Kaufmann (1997) 82–87

[105] Droste, S., Heutelbeck, D., Wegener, I.: Distributed hybrid genetic programming for learning boolean functions. In Schoenauer, M., ed.: Proceedings of the 6th Parallel Problem Solving from Nature (PPSN VI). Volume 1917 of Lecture Notes in Computer Science (LNCS)., Springer (2000) 181–190

[106] Jansen, T., Wegener, I.: Evolutionary algorithms – how to cope with plateaus of constant fitness and when to reject strings of the same fitness. IEEE Transactions on Evolutionary Computation **5**(6) (2001) 589–599

[107] Wegener, I., Witt, C.: On the optimization of monotone polynomials by the (1+1)ea and randomized local search. In: Genetic and Evolutionary Computation Conference (GECCO). Number 2723 in Lecture notes in computer sciences (LNCS) (2005) 622–633

[108] Wegener, I., Witt, C.: On the analysis of a simple evolutionary algorithm on quadratic pseudo-boolean functions. Journal of Discrete Algorithms (3) (2005) 61–78

[109] Droste, S., Jansen, T., Wegener, I.: A rigorous complexity analysis of the (1+1) evolutionary algorithm for linear functions with boolean inputs. In: Proceedings of the third IEEE International conference on Evolutionary computation (ICEC 98), Piscataway, NJ, IEEE press (1998) 499–504

[110] Droste, S., Jansen, T., Wegener, I.: A rigorous complexity analysis of the (1+1) evolutionary algorithm for separable functions with boolean inputs. Evolutionary Computation **6**(2) (1998) 185–196

[111] Droste, S., Jansen, T., Wegener, I.: On the analysis of the (1+1) evolutionary algorithm. Theoretical Computer Science **276**(1-2) (2002) 51 – 81

[112] Sirkeci-Mergen, B., Scaglione, A.: 10. In: Randomized cooperative transmission in large scale sensor networks. A. Swami, Q. Zhao, Y. Hong, and L. Tong, Wiley (2007)

[113] Laneman, J., Wornell, G.: Energy-efficient antenna sharing and relaying for wireless networks. In: Proceedings of the IEEE Wireless Communication Networking Conference. (2000) 294

[114] Lanemann, J., Tse, D., Wornell, G.: Cooperative diversity in wireless networks: Efficient protocols and outage behaviour. Transactions on inforamtion Theory **50**(12) (2004)

[115] Nabar, R., Bölscskei, H., Wornell, G.: Fading relay channels: Performance limits and space-time signal design. IEEE Journal on Selected Areas in Communications **22**(6) (2004) 1099–1109

[116] Gastpar, M., Vetterli, M.: On the capacity of large gaussian relay networks. IEEE Transactions on Information Theory **51**(3) (2005) 765–779

[117] Ahlswede, R., Cai, N., Li, S.Y., Yeung, R.: Network information flow. IEEE Transactions on Information Theory **46**(4) (2000) 1204–1216

[118] Li, S.Y., Son, S., Stankovic, J.: Linear network coding. IEEE Transactions on Information Theory **49**(2) (2003) 371–381

[119] Woldegebreal, D.H., Karl, H. In: Network-Coding-Based Cooperative Transmission in Wireless Sensor Networks: Diversity-Multiplexing Tradeoff and Coverage Area Extension. Volume 4913 of Lecture Notes in Computer Science (LNCS). Springer-Verlag (2008) 141–155

[120] Scaglione, A., Hong, Y.W.: Opportunistic large arrays. In: IEEE International Symposium on Advances in Wireless Communications. (2002)

[121] Salhotra, A., Scaglione, A.: Multiple access in connectionless networks using cooperative transmission. In: Allerton Conference. (2003)

[122] Paulraj, A., Nabar, R., Gore, D.: Introduction to Space-Time Wireless communications. Cambridge University Press (2003)

[123] Shuguang, C., Goldsmith, A.: Energy-efficiency of mimo and cooperative mimo techniques in sensor networks. IEEE Journal on Selected Areas in Communications **22**(6) (2004) 1089–1098

[124] Alamouti, S.M.: A simple transmit diversity technique for wireless communications. IEEE Journal on select areas in communications **16**(8) (1998)

[125] Gastpar, M., Vetterli, M.: On the capacity of wireless networks: the relay case. In: Proceedings of the IEEE Infocom. (2002) 1577–1586

[126] Hagmann, W.: Network synchronisation techniques for satellite communication systems. PhD thesis, USC, Los Angeles (1981)

[127] Best, R.: Phase-Locked Loops: Design, Simulation and Applications. McGraw-Hill (2003)

[128] Krohn, A.: Superimposed Radio Signals for Wireless Sensor Networks. PhD thesis, Technical University of Braunschweig (2007)

[129] Bennett, W.: Introduction to signal transmission. McGraw-Hill (1971)

[130] Sigg, S., Masri, R.M.E., Beigl, M.: A sharp asymptotic bound for feedback based closed-loop distributed adaptive beamforming in wireless sensor networks. Transactions on mobile computing (2009 (submitted))