Independent and Dominating Sets in Wireless Communication Graphs

Tim Nieberg

University of Twente
The Netherlands

EWI/TW/DWMP
P.O. Box 217, 7500 AE Enschede
The Netherlands.

This thesis was edited with NEdit and typeset with LaTeX2e.

Keywords: Wireless Communication, WSN, Disk Graph, Independent Set, Dominating Set, PTAS, Local Algorithm.

Printed by Wöhrmann Print Service.

ISBN 90-365-2331-1

INDEPENDENT AND DOMINATING SETS IN WIRELESS COMMUNICATION GRAPHS

PROEFSCHRIFT

ter verkrijging van
de graad van doctor aan de Universiteit Twente,
op gezag van de rector magnificus,
prof. dr. W.H.M. Zijm,
volgens besluit van het College voor Promoties
in het openbaar te verdedigen
op donderdag 6 april 2006 om 16.45 uur

door

Tim Nieberg

geboren op 22 juli 1976

te Hamm.

Dit proefschrift is goedgekeurd door

Prof. Dr.	G.J.	Woeginger	(promotor)
Dr.	J.L.	Hurink	(assistent-promotor)

Acknowledgements

Working towards a PhD is a long road with many bends and turns. It is impossible to do on one's own, and I had help and advice from a lot of people both from the academic and the 'real' world. In the end, it is this help that kept me going on this road, and made this trip the trip of my life. I am grateful to all, and there are many people which I am sure I forget to mention here, although not intentionally.

First and foremost, I want to thank Johann Hurink without whom I would not be where I am today—literally in Twente, and figuratively—. He took and accompagnied me to new heights—literally to 12324 ft., and figuratively—. In the last four years, he has been a great supervisor and friend who always had a helping hand and word. Johann has the great gift of giving the type of critical advice that one does not want to hear, that sometimes results in a lot of additional work, but which improves everything beyond one's own.

Gerhard Woeginger unfortunately left Twente half-way through the four years. Nevertheless, he continued to supervise me, and I enjoyed the many discussions we had. These were usually short, but the new ideas and approaches kept me busy for a long time to follow thereafter.

The DWMP group has been my second home, and the 10 o'clock coffee gatherings were an integral part of my daily life. Tobias Brüggemann faced the same road as I did, and he has been a great fellow all along through the ups and downs. I also want to thank Hilbrandt Baarsma, Paul Bonsma, Hajo Broersma, Kees Hoede, Jacob-Jan Paulus, Gerhard Post, Walter Kern, Georg Still, and of course Dini, the secret soul of the group.

Most of the time I spent at the Embedded Systems cluster, in the office with Supriyo Chatterjea, Stefan Dulman, Lodewijk van Hoesel, Tjerk Hofmeier, and Jian Wu. Thank you guys, for the great time. I also want to express my gratitude to Paul Havinga, and everyone from the various projects at the group, including EYES and SmartSurroundings.

The Distributed Computing Group of Roger Wattenhofer at the ETH Zurich welcomed me and gave me a great time doing research there, even though I must be the worst player at the 'Töggeli-Automat' they ever saw.

I want to thank the members of my graduation committee, and everyone that helped during the process of writing and preparing this thesis.

There's life outside the university. Although there is no clear boundary—you always take home problems—there are many friends and family that fortunately have no clue what you are actually researching. I want to thank my parents, Emma and Peter, and my brother and sister, Kevin and Jill Kristin. They are always there when you need them, and they ensured that I kept both feet on the ground.

Elli and Olaf are great friends to have, and during the last years the many trips to Hanover and Denmark were always accompagnied by lots of fun and laughter.

These four years would not have been possible without the unconditional support of Kristina. I want to thank her simply for always being there, in good and in bad, in relaxing and in stressful times.

I can still fix my bike—in fact, now even cars—, which has always been the measure to determine whether I should go on, or quit. To go on it is then, and I do so with all the joy I can imagine, awaiting the wonders that are yet to come...

Enschede, April 2006
Tim Nieberg

Abstract

Wireless ad hoc networks are advancing rapidly, both in research and more and more into our everyday lives. Wireless sensor networks are a prime example of a new technology that has gained a lot of attention in the literature, and that is going to enhance the way we view and interact with the environment. These ad hoc and sensor networks are modeled by communication graphs which give the communication links between some devices or nodes equipped with wireless transceivers or radios.

The nature of wireless transmissions does not lead to an arbitrary network topology, but creates a network with certain properties. These properties include the important structure of bounded growth. In this thesis, we look at the structures and resulting optimization problems of independent and dominating sets in on graphs that model wireless communication networks.

Independent and dominating sets are prominently used in the efficient organization of large-scale wireless ad hoc and sensor networks. In a communication graph, an independent set consists of vertices that cannot communicate with one another directly. Such a set is commonly used in clustering strategies, e.g. to obtain a hierarchical view of the network. A dominating set is given by a set of vertices so that every vertex in the graph is either in this set, or adjacent to a vertex from this set. Dominating sets of small cardinality are frequently used for backbone structures in communication networks, e.g. to obtain efficient multi-hop routing protocols.

For the optimization problems of seeking independent sets of large cardinality or weight (Maximum Independent Set problem), and seeking dominating sets of small cardinality (Minimum Dominating Set problem) on graphs of polynomially bounded growth, we present and discuss polynomial-time approximation schemes (PTAS). The algorithms presented are robust in the sense that they accept an undirected graph, and return a desired solution or a certificate that shows that the instance does not satisfy the structural properties of a wireless communication graph.

Wireless ad hoc and sensor networks usually lack a central control instance. Local, distributed algorithms are designed to operate in such a scenario. We propose a fast local algorithm that constructs a so-called maximal independent set, which is also dominating. We also extend the ideas behind the centrally executed PTAS towards a local approach. This gives a distributed approximation scheme for the Maximum Independent Set and Minimum Dominating Set problems on wireless communication graphs.

The second part of the thesis is devoted to an application of independent and dominating sets in a wireless sensor network. We present the design of an energy-efficient communication strategy for low-resource, large-scale wireless networks that is based on an integrated approach stemming over various layers of the communication stack. The transmission of data packets in this scheme is done in a scheduled manner, thus reducing the number of collisions due to simultaneous transmissions. The approach, called EMACs, also includes the creation and maintenance of a connected backbone in the network that is used for implicit sleep-state scheduling of the nodes to conserve additional energy. Some results obtained from a practical implementation of this scheme indicate that the EMACs communication scheme improves the network lifetime compared to existing schemes for wireless sensor networks.

This thesis answers and contributes to several open questions from the literature. The characterization of wireless communication graphs by the bounded growth property includes geometrically defined Unit Disk Graphs. By giving a PTAS that does not exploit geometric information for the Maximum Independent and Minimum Dominating Set problems on these graphs, we give a positive answer to the open problem of the existence of a PTAS for Unit Disk Graphs without geometric representation. Local, distributed maximal independent set computation has received a lot of attention in the literature, and a fast, i.e. poly-logarithmic distributed time approach is a longstanding open problem for communication networks in general. At least for wireless communication networks, we present the first such fast, local approach.

Samenvatting

Draadloze *ad-hoc* netwerken ontwikkelen snel, zowel in het wetenschappelijk onderzoek als in ons dagelijks leven. Draadloze *sensor* netwerken zijn een voorbeeld van een nieuwe technologie die veel aandacht in de literatuur krijgt, en zullen de manier waarmee wij naar onze omgeving kijken drastisch veranderen. Deze ad-hoc en sensor netwerken worden door communicatiegrafen gemodelleerd, waarbij de punten of knopen van deze graaf de apparaten, en de lijnen tussen de knopen de (directe) communicatieverbindingen tussen de apparaten weergeven. De centrale begrippen in dit proefschrift zijn *onafhankelijke* en *dominerende* verzamelingen in communicatiegrafen. Deze structuren zijn bepalend voor de efficiënte organisatie van draadloze ad-hoc en sensor netwerken. De aard van draadloze uitzending en ontvangst leidt niet tot een willekeurige topologie van het netwerk, maar creëert een graaf met bepaalde eigenschappen. Deze eigenschappen omvatten de belangrijke eigenschap van *beperkte groei.* In dit proefschrift bestuderen we structuren en resulterende optimaliseringsproblemen van onafhankelijke en dominerende verzamelingen in grafen van beperkte groei.

In een communicatiegraaf bestaat een onafhankelijke verzameling uit punten die niet rechtstreeks met elkaar kunnen communiceren. Zo'n verzameling wordt vaak in clusteringsstrategieën gebruikt; hierbij worden knopen gegroepeerd om bijvoorbeeld een hiërarchisch overzicht van het netwerk te verkrijgen.

Een dominerende verzameling is een verzameling punten zodat ieder punt van de communicatiegraaf ofwel tot die verzameling behoort, ofwel rechtstreeks communiceert met een punt uit die verzameling. Dominerende verzamelingen van kleine grootte zijn van belang voor ruggengraat structuren in communicatienetwerken; zo worden ze bijvoorbeeld gebruikt om efficiënte multi-hop routeringsprotocollen te verkrijgen.

Het eerste deel van het proefschrift onderzoekt de optimaliseringsproblemen Maximum Independent Set probleem (onafhankelijke verzamelingen van grote cardinaliteit of gewicht) en Minimum Dominating Set probleem (dominerende

verzamelingen van kleine cardinaliteit) op grafen van polynomiaal beperkte groei. Voor beide problemen construeren wij een polynomiale tijd approximatie schema (PTAS). De resulterende algoritmes van het PTAS zijn robuust in de zin dat zij werken op alle grafen, en ofwel een gewenste oplossing, ofwel een certificaat teruggeven dat aantoont dat aan de structurele eigenschappen van een communicatiegraaf niet wordt voldaan.

Gewoonlijk missen draadloze ad-hoc en sensor netwerken een centrale aansturing. Lokale, gedistribueerde algoritmes worden daarom ontworpen, die in deze situatie de communicatie kunnen verzorgen. Wij behandelen een snel lokaal algoritme, dat een maximaal onafhankelijk verzameling bepaalt, welke tegelijk een dominerende verzameling is. Wij breiden het idee achter het centraal uitgevoerde PTAS uit naar een lokale benadering. Dit geeft een gedistribueerd approximatie schema voor het Maximum Independent Set probleem en het Minimum Dominating Set probleem op draadloze communicatiegrafen.

Het tweede deel van het proefschrift is gewijd aan een toepassing van de onafhankelijke en dominerende verzamelingen in draadloze sensor netwerken. Voor functioneel beperkte draadloze netwerken ontwerpen we een energie-efficiënte communicatiestrategie. Deze is gebaseerd op een geïntegreerde benadering, die over verschillende lagen van de communicatiestapel werkt. De verzending van berichten in deze strategie geschiedt op een gecontroleerde wijze, waardoor het aantal conflicten tengevolge van gelijktijdige uitzendingen verminderd. Deze aanpak, EMACs, omvat ook de creatie en het onderhoud van een verbonden ruggengraat in het netwerk, dat voor impliciete slaap-toestand planning van de knopen wordt gebruikt. Resultaten die door een praktische uitvoering van deze aanpak verkregen worden, tonen aan dat EMACs de levenstijd van het netwerk verbetert, vergeleken met bestaande methodes voor communicatie in draadloze sensor netwerken.

Dit proefschrift beantwoordt enkele open vragen uit de literatuur. De karakterisering van communicatiegrafen door de beperkte groei, omvat de geometrisch gedefinieerde Eenheidsschijfgrafen (Unit Disk Graphs). Door het geven van een PTAS dat geen geometrische informatie nodig heeft voor de Maximum Independent Set en Minimum Dominating Set problemen op deze grafen, geven wij een positief antwoord op het open probleem van het bestaan van een PTAS voor Eenheidsschijfgrafen zonder geometrische voorstelling.

Ook lokaal, verdeelde berekening van maximale onafhankelijke verzamelingen heeft veel aandacht in de literatuur gekregen, en een polylogaritmisch gedistribuëerde tijd constructie algoritme is een oud open probleem voor algemene communicatienetwerken. Voor draadloze communicatienetwerken geven wij voor het eerst zo'n snelle, lokale benadering.

Contents

Chapter 1

Introduction

Wireless networks are advancing more and more into our everyday lives. Cellular telephony networks or wireless local area networks are two of many examples, and there are new networks with new applications coming almost every day. As the wireless devices get smaller and more embedded, up to a point where they are no longer directly present to one's eyes, they precede into the background of our lives and perform tasks unattended and without much interaction of users. Numerous small devices, deeply embedded in the environment, sense and interact with the environment and form a collaborative network by means of wireless multi-hop communication. Such a network realizes the vision of *Ubiquitous Computing* by creating a smart environment.

This thesis explores basic structures for efficient communication and organization in large-scale wireless networks. Such large networks usually require a certain structural organization to operate efficiently, and the most prominent structures behind many approaches are based on *independent* and *dominating* sets.

An example for efficient use of resources are cluster based control structures. They allow for a hierarchical view of the network which decreases the complexity of the underlying network, and can make a highly dynamic network appear more static. Clustering in wireless networks is usually done by grouping nearby nodes together, which are then controlled by a designated node called clusterhead. On the one hand, many clustering schemes work by identifying these control nodes so that they form an independent set. On the other hand, many cluster based applications benefit from an independent set based clustering structure.

Dominating sets play an important role, e.g. in global flooding to alleviate the so-called broadcast storm problem. A message broadcast only from a small size dominating set is an efficient way to ensure that all nodes in the network are reached, both in terms of energy and interference. Many ad-hoc routing algorithms rely on flooding to find routes in a dynamic wireless network, and can thus benefit from such a dominating set structure.

In this introduction chapter, we present wireless networks, main components of these, especially the wireless communication devices or nodes, and applications and challenges in these.

During the research that lead to this thesis, we were especially interested in networks composed of small, low-resource nodes, such as given in wireless ad-hoc and sensor networks. While most of the work presented throughout this thesis applies equally well to other wireless communication networks, the applications we present focus on these network settings which are now introduced.

1.1 Wireless Ad-Hoc and Sensor Networks

A wireless ad-hoc network consists of a collection of autonomous devices, called nodes, that are equipped with processing and wireless communication capabilities. These nodes can communicate with one another by sending and receiving messages in small data packets.

A broadcast sent by a node can be picked up by all other nodes within radio coverage (unless it is interfered by other transmissions at the same time). The communication is either directly when the nodes can receive each other's transmissions, or via intermediate relay nodes that forward messages when sender and recipient are outside each other's radio ranges. This creates a multi-hop communication networking structure referred to as ad-hoc network.

Such an ad-hoc network does not rely on any fixed network structure, that is, there is no central server or common infrastructure. Thus, the network usually has to be organized and coordinated by the devices themselves. Furthermore, the communication network has a dynamic topology. The wireless devices may be mobile, and the communication protocols thus have to adapt to a changing topology. Also, contrasted to wireline communication networks, the capacity of the wireless links is far lower.

Nodes can be of different types, i.e. with different computation, storage, and communication capabilities, creating a heterogeneous network. Usually, the nodes run on batteries. Since ad-hoc networks rely on forwarding data packets sent by other nodes, power consumption becomes a critical issue in most networks.

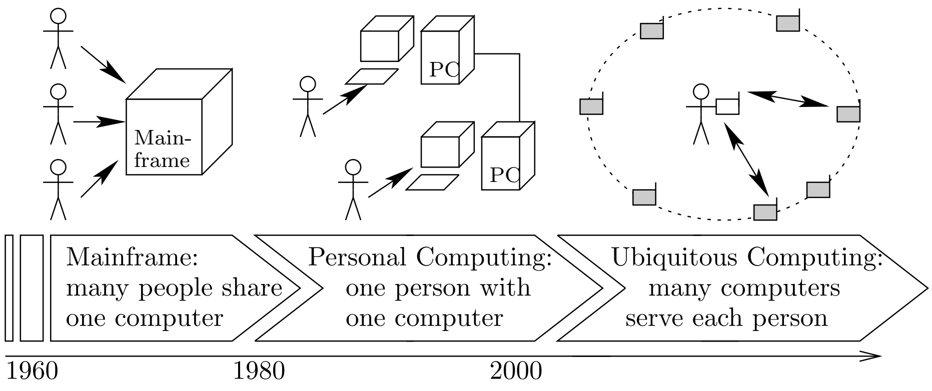


Figure 1.1: Evolution of computing.

A particular type of wireless ad-hoc network is given by a *wireless sensor network* (WSN). The purpose of such a network is physical environment monitoring. Each node is equipped with one or more sensors, whose readings are transported through the network towards special nodes called sinks that extract the data from the network, and make it available to the user.

WSNs are envisioned to consist of many nodes, their number ranging in the hundreds and thousands. The nodes themselves are very small in every respect, especially in form and in function. The resource constraints of the usually battery-operated devices, and the scale of the resulting networks, have severe implications in almost every aspect involved in designing and operating such a sensor network. Simple strategies to organize and operate the resulting network are called for. In this thesis, we also present practical approaches for efficient organization and operation of a WSN.

Not only in environmental monitoring, wireless sensor networks will enhance the usability of appliances. They provide condition based maintenance in applications spanning many areas including home, office, health-care, factory, vehicle, and metropolitan scenarios. The technology from this research field enables data collection, transportation, and in-network processing in a variety of situations, including context-aware personal assistance, home security, medical monitoring, machine failure diagnosis, traffic surveillance, and many more.

The above mentioned areas are known under many different names, for example Ubiquitous Computing, Ambient Intelligence, Smart Surroundings, Pervasive Computing, or simply the Third Paradigm of Computing (Figure 1.1, adapted from [59]). Efficient communication strategies in wireless ad-hoc and sensor networks are the enabling technologies for this new era.

1.1.1 Energy-Efficient Sensor Networks (EYES)

Most of the work presented in this thesis stems from research done on wireless sensor networks as part of the EYES project [17]. EYES is a three year European research project on self-organizing, and collaborative energy-efficient wireless sensor networks, running from 2002 till 2005. Its statement, given below, briefly characterizes the main challenges addressed by the project.

> The vision of ubiquitous computing requires the development of devices and technologies, which can be pervasive without being intrusive. The basic components of such a smart environment will be small nodes with sensing, computing, and wireless communications capabilities, able to organize flexibly into a network for data collection and delivery. Realizing such a network presents very significant challenges, especially at the architectural and protocol/software level. Major steps forward are required in the fields of communications protocol, data processing, and application support.
>
> Although sensor nodes will be equipped with a power supply (battery) and an embedded processor that makes them autonomous and self-aware, their functionality and capabilities will be very limited. Therefore, collaboration between nodes is essential to deliver smart services in a ubiquitous setting. In this project we investigate new algorithms for networking and distributed collaboration, and evaluate their feasibility through experimentation. These algorithms will be key for building self-organizing and collaborative sensor networks that show emergent behavior and can operate in a challenging environment where nodes move, fail, and energy is a scarce resource. [17]

The EYES project addresses the convergence of distributed information processing, wireless communications, and mobile computing.

One of the key approaches taken within the EYES project is to improve the functional lifetime of the sensor network using energy-efficient network protocols and routing techniques, and dynamic power management techniques. In this context, this thesis provides both theoretical background, as well as practical approaches, for efficient wireless communication structures and strategies.

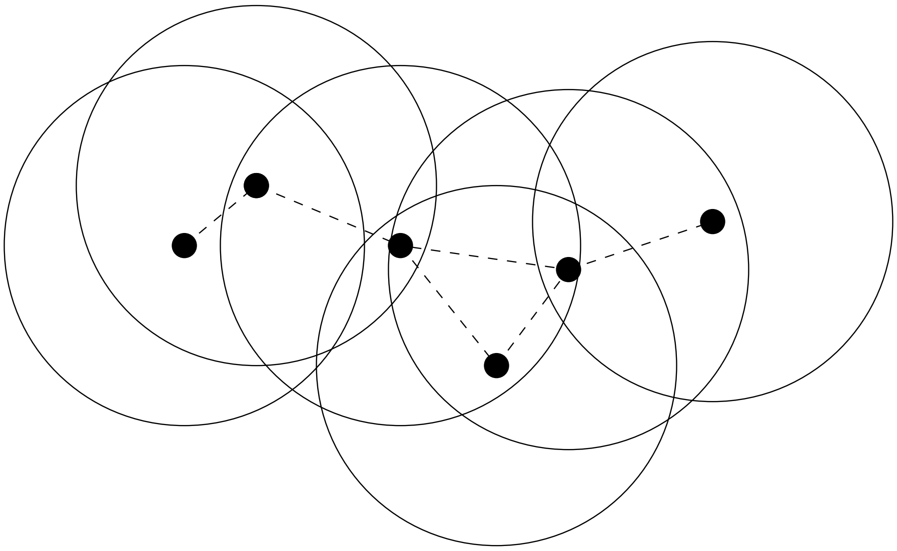


Figure 1.2: Unit Disk Graph model for wireless communication networks.

1.2 Wireless Communication

In the wide area of wireless ad-hoc and sensor networks, in this thesis, we focus on graphs and networks that are created by wireless communication between the nodes of an ad hoc network. Taking a closer look at wireless devices, and the communication between them, we present approaches that take the special structure of a wireless communication network into account.

Generally speaking, communication in a network is modeled by a graph $G = (V, E)$. In our case, the set V of vertices represents the wireless nodes, and the edges represent the possible direct communication between two nodes. In other words, there is an edge between two vertices if a transmission from one node can be received by the other. In the following, we will use the term *node* for the wireless devices, and the term *vertex* to denote them in graphs.

The ability to communicate between two nodes and, thus, the presence of an edge between two vertices depends on the position of the corresponding nodes, and their transmission power. Generally speaking, in wireless communication, two nodes that are not too far apart can communicate directly, while far separated nodes cannot. This fact is used to argue about the structure of a wireless

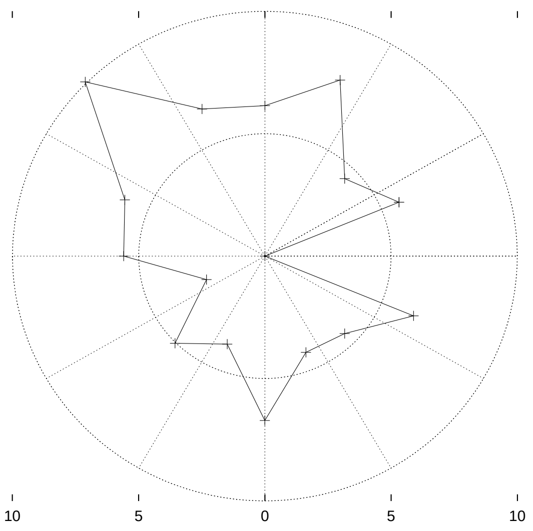


Figure 1.3: 80% message packet reception with respect to distance [m] (Office environment, node fixed to concrete pillar).

communication network from a graph-theoretic point of view. The goal is to represent the network by nodes and the geometric regions where their signal can be received.

A simple model for this would be a so-called Unit Disk Graph created by a set of equal-diameter disks representing the transmission areas. A message can be received by another node which is inside this area. An example of a Unit Disk Graph is given in Figure 1.2. However, this model is somewhat too simplistic to represent the real world.

In order to gain some understanding of realistic communication characteristics, we turn to a real world testbed. The following example from the EYES project gives some measurements obtained using sensor node prototypes. In Figure 1.3, the "shape" of a transmission area of a node in an office environment is presented. A message packet sent by a node placed at the origin will be received correctly in more than 80% of the cases when the receiving node is inside the given area. Clearly, the shape does not give a disk. In this thesis, later on, we present a more general concept of bounded growth that allows us to capture these effects, and capture the essence of a wireless communication network graph model.

Next to communication in wireless networks, interference during simultaneous transmissions is also of great importance. This interference can be modeled by a conflict graph, and this graph has a similar structure as the wireless communication graph. When presenting the models for wireless communication networks, we consider both ways to define such a network.

1.3 Efficient Network Organization

With respect to wireless communication networks, in the following, we are interested in the problems concerning the creation of subsets of nodes in the network.

The approaches we are considering in this thesis are for the creation of subsets of the nodes with certain properties:

- **Independence**:

  A subset of nodes is called *independent* if no two nodes in this subset are connected.

- **Domination**:

  A subset of nodes is called *dominating* if every node in the network is contained in this subset, or is connected to a node in it.

Nodes in an independent set do not interfere each other during simultaneous transmissions, and nodes in a dominating set can be used to efficiently reach the entire network by broadcasts from only these nodes. Note that a subset of nodes can satisfy both properties. Such a subset is called *maximal* independent set.

There are many different perspectives from which we may look at the mentioned structures and their construction by an algorithm. Considering the cardinalities of the respective subsets, we obtain optimization problems. Looking at the in-network creation of such subsets, it is already challenging to create these subsets by a locally executed algorithm in each node even when not considering cardinality explicitely. In the following chapters, we discuss independent and dominating sets from various angles, and in relation to graphs that model wireless communication networks.

Besides the above basic versions of the considered problems, there also exist weighted versions: each node is given a weight, and the goal is then to find a structure of minimum or maximum weight. A weight may correspondent to the capabilities of the nodes to perform additional duties. The weights can be determined taking into consideration aspects like residual energy of a node, its memory and processing power, and local figures like number of neighbors or mobility indicators. Usually, these weights are locally computed in each node, and they depend on the application the resulting structure is created for.

Independent and dominating sets in wireless networks are used in a variety of different applications, especially at the lower layers directly involved with communication strategies, and directly dealing with the topology of the communication network. Some of these applications are presented next.

- **Clustering**

  Clustering is often used in large-scale networks to reduce the complexity. On a topology level, clustering is done by grouping nodes inside a certain area, which are then controlled by a designated node called clusterhead. These clusterheads are chosen so that they satisfy both the independence and domination property. This results in a well-distributed structure that covers the entire network.

  For example, in [24], the use of independent set based clustering for the allocation of bandwidth and channels to support multimedia traffic is proposed. The question of cluster sizes, i.e. the number of nodes in each cluster, and their fair distribution are addressed in [45].

- **Routing and Flooding**

  In a dynamic multi-hop communication network, routing schemes that adapt to the changing topology are required. Each individual node is neither able to store the entire topology information, nor to keep updated information about the changes in the network topology. There are many on-demand routing algorithms designed for mobile ad-hoc networks. These algorithms create and sometimes maintain routes for data packets to be sent via relay nodes through the network to reach a destination. When trying to establish a new route, that is, trying to find the destination in the network together with a path leading there, these schemes have to rely on flooding the network to do so. Examples of such ad-hoc routing schemes are Dynamic Source Routing (DSR, [31]) and Ad-Hoc On-Demand Distance Vector routing (AODV, [52]), and many improvements based on these schemes.

  Basic, network-wide flooding causes the broadcast storm problem [43], resulting in excessive contention and collisions, i.e. a large communication protocol overhead. Using a dominating set of small size as a virtual backbone to propagate flooding messages overcomes this problem, and greatly reduces the number of messages needed, and thus the protocol overhead as well [2, 13, 55].

- **Sleeping Patterns**

  Wireless sensor networks are expected to be in operation for a long period of time, running on batteries. For example in event detection applications, there are long periods of inactivity, which allow the nodes to follow a sleep-state schedule where the radio hardware can be turned off for certain time intervals to save energy. However, the network has to remain functional, and most importantly connected so that a detected event can successfully be reported to the sink.

  An example of a connected dominating set based solution that exploits the redundancy present in the network is presented in Chapter 6.

Also from a theoretical point of view, independent and dominating sets with respect to graphs that model wireless communication networks are of outstanding interest. For example, in distributed computing, independent sets capture the important notion of symmetry breaking in a simple statement. The theoretical aspects of these structures are discussed, and introduced in detail in the following Chapter 2.

1.4 Contributions

In this thesis, we look at independent and dominating sets in wireless communication networks, both from a theoretical background and from practical implementations.

For wireless ad-hoc and sensor networks, many graphical models are proposed in the literature to represent the communication links in such networks. We review these mostly geometrically defined models, and relate them to a unified structural property called *bounded growth* which is common in all of these models. Bounded growth is defined without geometric information, which allows us to work with wireless communication networks based on adjacency information only.

We consider the optimization problems of creating large independent sets, and small dominating sets in wireless communication graphs. These problems are hard to solve to optimality, and we therefore consider approximative solutions. These are given by a *polynomial-time approximation scheme* (PTAS) that creates a near-optimal solutions, that is, solutions with an error of $(1+\varepsilon), \varepsilon > 0$, in efficient run time.

The question whether there exists a PTAS for the Maximum Independent Set and Minimum Dominating Set problems in Unit Disk Graphs that work without explicitely exploiting a geometric representation was an open problem. It was expressed since the first appearance of approximation schemes for these problems on geometric graphs in [29] (1985). We eventually give a positive answer to this question by presenting an approach that yields a PTAS for graphs with the polynomially bounded growth property, which includes Unit Disk Graphs.

Additionally, the approaches that give the approximation schemes for these problems can be extended towards robust algorithms in the sense that they accept any graph as input, and either return a $(1+\varepsilon)$-approximate solution, or a certificate showing that the input graph does not satisfy the bounded growth property of a wireless communication graph.

Wireless ad-hoc networks lack central control. Locally executed algorithms are thus of great importance. In the area of distributed computing, the problem of fast, i.e. poly-logarithmic time, local construction of maximal independent sets is a longstanding open problem. For the class of graphs with bounded growth, we present such a fast approach, again without relying on any geometric information. So, for bounded growth graphs, there is a positive answer to this question, as well.

The presented local, distributed algorithm yields a PTAS for the above optimization problems. By extending the ideas behind the centrally executed PTAS, we obtain the first *distributed approximation schemes* for the Maximum Independent Set and Minimum Dominating Set problems on wireless communication graphs.

In addition to these contributions on the theoretical background of independent and dominating sets in wireless communication graphs, we also discuss a practical implementation of an efficient communication protocol for wireless sensor networks. We show that a cross-layer approach for communication in WSNs is a very efficient way to achieve the goal of long network lifetime running on batteries.

1.5 Structure of the Thesis

In the first part of this thesis, in Chapters 2 – 5, we focus on the theoretical background of wireless communication graphs, and algorithmic approaches both from a global and local point of view.

In Chapter 2, the definitions of independent and dominating sets in networks are given together with basic properties of these structures. There, we focus on the structures and resulting problems from a graph-theoretic point of view and present the general models for computation and communication in networks. In order to make statements about the efficiency of the approaches discussed in this thesis, we provide a theoretical base for algorithms running in each wireless device (Section 2.1), and for communication patterns that allow these devices to collaborate by exchanging messages (Section 2.3). These models allow to abstract away from actual hardware and communication strategies, and therefore allow to give results that are independent of these. Chapter 2 is—so to say—the second half of the introduction.

Wireless networks, created by the possible direct communication links between the radios of several devices, result in specially structured graphs. These are presented in Chapter 3. Depending on the assumptions on the propagation of the radio waves being emitted by a wireless device, different graph models emerge, most of which are geometrically defined. However, all these models have a common structural property called bounded growth which is defined and discussed.

In Chapter 4, we consider the optimization problems of constructing independent sets of large size or weight, and dominating sets of small cardinality. For these optimization problems on wireless communication graphs, we present polynomial-time approximation schemes (PTASs) that are independent of geometric information. Robustness of the approaches is also discussed.

The focus in Chapter 5 is on local, distributed algorithms that are required in wireless ad-hoc networks. We present a fast, local approach for the construction of a maximal independent set in graphs of bounded growth. This approach, together with the ideas of the preceding chapter, is extended towards distributed approximation schemes that compute near-optimal independent and dominating sets using local information exchange only.

Chapter 6 is concerned with an energy-efficient communication strategy for wireless sensor networks. There, we focus on a practical application of an independent and dominating set. After shortly presenting the architecture of a WSN, we present a cross-layer approach that locally schedules collision-free communication, and creates a backbone that is used for the communication in the network. The approach is called EMACs (EYES Medium Access Control Scheme).

In Chapter 7, we conclude the thesis with a short summary of the key results and approaches.

Chapter 2

Definitions and Preliminaries

This chapter gives a detailed introduction into the problems considered in this thesis, as well as a description of the underlying models of computation and communication. Basic definitions are presented, and an overview of complexity classes and optimization problems in general is given.
In wireless networks, in-network computation and communication is crucial. In order to capture the characteristics of this paradigm, the $\mathcal{LOCAL}$ message passing model is introduced as a framework for distributed algorithms in wireless, ad-hoc networks.
The main structures of this thesis, independent and dominating sets in a graph, are defined in Section 2.2. Furthermore, basic properties, resulting problems and relations between these are discussed. Also, a simple greedy approach creating an independent and dominating set is presented.

This chapter provides the theoretical base of the thesis. We start by introducing a formal description of a "problem", an algorithm to solve it, a notion of the quality of a solution to it, and a notion of time it takes an algorithm to compute such a solution. In short, Section 2.1 sketches the basics of complexity theory as required for the rest of this thesis.

In Section 2.2, we define the structural properties of *independence* and *domination* for a subset of vertices in an undirected graph. The simple structure of a *maximal* independent set, which is both independent and dominating, is explained. We also introduce and discuss the resulting optimization problems, i.e. computing a subset with such a property of largest or smallest cardinality possible. These problems are the Maximum Independent Set and Minimum Dominating Set problem.

Large scale wireless communication networks like WSNs not only lack central control, but also each individual node lacks the capacity to store information about the entire topology of the network. In order to sufficiently describe and reason about algorithms which run locally in each node of such networks, we introduce the $\mathcal{LOCAL}$ message passing model in Section 2.3. This model captures these basic properties of wireless networking and in-network processing, while not being too concerned about the technical and hardware-specific details of communication.

In the last section of this chapter, a simple greedy strategy to create a maximal independent set, which is also a dominating set, is presented. This strategy serves as an important example also used later in this thesis, and we give both a centralized and local, distributed version of it.

2.1 Complexity and Approximability

In this section, we give some basic notions of problems, their complexity, and algorithms to solve them. A more complete overview can be found in [23]. In order to do optimization, and to analyze algorithms and complexity, we need to define the essence of optimization: a problem.

Definition 2.1. *A problem is given by a **problem description** Π, which is a pair $(\mathcal{I}, \mathcal{S})$ such that*

- *$\mathcal{I}$ denotes the set of instances of Π, and*
- *for each $x \in \mathcal{I}$, $\mathcal{S}(x)$ is the set of feasible solutions.*

We consider two types of problems called decision and optimization problems, which are given by refining the above general problem description.

Definition 2.2. *Let* Π *be a problem description whose set* $\mathcal{I}$ *of instances is partitioned into two sets* $\mathcal{Y}$ *and* $\mathcal{N} = \mathcal{I} \backslash \mathcal{Y}$*, called positive and negative instances as follows:*

$$x \in \mathcal{Y} \iff \mathcal{S}(x) \neq \varnothing.$$

Seeking an answer to the question of whether an instance $x \in \mathcal{I}$ *belongs to* $\mathcal{Y}$ *is called a* ***decision problem****.*

The aim of a decision problem is to recognize the positive instances, that is, those instances that have a feasible solution.

For *optimization problems*, we look at problem descriptions, where additionally each feasible solution is evaluated by an objective function as follows. For an instance $x \in \mathcal{I}$ together with a feasible solution $y \in \mathcal{S}(x)$, there is an objective function value (or measure) $f(x, y) \in \mathbb{N}$ of that solution.

For each instance, the objective function may take on different values for different feasible solutions. We are interested in solutions which minimize or maximize this value with respect to all feasible solutions for such an instance.

Definition 2.3. *In a problem description* Π*, for each pair of instance* $x \in \mathcal{I}$ *and feasible solution* $y \in \mathcal{S}(x)$*, let the objective function* $f(x, y)$ *assign a positive integer to this pair. The problem* $\Pi^{\min}$*:*

> *Given an instance* $x \in \mathcal{I}$*, find a solution* $y^* \in S(x)$ *such that for every* $y \in S(x)$*,*
>
> $$f(x, y^*) \leq f(x, y)$$
>
> *holds.*

is called a ***minimization problem****, and* y^* *is called an optimal solution for this problem.*

Analogously, a *maximization problem* is defined by finding a solution with maximum objective value over all feasible solutions. The question of minimization or maximization for an optimization problem is called *goal* of the problem. Note that in the above definition, the range of an objective function f can easily be relaxed to rational numbers.

It is easy to see that any optimization problem Π can be transformed in a straightforward way into a *corresponding decision problem.* This is done by looking at the following, modified set of feasible solutions in case of minimization.

Given an instance $x \in \mathcal{I}$ and a rational number τ, let

$$\mathcal{S}_\tau(x) := \{y \in \mathcal{S}(x) \mid f(x,y) \leq \tau\}$$

denote the set of feasible solutions for the corresponding decision problem. Clearly, for an instance x, the decision problem based on $\mathcal{S}_\tau(x)$ is equivalent to the following problem statement

> Given an instance $x \in \mathcal{I}$ and a rational number τ, is there a solution $y \in \mathcal{S}(x)$ so that $f(x,y) \leq \tau$?

The maximization case is obtained in a symmetric way.

For all the above problems, we can describe strategies and approaches which generate answers to the problems by algorithms. Generally speaking, an *algorithm* $\mathcal{A}$ for a problem Π is a finite sequence of instructions or calculations to be performed whenever an instance $x \in \mathcal{I}$ is presented to $\mathcal{A}$, and that returns some solution from $\mathcal{S}(x)$ when all these instructions have been performed. The solution returned is referred to as output of $\mathcal{A}$ for the instance, or $\mathcal{A}(x)$. For optimization problems, an output is usually considered together with its objective value $f(x, \mathcal{A}(x))$.

The instance $x \in \mathcal{I}$ is also called input, especially when considered in relation with an algorithm for the problem at hand, and the value given by the objective function $f(x,y)$ for a solution $y \in \mathcal{S}(x)$ is usually called cost.

For a problem Π, we define the length $|x|$ of an instance $x \in \mathcal{I}$ as the number of bits used to specify x in some fixed encoding. The time-complexity, i.e. the number of elementary steps it takes for an algorithm to return a solution, is measured as a function of the length of an input instance. The *computation time*, or run time, of an algorithm is measured by the number of basic operations it performs, e.g. additions, comparisons, multiplications etc. We suppose that all algorithms run on the same machine model, the *random access machine* (RAM), which is polynomially related to the Turing machine and other reasonable computational models.

The run time is usually expressed in asymptotic notation. For two functions $f : \mathbb{N} \to \mathbb{R}_+$ and $g : \mathbb{N} \to \mathbb{R}_+$, we say that "$f$ is in the order of g", written $f(n) = O(g(n))$ for short, if there exist $c, n_0 \in \mathbb{N}$ such that $f(n) \leq c \cdot g(n)$ for all $n \geq n_0$. While this O-notation gives an upper bound, we can analogously define $f(n) = \Omega(g(n))$ for lower bounds to say that "f is *at least* in the order of g". It is $f(n) = \Omega(g(n)) \iff g(n) = O(f(n))$.

If the run time of an algorithm $\mathcal{A}$ is bounded by a polynomial $p(|x|)$ for all possible instances $x \in \mathcal{I}$, thus, for any instance x and some fixed $k \in \mathbb{N}$, $\mathcal{A}$ is an algorithm with run time $O(|x|^k)$, then we say that $\mathcal{A}$ is *efficient*. A problem that can be solved by an efficient algorithm is referred to as polynomially solvable.

A problem that can be solved in polynomial time is generally considered *easy*. However, in the next chapters, we concentrate on problems which are *hard*. The notion of hard problems is formalized based on decision problems. The resulting complexity hierarchy of problems has been extended ever since the first appearance of the class *NP* together with *NP*-complete problems in the 1970s [11].

2.1.1 Decision Problems and the Class *NP*

The class *NP* consists of all decision problems that can be certified in polynomial-time: That is, a decision problem is in *NP* if there is a (necessarily polynomial size) “certificate” which shows that an input instance is a positive instance, and that there is a polynomial time algorithm which can verify this certificate with respect to the problem.

The class *P* consists of all those decision problems that can be recognized in polynomial time, i.e. for which the decision of an instance to be a positive instance can be reached in polynomial time.

Clearly, $P \subseteq NP$ holds. However, the question whether $P = NP$ or $P \neq NP$ is open. It is widely believed that $P \neq NP$, and a good reason for this is given by the class of *NP*-complete problems. The class of *NP*-complete problems has the following compelling property: if one can prove that a single *NP*-complete problem actually is in *P*, then $P = NP$. The reason for this comes from the fact that all *NP*-complete problems are polynomially reducible to each other.

Let $\Pi_1, \Pi_2 \in NP$. The problem Π_1 is *polynomially reducible* to Π_2 if there exists a polynomial time algorithm that for every instance x_1 of Π_1 produces an instance x_2 of Π_2 so that x_1 is a positive instance of Π_1 if and only if x_2 is a positive instance of Π_2.

Practically, this means that we can use an efficient algorithm for Π_2 to solve Π_1 efficiently: simply transform the instance of Π_1 into the respective instance of Π_2, and then solve this instance.

With the introduction of a polynomial reduction, we can formally introduce *NP*-complete problems [23].

Definition 2.4. *A decision problem* Π *is NP-**complete** if*

- Π *is in NP, and*
- *all other NP problems are polynomially reducible to* Π.

Polynomial reduction is a transitive relation on the class *NP*. If a problem Π_1 is polynomially reducible to Π_2 and Π_2 is reducible to Π_3, then Π_1 is also reducible to Π_3. This fact can be exploited to show that a new problem Π is *NP*-complete since it now suffices to show that Π is in *NP* and that there is an *NP*-complete problem which is polynomially reducible to Π. The start, i.e. the first problem shown to be *NP*-complete, was made by the satisfiability problem (SAT) [11], and today there are numerous problems that are known to be *NP*-complete.

2.1.2 *NP*-Optimization Problems

Similar to the class *NP* for decision problems, we introduce the class *NPO* for optimization problems [12]. For this class, we consider optimization problems for which the length of feasible solutions is polynomially bounded in terms of the length of an instance, and for which the value of the objective function can also be computed in polynomial time. More formally, an optimization problem Π is in the *class NPO* if

- all solutions are short, that is for every $x \in \mathcal{I}$, and every $y \in \mathcal{S}(x)$, we have $|y| \leq p(|x|)$ for some polynomial p,
- for any x and any y with $|y| \leq p(|x|)$, the question whether $y \in \mathcal{S}(x)$ can be decided in polynomial time, and
- given $x \in \mathcal{I}$ and $y \in \mathcal{S}(x)$, the objective function $f(x, y)$ is computable in polynomial time.

The class *PO* then corresponds to all optimization problems in *NPO* that can be solved to optimality in polynomial time. Again, we are interested in the *hard* problems, given by the following definition.

Definition 2.5. *A problem* Π *is said to be NP-**hard** if every problem in NP can be solved in polynomial time using a polynomial time algorithm that solves* Π *as a subroutine.*

Note that in the above definition, we no longer restrict Π to be a decision problem. In particular, this definition includes all optimization problems for which the corresponding decision problem is *NP*-complete. Furthermore, it is easy to see that if $P \neq NP$ holds, then $PO \neq NPO$ has to hold, as well.

In order to show that a problem Π is *NP*-hard, it suffices to show that an *NP*-complete problem Π' could be solved efficiently by efficiently solving Π. All other problems in *NP* are polynomially reducible to Π'.

From an algorithmic point of view, knowing that an *NPO* problem is *NP*-hard, we also know that we cannot compute an optimal solution in polynomial time, unless $P = NP$. In this case, there are two major options.

- On the one hand, we can abandon the idea of polynomial time, and still require an optimal solution from an algorithm. One way to achieve this would be complete enumeration, which may lead to exponential run time, in the worst case with an exponent only bounded by the polynomial bound on the size of all solutions.
- On the other hand, we can sacrifice optimality and start looking for approximate solutions which are computed by a polynomial time, i.e. efficient, algorithm.

The latter approach is taken in this thesis.

Let Π be an *NPO* problem. For any instance $x \in \mathcal{I}$, denote by $s^*(x)$ the cost of an optimal solution. The *performance ratio* of any feasible solution $y \in \mathcal{S}(x)$ is then given by

$$R(x,y) := \max\left\{\frac{f(x,y)}{s^*(x)}, \frac{s^*(x)}{f(x,y)}\right\}.$$

Note that the performance ratio is always greater than or equal to 1, independent of the goal: the first fraction is used for minimization, the second for maximization problems. Clearly, the closer $R(x,y)$ is to 1, the closer a solution is to an optimal solution. The performance ratio is also called *approximation ratio*.

Definition 2.6. *Let Π be an NPO problem, and let $\mathcal{A}$ be an algorithm that, for every instance $x \in \mathcal{I}$, returns a feasible solution $\mathcal{A}(x) \in \mathcal{S}(x)$. Given a function $r : \mathbb{N} \to [1,\infty)$, the algorithm $\mathcal{A}$ is an $r(n)$-**approximate algorithm** if for every instance $x \in \mathcal{I}$, the inequality*

$$R(x, \mathcal{A}(x)) \leq r(|x|)$$

holds. If there exists an $r(n)$-approximate polynomial time algorithm for Π, we say that Π is approximable within $r(n)$ (in polynomial time).

An algorithm which actually gives a constant performance ratio independent of the length of the instance is referred to as *constant-factor approximation.* An *NPO* problem Π belongs to the *class* APX if it is approximable within α (in polynomial time), for some constant $\alpha > 1$.

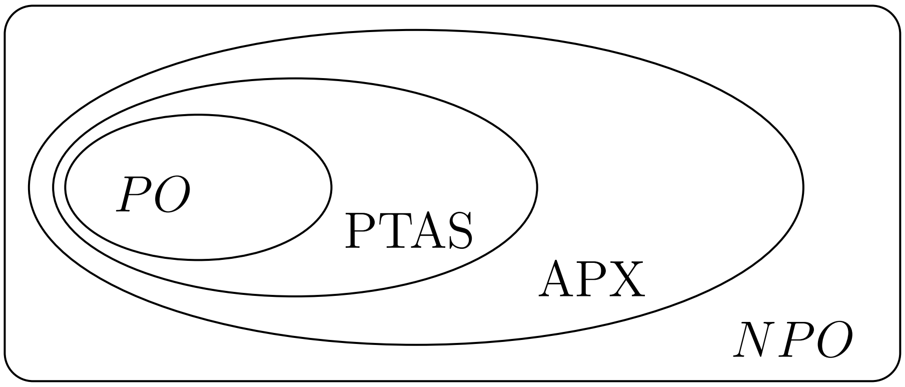


Figure 2.1: Relationships between some complexity classes for optimization problems discussed in this thesis.

Definition 2.7. *A family $\{\mathcal{A}_\varepsilon\}_{\varepsilon>0}$ of $(1+\varepsilon)$-approximation algorithms is called an* ***approximation scheme.***

If, for each fixed $\varepsilon > 0$, the $(1+\varepsilon)$-approximation algorithm $\mathcal{A}_\varepsilon$ of an approximation scheme $\{\mathcal{A}_\varepsilon\}_{\varepsilon>0}$ runs in polynomial time in the input size, we call $\{\mathcal{A}_\varepsilon\}_{\varepsilon>0}$ a ***Polynomial-Time Approximation Scheme (PTAS).***

An *NPO* problem that admits a PTAS is also in the *class* PTAS. We only demand the run time of a PTAS to be polynomial in the size of the input instance, and not in the desired performance ratio parameter $1/\varepsilon$. Computation times of $2^{1/\varepsilon} \cdot p(|x|)$ or $O(|x|^{1/\varepsilon})$ are allowed, obviously resulting in high run time when ε is close to 0.

Figure 2.1 gives the relations between the complexity classes for *NPO* problems [54]. It is easy to see that the inclusions given are strict if and only if $P \neq NP$. Especially to stress the difference between the classes APX and PTAS, there is also the notion of APX-complete problems. APX-complete problems are those problems in *NPO* that are the most difficult ones to approximate. Loosely speaking, an *NPO* problem is APX-complete if there exists a bound $\alpha > 1$ such that there exists no polynomial time α-approximation for this problem unless $P = NP$. Thus, if $P \neq NP$, for APX-complete problems, there cannot exist a polynomial-time approximation scheme.

There is a long list of other complexity classes and notions, in fact, a whole "complexity zoo" [14]. Here, we only presented the major classes that are of importance for this thesis. For a rigorous introduction to the theory of *NP*, we refer the reader to the classic book by Garey and Johnson [23]. A list of *NPO*-complete problems from various application areas is given by [12], including many further references.

2.2 Structures and Optimization on Graphs

In this section, we define several graph-theoretic structures, which are frequently used in efficient communication strategies for wireless networks, and the resulting optimization problems. These structures are independent and dominating sets of vertices in a graph. Before the respective definitions, we first introduce some notation which is used commonly throughout this thesis.

Communication networks are modeled as undirected graphs $G = (V, E)$. Here, the vertices V represent the communication devices or nodes, and two nodes are connected by an edge in $E \subseteq V \times V$ if they can communicate directly with one another. We set $n := |V|$, and denote by Δ the largest degree of a vertex in G. Furthermore, we assume the set of vertices to be ordered, for instance and simplicity $V = \{1, \ldots, n\}$. In this thesis, we are interested in algorithms that identify and create subsets of the vertices which satisfy certain structural properties. Also, as soon as we consider the cardinality or weight of such a subset as a measure for the quality of a solution, optimization problems arise.

Let $V' \subseteq V$ be a subset of vertices in $G = (V, E)$. In the following, we use $G[V']$ to denote the subgraph induced by V'. For a subgraph G' of G, we use $V(G')$ and $E(G')$ to refer to the vertices and edges of G' respectively.

In case of a weighted graph, that is, each vertex $v \in V$ is additionally given a weight w_v, we define the weight of a subset $V' \subseteq V$ by $W(V') := \sum_{v \in V'} w_v$.

For a subset $V' \subseteq V$, let $\max\{V'\}$ be the vertex $u \in V'$ with highest number, or weight in a weighted graph. In the latter case, we use the highest number to break a possible tie when vertices have the same weight.

Furthermore, we denote by $\Gamma(v)$ the closed neighborhood of a vertex $v \in V$, i.e.

$$\Gamma(v) := \{u \in V \mid (u, v) \in E\} \cup \{v\}.$$

Analogously, for $V' \subseteq V$, let $\Gamma(V') := \bigcup_{w \in V'} \Gamma(w)$ define the neighborhood of V'. In this context, we set $\Gamma(\varnothing) := \varnothing$. For $r \in \mathbb{N}$, we call $\Gamma_r(v) := \Gamma(\Gamma_{r-1}(v))$ the recursively defined r-th neighborhood of $v \in V$, where $\Gamma_0(v) := \{v\}$. Using the graph-theoretic distance $d_G(u, v)$, denoting the number of edges on a shortest path in G between vertices u and v, we can equivalently define the r-th neighborhood of v as

$$\Gamma_r(v) := \{u \in V \mid d_G(u,v) \leq r\}.$$

2.2.1 Independent Sets

Two vertices of a graph $G = (V, E)$ are called *independent* if they are not adjacent to one another. As a consequence, the subgraph $G[I]$ induced by an independent set I contains no edges.

Definition 2.8. *A subset $I \subseteq V$ is called* ***independent*** *if for every two vertices $u, v \in I$, there does not exist an edge $(u,v) \in E$.*

An independent set is called *maximal* if it cannot be extended by the addition of any other vertices from the graph. Note that maximality of an independent set is a purely structural property; we do not require such a set to be of large cardinality. A *maximal independent set* in G is further on abbreviated by MIS. For any graph, a MIS can easily be generated by a greedy approach. In Section 2.4, we will explain such an approach as an example.

When looking at the cardinality of an independent set, we obtain an optimization problem. The *Maximum Independent Set* (Max-IS) problem tries to find an independent set of maximum cardinality. Clearly, a maximum independent set is also maximal, but not vice versa.

In general, in an undirected graph, computing a Maximum Independent Set is *NP*-hard. Even more, it cannot be efficiently approximated within $n^{1-\varepsilon}$ (unless $P = NP$, [27]). There exists a polynomial time $O(n/\log^2 n)$-approximation using subgraph-removal techniques (see [7]).

2.2.2 Dominating Sets

A broadcast from a communication node is received by all its neighbors. This is captured in the notion of *domination* in a graph $G = (V, E)$.

Definition 2.9. *A subset $D \subseteq V$ of vertices is called* ***dominating*** *if every vertex in V is contained in the subset, or adjacent to a vertex in this set D.*

A dominating set can also be used for subsets of vertices $V' \subseteq V$, such a set D' then dominates V' if $V' \subseteq \Gamma(D')$ holds. There is an important relationship between maximal independent sets and dominating sets in a graph. Throughout this thesis, we will often rely on the following theorem.

Theorem 2.10. *Given a graph $G = (V, E)$, any maximal independent set $I \subseteq V$ is also a dominating set.*

Proof. Suppose there exists a non-dominated vertex in $V \setminus I$. This vertex could be added to I while keeping the independence property, thus violating maximality. $\square$

The *Minimum Dominating Set* (MIN-DS) problem asks for a dominating set of minimum cardinality. The problem is *NP*-hard on undirected graphs, and for efficient approximations, there is a lower bound of $(1 - \varepsilon) \ln n$, for $\varepsilon > 0$, on the approximation ratio (unless every *NP* problem is solvable in time $n^{O(\log \log n)}$, [21]).

There are also further variants of dominating sets in a graph $G = (V, E)$ by demanding additional properties for a dominating set $D \subseteq V$:

- *Minimum Connected Dominating Set* (MIN-CDS)
  If D consists of a single connected component, it is referred to as *Connected Dominating Set*. Of course, if G is not connected, such a structure does not exist.

- *Minimum Independent Dominating Set* (MIN-IDS)
  If we want to find a small subset of vertices that is both dominating and independent, we call the resulting problem Minimum Independent Dominating Set problem. In light of Theorem 2.10, this problem is sometimes also called *Minimum Maximal Independent Set* problem. Note that the Maximum Independent Dominating Set problem is equivalent to finding a Maximum Independent Set.

As with the MIN-DS problem, the MIN-CDS problem cannot be efficiently approximated within $(1 - \varepsilon) \ln n, \varepsilon > 0$, unless every problem in *NP* can be deterministically solved in time $n^{O(\log \log n)}$. The MIN-IDS problem cannot be efficiently approximated within a factor of $n^{1-\varepsilon}$ for any $\varepsilon > 0$ (unless $P = NP$, [26]).

The problems presented can be extended to vertex-weighted problems. While the constraints for the feasibility of a solution remain the same, the objective function is altered to include the weights of the vertices instead of the cardinality of the subsets. Further on, unless stated otherwise, we consider the unweighted versions of the problem.

2.3 Distributed and Local Algorithms

In this thesis, we are interested in efficient algorithms that create independent and dominating sets, both from the structural, e.g. maximal sets, and the optimization, e.g. maximum cardinality sets, point of view. Also, due to the wireless structure and resource-poor nature of the underlying communication network, we are interested in algorithms that account for this fact. We now go on to present a model for the local communication characteristics of wireless networks which we use to describe locally executed algorithms.

A prominent characteristic of wireless ad-hoc networks is the lack of central control. Even if a central base station is present in the communication network, the resulting communication and coordination overhead may be prohibitive to centrally organize and optimize such a network. In this part, we provide a framework, or model, for distributed and local computing that allows for capturing the effects of communication and coordination overhead, as well as the non-centralized nature of wireless networks: the $\mathcal{LOCAL}$ message passing model [40, 51].

Consider a simple, undirected graph $G = (V, E)$ which represents the point-to-point communication network. That is, the vertices are the (wireless) nodes, and two nodes are connected if and only if they can communicate with one another directly. Note that we only consider bidirectional communication, e.g. by ignoring unidirectional links.

For this communication network, we would like to point out that the graph is used two-fold: it represents both the communication network and the graph for which we seek solution subsets. In other words, G is the graph we work *in* and optimize *for*. This also follows from the application, as independent and dominating sets in communication networks are used *for* efficient communication strategies *in* the network itself.

A vertex in the graph has—possibly among others—two main parts we focus on: a processing part including CPU and memory, and a communication part which allows the vertex to send messages to its direct neighbors in G. We assume that each vertex $v \in V$ is given a unique identifier id_v at the hardware level. For concreteness and simplicity, we assume these identifiers to be ordered, i.e. $id_v \in \mathbb{N}$ for every $v \in V$, and furthermore, that each identifier is stored in the node using $O(\log n)$ bits. Usually, the explicit distinction between a vertex v and its identifier id_v is neglected, we simultaneously use v to denote both.

We now explain the communication characteristics of the $\mathcal{LOCAL}$ model. The networking on G operates in so-called global communication rounds. Basically, in one round, each vertex is able to communicate once with each of its neighbors by transmission of a message packet, and perform some computation. To be more precise, a round is divided into two parts. First, a vertex can perform some computations, and then communicate with its neighbors in G by sending and receiving message packets. In one round, a vertex can send at most one message to, and receive at most one message from *each* individual neighboring vertex. The order in which messages are sent and received in a round by the individual vertices is not specified, and assumed to occur simultaneously. A vertex that wants to reply to a message received from a neighbor can only do so in succeeding rounds.

As a consequence of this local, round-based communication pattern, for two vertices $u, v \in V$ with $d(u, v) = k$, it takes at least k rounds for a message sent by u to reach v via message forwarding of intermediate vertices. This also implies that a vertex, in a constant number of rounds, can only obtain information about the topology of its local neighborhood of constant radius.

We assume communication in the network to be synchronized, that is, in the $\mathcal{LOCAL}$ model, each vertex has the same notion of a round. The round-based synchronization is not to be confused with synchronization of time, a much more difficult task in distributed systems. In the $\mathcal{LOCAL}$ model, we only require the vertices to be in the same round during the execution of a distributed algorithm. This limitation is, however, not too severe. At the cost of higher message complexity, with the introduction of synchronizers [4], algorithms presented for the synchronous model can be employed in asynchronous settings, as well.

While the complexity of the computations performed locally at each vertex (within one round) can be explained using the notions described in Section 2.1, the $\mathcal{LOCAL}$ model mainly captures the communication part. In order to quantify this part, there are three complexity measures for local algorithms in this distributed model:

- The distributed *time complexity* of a local algorithm is the number of rounds until all vertices have terminated the algorithm.

  Typically, in such synchronous systems, all vertices wake up simultaneously, or start an algorithm at the same time. If we assume that a vertex starts the algorithm upon receiving the first message that belongs to the distributed algorithm at hand, we obtain an almost equivalent model. In this case, the time complexity is defined as the maximal time between the first vertex wake-up and overall termination.

- The number of messages sent during the execution is given by the *message complexity* of the algorithm, and it is usually given with respect to a single vertex of the network.

- Each message packet used by the algorithm transports information to its recipient. The amount of information contained in a message packet, that is, the length of such a packet, is called message size. The largest message needed to execute a local algorithm defines the *maximum message size*. In combination with the message complexity, this figure gives the overall exchanged amount of information in the network.

Up to now, in the $\mathcal{LOCAL}$ model, we allow that each vertex can address each neighbor directly by a distinct message in each round. Especially in wireless network settings, this is not very realistic. Sending a message in such networks is usually done by a broadcast which can be received by *all* adjacent vertices. In the following, in order to explicitely stress this broadcast version of the local model, we denote it by $\mathcal{LOCAL}_{\mathrm{BC}}$.

In applications, the size of message packets is limited, and thus this fact should be taken into account. The latter is especially interesting in the broadcast model, as a limited packet size may not allow distinct information for all neighboring vertices, e.g. when the neighborhood is larger than the packet size. When not considering the message size, and thus allowing message packets of arbitrary length, the $\mathcal{LOCAL}$ and $\mathcal{LOCAL}_{\mathrm{BC}}$ models are clearly equivalent: in this case, all messages to distinct neighboring vertices are combined into one larger message. This larger message is then broadcast in the respective round, and each neighboring vertex filters out the part concerning itself.

In the following, we assume that the message size is at least $\Omega(\log n)$, otherwise a vertex cannot even inform its neighbors about its identifier, and adjacent vertices would become almost indistinguishable, giving an anonymous network. We also suppose that each node has knowledge of the identifiers given to its neighbors. This can easily be achieved by a single broadcast of each node containing its identifier at the beginning of operation.

The synchronous, local message passing model seems to be a reasonable abstraction for communication-based algorithms in wireless ad-hoc networks, both from a theoretical and practical point of view.

- For the theory of local, distributed computing, it is the simplest model to describe algorithms for, and to reason about. Also, the focus is put only on the additional overhead that stems from locality and communication, independent of characteristics thereof.

- In practice, round-based communication can easily be achieved or is explicitely done. For example, TDMA-based medium access control imme-

diately maps to the synchronous message passing model. This can be seen in Chapter 6 when dealing with the EMACs protocol.
Also, in contrast to communication, computation is cheap in wireless communication, both with respect to time and energy consumption.

The main questions in local computing deal with the relation between a solution composed of local, partial solutions with respect to an optimal (global) solution for optimization, and how to compute these by local, distributed algorithms. The characteristics of the $\mathcal{LOCAL}$ model, the lack of central control and global information, limit the possibilities in computing. In this model, already the fast creation of a structure like a maximal independent set is a non-trivial task.

In this context, maximal independent sets play an important role since the creation of this structure captures the notion of symmetry breaking in a simple problem formulation. Being this prototypical, fast MIS creation has received particular attention in the literature.

The fastest known deterministic distributed algorithm is based on the notion of network decompositions introduced in [5], and is given in [50]. The authors obtain a deterministic $O(n^{(\sqrt{1/\log n})^d})$ algorithm, where d is a constant.

Furthermore, there is also an elegant randomized algorithm for MIS creation with *expected* time complexity of $O(\log n)$ presented in [39].

An important function, which is used quite often in the area of local algorithms is the "logstar"-function. This function is defined as

$$log^*n := \min\{i \in \mathbb{N} \mid \underbrace{\log\log\ldots\log}_{i} n \leq c\}$$

for some constant c.

A lower bound of $\Omega(\log^* n)$ for the distributed computation of maximal independent sets is given in [38]. This lower bound states that even on a ring C_n, at least time $\Omega(\log^* n)$ is required to construct a MIS.

For general graphs of bounded maximal degree Δ, a maximal independent set I can be computed locally in $O(\log^* n)$ communication rounds using messages of size $O(\log n)$ [38, 51]. Thus, in case Δ is bounded by a constant, there exists an asymptotically time optimal approach.

However, a deterministic, poly-logarithmic time distributed algorithm for MIS construction is a longstanding open problem.

Algorithm 1 Central-Greedy Maximal Independent Set.

```
Input: Undirected graph $G = (V, E)$
Output: Maximal independent set $I$
1: $I := \varnothing$;
2: while $V \neq \varnothing$ do
3:     Choose $v \in V$;
4:     $I := I \cup \{v\}$;
5:     $V := V \setminus \Gamma(v)$;
6: end while
```

2.4 Greedy MIS Construction

A maximal independent set is maximal with respect to the addition of vertices in a graph G. This characterization immediately suggests a simple greedy strategy to create such a set: add independent vertices to a partial solution until this is no longer possible. In this part, we explore this strategy both from a central and from a local perspective, and use the resulting algorithms as examples to explain the important complexity measures of the preceding parts.

We begin with a centralized, straightforward algorithm: pick a vertex, erase its neighborhood, and continue with the remaining graph. This approach is also presented by Algorithm 1.

During the execution of the algorithm, the set of not yet considered vertices gives the set of all vertices that could be added to I without violating the independence property of I. Algorithm 1 constructs a maximal independent set, since we always remove all conflicting vertices. Clearly, the approach yields an efficient algorithm.

Note that the greedy choice of the vertex $v \in V$ to be added next can be an arbitrary vertex from the candidate set V. This gives rise to several different strategies by basing the choice on, e.g., degree or weight of a vertex.

As an example of a local, distributed algorithm, we show how to run the greedy algorithm locally in each node of a communication network to create a maximal independent set. Many local algorithms encode *roles* of vertices using colors. In our approach, we have three roles (colors): undecided (grey), independent (black), and dominated (white) vertices.

Initially, all vertices are grey, and grey vertices have not yet taken a decision whether to join the partial solution. Eventually, all vertices should decide on their role in the solution, and either have joined the solution set given by the black colored vertices, or not (white color). The decision is done in an ordered

Algorithm 2 Local-Greedy Maximal Independent Set (code for vertex v).

```
Input: Undirected graph $G = (V, E)$
Output: Maximal independent set $\{v \in V \mid c(v) = \text{black}\}$.
1: $c(v)$ := grey;
2: while $c(v) = \text{grey}$ do
3:    if $\exists\, u \in \Gamma(v) \mid c(u) = \text{black}$ then
4:       $c(v)$ := white;
5:    else if $v = \max\{u \in \Gamma(v) \mid c(u) = \text{grey}\}$ then
6:       $c(v)$ := black;
7:    end if
8: end while
```

fashion, vertices with higher identifiers are allowed to decide first. A vertex waits with its decision until all vertices with higher identifier have decided.

The algorithm works as follows: every grey vertex joins the partial solution (black vertices) when it has the highest identifier among its grey neighbors and if none of its neighbors has already joined. Vertices adjacent to a black vertex immediately turn white. A change of color always results in a local broadcast that informs all neighbors about the new color. The local algorithm describing the decision process is also given by Algorithm 2.

Clearly, the algorithm terminates, as the grey vertex with highest identifier always changes its color and each black or white vertex never changes its color again. In order to obtain a MIS, after completion of the algorithm, the set of black vertices should be independent, and each white vertex should be neighbor to at least one black vertex.

Lemma 2.11. *The set of black vertices, as created by Algorithm 2, is a* MIS *on G.*

Proof. For the sake of contradiction, assume that there are two adjacent black vertices, say u and v. W.l.o.g. let $u < v$, then at the same time as v decided to turn black, u could not have taken this decision due to line 5 not holding for u. After this decision, u has to change its color to white due to line 3.

To see that the solution is maximal, observe that as long as there are non-dominated vertices (grey), there is also a vertex with the highest identifier among these which will turn to black. White vertices are adjacent to a black vertex. □

Let us now look at the complexity of this approach in the $\mathcal{LOCAL}$ message passing model. One round is given by a single execution of the while loop. In each round, at least the grey vertex with highest identifier changes its color and thus sends a message in this round.

There are exactly n color changes, therefore after at most $\Omega(n)$ communication rounds, there are no grey vertices left and the algorithm has thus terminated. However, in each round, more than just a single vertex or few vertices may change its color, possibly resulting in a faster distributed time complexity.

Unfortunately, for example by considering vertices along a line with increasing identifiers, there may actually be only a single point of activity during the execution, resulting in sequential time complexity of $O(n)$ communication rounds.

The message complexity (per vertex) is constant as a vertex only changes its color once, and therefore only has to broadcast this information once to all neighbors, which takes a single round.

A message containing information about a new color of the sender is of constant size. However, the algorithm assumes knowledge of the vertices' identifiers which requires messages of size $\Omega(\log n)$. All messages are broadcasts, the algorithm therefore is also suited without change for the $\mathcal{LOCAL}_{\mathrm{BC}}$ model.

Chapter 3

Wireless Communication Graphs

Wireless networks are created by the communication links between a collection of radio transceivers. The nature of wireless transmissions does not lead to an arbitrary, undirected graph, but to a structured graph which we discuss in this chapter. We introduce geometrically inspired graph models which are commonly used to represent wireless ad-hoc networks. All of these models share the property of (polynomially) bounded growth. This important property, which no longer depends on the geometry, is defined and discussed.
Bounded growth already allows for a constant bound on the approximation ratio of a solution given by a maximal independent set for the Max-IS and Min-DS problems. Such a solution can easily be created by the greedy strategies of the previous chapter.
Implications of the geometry behind the wireless graph models, for example recognition and geometric construction issues, are pointed out here. In this context, robust algorithms are also introduced.

A wireless, ad-hoc network is created by the communication links between a collection of radio transceivers. The nature of wireless transmissions does not lead to an arbitrary, undirected graph, but to a structured graph. This structure, called *bounded growth*, is introduced and discussed in this chapter.

We propose several geometrically inspired graph models for wireless communication graphs. These range from very simple Disk Graphs to more realistic Bounded Coverage Area Graphs, and most of these models are geometrically defined [46]. They follow the general intuition that vertices which are close to one another can communicate directly, while far separated ones cannot reach each other.

The bounded growth property of these models can be exploited to obtain efficient optimization strategies: Maximal independent sets are already constant-factor approximations to the Max-IS and Min-DS problems. In terms of encoding of the graphs, i.e. how the graph is presented as input to the algorithms, there are several possibilities. These go from giving the adjacency of each vertex to describing the geometric shapes and positions of the vertices, and by then looking at the intersection graph of the geometric objects. The latter is called *geometric representation*, and in practice requires a localization scheme that gives each wireless node its position. We will discuss the implications and advantages of these different input encodings.

In Section 3.1, we define bounded growth graphs. After that, we introduce prominent graph models that are used to model the communication in wireless ad-hoc and sensor networks, and give their relation to bounded growth. In Section 3.3, a constant-factor ratio for the cardinality between a maximal independent set and Maximum Independent and Minimum Dominating Sets are established for these particular graph classes. Before concluding this chapter, different representations of a graph, and various issues arising from these are discussed. Robust algorithms, which are described in Section 3.5, are a possibility to work around some problems with different representations of structured graphs.

3.1 Bounded Growth Graphs

Throughout this thesis, we are mostly interested in graph classes that model wireless communication networks. However, characteristics of such networks highly depend on the environment in which they operate, and we would like to abstract away from this environment. The smallest common structure of most wireless graph models is *bounded growth* as given by the following definition.

Definition 3.1. *Let $G = (V, E)$ be a graph. If there exists a function $f(.)$ such that every r-neighborhood in G contains at most $f(r)$ independent vertices, then G is f-**growth-bounded**. In this case, we call f the growth function.*

*Furthermore, we say that G has **polynomially** bounded growth if for some constant $c \geq 1$, $f(r) = O(r^c)$ is bounded by a polynomial of maximal degree c.*

Graphs of bounded growth are sometimes also called graphs of bounded (local) independence. Note that the growth function $f(.)$ only depends on the radius of the neighborhood, and not on the number of vertices in G. Thus, for constant r, the number of independent vertices in $\Gamma_r(v)$ is bounded by a constant for any $v \in V$.

Especially when considering subsets of vertices, like independent or dominating sets, the following lemma shows that the bounded growth property of a graph is closed under taking induced subgraphs.

Lemma 3.2. *Let $G = (V, E)$ be an f-bounded growth graph. Then, for any $V' \subset V$, the induced subgraph $G[V']$ is also of f-bounded growth.*

Proof. We prove the claim by contradiction. Suppose that there is a subset $V' \subset V$ such that the bound on the growth is violated for V'. That is, for some $r \in \mathbb{N}$, there exists a vertex $u \in V'$, and a neighborhood $\Gamma_r(u)$ in $G[V']$ such that there is an independent set $I \subset \Gamma_r(u)$ with cardinality $|I| > f(r)$.

Consider the addition of any $v \in V \setminus V'$ to V', and and the resulting induced graph $G[V' \cup \{v\}]$. It is easy to see that the independence property of I is still satisfied in $G[V' \cup \{v\}]$.

Furthermore, for every $u' \in \Gamma_r(u)$, it is $d_G(u, u') \leq d_{G[V']}(u, u') \leq r$.

Therefore, I is also an independent set with $|I| > f(r)$ in the r-th neighborhood of u when considering the original graph G. $\square$

It is easy to see that an f-bounded-growth graph does not contain a vertex-induced subgraph isomorphic to a $K_{1,f(1)+1}$. Such a subgraph would immediately violate the growth function f.

3.2 Geometric Intersection Graph Models

In the following, we propose several geometrically defined graph models that represent wireless networks in various degrees of granularity with respect to reality. We also apply and check Definition 3.1 on the geometric graphs to establish bounded growth for these models.

A wireless ad-hoc network is created by nodes that are equipped with a transceiver, and that are placed in the real world. The environment, especially the positions of the nodes, has to be accounted for, and it does not surprise that most wireless graph models are therefore defined for the Euclidean space. In the following, we use $\|.\|$ to denote the Euclidean distance in $\mathbb{R}^2$.

Usually, the models result from geometric intersection or containment graphs which give the general idea behind these models. Next, we therefore introduce these graph models in general, and then specify additional characteristics in order to justify them for the purpose of modeling wireless communication networks.

We assume that the vertices of the graph, i.e. in our case the wireless nodes, are placed in the 2-dimensional Euclidean plane. In other words, there exists a mapping $p : V \to \mathbb{R}^2$ which gives each vertex $v \in V$ its location $p_v \in \mathbb{R}^2$. Furthermore, each wireless node has a certain area which is covered by its radio. For every $v \in V$, let this area be represented by $A_v \subset \mathbb{R}^2$. As a consequence, another vertex $u \in V$ can receive a transmission, and thus a message from v, if and only if $p_u \in A_v$ holds.

There are two ways of defining the edges of a wireless graph, representing the possible communication or interference between wireless nodes. The first way is the *containment model*, where the set of edges is characterized by

$$(u, v) \in E \iff p_u \in A_v.$$

This model gives the possible direct communication between nodes, and results in a directed graph model.

If we only look at the coverage areas of the models, we can also define the *intersection model* as follows:

$$(u, v) \in E \iff A_v \cap A_u \neq \varnothing.$$

With this symmetric model, interference during simultaneous transmissions can be explored. If two nodes transmit at the same time, a third node in the non-empty intersection receives both transmissions simultaneously, and may thus not be able to reconstruct the messages.

We now define the resulting graph models as follows:

Definition 3.3. *Consider a set V of vertices, and for each $v \in V$ its location $p_v \in \mathbb{R}^2$ and coverage area $A_v \subset \mathbb{R}^2$. Then, the resulting intersection graph is called* ***intersection Coverage Area Graph****, and the resulting containment graph is called* ***containment Coverage Area Graph****.*

We abbreviate Coverage Area Graph by CAG. The intersection graph is undirected. If we consider each undirected edge to be a two-way edge between the respective vertices, it is easy to see that the containment graph is completely contained in the intersection graph for the same set of vertices and coverage areas. In the following, we refer to a set of positions and corresponding coverage areas as *geometric representation* of an intersection or containment graph. It is easy to see that such a representation results in a well-defined graph.

Throughout the following, we do not differentiate strictly between the containment and intersection graph models, and consider all edges to be undirected. This assumption also follows from the applications, as bidirectionality is usually assumed for wireless communications due to the interpretation of the edges. The manner in which the edges are interpreted depend on the application, e.g.

- if communication is considered, bidirectional communication is usually assumed, as this allows for direct acknowledgments of successful transmissions of message packets. In this case, we simply remove uni-directional links from the graph. This is gives the following characterization of an edge:
$$(u, v) \in E \iff (p_u \in A_v) \wedge (p_v \in A_u).$$
- if interference is considered, all edges incident to a vertex $v \in V$ represent links to vertices which can receive a message sent by v. Assuming each edge as being undirected models this case.

The results presented in the following chapters hold for both interpretations and restrictions of the edge set resulting from the wireless graph models.

3.2.1 Unit Disk Graphs

In practice and real world settings, the coverage areas do not take on arbitrary shapes, but follow the laws of physics, especially the laws of radio wave propagation with respect to the environment the network operates in. In the following, we investigate the coverage areas further, and depending on the assumptions on the environment, we obtain several specific graph models for wireless communication networks.

The most basic model used for wireless communication is a Unit Disk Graph: Suppose that all wireless nodes are equal, and are placed in an ideal environment. That is, all nodes send with the same transmission radius, and have the same circular coverage area. A graph $G = (V, E)$ is a *Unit Disk Graph* (UDG) if is the intersection graph of disks of equal diameter in the Euclidean plane. In

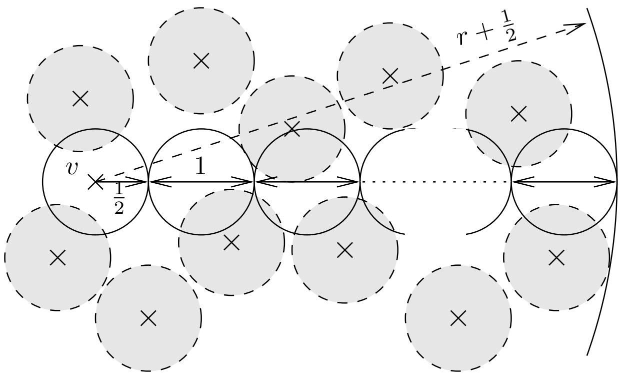


Figure 3.1: Proof of Lemma 3.5.

the following, by proper scaling, we assume the diameter of the disks to be of unit length. In other words:

Definition 3.4. *A graph $G = (V, E)$ is a* ***Unit Disk Graph*** *if there exists a map $p : V \to \mathbb{R}^2$ satisfying*

$$(u, v) \in E \iff \|p_u - p_v\| \le 1.$$

The above definition actually characterizes both intersection and containment graph. Scaling the diameters of the disks by a factor of 2 turns a Unit Disk *Intersection* Graph into an equivalent equal-diameter Disk *Containment* Graph, and vice versa. For such a UDG, containment and intersection graph are basically the same, and all edges are bidirectional. Unit Disk Graphs have polynomially bounded growth, which follows from a simple geometric packing argument also given in Figure 3.1.

Lemma 3.5. *Let $G = (V, E)$ be a Unit Disk Graph. Then, G is of (polynomially) $(2r + 1)^2$ bounded growth.*

Proof. From the above definition of a UDG, we conclude that any $w \in \Gamma_r(v)$, $v \in V$, satisfies

$$\|p_w - p_v\| \le r.$$

Let $I \subset \Gamma_r(v)$ denote an independent set in the r-neighborhood of v. The unit disks corresponding to vertices in I are pairwise disjoint, and are all contained in a disk of larger radius R with $R = (r + 1/2)$ around $f(v)$. This implies

$$|I| \leq \frac{\pi R^2}{\pi(1/2)^2} = (2r+1)^2$$

as claimed. $\square$

3.2.2 Bounded and Quasi Disk Graphs

If the wireless nodes are able to adjust their transmission power, still in ideal settings, then different circular coverage areas emerge.

Suppose that the maximum transmission range of a vertex is given by $c^+ > 0$. Furthermore, in order to achieve any communication, a minimum signal strength is needed, which we assume to create a circular coverage area of radius $c^- > 0$ around the position of each vertex. This yields a set of disks of different radii, and the resulting intersection graph is referred to as *Disk Graph.* The parameter $b := c^+/c^-$ gives gives the ratio between maximum and minimum range of the wireless nodes.

Definition 3.6. *A graph $G = (V, E)$ is a **Disk Graph** if there exist a map $p : V \to \mathbb{R}^2$ and a map $r : V \to \mathbb{R}^+$ satisfying*

$$(u, v) \in E \iff \|p_u - p_v\| \leq r_v + r_u.$$

*In this case, by setting $b := \frac{\max_{v\in V}\{r_v\}}{\min_{v\in V}\{r_v\}}$, we also call G a b-**Bounded Disk Graph** (BDG).*

Clearly, if the parameter b is constant, we again obtain a graph of polynomially bounded growth. This can easily be seen by adapting the geometric packing argumentation used in the proof of Lemma 3.5. The polynomial growth function is larger than that of a Unit Disk Graph by a factor of $O(b^2)$.

While Unit and Bounded Disk Graphs are widely used to obtain strong theoretic results for graph algorithms, one might argue that they are not very realistic since ideal assumptions are made for the radio propagation.

Refining the idea behind Bounded Disk Graphs by no longer limiting the reasons for different radii to the transmission power, but also to environmental reasons like objects, we can define a *Quasi Disk Graph* (QDG, [37]) as follows.

Definition 3.7. *A graph $G = (V, E)$ is a **Quasi Disk Graph** if there exist two values $0 < c^- \leq c^+$ and a map $p : V \to \mathbb{R}^2$ satisfying*

$$(u, v) \in E \text{ if } \|p_u - p_v\| \leq c^- \text{ and } (u, v) \notin E \text{ if } \|p_u - p_v\| > c^+.$$

In such a Quasi Disk Graph, there are two disks D^+ and D^- of radius c^+ and c^- respectively that can placed around each vertex position $p_v, v \in V$, such that

$$D^- \subseteq A_v \subseteq D^+$$

holds for the coverage area A_v of each vertex. Then, a more intuitive characterization of a Quasi Disk Graph based on transmissions is as follows.

- Two vertices $u, v \in V$ which are sufficiently close to each other, that is $\|p_u - p_v\| \leq c^-$ holds, always receive each other's messages.
- Two vertices $\bar{u}, \bar{v} \in V$ that are too far apart, i.e. $\|p_{\bar{u}} - p_{\bar{v}}\| > c^+$, cannot communicate directly.

If $c^- < \|p_u - p_v\| \leq c^+$ holds for two vertices $u, v \in V$, the existence of an edge is not explicitely defined, but depends on the concrete shapes of A_v and A_u. The coverage areas may take any shape within the bounding disks D_v^+ and D_v^-, and may be limited, e.g. due to objects in the vicinity of a vertex $v \in V$. Furthermore, effects like fading and the resulting unreliable transmission characteristics can be incorporated into this model.

Again, by slightly adjusting the geometric packing argumentation in the proof of Lemma 3.5, it is clear that a QDG (with parameter $b := c^+/c^-$) also has polynomially bounded growth, again with an additional factor of $O(b^2)$ for the Unit Disk Graph growth polynomial.

3.2.3 Bounded Coverage Area Graphs

In practice, it may not be possible to give a radius on the transmission range where coverage can be guaranteed for all wireless nodes, e.g. when these are mounted on concrete walls. Leaving the idea of circles that reflect the coverage area, we only consider the area itself: We assume that the *size* (or volume) of the coverage area of each vertex is bounded from below, and that it does not stretch too far from each vertex position.

Let $G = (V, E)$ be a Coverage Area Graph, and for each vertex $v \in V$, let $p_v \in \mathbb{R}^2$ be the vertex position, and let $A_v \subset \mathbb{R}^2$ be the coverage area. For each vertex $v \in V$, let $r_v > 0$ denote the minimum radius of a disk D_v that is centered at p_v and that completely covers A_v.

We then define $c^+ := \max_{v \in V}\{r_v\}$ to be radius of the largest such disk, and we define $a := \min_{v \in V}\{\text{vol}(A_v)\}$ to be the minimum size (or volume) of a coverage area. A parameter $b^2 := (c^+)^2/a$ then describes a bound on the relation between maximum and minimum coverage area.

We can again use a geometric packing argumentation to see that the intersection or containment Coverage Area Graph G has polynomially bounded growth with a growth function of $O(b^2 \cdot r^2)$.

As with the QDG, the Bounded Coverage Area Graph model also allows for modeling the influence of objects on the radio propagation, as well as fading and unreliable transmissions. The main characteristic of this model is an area A_v of strong signal, where a transmission is certainly received, and the fact that vertices which are far apart cannot communicate. Note that this model includes the measured transmission characteristics of the real world testbed presented in Section 1.2. There, in Figure 1.3 on page 6, the coverage area A_v of a particular wireless sensor node is shown.

It is easy to see the following relationship between these geometrically defined graph models.

Lemma 3.8. *By slightly abusing notation to denote the respective classes of graphs, it is*

$$UDG \subset BDG \subset QDG \subset CAG.$$

With respect to the optimization problems of seeking independent sets of large cardinality, and seeking dominating sets of small cardinality, in [10], it is shown that both the Max-IS and Min-DS problems remain *NP*-hard even on Unit Disk Graphs.

3.2.4 Graphs Based on Metric Spaces

While wireless networks operate in the Euclidean space, we can extend the previous graph models to intersection graphs induced by other metrics. Analogously to Unit Disk Graphs, we immediately obtain the following characterization of Unit Ball Graphs.

Definition 3.9. *Let $M = (X, d)$ be a metric space with a distance function $d : X^2 \to \mathbb{R}$. A graph $G = (V, E)$ is a* ***Unit Ball Graph*** *(UBG) if there exists a mapping $p : V \to X$ such that*

$$(u, v) \in E \iff d(p_u, p_v) \leq 1$$

holds. The pair (M, p) is called representation of G.

However, in this context, note that any undirected graph is such a Unit Ball Graph by taking the shortest path distance as metric on $V = X$. So, a UBG is not necessarily growth bounded.

Nevertheless, further restricting the metric space, we can identify a large class of metric Unit Ball Graphs with polynomially bounded growth. Such a restriction uses a bound on the "growth" of the metric space M, which is defined as follows [3, 25].

Definition 3.10. *Let $M = (X, d)$ be a metric space. The **doubling dimension** ρ of M is the smallest ρ such that every ball of radius r can be completely covered by at most 2^ρ balls of radius $r/2$. If ρ is bounded by a constant, we say that M is **doubling**.*

Analogously, we refer to a Unit Ball Graph as *doubling* if there exists a representation where the underlying metric space is doubling. The following lemma now shows that doubling UBGs are growth-bounded.

Lemma 3.11. *Let $G = (V, E)$ be a Unit Ball Graph with a representation (M, p) such that the metric space $M = (X, d)$ has doubling dimension ρ. Then, G has f-bounded growth with $f(r) = O(r^\rho)$.*

Proof. For a vertex $v \in V$, and a radius $r \geq 0$, consider the neighborhood $\Gamma_r(v)$, and let $I \subset \Gamma_r(v)$ denote an independent set therein.

Both $\Gamma_r(v)$ and I are contained in a ball $\{v' \in V \mid d(p_v, p_{v'}) \leq r\}$ of radius r around p_v. Also, for every $u \in I$, the ball with radius $1/2$ around p_u, given by $\{v' \in V \mid d(p_u, p_{v'}) \leq 1/2\}$, does not contain another vertex from I. In other words, for all $u \in I$, these balls are mutually disjoint.

The number of balls of radius $1/2$ needed to cover the ball of radius r around p_v, and thus $\Gamma_r(v)$, is at most $2^{\rho \log(2r)} = O(r^\rho)$, and the claim follows. $\square$

Clearly, if the metric space M is doubling, i.e. $\rho = O(1)$, then the induced doubling Unit Ball Graph has polynomially bounded growth $p(r) = O(r^\rho)$.

Also, the other types of geometric graph models of Section 3.2 can be generalized to a doubling metric space. This more general characterization allows us to use a distance function not only based on geometric distance, but also on characteristics of wave propagation of the wireless medium. We can thus relate signal strength, distance, and transmission characteristics to obtain a suitable metric intersection graph model for the wireless communication network.

Constant doubling dimensions are common not only in wireless networking, but also in communication networks, e.g. concerning latency in peer-to-peer networks or the Internet (see c.f. [25, 32, 33]). However, the communication links in such networks usually do not respect the Unit Ball Graph definition, and are characterized differently.

3.3 Simple Approximation Guarantees

In this section we deal with the construction of independent and dominating sets in graphs of bounded growth. As we show in this section, the simple, efficient approach of (greedily) computing a maximal independent set in such a structured graph turns out to be quite effective. Recall that a MIS in a graph is a feasible solution to both the Max-IS and Min-DS problems.

From the definition of bounded growth, it immediately follows that every f-bounded growth graph cannot contain more than $f(1)$ independent vertices in each first order neighborhood. This property can be exploited to obtain the following two lemmas, and the resulting constant-factor approximations given by a MIS [28].

Lemma 3.12. *Let $G = (V, E)$ be an f-growth-bounded graph, and let $I \subset V$ be a maximal independent set in G. Furthermore, let I^* be a Maximum Independent Set in G. Then,*

$$|I^*| \geq |I| \geq \frac{|I^*|}{f(1)}$$

holds.

Proof. We prove the lemma by comparing I with the optimal solution I^* using a charging argumentation. Each vertex $i \in I^*$ is charged to a vertex from I as follows:

- If $i \in I$, then i is charged to itself.
- If $i \notin I$, then i is charged to an adjacent vertex from I. Note that such a vertex exists due to the domination property of I.

It is easy to see that each vertex $v \in I$ is charged at most $f(1)$ times by vertices from the optimal solution I^*. The claim follows immediately. □

Lemma 3.13. *Let $G = (V, E)$ be an f-growth-bounded graph, and let $I \subset V$ be a maximal independent set in G. Furthermore, let D^* be a Minimum Dominating Set in G. Then,*

$$f(1) \cdot |D^*| \geq |I| \geq |D^*|$$

holds.

Proof. For the dominating set D^*, every vertex in D^* cannot dominate more than $f(1)$ vertices from I. Hence, $|D^*| \geq |I|/f(1)$ by a charging argumentation similar to the one used in the proof of Lemma 3.12, and the claim follows. □

A maximal independent set is thus a constant-factor approximation for the MAX-IS and MIN-DS problems on graphs of bounded growth. We point out that any Maximum Independent Set is also maximal. As an interesting remark, note that the cardinalities of optimal solutions to the MAX-IS and MIN-DS problems are therefore only a factor of $f(1)$ apart from each other in graphs with f-bounded growth.

A maximal independent set can be created by a greedy strategy as presented in Section 2.4. When considering a greedy strategy, the choice of the next vertex to be added to a partial solution can be arbitrary. However, for certain classes of graphs, a more sophisticated choice improves the approximation guarantee below the value of $f(1)$ for the MAX-IS problem.

An example for such an improved choice can be found in Unit Disk Graphs. For a UDG $G = (V, E)$, let $p_v = (x_v, y_v) \in \mathbb{R}^2$ denote the position of each vertex $v \in V$ in a geometric representation. A possible ordered choice of vertices to be added to a partial solution I is given by the following criterion:

$$\text{Choose } \bar{v} \in V \text{ such that } x_{\bar{v}} = \min\{x_u \mid u \in V \setminus \Gamma(I)\}.$$

In other words, we always choose a *leftmost* vertex which is neither in the partial solution, nor dominated by it. Further on we refer to this approach as *Leftmost-Greedy* for Unit Disk Graphs.

For the chosen vertex, we now have the following geometric observation, which is also presented in Figure 3.2 [41].

Lemma 3.14. *Let $G = (V, E)$ be a UDG with representation $p_v = (x_v, y_v)$ for every $v \in V$, and be $\bar{v} \in V$ such that $x_{\bar{v}} = \min\{x_u \mid u \in V\}$. Then, the cardinality of an independent set in $\Gamma(\bar{v})$ is at most 3.*

Proof. Let $v_1, v_2 \in \Gamma(\bar{v})$ be two vertices such that $(v_1, v_2) \notin E$, that is, both $\|p_{\bar{v}} - p_{v_i}\| \leq 1, i = 1, 2$, and $\|p_{v_1} - p_{v_2}\| > 1$ hold. Looking at the two rays emanating from $p_{\bar{v}}$ to p_{v_1} and p_{v_2}, a straightforward geometric argumentation shows that the angle between these two rays has to be larger than $\pi/3$. Otherwise, two circles of unit diameter centered at p_{v_1} and p_{v_2} intersect.

With this observation, suppose there are 4 independent vertices in the neighborhood of $\bar{v}$, say $v_1, \dots, v_4 \in \Gamma(\bar{v})$. W.l.o.g. let the vertices be ordered as seen from $p_{\bar{v}}$. All angles between rays to the positions $p_{v_1}, \dots, p_{v_4}$ have to be larger than $\pi/3$. Thus, the overall angle at $p_{\bar{v}}$ between p_{v_1} and p_{v_4} is larger than π, and this contradicts $x_{\bar{v}}$ being the smallest x-coordinate. □

By a small modification to the analysis of Lemma 3.12, that is, charging each vertex from the partial solution at the time it is chosen by the Leftmost-Greedy,

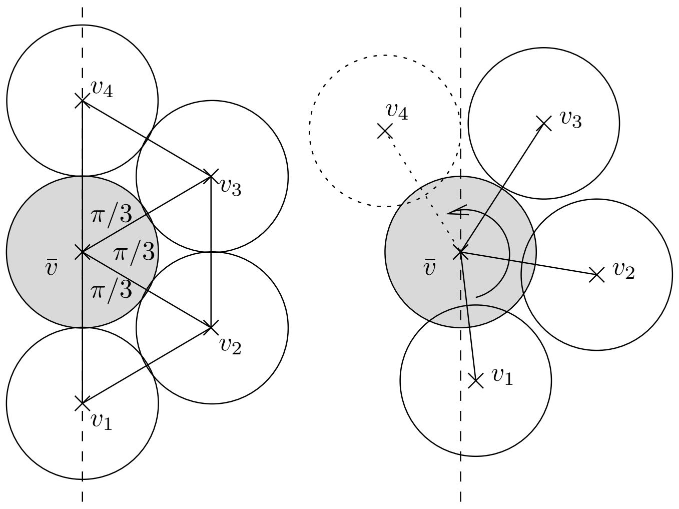


Figure 3.2: *Leftmost-Greedy*: the vertex $\bar{v}$ with smallest x-coordinate cannot have more than 3 independent neighbors.

each vertex is charged at most by three independent vertices from an optimal solution in its neighborhood. Overall, we obtain a 3-approximation with this approach.

Another example of a better approximation guarantee by a more sophisticated greedy choice can be found in Disk Graphs of arbitrary radius. Let r_v denote the radius of the respective disk centered at the position p_v of each vertex $v \in V$. A possible ordered choice of vertices to be added next to a partial solution I during a greedy approach is given by:

$$\text{Choose } \bar{v} \in V \text{ such that } r_{\bar{v}} = \min\{r_u \mid u \in V \setminus \Gamma(I)\}.$$

Similar to the proof of Lemma 3.14, we can geometrically motivate that a center disk cannot be surrounded by more than 5 disks of equal or larger radius which on one hand intersect with this central disks, but on the other hand do not intersect with each other (see Figure 3.3). Therefore, we can adjust the argumentation so that a disk chosen in this ordered greedy approach is charged by at most 5 disks with larger radius. This *Smallest-Radius-Greedy* approach then gives a 5-approximation for the MAX-IS problem on Disk Graphs, and it does so for every bound on the ratio between largest and smallest disks in the

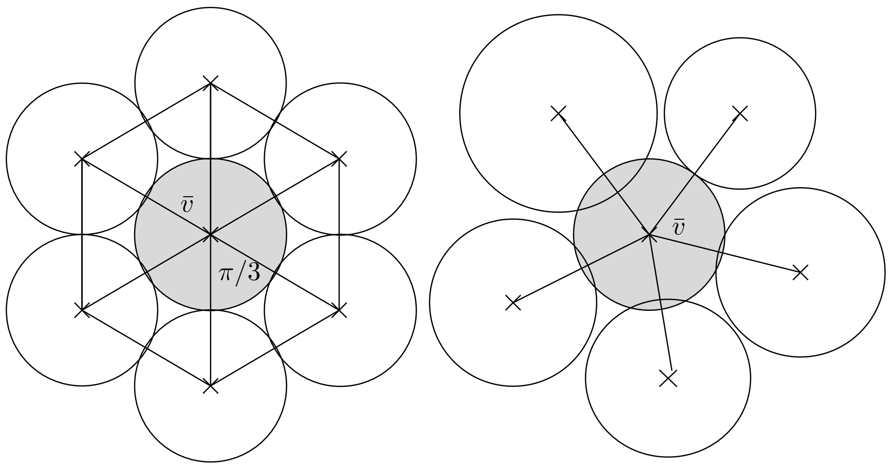


Figure 3.3: *Smallest-Radius-Greedy*: the vertex $\bar{v}$ with smallest radius cannot have more than 5 independent neighbors.

representation. Note that this bound is reflected in the approximation guarantee of an arbitrary maximal independent set with respect to an optimal solution (Section 3.2.2). With the Smallest-Radius-Greedy we thus reduce the approximation guarantee from $f(1) = O(b^2)$ to 5, independent of the value for b.

In both of the above examples, we have explicitely exploited geometric information, either the positions or radii of the disks corresponding to the vertices in a geometric representation. This raises a natural question

- whether the approaches inherently depend on this information, and
- how to make this information available to a (greedy) algorithm in case we are just given adjacency information of the graph, and the general knowledge that the (Unit) Disk Graph structure is satisfied.

While the latter question is discussed in the following part, we briefly explain how to achieve the above approximation ratio without the geometric knowledge. Taking a closer look at the geometric proof of Lemma 3.14, for a Unit Disk Graph, we may assume that a geometric representation exists without being given as part of the input to the greedy algorithm. For the approach, we therefore only need to find a vertex that does not have more than three independent vertices in its first order neighborhood. This can be done in time $O(n^5)$ by a complete search over all vertices, together with all choices of 4 neighboring vertices which may then not be mutually independent.

For the general Disk Graph case, we can find a vertex with no more than 5 independent neighbors to be added in time $O(n^7)$. The complexities of $O(n^5)$ and $O(n^7)$ can be improved by matrix multiplication techniques as proposed in [15].

The above approaches without representation take significantly longer than explicitly exploiting a representation. When the geometric information for the ordered greedy strategies is available, we can preprocess the input graph and sort the vertices accordingly to obtain the required order. We therefore explore the idea of constructing a geometric representation from adjacency information in the following part.

3.4 Graph Representations and Localization

Important problems in the area of geometrically defined graphs are the questions concerning recognition and reconstruction, both from a theoretical and a practical point of view. Generally speaking, these problems start with an undirected graph, where only adjacency information of the edges is given, and then seek for geometric information, e.g. possible positions of the vertices.

Consider the case of Unit Disk Graphs, where the edges can be encoded by giving each vertex a position in the Euclidean space, and edges are present between vertices if and only if the unit diameter disks at these positions intersect. Note that in the Leftmost-Greedy algorithm of the previous section, we assumed that such a representation exists. However, we do not need to exploit any information of such a representation in the algorithm to obtain the approximation guarantee of 3.

This scenario gives rise to one of the most fundamental problems for geometrically defined graphs (and structured subclasses of graphs in general), the recognition problem: given a graph, verify if this graph satisfies the geometric properties that define the structure.

When only adjacency information is available from a Unit Disk Graph G, it becomes hard to distinguish G from an undirected graph without underlying structure. In fact, an efficient algorithm to recognize Unit Disk Graphs can be used to efficiently solve *NP*-complete problems.

Theorem 3.15. *The problem of deciding whether a given graph $G = (V, E)$ is a Unit Disk Graph is NP-hard.*

The above theorem is proven in [8]: Given an instance C of SAT, a corresponding graph G_C is constructed in polynomial time such that G_C has a feasible representation if and only if C is satisfiable, i.e. C is a positive instance. However, even though UDG recognition is a decision problem, the question of a polynomial size certificate is still open, so that the claim of UDG recognition being *NP*-complete cannot be justified.

Any polynomial time algorithm computing a representation for a UDG could be used in a straightforward way to solve the recognition problem. Thus, we immediately get the following corollary to Theorem 3.15.

Corollary 3.16. *For a UDG $G = (V, E)$, if computing a feasible representation $p : V \to \mathbb{R}^2$ can be done in polynomial time, then $P = NP$ holds.*

Due to this fact, approximations to the problem of computing a representation, which are usually referred to as *embedding* of a graph come into the picture. Consider a general, undirected graph $G = (V, E)$ together with some mapping $p : V \to \mathbb{R}^2$ which gives each vertex a position in the Euclidean space. We define the quality q_p of the embedding p as

$$q_p(G) := \frac{\max_{(u,v)\in E} \|p_u - p_v\|}{\min_{(\bar{u},\bar{v})\notin E} \|p_{\bar{u}} - p_{\bar{v}}\|}.$$

In case of the geometrically defined wireless communication graphs of the preceding section, this measure captures the essence of wireless communications: the smaller q_p, the closer we get to the characteristic that vertices which are close to one another can communicate directly, and vertices far apart cannot.

It is easy to see that an embedding p with quality $q_p(G) \geq 1$ defines a Quasi Disk Graph with parameter $b = q_p$ as defined in Section 3.2.2. For a Unit Disk Graph $G = (V, E)$, it is also easy to verify that an embedding p^* is a representation of G if and only if $q_{p^*}(G) < 1$ holds.

A more detailed look at the construction of the graph G_C used in the proof of Theorem 3.15, this time in light of the above measure, gives the following result [34].

Theorem 3.17. *Let $G = (V, E)$ be a Unit Disk Graph. If $P \neq NP$, then there is no efficient algorithm that computes an embedding $p : V \to \mathbb{R}^2$ with quality $q_p(G) \leq \sqrt{3/2} - \epsilon$, where ϵ tends to 0 as $|V| \to \infty$.*

On the other hand, in [42], an $O(\log^{2.5} n\sqrt{\log\log n})$-approximation algorithm for the embedding of a Unit Disk Graph is proposed.

Despite being challenging from a theoretical point of view, also in practice, these questions play an important role in wireless networks. In this area, giving each wireless device its position in the physical environment is known as localization. However, practicable, reliable localization in wireless networks is a non-trivial task [56], and there are numerous in-network approaches to this problem proposed in the literature. These are then usually evaluated numerically in ideal settings. Furthermore, for example after the initial employment of a wireless sensor network, positional information may not yet be available. Nevertheless, in this setting, efficient communication structures—including independent and dominating sets—need to be set up as well. Of course, a rather quick—yet expensive—solution is to equip each device with a GPS or Galileo receiver.

3.5 Robust Algorithms

From the above discussion, it becomes clear that in wireless communication graphs, the question of geometric representation or localization is an important one. The information how a graph is presented as input to an algorithm, e.g. by its adjacency information only, or by its complete geometric representation, determine significant differences in the design of optimization algorithms.

In this context, when a geometric representation is not available, the notion of *robustness* comes into play. Generally speaking, a robust algorithm must produce correct output regardless of the input [53]. More precisely, robust algorithms on a subset of instances of a problem are defined as follows.

Definition 3.18. *Let $\mathcal{A}$ be an algorithm defined on a set $\mathcal{I}$ of instances, and let f be a function defined on $\mathcal{I}$. Furthermore, let $\mathcal{U} \subseteq \mathcal{I}$.*

*Then, we say that $\mathcal{A}$ **computes** f **robustly** (on $\mathcal{U} \subseteq \mathcal{I}$) if*

- *for all instances $x \in \mathcal{U}$, the algorithm $\mathcal{A}$ returns $f(x)$, and*
- *for all instance $x \in \mathcal{I} \setminus \mathcal{U}$, the algorithm $\mathcal{A}$ returns either $f(x)$, or a certificate showing that $x \notin \mathcal{U}$.*

Of course, the notion of a robust algorithm is especially interesting when $\mathcal{A}$ has polynomial running time with respect to the size of the input, and the decision whether an instance belongs to the subclass $\mathcal{U} \subseteq \mathcal{I}$ is not easy to decide.

For example, keeping Theorem 3.15 in mind, an algorithm $\mathcal{A}$ may robustly compute some function f on Unit Disk Graphs, and accept any undirected graph as input. An output then either gives a value for f, or a certificate that the input is not a Unit Disk Graph. Note that, in case a value for f is returned, we may *not* conclude that the input is a Unit Disk Graph. However, $\mathcal{A}$ computed the correct value for f.

Furthermore, robust algorithms can be combined in sequence, where the computed solution is used as input for further algorithms, that is, as building blocks of more general algorithms. In other words, robustness is preserved by composition. This is to be contrasted to the non-robust case where an algorithm need not produce an output or even terminate if the input does not satisfy the additional structural properties, and some produced output may not even be a feasible solution to the problem at hand.

3.6 Conclusions

In this chapter, we have reviewed and proposed geometrically inspired containment and intersection graph models for communication in wireless, ad-hoc networks. These range from ideal assumptions, resulting in a Unit Disk Graph, to more realistic approaches, resulting in Bounded Area Graphs. All these graph models have in common the structure of polynomially bounded growth if the respective parameters are bounded.

The bounded growth property allows for a constant-factor bound on the ratio between the cardinalities of maximal independent sets and optimal solutions to the MAX-IS and MIN-DS problems respectively. Furthermore, since a Maximum Independent Set is also maximal, this ratio also holds between optimal solutions to these two problems on the same instance.

For the geometric intersection graphs, possibly exploiting geometric information, we can further improve the approximation ratio given by a maximal independent set for some classes of graphs. We have seen that having a geometric representation available differs greatly from the case that we only have adjacency information. The questions of embedding, or localization, were discussed. We also introduced the notion of robust algorithms. These type of algorithms present a way to deal with optimization on the presented subclasses of undirected graphs.

In the following, since bounded growth is a structural property of a graph, we assume the growth function or polynomial to be known, and not part of the input. This is reasonable as the actual graph is an instance of the optimization problems we are considering, whereas the growth function is defined for possible instances reflecting wireless communication graphs.

Chapter 4

Polynomial-Time Approximation Schemes

This chapter presents polynomial-time approximation schemes for Maximum Independent Set and Minimum Dominating Set problems on polynomially Bounded Growth Graphs.
We briefly introduce the algorithms for Unit Disk Graphs which rely on a given geometric representation. The main focus, however, lies on an approach called local neighborhood-expansion scheme, that no longer relies on a geometric representation but only requires polynomially bounded growth.
The presented approach can be extended towards a robust algorithm that works on any undirected graph and either returns a subset of desired quality, or gives a polynomial time certificate showing that the instance does not reflect a graph of polynomially bounded growth.

In this chapter, we present polynomial-time approximation schemes for the Max-IS and Min-DS problems on graphs of polynomially bounded growth. Recall that a PTAS is a family of algorithms which in addition to an instance yield a desired approximation guarantee $(1 + \varepsilon)$, given by a parameter $\varepsilon > 0$ as part of the input. The run time, for fixed ε, has to be polynomial.

We start by introducing main concepts that yield existing PTASs on geometrically defined graphs such as Unit Disk Graphs, and that explicitely require geometric information. While these approaches have been known for some time, the question whether there is also an approach for the case that a geometric representation is not available, was an open problem. As we have seen in Section 3.4, this non-geometric case is significantly more difficult due to the hardness of computing a feasible representation. We thus abandon the geometry thereafter and focus on graphs of polynomially bounded growth, which also includes an answer to this question.

For these polynomial BGGs, we present an approach based on adjacency information only that yields PTASs for the Max-IS and Min-DS problems. The resulting algorithms work free of any geometric information, and thus give a positive answer to the above question of a PTAS without representation. The approach for both problems constructs optimal, partial solutions for locally bounded neighborhoods, and combines these to create a globally feasible solution which satisfies the desired approximation guarantee. The algorithms, also for the weighted version of the Max-IS problem, are explained in Section 4.2, and follow the ideas of [48] and [47].

Besides the independence from a geometric representation, an additional advantage of the PTAS lies in the fact that we can extend the algorithm towards a *robust* approach. We have a polynomial time algorithm which either approximates the problem, or solves the recognition problem. Note that the algorithm does *not* solve the recognition problem in general. If the graph is not geometrically defined, it may also return a desired, $(1 + \varepsilon)$-approximate solution. This is further discussed in Section 4.3.

4.1 Schemes Based on Geometric Separation

Most of the existing work on approximation schemes for wireless communication graphs is reported for (Unit) Disk Graphs, and assume the graph given by its geometric representation. We would like to point out again that both Max-IS and Min-DS problems are *NP*-hard even when the input is restricted to Unit Disk Graphs with a given representation [10]. However, when the representation of a UDG is given as part of the input, we can use *geometric separation* to obtain a PTAS for many problems on a UDG. This is done by dividing the graph into smaller subgraphs, for which an optimal partial solution can be obtained efficiently.

Consider a problem on a geometrically defined graph that can be solved by partitioning the graph for a divide-and-conquer approach. Such an approach first partitions the graph depending on some parameter. The parameter is usually chosen so that the range of this parameter is bounded, giving a bounded number of possible ways to partition the graph. Then, for each choice of the parameter, the respective (sub-) problem is solved giving a feasible candidate solution to the global problem. The overall solution is then given by taking the best of the candidate solutions, which then gives a good approximation.

A special separation strategy called *shifting* allows for a bound on the performance ratio of a simple divide-and-conquer approach. This strategy is, for example, used to obtain approximation schemes for problems restricted to planar graphs [6], or for certain geometric packing and covering problems [29].

Due to its simplicity, we present the basic shifting strategy for the Maximum Independent Set problem [30], and then discuss how to modify this approach for the Minimum Dominating Set problem. Several improvements and extensions are also discussed.

4.1.1 Maximum Independent Set

Let the Unit Disk Graph $G = (V, E)$ be given by its geometric representation, i.e. the set of center points $\{p_v \in \mathbb{R}^2 \mid v \in V\}$ of the unit disks that create the intersection graph, and let $A_G \subseteq \mathbb{R}^2$ denote the total rectangular area needed to draw the graph. Given $\varepsilon > 0$, let k be the smallest integer such that $(\frac{k}{k+1})^2 \geq \frac{1}{1+\varepsilon}$ holds. For each $0 \leq i \leq k$ and $0 \leq j \leq k$, consider the following separation of G.

We divide A_G into vertical and horizontal strips according to left-open unit intervals $]l, l+1], l \in \mathbb{N}$. We assume these intervals to be enumerated according to the distance l from the origin. Let $G_{i,j}$ then denote the subgraphs obtained by removing all vertices whose center-points fall into the vertical intervals congruent to i modulo $(k+1)$ and into the horizontal intervals congruent to j modulo $(k+1)$. The graph $G_{i,j}$ thus consists of independent subgraphs, where the respective vertices' center points are all drawn inside $k \times k$ sized boxes. Figure 4.1 gives an example of such a separated Unit Disk Graph, the removed disks are dashed.

A candidate independent set $I_{i,j}$ in $G_{i,j}$ can now be obtained by calculating

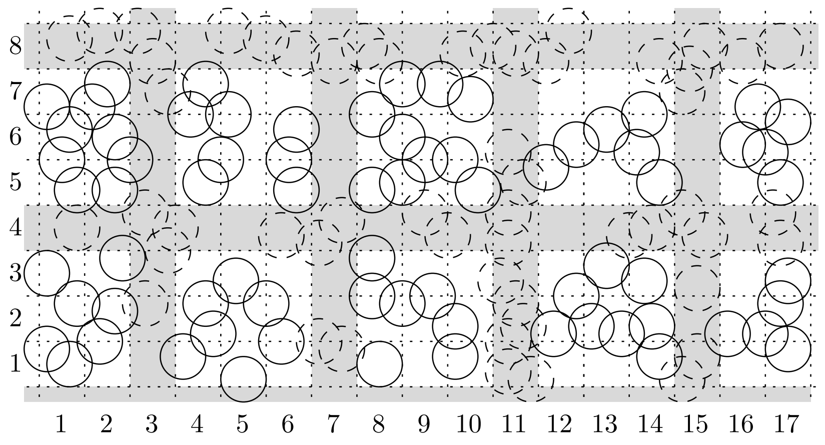


Figure 4.1: Separated representation of a UDG ($k = 3, i = 3, j = 0$), Max-IS case.

a Maximum Independent Set in all boxes separately. For each box, the size of any independent set therein is bounded by the area $O(k^2)$ by a simple packing argument of non-overlapping disks similar to Theorem 3.5. Therefore, an optimal solution $I_{i,j}$ can be computed in $n^{O(k^2)}$, e.g. by complete enumeration.

Each of the solutions $I_{i,j}$ forms a candidate for an approximation of the Maximum Independent Set in G. Let I be the largest set among these candidate sets, i.e. $|I| := \max_{i,j} |I_{i,j}|$, which is returned as overall solution. A detailed summary of the approach to compute I is given by Algorithm 3, and the following theorem establishes the $(1 + \varepsilon)$ approximation guarantee.

Theorem 4.1. *Let I^* be a Maximum Independent Set in a UDG $G = (V, E)$. The set I computed by Algorithm 3 is independent and satisfies*

$$(1 + \varepsilon) \cdot |I| \geq |I^*|.$$

Proof. Clearly, by the separation of the boxes, each $I_{i,j}$ consists of separated independent sets inside the boxes, and the solution I is thus an independent set.

Consider the vertical separation, and let G_i be the resulting graph For each $i = 0, \dots, k$, let S_i denote the set of removed vertices, i.e. $S_i := V(G \setminus G_i)$. Observe that, for $0 \leq i_1 < i_2 \leq k$, we have $S_{i_1} \cap S_{i_2} = \varnothing$, and $\bigcup_{i=0}^{k} S_i = V$.

Algorithm 3 PTAS Maximum Independent Set (with representation).

```
Input: Representation $\{p_v = (x_v, y_v) \mid v \in V\}$ of a UDG $G = (V, E)$, $\varepsilon > 0$
Output: $(1+\varepsilon)$-approx. Maximum Independent Set $I$
 1: Compute minimum $k \in \mathbb{N}$ such that $(\frac{k}{k+1})^2 \geq \frac{1}{1+\varepsilon}$;
 2: $I := \varnothing$;
 3: for $i := 0$ to $k$ do
 4:    Construct $G_i$ by removing all vertices with $x_v \in \,]l, l+1]$ from $G$,
       $l \equiv i \mod (k+1)$;
 5:    for $j := 0$ to $k$ do
 6:       Construct $G_{i,j}$ by removing all vertices with $y_v \in \,]l, l+1]$ from $G_i$,
          $l \equiv j \mod (k+1)$;
 7:       Compute Maximum Independent Set $I_{i,j}$ in $G_{i,j}$;
 8:       if $|I| < |I_{i,j}|$ then $I := I_{i,j}$;
 9:    end for
10: end for
```

With respect to an optimal solution I^*, we then obtain

$$|S_0 \cap I^*| + |S_1 \cap I^*| + \cdots + |S_k \cap I^*| = |I^*|.$$

Therefore, looking at the average value over the $k+1$ summands, it is

$$\min_{0 \leq i \leq k} |S_i \cap I^*| \leq \frac{|I^*|}{k+1},$$

and

$$\max_{0 \leq i \leq k} |V(G_i) \cap I^*| = |I^*| - \min_{0 \leq i \leq k} |S_i \cap I^*| \geq \frac{k}{k+1} |I^*|.$$

By the same argumentation, this time for fixed G_i and the horizontal separation $G_{i,j}, j = 0, \ldots, k$, we have

$$\max_{0 \leq j \leq k} |V(G_{i,j}) \cap I^*| \geq \frac{k}{k+1} |V(G_i) \cap I^*|.$$

Combining these inequations gives

$$\max_{0 \leq i,j \leq k} |V(G_{i,j}) \cap I^*| \geq \left(\frac{k}{k+1}\right)^2 |I^*|.$$

Since the candidate solutions $I_{i,j}$ are optimal, we have $|I_{i,j}| \geq |V(G_{i,j}) \cap I^*|$, and the claim follows. □

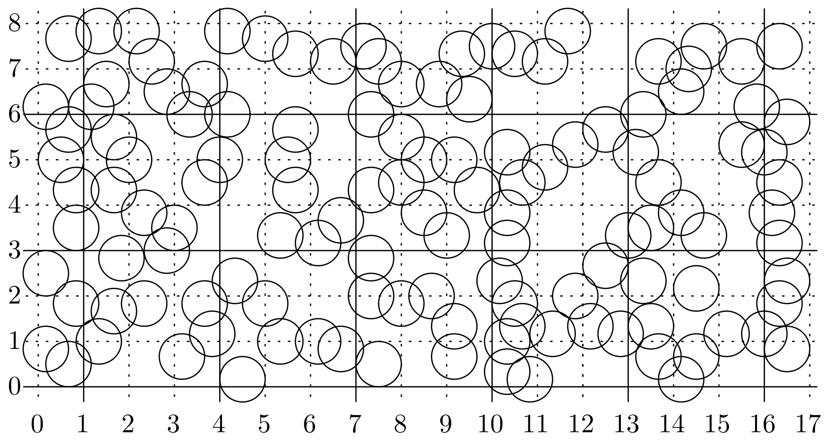


Figure 4.2: Separated representation of a UDG ($k = 3, i = 1, j = 0$), MIN-DS case.

Clearly, Algorithm 3 has polynomial run time, which is dominated by the computation of the $(k+1)^2$ different candidate solutions $I_{i,j}$ in time $n^{O(k^2)}$. Note that k is bounded depending on ε. Also, an extension to higher dimensions is straightforward. Furthermore, it is easy to adapt the algorithm and the above proof to the weighted version of the problem, by replacing the cardinality of the subsets with their respective weight.

There are several improvements to the above generic separation approach. By applying dynamic programming techniques along one of the separating dimensions, the time bound can be reduced to $n^{O(k)}$ by giving an optimal solution in G_i for each $i = 0, \ldots, k$. This idea can be extended to geometric objects in $\mathbb{R}^d, d \geq 2$, whose projection to the first $d-1$ coordinates have roughly the same size, e.g. for map labeling problems treated in [1]. Combined with a dynamic programming approach which simultaneously considers disks of different size, the shifting strategy also yields a PTAS for General Disk Graphs [16], thus no longer requiring the disks to have roughly the same radii.

4.1.2 Minimum Dominating Set

The geometric separation approach can also be used to obtain a PTAS for the Minimum Dominating Set problem on UDGs. We now present a basic $(\frac{k+2}{k})^2$-approximation algorithm also using the shifting strategy. The parameter $k \in \mathbb{N}$

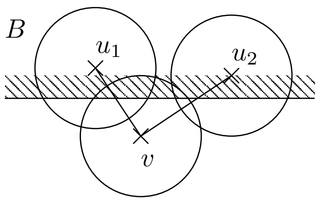


Figure 4.3: Using $v \notin B$ as dominating vertex for $u_1, u_2 \in B$.

is then chosen so that $(\frac{k+2}{k})^2 \leq (1+\varepsilon)$ holds.

For the dominating set approach, there are some subtle differences to a PTAS for maximum independent sets:

- Geometric separation cannot be done by removing vertices from horizontal and vertical strips. A subset of a dominating set no longer needs to be dominating. Therefore, we separate along horizontal lines at $l \equiv i \mod k$ and vertical lines at $t \equiv j \mod k$ (Figure 4.2). This yields boxes of the form $]l, l+k] \times]t, t+k]$, and for fixed i, j, the boxes form a partition of G. This way, it is ensured that each vertex in G is dominated.

- A minimum cardinality set dominating the vertices of a box B, say e.g. $B =]l, l+k] \times]t, t+k]$ for some l and t, may contain vertices from adjacent strips: A vertex $v \in V$, with $p_v \notin B$, but whose unit disk intersects with disks centered inside B can be used to dominate these vertices, see Figure 4.3.

So, for the computation of an optimal dominating set for a $k \times k$ sized box, we have to consider adjacent vertices, as well. For the construction of a partial solution, we expand each box by one unit in each direction, that is, we may use vertices from this expanded $(k+2) \times (k+2)$ sized box to dominate vertices inside the $k \times k$ sized box.

The construction of the candidate solution $D_{i,j}$ for a set of boxes $\mathcal{B}_{i,j}$ can still be done efficiently. Any maximal independent set in a $k \times k$ sized box is of cardinality $O(k^2)$, and by Theorem 2.10, such a set is also dominating. Thus, even considering an enlarged box for an optimal dominating set, the cardinality of such a set remains $O(k^2)$, and can be constructed, e.g. by complete enumeration in $n^{O(k^2)}$. Due to k being bounded depending on ε, the overall run time of the algorithm is thus polynomial.

Algorithm 4 presents the modified PTAS for the Min-DS problem on UDGs

Algorithm 4 PTAS Minimum Dominating Set (with representation).

```
Input: Representation $\{p_v = (x_v, y_v) \mid v \in V\}$ of a UDG $G = (V, E)$, $\varepsilon > 0$
Output: $(1+\varepsilon)$-approx. Minimum Dominating Set $D$
1: Compute minimum $k \in \mathbb{N}$ such that $(\frac{k+2}{k})^2 \leq 1+\varepsilon$;
2: $D := \varnothing$;
3: for $i := 0$ to $k-1$ do
4:     for $j := 0$ to $k-1$ do
5:         Partition $V$ along the $k \times k$ sized boxes yielding sets $\mathcal{B}_{i,j} :=$
           $\{v \in V \mid p_v \in ]l, l+k] \times ]t, t+k], l \equiv i \mod k, t \equiv j \mod k\}$;
6:         For each $B \in \mathcal{B}_{i,j}$, compute a Minimum Dominating Set for $B$ in $G$,
           and combine these to $D_{i,j}$;
7:         if $|D| > |D_{i,j}|$ then $D := D_{i,j}$;
8:     end for
9: end for
```

with representation. For the quality of the solution, we have the following theorem.

Theorem 4.2. *For a UDG $G = (V, E)$, let D^* denote a Minimum Dominating Set of G. Then, Algorithm 4 constructs a dominating set D of cardinality*

$$|D| < (1+\varepsilon) \cdot |D^*|.$$

Proof. It is easy to see that D dominates V since the $k \times k$ sized boxes for each choice of i and j are cover the entire graph.

Consider the vertical separation only. Then, for some iteration $0 \leq i' < k$, at most $\frac{2}{k} \cdot |D^*|$ vertices fall into the additional unit intervals left and right of the lines that partition the graph. This is due to the fact that each vertical unit interval, over all k iterations, is considered at most once left of each strip, and at most once right of each vertical strip of width k.

The cardinality of the dominating set for all these strips, given by the union of the optimal dominating sets for each extended strip $]l-1, l+k+1]$ with $l \equiv i' \mod k$, is then no more than $|D^*| + \frac{2}{k} \cdot |D^*| = \frac{k+2}{k} \cdot |D^*|$.

By the same argumentation, we can obtain a $(\frac{k+2}{k})$-approximate dominating set inside each of the above horizontal strips, and sequential application yields the claim. □

Again, from the proof, an extension to higher dimensions is straightforward. A slightly modified algorithm with adjacent $k \times k$ sized boxes gives a PTAS for the Minimum Vertex Cover problem [30]. For the weighted dominating set, this approach is no longer applicable due to the fact that the cardinality of a weighted dominating set inside a box can no longer be bounded. In [9], the above approach is extended towards the Minimum Connected Dominating Set problem.

4.2 Local Neighborhood Based Schemes

As seen in Section 3.4, the case when no geometric representation is available is significantly different already for UDGs. We cannot easily make this information available to an algorithm. Next, we go on to present an approach that no longer relies on positional information of the vertices, but assumes only knowledge of the adjacency of each vertex in the graph. Additionally, we no longer restrict the discussion to UDGs, but assume the graph $G = (V, E)$ to be of polynomially bounded growth, thus completely abandoning any underlying geometric structure.

The following discussion follows [48] and [47], where robust PTASs for the MAX-IS and MIN-DS problems respectively are presented for UDGs without geometric representation. In order to simplify the presentation of the approximation schemes, we introduce some definitions and notation, including the basic definition of collections of d-separated neighborhoods in G.

Definition 4.3. *For a graph $G = (V, E)$, let $\mathcal{S} = \{S_1, \ldots, S_k\}$ be a collection of subsets of vertices $S_i \subseteq V, i = 1, \ldots, k$, with the following property:*

$$\textit{for any two vertices } s \in S_i \textit{ and } \bar{s} \in S_j, i \neq j,\ d_G(s, \bar{s}) > d.$$

*We refer to $\mathcal{S}$ as a **d-separated** collection of subsets.*

It is easy to see that the subsets of any d-separated collection, $d \geq 0$, are mutually disjoint. We call a d-separated collection $\mathcal{S} = \{S_1, \ldots, S_k\}$ *exhaustive* if for every vertex $v \in V$, there exists a vertex $s \in \bigcup S_i$ such that $d_G(v, s) \leq d$. An example of an exhaustive 2-separated collection is given in Figure 4.4. The grey areas mark the different subsets that make up the collection; vertices which are not part of it, and thus separate the subsets, are white.

Denote by $\mathcal{P}(V)$ the set of all subsets of vertices. We then define two functions $I : \mathcal{P}(V) \to \mathcal{P}(V)$ and $D : \mathcal{P}(V) \to \mathcal{P}(V)$ which return an independent set of maximum cardinality and a dominating set of minimum cardinality respectively for the subset given as argument. For a subset $V' \subseteq V$, the set

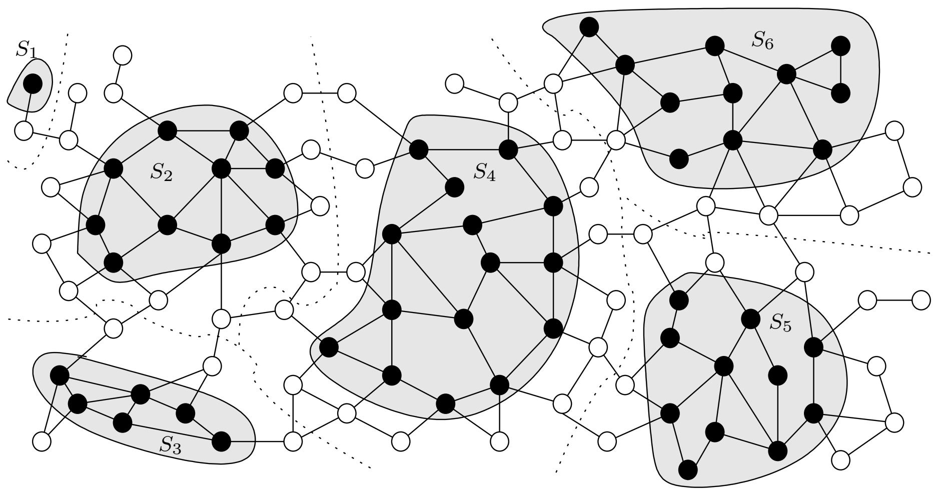


Figure 4.4: Example of a 2-separated collection $\mathcal{S} = \{S_1, \dots, S_6\}$

$I(V')$ is independent in V', and $D(V')$ dominates V'. For $D(V')$, the inclusion $D(V') \subseteq V'$ needs not to hold. Therefore, in the following, the function $D(.)$ is always computed with respect to the entire underlying graph G. However, it is easy to see that $V' \subseteq \Gamma(D(V'))$ and that $D(V') \subseteq \Gamma(V')$ hold.

Using the above definitions, we are thus interested in a polynomial time approximation of $I(V)$ and $D(V)$ within a factor of $1+\varepsilon$ for any given $\varepsilon > 0$. Due to the intractability of the corresponding optimization problems, we resort to an implicit divide-and-conquer approach using local neighborhoods and partial solutions therein.

For some vertex $v \in V$, and its r-th neighborhood $\Gamma_r(v)$, we use $I_r(v)$ and $D_r(v)$ to denote the independent set $I(\Gamma_r(v))$ and the dominating set $D(\Gamma_r(v))$ for $\Gamma_r(v)$. Further on, in case the vertex v is unambiguous, we omit indexing the respective neighborhoods and subsets with this central vertex.

Suppose the radius r of a neighborhood Γ_r is bounded. Then by the definition of Bounded Growth Graphs, and the fact that any maximal independent set is also dominating (Theorem 2.10),

$$|D_r| \leq |I_r| \leq p(r)$$

holds. With this bound on the cardinality of the locally optimal solutions, it becomes clear that we can obtain both optimal solutions I_r and D_r in time $n^{O(p(r))}$ for this neighborhood Γ_r.

4.2.1 Maximum Independent Set

We now present an approach that gives the PTAS for the Maximum Independent Set problem on a polynomial BGG $G = (V, E)$. The basic idea of is simple. We start with an arbitrary vertex $v \in V$, and consider for $r = 0, 1, 2, \ldots$, the r-th neighborhoods $\Gamma_r(v) = \Gamma_r$ and optimal independent sets $I_r(v) = I_r \subseteq \Gamma_r$ therein.

We keep expanding the neighborhoods as long as

$$|I_{r+1}| > (1 + \varepsilon) \cdot |I_r|$$

holds. Let $\bar{r}$ denote the smallest $r \geq 0$ for which this condition is violated. Such an $\bar{r}$ indeed exists, and it is bounded by a constant that only depends on ε.

Lemma 4.4. *Let $G = (V, E)$ be a graph of polynomially p-bounded growth. There exists a constant $c = c(\varepsilon)$ such that $\bar{r} \leq c$.*

Proof. Let $r < \bar{r}$. By definition of $\bar{r}$, we then have for r

$$|I_r| > (1 + \varepsilon)|I_{r-1}| > \cdots > (1 + \varepsilon)^r |I_0| = (1 + \varepsilon)^r.$$

Since the graph G is of polynomial bounded growth, we also have $|I_r| \leq p(r)$. By comparison, i.e.

$$p(r) \geq |I_r| > (1 + \varepsilon)^r,$$

the claim follows. □

To achieve an independent set for the graph G, we iteratively apply the above scheme. Each time the expansion is stopped, we remove the neighborhood $\Gamma_{\bar{r}+1}$ from G, and combine $I_{\bar{r}}$ with the partial solution I obtained thus far. Since $I_{\bar{r}} \subseteq \Gamma_r$, while we remove $\Gamma_{\bar{r}+1}$, the set $\Gamma_{\bar{r}}$ is 1-separated to all sets to be calculated in the remaining process. This implies that the created sets $\Gamma_{\bar{r}}$ during the complete algorithm form a 1-separated collection. This collection of subsets is also exhaustive since we continue creating partial solutions until the remaining graph is empty.

A detailed summary of the above approach is given by Algorithm 5, and we now prove its correctness and polynomial complexity starting with the independence property of the returned solution.

Algorithm 5 PTAS Maximum Independent Set.

```
Input: $G = (V, E)$ polynomial BGG, $\varepsilon > 0$
Output: $(1 + \varepsilon)$-approx. Maximum Independent Set $I$
 1: $I := \varnothing$;
 2: while $V \neq \varnothing$ do
 3:     Pick $v \in V$;
 4:     $r := 0$;
 5:     while $|I_{r+1}(v)| > (1 + \varepsilon) \cdot |I_r(v)|$ do
 6:         $r := r + 1$;
 7:     end while
 8:     $I := I \cup I_r(v)$;
 9:     $V := V \setminus \Gamma_{r+1}(v)$;
10: end while
```

Lemma 4.5. *The solution I created by Algorithm 5 forms an independent set in the graph G.*

Proof. This follows directly from the separation of the neighborhoods considered: each $v \in V \setminus \Gamma_{\bar{r}+1}$ has no neighbor in $\Gamma_{\bar{r}}$, and thus not in $I_{\bar{r}} \subseteq \Gamma_{\bar{r}}$. □

In order to motivate that all operations can be completed in polynomial run time, note that we only need to consider a single iteration, as the number of new central vertices picked in the algorithm is limited by n. Due to the definition of a polynomial BGG, and the constant established by Lemma 4.4, every operation can be completed in polynomial time, with the degree only depending on $\varepsilon > 0$, but not on the size of the graph. It remains to show that the cardinality of the independent set I meets the desired approximation guarantee of $(1 + \varepsilon)$.

Theorem 4.6. *Let $I^* := I(V)$ be a Maximum Independent Set in a polynomial BGG $G = (V, E)$. The independent set I computed by Algorithm 5 satisfies*

$$(1 + \varepsilon) \cdot |I| \geq |I^*|.$$

Proof. Suppose inductively that Algorithm 5 computes a $(1 + \varepsilon)$-approximate independent set $I' \subseteq V \setminus \Gamma_{\bar{r}+1}$ for $G' = G[V \setminus \Gamma_{\bar{r}+1}]$.

By definition of $\bar{r}$, we have

$$|I_{\bar{r}+1}| \leq (1 + \varepsilon) \cdot |I_{\bar{r}}|.$$

Since the cardinality of the part of the optimal set I^* which lies in $G[\Gamma_{\bar{r}+1}]$ is bounded by the cardinality of $I_{\bar{r}+1}$, we get

$$|\Gamma_{\bar{r}+1} \cap I^*| \leq |I_{\bar{r}+1}| \leq (1+\varepsilon) \cdot |I_{\bar{r}}|.$$

Further, by inductive assumption, I' is $(1+\varepsilon)$-approximately optimal for G'. Therefore,

$$|V(G') \cap I^*| \leq |I(V(G'))| \leq (1+\varepsilon) \cdot |I'|.$$

Adding the two inequalities, we obtain

$$|I^*| \leq (1+\varepsilon) \cdot (|I_{\bar{r}}| + |I'|) = (1+\varepsilon) \cdot |I|.$$

□

Note that the Algorithm 3 actually returns a $(1+\varepsilon)$-approximate independent set for any undirected graph given as input. We have only used the specific structure of polynomial BGGs in Lemma 4.4 in order to bound the radius of the largest neighborhood we need to consider during execution. The criterion to stop expanding a neighborhood is met eventually in any undirected graph, however, possibly while considering an $O(n)$-neighborhood or eventually G itself.

Coming back to the run time of Algorithm 5, we see that the time needed for completion of the algorithm is dominated by the constant $c = c(\varepsilon)$ of Lemma 4.4. To be more precise, the run time dominated by the time needed to construct an optimal solution for $\Gamma_{\bar{r}+1}(v), \bar{r} \leq c$.

Using the inequality $\log(1+\varepsilon) > 1/2 \cdot \varepsilon$ for sufficiently small ε, we can bound the constant c by $O(1/\varepsilon \log 1/\varepsilon)$. The overall run time of the approach is thus $n^{O(1/\varepsilon \log 1/\varepsilon)}$.

4.2.1.1 Maximum Weight Independent Set

The previous approximation scheme can easily be adapted for the case that each vertex $v \in V$ is given a weight w_v. Without loss of generality, we assume $w_v > 0$ for every $v \in V$. Recall that in this case, we are interested in an independent set $I \subseteq V$ of high total weight $W(I) := \sum_{i \in I} w_i$ in G. We now present the necessary modifications to the previous algorithm in order to obtain an independent set of weight at least $(1+\varepsilon)^{-1}$ the maximum total weight of any independent set in the polynomial BGG.

The algorithm again follows the idea of expanding the local neighborhood of a central vertex v. This time, however, the central vertex v is chosen to be a vertex with maximum weight $w_v = \max\{w_i \mid i \in V\}$ in the remaining graph G. Then, we compute an independent set $I_r \subseteq \Gamma_r(v)$ of *maximum weight* for increasing radii r until the criterion

$$W(I_{r+1}) > (1+\varepsilon) \cdot W(I_r)$$

is violated. Let $\bar{r}$ denote the smallest $r \geq 0$ for which this is the case.

Lemma 4.7. *Let $G = (V, E)$ be a p-BGG. There exists a constant $c = c(\varepsilon)$ such that $\bar{r} \leq c$.*

Proof. Adapting the proof of Lemma 4.4, for $r < \bar{r}$ we get

$$W(I_r) = \sum_{i \in I_r} w_i \leq \sum_{i \in I_r} w_{\max} = |I_r| \cdot w_{\max},$$

and

$$W(I_r) > (1+\varepsilon) \cdot W(I_{r-1}) > \ldots > (1+\varepsilon)^r \cdot W(I_0) = (1+\varepsilon)^r \cdot w_{\max}$$

respectively. Since $|I_r| \leq p(r)$, combining the above two inequalities again yields the claim. □

As a consequence, the running time of this algorithm remains polynomial in the weighted case. Furthermore, the approximation ratio can be guaranteed by the following theorem.

Theorem 4.8. *The adapted algorithm, i.e. choosing a central vertex of maximum weight in G in each round, creates an independent set I of weight*

$$(1+\varepsilon) \cdot W(I) \geq W(I^*),$$

where I^* denotes an optimal solution to the Maximum Weight Independent Set problem on G.

Proof. Let $V' := V \setminus \Gamma_{\bar{r}+1}(v)$, and assume inductively that $I' \subseteq V'$ is a $(1+\varepsilon)$-approximate weighted independent set in $G[V']$. Clearly, $I_{\bar{r}} \cup I'$ is an independent set in G.

For the weighted independent set in the neighborhood $\Gamma_{\bar{r}+1}(v)$, we have

$$W(I^* \cap \Gamma_{\bar{r}+1}(v)) \leq W(I_{\bar{r}+1}) \leq (1+\varepsilon) \cdot W(I_{\bar{r}})$$

from the criterion to stop expanding the neighborhoods.

For the weight $W(I) = W(I_{\bar{r}} \cup I')$ of the overall solution set I returned by the algorithm we then get

$$\begin{aligned} W(I^*) &= W((I^* \cap \Gamma_{\bar{r}+1}(v) \cup (I^* \cap V')) \\ &= W(I^* \cap \Gamma_{\bar{r}+1}(v)) + W(I^* \cap V') \\ &\leq (1+\varepsilon) \cdot W(I_{\bar{r}}) + (1+\varepsilon) \cdot W(I') \\ &= (1+\varepsilon) \cdot W(I_{\bar{r}} \cup I') \\ &= (1+\varepsilon) \cdot W(I). \end{aligned}$$

□

Summarizing, we have the following corollary for independent sets on graphs of polynomially bounded growth.

Corollary 4.9. *There exists a PTAS for the Maximum (Weight) Independent Set problem on graphs of polynomially bounded growth. In addition to the desired approximation guarantee, this PTAS requires only adjacency information of the input graph.*

4.2.2 Minimum Dominating Set

In this part, we present a polynomial-time approximation scheme for the Minimum Dominating Set problem on graphs of polynomially p-bounded growth. The approach follows the implicit separation idea of the PTAS for the MAX-IS problem on this class of graphs presented in Section 4.2.1. However, while in the previous section, the separation strategy and the overall approximation followed by rather simple arguments, for the MIN-DS problem, some attention has to be paid to the manner the local neighborhoods are created and put together.

Again, we use local neighborhoods of bounded radius, and optimal partial solutions therein, to obtain a PTAS. Recall that for neighborhoods $\Gamma_r(v), v \in V$, in graphs of polynomially p-bounded growth, we can obtain a dominating set $D_r(v) = D(\Gamma_r(v))$ of minimum cardinality in time $n^{O(p(r))}$.

The main part of the algorithm now consists of iteratively constructing dominating sets for the local neighborhoods Γ_r, and to stop if the cardinality of these dominating sets does not grow too much any more. To be more precise, we stop expanding the radius r of the neighborhoods if

$$|D_{r+2}(v)| > (1+\varepsilon) \cdot |D_r(v)|$$

is violated.

Algorithm 6 PTAS Minimum Dominating Set.

```
Input: $G = (V, E)$ polynomial BGG, $\varepsilon > 0$
Output: $(1 + \varepsilon)$-approx. Minimum Dominating Set $D$
1: $D := \varnothing$;
2: while $V \neq \varnothing$ do
3:     Pick $v \in V$;
4:     $r := 0$;
5:     while $|D_{r+2}(v)| > (1 + \varepsilon) \cdot |D_r(v)|$ do
6:         $r := r + 1$;
7:     end while
8:     $D := D \cup D_{r+2}(v)$;
9:     $V := V \setminus \Gamma_{r+2}(v)$;
10: end while
```

Recall the possibility of vertices outside a subset being able to dominate vertices inside this subset. The local dominating sets are always created with respect to G, for neighborhoods Γ_r, we have $D_r \subseteq \Gamma_{r+1}$. Therefore, in order to have sufficient separation, we need to consider a 2-separated structure given by the neighborhoods.

The approach constructs a 2-separated collection of neighborhoods given by the $\Gamma_{\bar{r}}(v)$, where $\bar{r}$ denotes the radius upon stopping to expand the neighborhoods. At this point, we remove $\Gamma_{\bar{r}+2}$ from G and keep $D_{\bar{r}+2}$ as part of the solution. The overall separation can then easily be seen by inductive argumentation: each neighborhood removed from the remaining set V of vertices satisfies the separation property with respect to previously removed neighborhoods and the resulting reduced set V for the following iteration. The resulting algorithm from the above approach is given by Algorithm 6.

The following lemma then gives a lower bound for partial dominating sets of 2-separated collections with respect to an optimal dominating set for a graph G.

Lemma 4.10. *Let $D^* := D(V)$ be a Minimum Dominating Set in a graph $G = (V, E)$. For a 2-separated collection $\mathcal{S} = \{S_1, \ldots, S_k\}$ in G, it is*

$$|D^*| \geq \sum_{i=1}^{k} |D(S_i)|.$$

Proof. For each subset $S_i \in \mathcal{S}$, consider the neighborhood $\Gamma(S_i)$. As a direct consequence of the definition of 2-separated subsets, these neighborhoods are pairwise disjoint. Furthermore, any vertex outside $\Gamma(S_i)$ has distance more than one to all vertices in S_i. Therefore, $D^* \cap \Gamma(S_i)$ dominates S_i.

On the other hand, also $D(S_i) \subseteq \Gamma(S_i)$ dominates S_i using a minimum number of vertices. Therefore, we obtain

$$|D^* \cap \Gamma(S_i)| \geq |D(S_i)|.$$

Combining this observation for all subsets of $\mathcal{S}$, we get

$$|D^*| \geq \sum_{i=1}^{k} |D^* \cap \Gamma(S_i)| \geq \sum_{i=1}^{k} |D(S_i)|,$$

as claimed. □

Looking at such a 2-separated collection, together with related subsets and bounded cardinality dominating sets for these, we extend the above Lemma 4.10 to receive an upper bound.

Corollary 4.11. *Let $\mathcal{S} = \{S_1, \ldots, S_k\}$ be a 2-separated collection in $G = (V, E)$, and let $T_1, \ldots, T_k$ be subsets of V with $S_i \subseteq T_i$ for all $i = 1, \ldots, k$. If there exists a bound $\rho \geq 1$ such that*

$$|D(T_i)| \leq \rho \cdot |D(S_i)|$$

holds for all $i = 1, \ldots, k$, then the set $T := \bigcup_{i=1}^{k} D(T_i)$ satisfies

$$|T| \leq \rho \cdot |D^*|,$$

where D^* denotes a Minimum Dominating Set in G.

Proof. $|\bigcup_{i=1}^{k} D(T_i)| \leq \sum_{i=1}^{k} |D(T_i)| \leq \rho \cdot \sum_{i=1}^{k} |D(S_i)| \leq \rho \cdot |D^*|$. □

The partial solutions taken from the respective $(\bar{r}+2)$-neighborhoods $\Gamma_{\bar{r}+2}$ in each iteration thus satisfy the desired approximation guarantee. Furthermore, the following lemma establishes the overall domination property for the partial solutions $D(\Gamma_{\bar{r}+2})$.

Lemma 4.12. *The set D returned by Algorithm 6 dominates $G = (V, E)$.*

Proof. Let N_i denote the neighborhoods $\Gamma_{\bar{r}+2}(v)$ removed from V in each iteration $i = 1, \ldots, k$ of the algorithm. Then, since we stop the algorithm when $V = \varnothing$ is reached, it is $\bigcup_{i=1}^{k} N_i = V$, and each of these neighborhoods N_i is dominated by $D(N_i)$. Therefore, $\bigcup_{i=1}^{k} D(N_i)$ dominates G. □

So far, we see that the solution set D returned by Algorithm 6 is a global $(1+\varepsilon)$-approximate dominating set for the input graph G. At this point, it is noteworthy to remind that this approximation guarantee is valid for any undirected graph G, even if it does not satisfy the bounded growth condition.

It remains to show that the approximation algorithm has polynomial run time. Clearly, the number of times we have to pick a new central vertex, and construct local neighborhoods and dominating sets for these is bounded by $n = |V|$. We may thus limit the further discussion to one iteration only. Since a vertex-induced subgraph of a BGG again is a BGG, we focus w.l.o.g. on the polynomial BGG $G = (V, E)$ in the first iteration, and again show that the radius of the largest neighborhood we need to consider is bounded by a constant.

Lemma 4.13. *Let $G = (V, E)$ be a graph of polynomially p-bounded growth graph. There exists a constant $c = c(\varepsilon)$ such that the radius r of any neighborhood $\Gamma_r(v)$ considered in the algorithm is bounded by c, i.e. $r \leq c$.*

Proof. It is $|D(\Gamma_0(v))| = |D(\Gamma_1(v))| = 1$, as the central vertex v dominates itself and all its neighbors.

Consider the criterion for stopping to expand the neighborhoods in Algorithm 6, and consider the inequality $|D(\Gamma_{r+2})| > (1+\varepsilon) \cdot |D(\Gamma_r)|$ for all $r \leq \bar{r}$.

First, if r is an even number, we have

$$\begin{aligned} p(r+2) \geq |D(\Gamma_{r+2})| &> (1+\varepsilon) \cdot |D(\Gamma_r)| \\ &> \cdots > (1+\varepsilon)^{r/2} \cdot |D(\Gamma_0)| = (\sqrt{1+\varepsilon})^r. \end{aligned}$$

Second, if r is odd, we get

$$\begin{aligned} p(r+2) \geq |D(\Gamma_{r+2})| &> (1+\varepsilon) \cdot |D(\Gamma_r)| \\ &> \cdots > (1+\varepsilon)^{r/2} \cdot |D(\Gamma_1)| = (\sqrt{1+\varepsilon})^r. \end{aligned}$$

Since $\varepsilon > 0$, and thus $\sqrt{1+\varepsilon} > 1$, in both cases, the inequalities have to be violated eventually. Thus, the bound c on the largest radius of the neighborhoods depends only on the approximation guarantee ε. □

If the input graph of Algorithm 6 is of polynomially bounded growth, each iteration has polynomial running time. The complexity of the computation of $D(\Gamma_r(v))$ dictates the overall run time of the algorithm. This can be achieved in $n^{O(p(c))}$.

We close this part with the following summary.

Corollary 4.14. *There exists a PTAS for the* Min-DS *problem on graphs of polynomially bounded growth. In addition to the desired approximation guarantee, the respective approximation schemes require only adjacency information of the input graph.*

4.3 Robustness

We now present a simple way to make the above approximation schemes *robust*. In this case, the robust algorithm accepts any undirected graph as valid input, and either returns a desired approximate solution, or outputs a polynomial certificate showing that the input graph does not satisfy the structural assumption of p-bounded growth.

Recall that an algorithm $\mathcal{A}$ computes a function f robustly on $\mathcal{U} \subseteq \mathcal{I}$ if, for any instance $x \in \mathcal{I}$, it either returns $f(x)$ or a certificate showing that $x \notin \mathcal{U}$. In our situation, $\mathcal{I}$ is the set of all undirected graphs, and $\mathcal{U}$ are all those graphs that have polynomially bounded growth. The function f computes a $(1+\varepsilon)$-approximate subset of the vertices, depending on the problem.

In the previous section, we have seen that the introduced approximation algorithms actually yield a PTAS when the instance reflects a graph of polynomially p-bounded growth. We thus continue the discussion only for the case that the undirected graph $G = (V, E)$ presented to the algorithm does not satisfy the characterization of a polynomially BGG.

Observe that both approximation algorithms return an independent or dominating subset, I or D respectively, for G also in the case that G is not a BGG. We only use specific properties of BGGs to derive the polynomial run time of the algorithms. In both problems, the polynomial run time of the approximation algorithms results from the bound $p(r)$ on the size of any independent set, and an optimal dominating set in the r-th neighborhood, i.e. $|D_r| \leq |I_r| \leq p(r)$. If, during execution of the algorithm, a neighborhood Γ_r contains an independent set of size larger than $p(r)$, we can use this neighborhood as a polynomial certificate showing non-membership in the class of p-Bounded Growth Graphs.

By definition of polynomial BGGs, any independent set in Γ_r has to satisfy the given bound on the cardinality. For dominating sets, this bound obviously does not hold. However, before considering dominating sets for a neighborhood Γ_r, we can use the greedy approach of Section 2.4 to quickly compute a maximal independent set as a starting point. This independent set is then used to guarantee the polynomial run time of the algorithm, making the approach for the Min-DS approximation scheme robust in the above sense, as well.

We can thus apply the approximation schemes to any undirected graph which is *believed* to be of polynomially bounded growth, without risk of failure, i.e. exponential running time, if this assumption is wrong. In wireless ad-hoc networks, this gives the advantage that we can indirectly account for all the uncertainties and the dynamic behavior which govern the resulting applications: there may exist wireless communication graphs in very harsh environments for which the polynomial BGG assumption does not hold. In this case, we at least receive according feedback.

4.4 Conclusions

In this chapter, we presented polynomial-time approximation schemes for the Max-IS and Min-DS problems on wireless communication graphs. For geometrically defined intersection graphs, existing algorithms exploit the geometric information and use a separation strategy to construct several, global candidate solutions. Among these, a solution which satisfies the desired approximation guarantee is found and returned. The problem of computing such a solution without relying on geometric data, but on adjacency information only, used to be an open problem.

With the direct approach of computing locally optimal, partial solutions in bounded neighborhoods, we obtain approximation schemes that are independent of the geometry, and that work for graphs of polynomially bounded growth. The idea behind this new strategy is to create a global solution based on local solutions for expanding neighborhoods until a desired quality is met. These local solutions are then combined in such a way that preserves feasibility and so that the desired approximation ratio of $(1 + \varepsilon)$ is satisfied. We showed that the diameters of the neighborhoods we need to consider for this approach are bounded by some constant that only depends on the parameter ε, and not on the size of the graph.

The approach results in PTASs for the Maximum (Weight) Independent Set and Minimum Dominating Set problems on these graphs. The overall run time of the resulting $(1 + \varepsilon)$-approximation algorithms is $n^{O(1/\varepsilon \log 1/\varepsilon)}$.

In light of the intractability to verify whether a given undirected graph is, e.g., a Unit Disk Graph, we also showed how to make the approximation algorithms robust. A robust algorithm must always terminate and produce meaningful output. In our case, the algorithm accepts any undirected graph, and returns a solution set that satisfies the desired approximation guarantee, or a certificate that allows the polynomial time verification of non-membership in the class of polynomially bounded growth graphs.

For the practical use of these approximation schemes in wireless communication networks, in addition to not relying on localization of the wireless nodes, robustness ensures the desired operation by always giving a useful output in polynomial time.

Chapter 5

Distributed and Local Algorithms

In this chapter, we turn our attention towards local, distributed algorithms for independent and dominating sets on graphs of bounded growth.
In particular, we present a deterministic approach for the $\mathcal{LOCAL}$ model to construct a maximal independent set in $O(\log \Delta \log^* n)$ communication rounds for bounded growth graphs.
The result is exploited to transform the approximation schemes of the previous chapter into local, distributed algorithms for graphs whose growth is polynomially bounded. The communication complexity of this local approach is also dominated by the above MIS construction.

In this chapter, we focus on global structures that are constructed based on local information and using local message exchange. We present and discuss approaches in the $\mathcal{LOCAL}$ message passing model, where each vertex in the graph can perform some computations, and communicate only to its direct neighbors by exchanging messages based on rounds.

The first part of this chapter deals with the local construction of maximal independent sets in graphs of bounded growth. While from a central perspective, this is a rather easy task, e.g. by the greedy approach of Section 2.4, an efficient distributed approach is somewhat harder. Of course, the localized greedy version constructs a MIS in up to $O(n)$ communication rounds. Especially in large-scale wireless networks, this time bound is rather large. We show an approach that works significantly faster. In case of a Unit Disk Graph *with* geometric representation, we can exploit this geometric information to create a MIS in a constant number of rounds. For Bounded Growth Graphs, without representation, we present an approach that only requires $O(\log \Delta \log^* n)$ rounds of communications. This approach then depends only slightly on the size of the network. The description of the local algorithm to construct a MIS follows [35].

In the second part, localized versions of the approximation algorithms of the previous chapter are presented. These approximation schemes seem to be perfectly suited for a distributed algorithm: the global solution is composed of small, local solutions for neighborhoods of bounded diameter. However, these local solutions have to be computed such that they do not interfere too much with each other. In Section 5.2, we show a distributed approach that allows for the local, non-interfering construction of local, neighborhood based solutions. Section 5.2 follows [36].

Throughout this chapter, we characterize a subset of vertices by its density in the graph. This notion of density is defined as follows.

Definition 5.1. *Let $S \subseteq V$ be a subset of the vertices of a graph $G = (V, E)$ such that*

$$\bigcup_{s \in S} \Gamma_r(s) = V.$$

Such a set S is then called ***r-ruling****.*

In other words, in an r-ruling set S in $G = (V, E)$, for each vertex $u \in V \setminus S$, there is a vertex $v \in S$ with $d_G(u, v) \leq r$. If the set S in the above definition is also independent, we refer to it as an *independent* r-ruling set. Note in this context that a maximal independent set is a 1-ruling independent set.

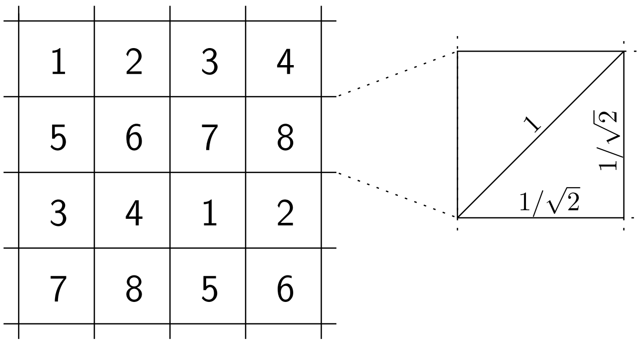


Figure 5.1: Coloring for the MIS-construction on a UDG.

5.1 Local Construction of Maximal Independent Sets

Local MIS construction is an important problem, which we discuss in this part for graphs of bounded growth. We begin by looking at the geometrically defined intersection graphs. As we have already seen in Chapter 4, the case when the geometric information of such a graph is available, separation into smaller subproblems can be used to obtain efficient approaches.

With a geometric representation given, a fast, local MIS-construction is straightforward as follows. For simplicity, consider a Unit Disk Graph, and suppose that each vertex knows its position. We then divide the area needed to draw the graph into squares of side length $1/\sqrt{2}$. These left-open squares are enumerated, and each square is then colored with 8 colors according to the scheme given in Figure 5.1. Note that each vertex can determine its color without involving its neighbors based on position.

Looking at the respective vertices inside a single square, it is easy to see that each square induces a clique, and that vertices in two different squares of the same color are not connected.

We now locally choose, in sequence given by the colors, in each square at most one non-dominated vertex to be added to a partial solution. Vertices adjacent to the partial solution are considered dominated. Initially, all vertices are non-dominated, and once a vertex is dominated, it no longer participates in the process. Just like in the greedy MIS creation (Section 2.4), a vertex that becomes dominated informs all neighbors about this change by a respective message.

Starting with all vertices of color $i = 1$, each non-dominated vertex of color i locally determines whether it has the highest identifier among its non-dominated neighbors of the same color. If this is the case, it joins the partial solution, and informs all its neighbors about this decision. Since vertices of equal color form local cliques, this entire process can be done in a single round based on information already present at each vertex. Also, after this round and an additional round for now newly dominated vertices to inform their neighbors, all vertices of color i are either in the partial solution, or dominated by it.

Clearly, this approach results in a global, maximal independent set. In case of other geometrically defined intersection graphs, we may have to adjust the coloring accordingly, and it is easy to see that the number of colors determines the overall number of rounds: each color results in two rounds of communication, one round to announce the locally added vertex to the partial solution, and one round for the other vertices to inform their neighborhood about becoming dominated.

Coming back to the non-geometric case, we now describe and analyze a distributed maximal independent set construction for graphs of f-bounded growth. This algorithm works in three subsequent phases by

- constructing an $O(\log \Delta)$-ruling independent set (Section 5.1.1),
- turning this set into a 3-ruling independent set (Section 5.1.2), and
- constructing a MIS from the 3-ruling independent set of the previous stage (Section 5.1.3).

In the following, we also establish that the overall algorithm terminates after $O(\log \Delta \log^* n)$ rounds for BGGs, and requires only short messages of size $O(\log n)$.

5.1.1 Constructing a Sparse Independent Set

The first phase of our MIS construction is a distributed algorithm which locally computes an $O(\log \Delta)$-ruling independent set S. A detailed description of the first phase is given by Algorithm 7. Before analyzing the algorithm, we give an informal description of the code.

At the beginning, S is empty and all vertices actively participate in the construction (denoted by the black color $b(v)$ for $v \in V$). Vertices are colored black as long as they have not decided whether to join the independent set S.

Algorithm 7 Sparse Independent Set (code for vertex v).

```
Input: $G = (V, E)$.
Output: Sparse independent set $S$.
 1: $S := \varnothing, b(v) :=$ black;
 2: while $b(v) =$ black do
 3:    if $\exists\, u \in \Gamma(v) \setminus \{v\} \mid b(u) =$ black then
 4:       $d(v) := \min\{u \in \Gamma(v) \setminus \{v\} \mid b(u) = \text{black}\}$;
 5:       Inform neighbor $d(v)$;
 6:       $A_v := \{u \in \Gamma(v) \mid d(u) = v\}$;
 7:       if $A_v \neq \varnothing$ then
 8:          $p(v) := \min A_v$;
 9:          Inform neighbor $p(v)$;
10:       end if
11:       $B_v := \{u \in \Gamma(v) \mid p(u) = v\}$;
12:       if $(A_v = \varnothing) \wedge (B_v = \varnothing)$ then
13:          $b(v) :=$ white;
14:       else
15:          Construct maximal independent set $I$ on $\overline{G} = (\overline{V}, \overline{E})$
             with $\overline{V} := \{u \in V \mid b(u) = \text{black}\}$
             and $\overline{E} := \{(u, p(u)) \mid u \in \overline{V} \wedge A_u \neq \varnothing\}$;
16:          if $v \notin I$ then $b(v) :=$ white end if
17:       end if
18:    else
19:       $S := S \cup \{v\}, b(v) :=$ white;
20:    end if
21: end while
```

As soon as a vertex turns white, it has either joined S, or it has decided not to join S.

From a general perspective, the aim is to repeatedly eliminate black vertices from the network until single, locally independent vertices are left. We do so with the help of an edge-induced subgraph $\overline{G}$ of bounded degree. The edges of $\overline{G}$, connecting some black vertices in G are chosen as follows (see also Figure 5.2). First, each black vertex v chooses another black neighbor which we denote by $d(v)$. Then, each u which has been chosen by at least one neighbor v, selects a neighbor $p(u)$ for which $d(p(u)) = u$ holds. The edge set of $\overline{G}$ then consists of all edges of the form $(u, p(u))$. In $\overline{G}$, each vertex has degree ≤ 2 (see Lemma 5.2).

All black vertices that have no edge in $\overline{G}$ incident to them become white, and thus no longer participate in the process.

On $\overline{G}$, we can efficiently construct a maximal independent set I (explanation follows below). The set I is now used to change the color of vertices: all vertices from $\overline{G}$ that are not in the independent set I change their color to white. This way, the number of black vertices is reduced by at least a constant factor every time we execute this process (see Lemma 5.4 and Lemma 5.5). As soon as a black vertex has no more black neighbors, it joins the independent set S, and also changes its color to white. The basic steps are also explained by the example in Figure 5.2.

We now consider a single execution of the while loop (lines 3–21) which encodes the above approach. Further on, we call such a single while loop execution a single iteration of Algorithm 7.

Lemma 5.2. *In the graph $\overline{G} = (\overline{V}, \overline{E})$, every vertex has degree at most 2.*

Proof. Consider $v \in \overline{V}$, then there are at most two vertices adjacent to v by an edge in $\overline{E}$, namely $d(v)$ if $p(d(v)) = v$, and $p(v)$. □

The bounded degree of $\overline{G}$ allows for the local construction of the MIS I on $\overline{G}$ in $O(\log^* n)$ rounds using methods described in, e.g., [38, 51]. Each iteration of the while loop can therefore be achieved in $O(\log^* n)$ communication rounds, and this construction uses only messages of size $O(\log n)$.

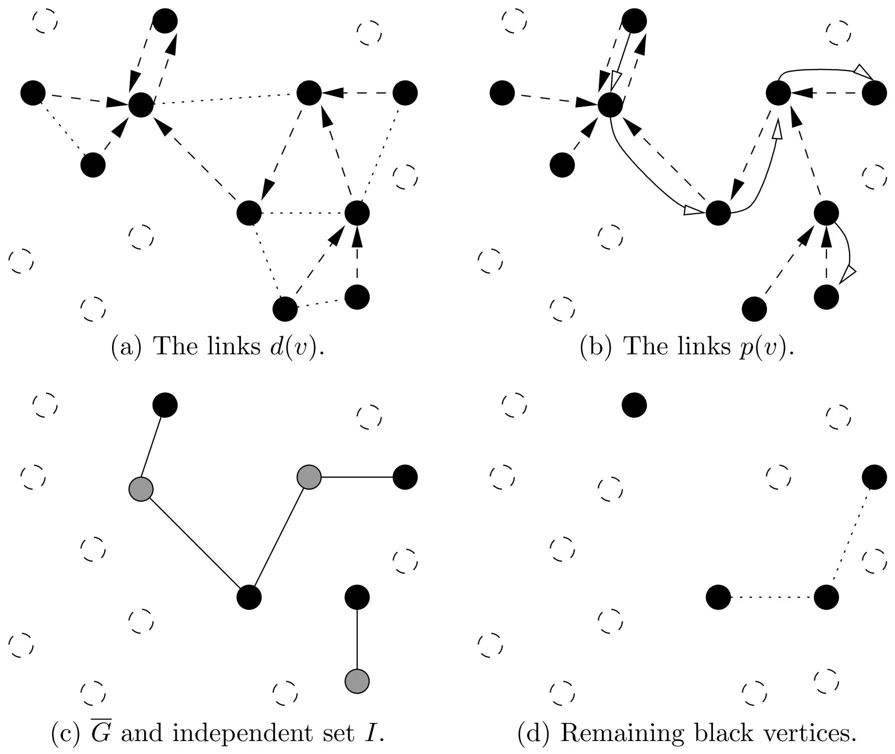
The remaining black vertices now satisfy the following property.

Lemma 5.3. *Let $V_B := \{v \in V \mid b(v) = black\}$. After k iterations of Algorithm 7, $S \cup V_B$ is a $2k$-ruling set in G.*

Proof. We prove the lemma by induction over the number k of while loop iterations. Initially all vertices are black, thus the lemma is satisfied for $k = 0$.

For the induction step, we show that if a vertex v becomes white in an iteration of the while loop, either v joins S or there is an black vertex at distance at most 2 from v which remains black at least until the next while loop iteration. A black vertex v can turn white at three points corresponding to the following three situation.

- The vertex v joins the set S (line 20), and the condition of the lemma is met.
- If the vertex v is not in the independent set I, but incident with another black vertex in $\overline{G}$ (line 16), v has a neighbor u that belongs to I due to the domination property of I. This vertex u remains black, with $d_G(u, v) = 1$.


(a) The links $d(v)$. (b) The links $p(v)$.

(c) $\overline{G}$ and independent set I. (d) Remaining black vertices.

Figure 5.2: One iteration of Algorithm 7. The white, dashed vertices do not participate in the execution.

- The last remaining case is that the vertex v has some black neighbor in G, but in $\overline{G}$, the vertex v is isolated (line 13). Since v has a black neighbor, it has chosen a vertex $u = d(v)$ (line 4). This vertex u, with $A_u \neq \varnothing$, has chosen a vertex $p(u)$, and therefore is a vertex of $\overline{G}$. Because all vertices of the MIS I in $\overline{G}$ remain black, either u or a neighbor w of u is still black after completing the iteration.

We see that there is a black vertex at distance at most two from v, and by induction the stated property is valid. □

Clearly, Algorithm 7 terminates once there are no black vertices left in G. The following two lemmas give bounds on the number of rounds needed to complete, and to explain the resulting structure in G for general and f-growth bounded graphs, respectively.

Lemma 5.4. *On an undirected graph $G = (V, E)$, Algorithm 7 terminates after $O(\log n)$ iterations with an $O(\log n)$-ruling independent set S.*

Proof. Let n_B be the number of black vertices at the beginning of an iteration. We prove that in one iteration, at least $n_B/3$ vertices turn white which proves the bound on the number of iterations. The second part of the claim then follows by Lemma 5.3.

Let $\overline{n} \leq n_B$ be the number of vertices of $\overline{G}$ in the considered iteration. All vertices which are not part of $\overline{G}$ become white. It therefore suffices to prove that at least one third of the vertices of $\overline{G}$ change color.

Since $\overline{G}$ is an edge-induced subgraph of G, all vertices of $\overline{G}$ have at least degree 1, that is, $\overline{G}$ does not contain isolated vertices. Because the maximum degree of a vertex in $\overline{G}$ is 2 (Lemma 5.2), the MIS I c consists of at most $2\overline{n}/3$ vertices. Hence, at least $\overline{n}/3$ vertices become white because they are not in the independent set I. □

We now turn our attention to f-growth bounded graphs, and show that in this case, the run time can even be reduced to $O(\log \Delta)$ iterations, where Δ denotes the maximal vertex degree in G.

Lemma 5.5. *Let G be f-growth-bounded. Algorithm 7 terminates after $O(\log \Delta)$ iterations with an $O(\log \Delta)$-ruling independent set S.*

Proof. Let M be a maximal independent set in G, which we use to create the following cluster graph. We associate a cluster C_u with each vertex $u \in M$, and each vertex $v \notin M$ is assigned to the cluster C_u of an adjacent vertex $u \in M$. Each cluster then contains at most $\Delta + 1$ members. The cluster graph $\mathcal{G}_C$ is then given by the vertex set $\{C_u \mid u \in M\}$, and two clusters are connected if and only if there is an edge in G connecting two vertices of the two clusters.

Clearly, if two clusters C_u and C_v are connected in $\mathcal{G}_C$, then there exists a path of length less than or equal to 3 in G between u and v. Since G is f-growth bounded, there are no more than $f(3) = O(1)$ independent vertices at distance at most 3 from any vertex $v \in V$. Therefore, the maximum degree of $\mathcal{G}_C$ is bounded by $\Delta_{\mathcal{G}_C} \leq f(3)$.

We now show that the maximum number of black vertices per cluster is reduced by a factor of 2 in a constant number of iterations of Algorithm 7. To be more precise, let α be the maximum number of black vertices of a cluster at iteration t. We show that there is a constant k so that at iteration $t + k$, each

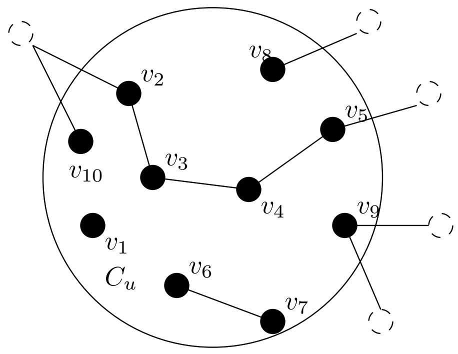


Figure 5.3: Cluster C_u with the edges in $\overline{G}$.

cluster only contains at most $\alpha/2$ black vertices. Since $\alpha \leq \Delta + 1$ at $t = 0$, this immediately yields the claim of the lemma.

Now, we consider to a single iteration, and a single cluster C_u with c black vertices. Since we are only interested in the black vertices, we omit all white vertices for now. We partition the c vertices of C_u into 3 groups $\mathcal{C}_{\text{in}}$,$\mathcal{C}_{\text{out}}$, and $\mathcal{C}_{\text{iso}}$ according to their neighbors in $\overline{G}$. All vertices in C_u that are not in $\overline{G}$, i.e. that are isolated, are in $\mathcal{C}_{\text{iso}}$, all vertices which are in $\overline{G}$ adjacent to another vertex from C_u are in the set $\mathcal{C}_{\text{in}}$, and the remaining vertices which are only connected to other vertices outside C_u in $\overline{G}$ are grouped together in $\mathcal{C}_{\text{out}}$. The three groups are also explained in the example in Figure 5.3. Here, we have $\mathcal{C}_{\text{iso}} = \{v_1\}$, $\mathcal{C}_{\text{in}} = \{v_2, \ldots, v_7\}$, and $\mathcal{C}_{\text{out}} = \{v_8, v_9, v_{10}\}$; adjacent black vertices not in C_u are dashed.

Looking at a MIS I on $\overline{G}$, we further divide the vertices in $\mathcal{C}_{\text{out}}$ into the vertices which remain black, i.e. that belong to the independent set I, denoted by $\mathcal{C}^+_{\text{out}}$, and those who turn white, denoted by $\mathcal{C}^-_{\text{out}}$. With the above sets, let $c_{\text{in}} := |\mathcal{C}_{\text{iso}}| + |\mathcal{C}_{\text{in}}| + |\mathcal{C}^-_{\text{out}}|$ and $c_{\text{out}} := |\mathcal{C}^+_{\text{out}}|$. Then, we see that $c = c_{\text{in}} + c_{\text{out}}$ clearly holds.

Due to $\Delta_{\overline{G}} \leq 2$, during the construction of the MIS I on $\overline{G}$, at least a third of the vertices in $\mathcal{C}_{\text{in}}$ turn white. All vertices in $\mathcal{C}_{\text{iso}}$ and $\mathcal{C}^-_{\text{out}}$ become white. Therefore, in one iteration, at least $c_{\text{in}}/3$ vertices in C_u turn their color to white.

The vertices in $\mathcal{C}^+_{\text{out}}$ belong to the MIS I, and therefore all vertices outside the cluster C_u that are adjacent to a vertex in $\mathcal{C}^+_{\text{out}}$ become white. Since in $\overline{G}$, each vertex of $\mathcal{C}^+_{\text{out}}$ is adjacent to at least one vertex outside C_u, and each vertex outside C_u is adjacent to at most 2 vertices in $\mathcal{C}_{\text{out}}^+$, at least $c_{\text{out}}/2$ vertices of clusters outside C_u which are connected to C_u become white.

Coming back to a sequence of consecutive iterations, assume that after k iterations, there are still $\alpha/2$ black vertices in C_u. Let $c^{(j)}, c_{\text{in}}^{(j)}$, and $c_{\text{out}}^{(j)}$ denote the respective values of c, c_{in}, and c_{out} for the j-th iteration. Since there are at most α black vertices at the beginning, we have

$$\frac{1}{3} \cdot \sum_{j=1}^{k} c_{\text{in}}^{(j)} \leq \frac{\alpha}{2}$$

because otherwise at least $\alpha/2$ vertices would have changed their color to white. The number of vertices in the adjacent clusters of C_u in $\mathcal{G}_C$ that have become white then is at least

$$\begin{aligned}
\frac{1}{2} \cdot \sum_{j=1}^{k} c_{\text{out}}^{(j)} &= \frac{1}{2} \cdot \sum_{j=1}^{k} \left(c^{(j)} - c_{\text{in}}^{(j)}\right) \\
&\geq \frac{1}{2} \cdot \sum_{j=1}^{k} \frac{\alpha}{2} - \frac{1}{2} \cdot \sum_{j=1}^{k} c_{\text{in}}^{(j)} \\
&\geq k \cdot \frac{\alpha}{4} - \frac{1}{2} \cdot \frac{3\alpha}{2} \\
&= \frac{k-3}{4} \cdot \alpha.
\end{aligned}$$

At the beginning of iteration t, there are no more than $\Delta_{\mathcal{G}_C} \cdot \alpha$ black vertices all the adjacent clusters to C_u. Combining this with the above inequality, we get that after $O(\Delta_{\mathcal{G}_C}) = O(f(3)) = O(1)$ iterations, there are no black vertices left in the neighboring clusters. From that point on, at least a third of the black vertices in C_u turn white in each further iteration.

Overall, this leads to at most $O(\log \alpha) = O(\log \Delta)$ iterations of the while loop. □

Summarizing the above properties, we obtain the following theorem that concludes the first phase of the overall local MIS construction. Note that the information sent during the execution of the algorithm consists of vertex identifiers and color changes only. The construction of a maximal independent set on $\overline{G}$ in each iteration requires $O(\log^* n)$ time and messages of size $O(\log n)$, and this dominates the overall time for a single iteration of the while loop of Algorithm 7.

Algorithm 8 Dense Independent Set.

```
Input: $t$-ruling independent set $S$ on $G = (V, E)$ $(t > 3)$.
Output: 3-ruling independent set $S$.
1: while $S$ is not 3-ruling do
2:     for each $u \in S$ do
3:         Compute $\hat{S}_u \subseteq \Gamma_4(u)$ such that $S \cup \hat{S}_u$ is an independent set
           and such that $S \cup \hat{S}_u$ dominates $\Gamma_3(u)$;
4:         Construct MIS $I$ on $G[\bigcup_{u \in S} \hat{S}_u]$;
5:         $S := S \cup I$;
6:     end for
7: end while
```

Theorem 5.6. *Algorithm 7 is a local, distributed algorithm which computes an independent set S in G.*

- *On an f-bounded growth graph, S is an $O(\log \Delta)$-ruling set, and the algorithm terminates after $O(\log \Delta \log^* n)$ communication rounds.*
- *For general graphs, the algorithm terminates in $O(\log n \log^* n)$ rounds with an $O(\log n)$-ruling set S.*

Furthermore, all messages are of size $O(\log n)$.

5.1.2 Making a 3-ruling Independent Set

In this second phase of the local MIS computation, we show how to turn the relatively sparse, $O(\log \Delta)$-ruling independent set S into a 3-ruling independent set. More precisely, for a BGG $G = (V, E)$, we show for a t-ruling independent set S $(t > 3)$ how to transform S into a 3-ruling independent set in $O(t \cdot \log^* n)$ communication rounds. This is done using small messages of size $O(\log n)$. Algorithm 8 describes the method to achieve this.

The basic idea behind the algorithm is to enlarge the independent set around already existing vertices to make it denser in each step. Each vertex in the existing independent set S adds a set of independent vertices $\hat{S}_u$ to the set so that the 3-rd neighborhood is dominated by $S \cup \hat{S}_u$. However, since the addition of vertices is not globally controlled, it is not guaranteed that the overall resulting set $S \cup \bigcup_{u \in S} \hat{S}_u$ is again independent. Therefore, before going on, the independence property of S needs to be restored by computing a maximal independent set I on the newly added vertices, and adding I to S.

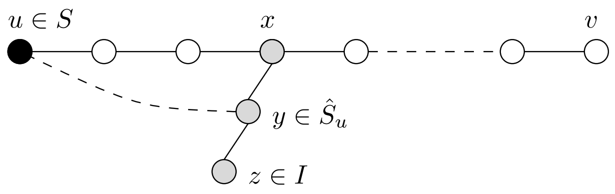


Figure 5.4: Proof of Lemma 5.7. It is $d_G(u, v) = t$. Note that $x = y$ or $y = z$ may be, but $d_G(v, z) \leq t - 1$ then holds, as well.

The following lemma shows that in each iteration, the maximum distance of any vertex to a vertex in S is reduced by at least 1.

Lemma 5.7. *Let S be a t-ruling independent set ($t > 3$) on a BGG. After one iteration of Algorithm 8 (lines 2–6), S is a $(t-1)$-ruling independent set.*

Proof. The sets $\hat{S}_u$ are constructed such that vertices in S and $\hat{S}_u$ are non-adjacent. Furthermore, the set $I \subseteq \bigcup_{u \in S} \hat{S}_u$ is independent. This shows that the set S remains an independent set throughout the execution of the algorithm.

In order to prove that the maximum distance from a vertex to another vertex in S decreases, consider a vertex $v \in V$ for which the distance to the closest vertex $u \in S$ is $t > 3$. We show that after one iteration of the while loop, the distance from v to another vertex in the expanded set $S \cup I$ is at most $t - 1$.

The set $\hat{S}_u$ is explicitely constructed to ensure that every vertex $w \in \Gamma_3(u)$ has a neighbor in $S \cup \hat{S}_u$. On a shortest path from v to u, which has length t, let x be the vertex at distance exactly 3 from u, i.e. that has a distance of $t-3$ from v (see also Figure 5.4). By construction of $\hat{S}_u$, there must be a vertex $y \in \hat{S}_u$ with $(x, y) \in E$ (or $x = y$). This vertex y has distance at most $t-2$ from v. Note that y cannot be in S since this would contradict v being of distance t from a vertex in S. For the same reason, no neighboring vertex from y can be contained in S. During the construction of the maximal independent set I, either y or a neighboring vertex z (with $(y, z) \in E$) are added to I. This vertex z has at most a distance of $t - 1$ from v. Therefore, the distance between v and $S \cup I$ is at most $t - 1$. The claim follows. □

Now, we go on to show that Algorithm 8 can indeed be implemented efficiently by a distributed approach for graphs of bounded growth. The following lemma also shows that this can be achieved using only small messages.

Lemma 5.8. *Let G be a growth bounded graph. On G, Algorithm 8 can be executed by a distributed approach within $O(t \log^* n)$ rounds of communication, and using messages of size $O(\log n)$.*

Proof. By Lemma 5.7, the algorithm terminates after less than t iterations of the while loop. It thus remains to show that each iteration can be achieved in $O(\log^* n)$ rounds using small messages.

First, we consider the local construction of $\hat{S}_u$ for some vertex $u \in S$. This is done by a greedy approach that successively adds candidate vertices to $\hat{S}_u$ as follows. A vertex $v \in \Gamma_4(u)$ can potentially join $\hat{S}_u$ if it has no neighbor in $S \cup \hat{S}_u$ and if it has an uncovered neighbor $w \in \Gamma_3(u)$, that is, w is not adjacent to any vertex from $S \cup \hat{S}_u$. We call such a vertex candidate. For the addition of candidates to $\hat{S}_u$, we then follow a simple greedy strategy based on the identifier to break ties: we add a candidate v if it has a lower identifier than all its adjacent candidates.

Now, for a vertex v to figure out if it is a candidate and whether it has locally the lowest identifier can be done in 3 rounds:

1. All vertices in $S \cup \hat{S}_u$ inform their neighbors about their membership.

2. All dominated vertices in $\Gamma_3(u)$ repeat this notification, however this time stating that they are adjacent to a vertex in $S \cup \hat{S}_u$.

3. A vertex can now decide on becoming a candidate. All candidates inform their neighborhood about their identifier.

We call these three rounds a step. In each step, at least the candidate with overall lowest identifier joins the set $\bar{S}_u$. Due to G being growth bounded, there are at most $f(4) = O(1)$ independent vertices in $\Gamma_4(u)$. Hence, the cardinality of $\hat{S}_u$ is bounded by a constant, and therefore also the number of steps to complete the greedy creation of it.

Note that if we allow the messages exchanged to be of arbitrary size, 8 communication rounds would suffice to construct $\hat{S}_u$: the central vertex $u \in S$ collects all information about its 4-th neighborhood, centrally computes $\hat{S}_u$, and informs all respective vertices.

Now, it remains to show that the construction of the MIS I (line 4) can be done using $O(\log^* n)$ rounds on the graph $G' := G[\bigcup_{u \in S} \hat{S}_u]$. Consider a vertex v from G', i.e., $v \in \hat{S}_u$ for some $u \in S$. Further, let w be a neighbor of v in G'. Then, $w \in \hat{S}_{\bar{u}}$ for some $\bar{u} \in S \setminus \{u\}$. Since $\hat{S}_{\bar{u}} \subseteq \Gamma_4(\bar{u})$, we have $d_G(v, \bar{u}) \leq 5$. Because G is f-growth bounded, there are at most $f(5)$ possible vertices $\bar{u} \in S$ which can be the cause for a neighbor w of v. Furthermore, the number of possible neighbors of v in each $\hat{S}_{\bar{u}}$ is bounded by $f(1)$. Therefore, the maximum degree of G' can be bounded by $f(5) \cdot f(1) = O(1)$. For graphs of bounded degree, a maximal independent set I can be computed locally in $O(\log^* n)$ communication rounds using messages of size $O(\log n)$ [38, 51]. □

To conclude this phase of the local MIS construction, we summarize Algorithm 8 as follows.

Theorem 5.9. *Let $G = (V, E)$ be of bounded growth. Then, a t-ruling independent set can be transformed into a 3-ruling independent set in $O(t \log^* n)$ communication rounds using messages of size $O(\log n)$.*

5.1.3 Turning the 3-ruling Independent Set Into a MIS

In order to turn a 3-ruling independent set S into a maximal independent set, we locally construct a cluster graph $\mathcal{G}$ induced by S as follows. For each $u \in S$, the corresponding cluster C_u is defined as the set of vertices $v \in V$ for which u is the nearest vertex in S; ties are broken arbitrarily. The cluster graph $\mathcal{G}$ is then given by the vertex set $\{C_u \mid u \in S\}$ of the clusters, and two clusters are connected if and only if there is an edge in G connecting vertices of the two clusters.

Since S is a 3-ruling set, the distance in G between the centers of two adjacent clusters in $\mathcal{G}$ is at most 7. The maximal degree of $\mathcal{G}$ is therefore bounded by $f(7)$ if G is f-growth bounded. Using, e.g., local algorithms presented in [38, 51], we can color the cluster graph $\mathcal{G}$ using at most $f(7) + 1$ colors. This can be achieved with small messages of size $O(\log n)$.

Each vertex $v \in V$ is now colored with the color $\chi_v = \chi_u$ of the cluster C_u it belongs to. Using this color, v is now given a weight composed of its color and its identifier, denoted by $w_v := < \chi_v, id_v >$. This weight induces a lexicographic order on V where a vertex $v \in V$ is placed before another $u \in V$ if it has a lower color, or, in case of equal color, has a lower identifier.

Starting from S as an initial solution, we run the Local-Greedy Algorithm of Section 2.4 with these weights to obtain a Maximal Independent Set in G. Generally speaking, vertices of each color sequentially add their contribution to the MIS. The run time for this weighted graph is given by the following theorem.

Lemma 5.10. *Let $G = (V, E)$ be an f-growth bounded graph, S be a 3-ruling independent set in G, and assume that every vertex $v \in V$ is given a weight w_v*

composed of its color in the cluster graph $\mathcal{G}$ and its identifier.
Then, the local-greedy approach given by Algorithm 2 (Section 2.4) locally adds independent vertices up to a maximal independent set in $O(f(7) \cdot f(3))$ communication rounds.

Proof. Each cluster $C_u, u \in S$, is a subset of $\Gamma_3(u)$, thus there can be at most $f(3)$ independent vertices per cluster. Considering all clusters with the overall lowest color, these are non-adjacent. Due to the order induced by the weights, the uncovered vertex with lowest identifier in each C_u of this color can join the partial solution. Thus, after $O(f(3))$ communication rounds, all vertices of this color are either in the MIS or adjacent to it.

Subsequently, as soon as there are no uncovered vertices left with color i, the above argument holds for color $i+1$. Therefore, after $O(f(3) \cdot f(7)) = O(1)$ rounds, all vertices are either in the MIS or covered by a vertex from it. □

Combining all the three phases of the preceding parts, we obtain the main theorem of this section on the distributed MIS construction on bounded growth graphs.

Theorem 5.11. *Let G be a graph of f-bounded growth. There is a deterministic, distributed algorithm which constructs a maximal independent set on G. In the $\mathcal{LOCAL}$ message passing model, this algorithm requires $O(\log \Delta \log^* n)$ communication rounds to terminate, and all messages are of size $O(\log n)$.*

5.2 Local Approximation Schemes

In this part, we present algorithms that yield efficient, local approaches for the construction of $(1+\varepsilon)$-approximate solutions to the Max-IS and Min-DS problems on a polynomially bounded growth graph $G = (V, E)$ with growth polynomial p.

These algorithms are based on the approximation schemes presented in the previous chapter. Of course, the approach taken in Chapter 4 already exhibits several local properties by only considering the bounded neighborhood of a vertex during the construction of partial solutions. This property can be exploited and extended towards a distributed, in-network approach in a rather straightforward way: always let the vertex with overall lowest identifier gain knowledge about its neighborhood, and construct a solution as given by the PTAS; then go on with the remaining vertices. However, while this approach certainly can be achieved by local message exchange, it is not very efficient. We therefore consider a more parallelized, local algorithm.

The main idea behind the local approach is to quickly identify vertices which are separated far enough so that their neighborhoods do not overlap during the construction of partial solutions. These vertices then construct local solutions in parallel. In order to eventually cover the whole graph with partial solutions, we first identify potential central vertices by a maximal independent set in the graph, then label these vertices so that two vertices of the same label can perform in parallel. Such a labeling also gives an order in which additional local solutions are created in a non-overlapping way if needed to cover the entire graph.

Before presenting the algorithm, we explain the above structure which we call a *colored interference graph*, and which forms the core of the algorithm that gives the local approximation schemes. The vertices of this interference graph are the potential central vertices, and their colors exactly represent their respective labels.

From Lemmas 4.4 and 4.13 for the central approximation schemes, we know that each partial solution $I_{\bar{r}}(v)$ and $D_{\bar{r}}(v)$ respectively is inside a bounded neighborhood around the central vertex $v \in V$ if the graph is of polynomially bounded growth. The radius c of the largest neighborhood can be bounded by a constant that only depends on the desired approximation factor ε. In the following, we assume this constant c to be known.

Clearly, for the independent set PTAS, two partial solutions $I_{\bar{r}_1}(v_1)$ and $I_{\bar{r}_2}(v_2)$ do not interfere if $d_G(v_1, v_2) \geq \bar{r}_1 + \bar{r}_2 + 1$ holds in the original graph G. For the Min-DS case, two partial solutions $D_{\bar{r}_1+2}(v_1)$ and $D_{\bar{r}_2+2}(v_2)$ do not interfere if $d_G(v_1, v_2) \geq \bar{r}_1 + \bar{r}_2 + 7$ holds in G. For simplicity, we go on considering two neighborhoods to be *non-interfering* if their central vertices u and v are at least of distance $d_G(u, v) \geq 2c + 7$. Such two vertices can compute partial solutions in parallel, and since these solutions are non-interfering, they can be added to a global solution without violating feasibility.

On the other hand, identifying such a mutually non-interfering subset of vertices, and then computing a partial solution does not give a complete global solution. We need to add partial solutions until the entire graph is considered. This process can be incorporated in the parallel computing scheme by the already existing information on interference.

Both non-interference, and the sequence in which vertices add partial solutions is encoded by a coloring of the vertices. This is done such that two vertices of the same color do not interfere, and the coloring itself gives an order for the addition of partial solutions. This coloring is locally constructed with the help of a maximal independent set $\mathcal{I}$ and for a resulting interference graph $\mathcal{G} = \{\mathcal{I}, \mathcal{E}_{2c+7}\}$ as follows.

A vertex of $\mathcal{G}$ is given by a vertex from the independent set $\mathcal{I}$, and two vertices are connected if their distance in the original graph G is less than or equal to $2c + 7$. In other words, for $u, v \in \mathcal{I}$,

$$(u, v) \in \mathcal{E}_{2c+7} \iff d_G(u, v) \leq 2c + 7.$$

Note that the maximum degree $\Delta_{\mathcal{G}}$ in $\mathcal{G}$ is bounded by $p(2c+7) = O(1)$ due to the vertex set $\mathcal{I}$ being independent in G. Additionally, two adjacent vertices in $\mathcal{G}$ can exchange messages with one another in a constant number of communication rounds. We can thus obtain a feasible coloring with no more than $\Delta_{\mathcal{G}} + 1$ colors in $O(\log^* n)$ rounds [38, 51].

The above structure and ideas are schematically presented in Figure 5.5. There are four vertices v_i, v_j, v_k, and v_l highlighted as vertices of the maximal independent set $\mathcal{I}$. The radii of their $(c + 3)$-neighborhoods in G, and the neighborhoods for which respective partial solutions have been computed are sketched. It can be seen that the vertex pairs v_i, v_j and v_k, v_l of are non-interfering. Therefore, $v_i(v_k)$ and $v_j(v_l)$ may have the same color. Furthermore, if the color of v_i and v_j is lower than that of v_k and v_l, the neighborhoods Γ_i and Γ_j are removed from G before considering Γ_k and Γ_l. In this respect, non-interference remains.

On a more technical side, note that thus far, we have only considered vertices from the MIS $\mathcal{I}$ as possible center vertices of partial solutions. However, for each vertex in G, we need to ensure that it is also considered when not participating in a partial solution stemming from an independent vertex. Due to the maximality of $\mathcal{I}$, each such vertex is adjacent to an independent vertex. We therefore consider not only $v \in \mathcal{I}$ during the approach, but the entire neighborhood $\Gamma(v)$ in G. Coordinated by $v \in \mathcal{I}$, more local solutions are created in sequence with central vertices in this neighborhood if needed.

There are at most $p(1) = O(1)$ independent vertices in a neighborhood of a vertex $v \in \mathcal{I}$. Looking at the approach for the Max-IS problem, we always remove the $(\bar{r} + 1)$-neighborhood, $\bar{r} \geq 0$, from the set of vertices, thus after at most $f(1)$ subsequent partial solutions, $\Gamma(v)$ is considered completely. In case of a dominating set, since the $(\bar{r} + 2)$-neighborhood is removed for a partial solution, starting the local neighborhood expansion process at any $u \in \Gamma(v)$ results in all vertices from $\Gamma(v)$ being considered.

Algorithm 9 more formally gives the above approach for both the Max-IS and Min-DS problems. From the preceding discussion, and the local construction of a maximal independent set in a graph of bounded growth of Section 5.1, we see that the colored interference graph can be computed in $O(\log^* n \log \Delta)$ communication rounds for polynomial BGGs.

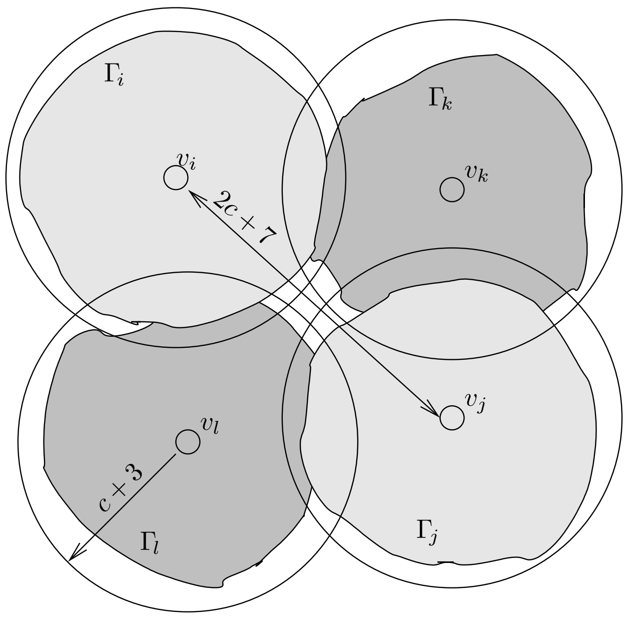


Figure 5.5: Schematic overview of the local approximation schemes.

Note that the central vertices of each partial solution are able to communicate with all other vertices in the $O(c)$-neighborhood in $O(c)$ communication rounds. The construction of the partial solutions in the second part of the algorithm can be achieved in $O(c \cdot p(1) \cdot \Delta_{\mathcal{G}}) = O(1)$ rounds due to the parallel computations of vertices with equal color. The overall communication complexity of Algorithm 9 is thus dominated by the local MIS construction.

Explicitely taking the parameter ε into account, we see that $p(c) = 1/\varepsilon^{O(1)}$, and therefore $\Delta_{\mathcal{G}} = 1/\varepsilon^{O(1)}$ holds. The overall message time complexity is then given by $O(\log^* n \log \Delta + \log^* n/\varepsilon^{O(1)})$.

We summarize this section by the following theorem.

Theorem 5.12. *Let $G = (V, E)$ be a graph of polynomially bounded growth. There exists a distributed $(1+\varepsilon)$-approximation algorithm in then $\mathcal{LOCAL}$ message passing model for the* Max-IS *and* Min-DS *problems on G. The number of communication rounds required is $O(\log^* n \log \Delta + \log^* n/\varepsilon^{O(1)})$.*

As a final remark, we point out that all computations to be performed locally

Algorithm 9 Local Approximation Scheme.

```
Input: $G = (V, E)$ polynomial BGG, $\varepsilon > 0$.
Output: $(1 + \varepsilon)$-approx. solution set $S$.
1: $S := \varnothing$;
2: Compute maximal independent set $\mathcal{I}$ on $G$;
3: Construct interference graph $\mathcal{G} = \{\mathcal{I}, \mathcal{E}_{2c+7}\}$;
4: Color $\mathcal{G}$ using $\Delta_{\mathcal{G}} + 1$ colors;
5: for $k = 1$ to $\Delta_{\mathcal{G}} + 1$ do
6:     for every $v \in \mathcal{I}$ with color $k$ do
7:         while $\Gamma(v) \cap V \neq \varnothing$ do
8:             For some $u \in \Gamma(v) \cap V$, compute solution $S_{\bar{r}}$ for the neighborhood
               $\Gamma_{\bar{r}}(u) \cap V$ according to the PTAS in Section 4.2;
9:             Inform vertices in $\Gamma_{c+3}(v)$ about $\bar{r}$ and $S_{\bar{r}}$, respective vertices are
               removed from $V$;
10:            $S := S \cup S_{\bar{r}}(u)$;
11:        end while
12:    end for
13: end for
```

at each vertex are also polynomially bounded. (see c.f. Chapter 4).

5.3 Conclusions

In this chapter, we have studied local, distributed approaches for the construction of independent, and dominating sets in graphs of bounded growth. The approaches presented do not rely on positional information.

The local construction of a maximal independent set for growth bounded graphs is discussed. By first computing a sparse, independent set, and then adding vertices to make this set denser and eventually maximal, we obtain local approach that completes in $O(\log^* n \log \Delta)$ communication rounds.

Using a MIS as a starting point, we also presented two approaches that create a $(1 + \varepsilon)$-approximate independent, and dominating set in graphs of polynomially bounded growth, respectively ($\varepsilon > 0$). The idea behind the approaches follows the approach presented in Chapter 4, i.e. creating partial solutions in neighborhoods of bounded radius. The respective algorithms require $O(\log^* n \log \Delta + \log^* n/\varepsilon^{O(1)})$ communication rounds.

Chapter 6

A Communication Strategy for Wireless Sensor Networks

In this chapter, we present EMACs, an integrated approach for a communication strategy designed for wireless sensor networks. The resource limitations of such networks, especially in terms of energy, require a collaborative approach spanning over various layers of the classical communication stack to be efficient. We combine a scheduled medium access control mechanism that allows collision free transmissions of neighboring wireless devices together with an implicit backbone creation that identifies redundant sensor nodes. These redundant nodes can then save additional energy by following a sleeping pattern.
The protocol is completely local, and based on time division multiple access, where nodes periodically broadcast short messages containing network and synchronization information. During other times, nodes can turn off their communication part to save energy.
The EMACs protocol is developed as part of the EYES project on energy-efficient sensor networks. We also present results obtained for the approach taken, both in simulation and in a testbed implementation on wireless node prototypes.

This chapter presents a cross-layered approach for networking in wireless sensor networks called EMACs (EYES Medium Access Control Scheme). We show how a tightly integrated set of networking protocols forms a good solution to reach the target of an energy-efficient WSN. We combine medium access control organization with an implicit sleep-state scheduling based on independent and dominating sets.

We begin the next section with a short introduction to the architecture of wireless sensor networks as envisioned in the EYES project. Especially, medium access control schemes are introduced that form the core of the communication approach. Then, the actual EMACs approach is presented in detail by taking the local viewpoint of a single sensor node.

Basically, a wireless node has three modes of operation: active, passive, and sleeping. When a node is active, it contributes to the network by taking part in forwarding message packets, in accepting packets from other active and passive nodes, and in keeping the network synchronized in time. Passive nodes on the other hand conserve energy by only keeping track of a single, neighboring active node; at other times, they turn off their radio to save energy. A sleeping node drops out of the communication network for an agreed amount of time, and upon waking up again has to re-synchronize before participating in the network again.

The set of active nodes creates a connected, mesh-like structure that dominates the entire communication network. This way, a message sent by any node can be routed successfully to the respective destination. Transmissions between active nodes are scheduled in time. In this approach, called time-division multiple access (TDMA), time is divided into intervals called time slots, and these time slots are distributed among the nodes. At any point in time, only one node is locally allowed to transmit or receive a message packet, collisions and resulting retransmissions are thus avoided. The distribution of time slots is done so that simultaneous transmissions are possible if the nodes transmitting in them are separated far enough not to cause interference. This reuse of time slots is explained later on in more detail.

In succeeding sections, we present the local algorithms that let a node decide locally on its role, i.e. being active or not, and that let a node choose a non-conflicting time slot. We explain in detail how a time slot is organized to maintain a synchronized, but dynamic network. In the following, we assume the communication graph of the wireless sensor network to be of bounded growth.

We close the chapter with a short discussion on results on the energy consumption of the proposed protocol by a comparative simulation, and report on some lessons learned from a practical implementation on actual hardware, the

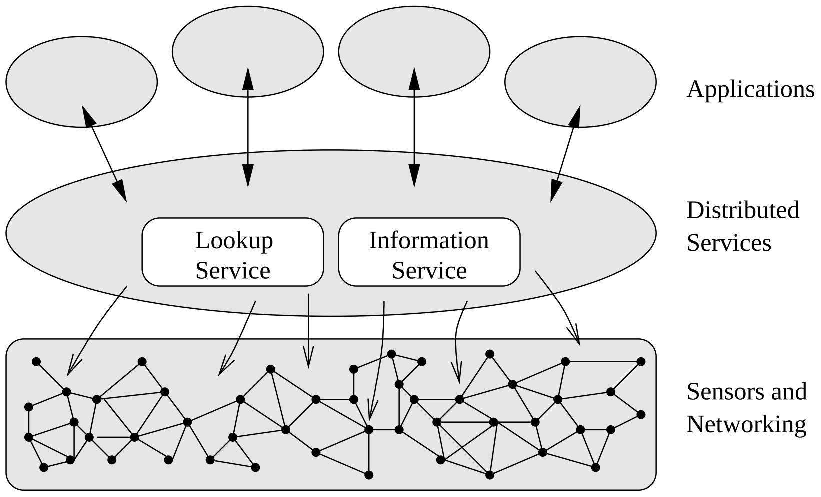


Figure 6.1: Wireless sensor network architecture.

EYES prototype sensor nodes. However, since the focus of this chapter is on the structure itself, and how to create and maintain it, we do not present the entire simulation set, but only an excerpt. For a complete overview of results of the scheme, also taking improved routing protocols into account, see [58, 57, 44].

6.1 Wireless Sensor Network Architecture

Ongoing advances in sensor technology, low power analog and digital electronics, and low-power radio design have enabled the development of cheap, small, low-power sensor nodes that integrate sensing, processing, and wireless communication capabilities.

From collaboration between large groups of such sensor nodes, intelligent behavior can emerge, and ultimately the limited capabilities of single sensor nodes can be surpassed. Sensor nodes are then able to spontaneously create an ad-hoc network, as well as dynamically adapt to device mobility and failure, and react to changes in task and network requirements.

For a wireless sensor network, from a very broad perspective, we look at the following architecture where applications rely on a two-layer structure, with specific, well-separated tasks (see Figure 6.1, [18]).

- The lower layer, called *Sensors and Networking Layer*, provides interfaces of each node to the environment both for in-network communication and sensing. It contains basic communication protocols regulating access to the wireless medium, transportation of message packets to neighboring nodes, and ad-hoc routing protocols. All these protocols must be able to adapt to the dynamic behavior of the network, and account for the limited resources of each individual node as well as the resulting network in general. In this layer, energy-efficiency is crucial. This layer works on and with the communication graph created by the wireless sensor nodes.

- The upper layer, called *Distributed Services Layer*, provides general services that support sensor network applications. These services are usually independent of specific sensor nodes, and provide a less dynamic view of the network as a whole. Specifically, there are two major services to be identified at this layer, a lookup and an information service. The lookup service identifies nodes based on their roles, capabilities, and produced data. It supports mobility, instantiation, and reconfiguration issues of the network. The information service deals with all aspects of the actual data, including access, manipulation, dissemination, and most importantly aggregation of possibly vast quantities of low-level information from the individual sensors. This layer abstracts away from the wireless communication graph, towards graph structures based on logical data structures.

On top of this architecture, applications can be created, usually from a global network perspective and without consideration of actual communication issues.

Sensor networks are expected to be left unattended for long periods of time. Each sensor running on batteries, this requires highly energy efficient approaches at all layers. In the following, we are interested in the Sensors and Networking Layer, and there mainly in the parts that deal with inter-node communication. In contrast to traditional communication protocol stacks that assume an excess of resources, and that can spare the energy and memory to send many messages, we consider an integrated approach that mitigates this overhead in energy usage.

6.1.1 Medium Access Control

The center of the following discussion is *medium access control* (MAC) and the resulting scheduled transmissions of data packets to and from a wireless node. Generally speaking, medium access protocols are used to control transmission requests, authentication and other overheads in a communication network. In

Operation	Energy
Transmit	$21\,mW$
Receive	$14.4\,mW$
Standby	$15\,\mu W$

Table 6.1: Transceiver data (RFM TR 1001).

other words, the task of such a protocol is to ensure that no collisions occur so that message packages can be successfully sent by all nodes in a wireless network. A MAC protocol deals with communication between nodes that are within radio range of each other, multihop communication between nodes further separated in the network is taken care of by routing protocols.

Seen from a local perspective, a MAC scheme controls the use of the radio within a node. During the design of the networking protocol, we assume a single channel transceiver, that is, all nodes transmit at the same frequency. Such a transceiver basically has three general states of operation: transmit, receive, and standby. Considering the actual hardware used for wireless sensor nodes, transmitting consumes more power than receiving and the standby power consumption is lower than the other two by a large factor. As an example we give the specifications of the RFM TR 1001 radio transceiver used in the EYES wireless sensor node prototypes. The respective values are given in Table 6.1.

Current MAC protocols for wireless sensor network settings can broadly be divided into contention based and scheduled approaches, which we briefly characterize next.

6.1.1.1 Carrier Sense Multiple Access (CSMA)

Contention based MAC schemes are reactive schemes, nodes that do not transmit are constantly listening to the channel. A node that wants to transmit a packet first listens whether there is already a transmission going on, i.e. it senses the carrier or wireless medium, and then addresses the recipient to initiate some handshake mechanism that results in the actual broadcast of the packet. Successful reception is also acknowledged by the recipient. Collisions during this process usually result in a random back-off time that is waited until another try. For WSNs, a prominent scheme that falls into this category is the so-called SMAC (Sensor-MAC) protocol first introduced in [60]. The protocol is *carrier sense multiple access with collision detection* (CSMA/cd) based, and these type of MAC schemes do not require tight time synchronization of the nodes. On the other hand, constant listening to the channel requires the transceiver of the nodes to be active most of the times. To overcome this problem, the SMAC protocol also incorporates sleeping-patterns for the nodes to reduce energy usage. We briefly describe this protocol next since we use it for comparison later on.

Basically, in SMAC, nodes locally choose periods of time in which they are not listening for incoming messages. After such a sleep period, the nodes wake up and listen for communication addressed to them, or initiate communication themselves. This requires local synchronization of the sleep and listen periods. Therefore, each node maintains a list with the schedules of its neighbors. A node that has locally fixed its sleeping schedule exchanges this with its neighbors periodically in the listen periods as follows.

- A node, upon entering the network, listens for a defined amount of time to learn about the schedules of its neighbors. If no such information is received, it chooses a random point in time to enter the sleep phase and broadcasts this information in a so-called synchronization packet. This node is then called synchronizer, and defines a schedule in the network.

- If a node receives a synchronization message during this initial time, it adjusts the starting time of its sleep periods accordingly, and follows the schedule of the synchronizer node. Such a node is called a follower. A random time interval is waited before such a follower broadcasts its synchronization message in order to prevent collisions from other followers that are triggered by the same synchronizer.

- If a node that already follows a certain schedule learns of neighbors that follow a different schedule, it keeps its own schedule, but additionally wakes up according to the new schedule. However, it does not include this information in its own synchronization packet to prevent propagation, and rescheduling of the entire network.

Clearly, nodes between different regions with different sleep/listen periods have less sleep time compared to others, and their energy consumption is therefore higher. The protocol gives no guarantees for overall connectivity of the network, this property highly depends on the parameter settings like sleep interval length and on characteristics of the wireless network topology like node density.

6.1.1.2 Time Division Multiple Access (TDMA)

Another approach to allow multiple wireless sensor nodes to share the same frequency is to divide it into different time slots, i.e. by giving each node a reserved time interval in which this node is allowed to send while neighboring nodes are listening. This medium access scheme approach is called *time division multiple access* (TDMA).

Due to the scheduled nature of such an approach, a node can turn its transceiver to standby at all times it is not allowed to send, or is not required to listen. During the time slots, interference due to several nodes transmitting at the same time does not occur. However, in wireless networks, such an approach requires tight synchronization of time for the nodes, and the transmission schedule has to be agreed on beforehand, or is centrally assigned. Also, the assignment of the time slots is not an easy task. Generally speaking, scalability of TDMA based approaches is not as good as that of contention based schemes.

In the following we discuss the TDMA based MAC scheme designed for the EYES wireless sensor network.

6.2 EYES Medium Access Protocol

In this section we introduce the EYES Medium Access Protocol (EMACs, [57, 58]), a cyclic TDMA based protocol with completely distributed and local operation. The scheme, together with an implicit backbone structure presented thereafter in Section 6.3, gives an energy-efficient and scalable solution which is specifically designed for wireless sensor networks.

In the EMACs approach, the set of wireless nodes is divided into *active* and *passive* nodes. The active nodes form a backbone of the communication network, that is, they are a connected dominating set. Passive nodes are adjacent to at least one active node in order to make use of the backbone structure. After presenting the MAC scheme, we discuss the local decision algorithm that creates this structure, and discuss implications and advantages of this backbone based approach. For now, we assume the active nodes to form a connected dominating set, and that each node knows its status. At the start of the network, and also when new nodes are added to the network, nodes are active, and may later on decide to become passive.

The EMACs scheme operates on a cyclic schedule (see Figure 6.2). Time is divided into frames, and each frame is further divided into *time slots*. Nodes can use the time slots to transfer data without having to content for the medium, or

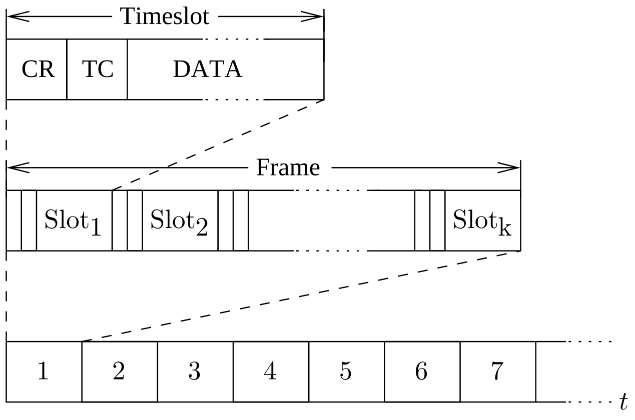


Figure 6.2: TDMA organization in EMACs.

having to waste energy due to collisions of transmissions. Locally, this requires that each time slot is only controlled by one node, the so-called time slot *owner*. In each frame, each active node should own at least one time slot. The number of time slots per frame is fixed, and each frame basically has the same schedule which is repeated giving a cyclic schedule.

Only active nodes own and control a time slot. A passive node does not own a time slot, but communicates via an active neighboring node. It chooses one active neighbor to which it synchronizes, and with which it handles its communications. This allows for a significant energy conservation, as the rest of the time, this node can switch its transceiver to standby.

6.2.1 Time Slot Structure

A node which owns a time slot performs three tasks in the time alloted for the slot. For these tasks, each time slot is divided into three sections, a *Communication Request* (CR), a *Traffic Control* (TC), and a *DATA* section (see Figure 6.2). These are now described in detail.

- **Communication Request Section (CR)**

  For a short fraction of the time, a time slot owner listens to incoming requests from nodes in its neighborhood that want to communicate with it. This is especially important for communication with passive neighbors.

Passive nodes do not own and control their own time slot, and therefore are also unable to receive any requests for communication. However, these nodes can still transmit data via neighboring active nodes by placing a request in the CR section of such a node.

Collisions of requests may occur, but this is detected by the time slot owner, that informs its neighborhood about this fact in the TC section. These collisions can then be resolved by contention based approaches, e.g. by random back-off times for the next request. Since the data rate in wireless sensor networks is low, collisions in the CR section are expected to occur seldom.

A special type of communication request, which is transmitted by a node that wants to join the network and become active is the *node announcement*. With such a message, the new node informs all active neighbors of its existence and chosen time slot within a frame.

- **Traffic Control Section (TC)**

  The most important part of each time slot is the Traffic Control section. Here, a short, timed broadcast is sent by the time slot owner in every frame. Since the beginning of each TC section is timed precisely, this section is—so to say—the distributed heartbeat of the network.

  In more detail, the broadcast message in the TC section contains the identifier of the time slot owner, a possible acknowledgment to a preceding request, as well as control and synchronization information. Most importantly, the TC message gives the so-called *slot schedule table* of the time slot owner, that is a list of all time slots that are occupied by other active neighbors. Nodes have to listen to the TC section of all neighboring active nodes, and thereby have good knowledge about their local neighborhood. It is straightforward to see that a node listening to all slot schedules of its active neighbors knows all occupied time slots within its 2-neighborhood. This information is needed to choose a non-conflicting time slot, which we describe in the next section.

  The TC section also informs neighbors on how the directly following DATA section is to be used.

- **Data Section (DATA)**

  The remainder of the time slot, the DATA section, is reserved for the actual transfer of message packets. The format and length (which is bounded by the CR-section of the following time slot) is free and is decided by the actual data that needs to be sent, as given by higher layers. This section is rather long compared to the first two sections.

Seen from the time slot owner, the DATA section can be used for direct transmissions to a specific neighbor, for a broadcast to be received by all neighbors, and also for incoming message packets, e.g. from passive neighbors. The concrete use is specified in the TC section.

Due to the scheduled nature of the approach, nodes can switch their wireless transceivers to standby during all times that they are not involved in the communication. That is, during CR sections when they have no request and do not own the respective time slot, and during the DATA sections they do not participate in. In this context, note that the TC sections that nodes have to listen in are rather short. Passive nodes only need to listen to one TC section per frame, since they only need to synchronize with their chosen active neighbor. This allows them to save even more energy.

6.2.2 Distributed Time Slot Allocation

Wireless sensor networks lack central control. The allocation of time slots to individual nodes therefore cannot be centrally assigned. A node has to choose a time slot locally based on local information. This decision process for a node that needs to allocate its time slot is described next.

As explained above, each active node transmits a slot schedule table in its TC section. This list contains all occupied time slots of its one-hop active neighbors, and its own slot. Note that this information can be efficiently encoded by a bit-string of length equal to the number of time slots in a frame.

New nodes entering the active state, after listening to a complete frame, have all information needed to choose a *non-conflicting* time slot, that is, a time slot which is not used by another active node within two hops from this node. After three hops, a time slot can be reused, as this does not result in interference.

The new node then chooses the first non-conflicting time slot. A closer look at this process reveals its likeness to an online-approach to graph coloring in $G^2 = (V, E^2)$ (see e.g. [22]). The second power G^2 of the communication graph is defined on the same set of vertices, while there is an edge between two vertices if their distance in the original graph G is less than or equal to 2. This online approach basically models a greedy, first-fit coloring.

However, when two or more nodes choose the same time slot in the same frame, conflicts may occur. Such an interference is noticed by neighboring active nodes, who can then use their own time slot, and the TC section in particular,

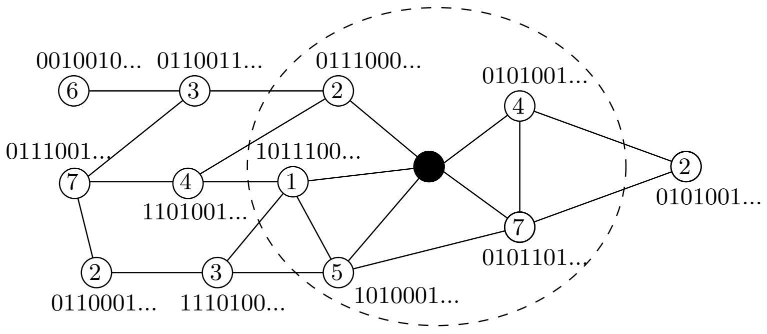


Figure 6.3: Time slot allocation example.

to force these new nodes to drop their time slot again, and after some random back-off time to re-allocate a time slot. Also, it may happen that a newly woken node has two neighbors with the same time slot, e.g. when this node connects two components. In this case, the CR section of these conflicting neighbors is used to report this fact to them, and new time slots are also chosen in these neighbors.

The process of deciding on a time slot is also described by an example given in Figure 6.3. The numbers at the nodes denote the respective time slots, and the time slot schedules are also given. Note that this schedule always includes a node's own slot. Suppose that the black node enters the network, and learns about the slots and schedules of the neighbors. After combining the time slot schedules by an or-operation over the bit-strings yielding '1111101...', the node can choose time slot 6 without causing collisions locally. This choice then also affects the time slot schedules of the neighboring nodes, which adjust accordingly by setting time slot 6 to occupied.

The number of time slots is given by the frame length, i.e. the number of slots per frame. This frame length is globally fixed, and it may thus happen in dense networks that a new node cannot choose a non-conflicting time slot. In this case, the node postpones the decision until a time slot becomes available, or it becomes passive and needs not to own a time slot. As we show in the next section, for wireless communication graphs, with a bounded number of slots per frame, we can achieve a globally connected backbone for the communication network by the active nodes. This also implies that the above problem of a node having to postpone its time slot decision can be resolved.

6.3 Active and Passive Nodes

In the EMACs protocol, we partition the set of wireless nodes into active and passive nodes. In this section, we now explain how this separation is actually done by a local decision process in each node. Recall that we assumed the set of active nodes to be a connected dominating set in the wireless communication network.

The main idea behind the structure, and the algorithm follows the argumentation of wireless communication graphs, and the models presented in Chapter 3: when many nodes are placed close to each other, we obtain a highly connected graph, and vice versa. This also implies that in highly connected areas of the graph, there is a lot of redundancy. Therefore many nodes can become passive to save energy in this case, while the connected structure given by the graph induced by the active nodes suffices for a functional wireless sensor network.

Each node that enters the network, e.g. by waking up or being deployed, has to decide whether it is needed as part of the backbone, which we refer to as *connected active set* further on. In this set, the active nodes may take on several, different roles presented next before explaining the actual decision algorithm that relies on these roles.

6.3.1 Roles of Active Nodes

The connected active set is divided into groups of nodes with different roles. These roles are

- *anchor* nodes which form the base of the set,
- *bridge* nodes which are needed for connectivity,
- *undecided* active nodes which have not taken a decision, and
- *nonmember* active nodes which are active stemming from a decision at higher layers of the wireless sensor applications.

In more detail, the set of anchors is created to form a maximal independent set in the communication network. Thus, the set of anchors is a dominating set covering the network, and no two anchors are neighbors.

The bridges are then connections between nearby anchors. In this context, we call two anchors nearby if they are within a distance of three hops of one another. Due to the maximality of the independent set given by the anchor nodes in the communication network G, we have the following lemma establishing overall connectivity of the active set using anchor and bridge nodes as described.

Lemma 6.1. *Let $G = (V, E)$ be connected, and let $I \subseteq V$ be a maximal independent set in G. Furthermore, let $B \subseteq V$ be a set of vertices such that every two vertices $u, v \in I$ from the independent set with $d_G(u, v) \leq 3$ are connected in $G[I \cup B]$, the graph induced by $I \cup B$. Then, $I \cup B$ is a connected dominating set in G.*

Proof. The domination property is already satisfied by the MIS I. Consider two vertices $i, j \in I \cup B$, and let $P = (i_0, i_1, \ldots, i_k)$ denote a path from $i = i_0$ to $j = i_k$ in G. For each $i_t \in P, t = 0, \ldots, k$ choose a vertex $\bar{i}_t \in I$ from the maximal independent set I such that $(i_t, \bar{i}_t) \in E$ or $i_t = \bar{i}_t \in I$ holds.

Since $(\bar{i}_{t-1}, i_{t-1}, i_t, \bar{i}_t)$ is a path in G, we have $d_G(\bar{i}_{t-1}, \bar{i}_t) \leq 3, i = 1, \ldots, k$. Thus, $\bar{i}_{t-1}$ and $\bar{i}_t$ are connected in $G[I \cup B]$, and also a path between i and j exists in $G[I \cup B]$. □

There are two types of bridges. Recall that active nodes own a time slot and periodically transmit a TC section including synchronization information and the role of the node. A node that receives the TC sections of two or more anchor nodes is called *direct* bridge if it is active. When two nodes are needed to connect two nearby anchors of distance 3, these two nodes form a *distributed* bridging pair. A bridge node always informs its neighborhood about the anchor nodes it connects. This way, redundant, local bridges need not to be set up.

A special role is given to undecided active nodes, this role is mainly used when a node enters the network and has not found a neighboring anchor.

Furthermore, we allow nodes that are not needed in the above structure to be active as well. For example, this may be the case for special sensor nodes that perform important tasks for higher layers or are explicitely needed in applications. These nodes are called *nonmember active.*

6.3.2 Local Decision Algorithm

Each node that enters the network, e.g. by waking up or being deployed has to decide whether it is needed as part of the connected active set. This is achieved by a local algorithm described in this part. Additionally, the decision process is performed when a change in the local topology given by the active nodes occurs. This is detected from the TC sections transmitted by all active neighbors in each frame.

For a node $v \in V$, the decision process is as follows (an overview of the process is also given in Figure 6.4):

1. **Anchor node decision**

   When there are neighboring anchor nodes, v cannot become an anchor itself. However, when there is no anchor identified in $\Gamma(v)$, the identifiers of the nodes are used in an approach identical to the local greedy strategy of Section 2.4. The node v determines whether it is the node with the lowest identifier in its neighborhood, and if this is the case, it becomes an anchor. Otherwise, it waits for undecided nodes with lower identifier to decide first.

2. **Direct bridge decision**

   When there are two or more anchors in $\Gamma(v)$, the node v identifies all adjacent bridges based on the control information in the TC message. In case a pair of anchors is not yet connected, it performs this part by becoming a respective direct bridge.

3. **Distributed bridge decision**

   If v does not change to an anchor or a direct bridge, it may still become a distributed bridge. This decision cannot be taken according to local neighborhood information only. The node v transmits the identifier of the neighboring anchor node with lowest identifier in its TC section of the following frame. This way, a distributed bridging detection becomes possible, and another node that can complete a distributed bridging pair answers accordingly such that this connection of the anchors can be formed.

4. **Becoming passive**

   At last, if v is neither needed as anchor, nor bridge, it may become passive, and no longer claim a time slot. It may also remain active as nonmember active node in the communication network.

Especially during the first decision on becoming an anchor node, we see that the node with lowest identifier can always take a decision that results in a role other than undecided active. The undecided active role is thus only a temporary one.

6.3.3 Connected Active Set Structure

In this part, we explain the structure and the resulting properties given by the connected active set in the wireless communication network. Recall that a wireless communication network is modeled by a graph of f-bounded growth.

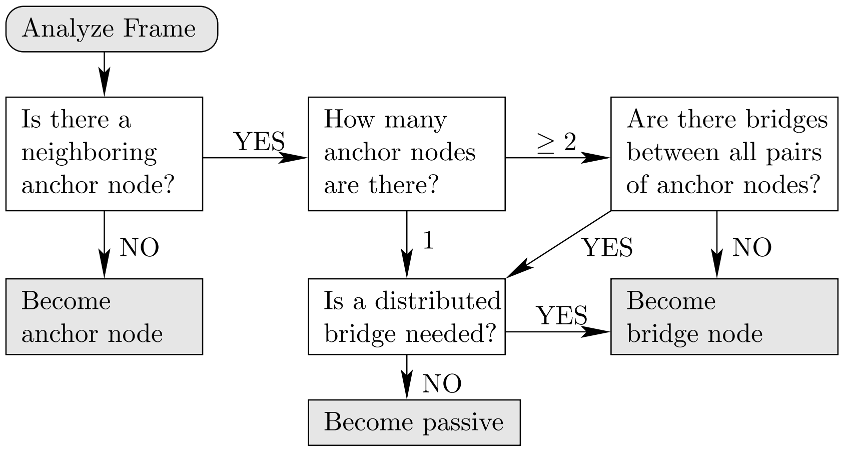

Analyze Frame
Is there a neighboring anchor node?
YES
How many anchor nodes are there?
≥ 2
Are there bridges between all pairs of anchor nodes?
NO
1
YES
NO
Become anchor node
Is a distributed bridge needed?
YES
Become bridge node
NO
Become passive

Figure 6.4: Overview of the local decision algorithm.

A closer look at the construction of the connected active set in the communication network shows that by connecting all nearby anchor nodes, we create a mesh-like structured backbone. This structure does therefore not hinder backbone based routing protocols by introducing long, unnecessary detours. Additionally, the maximal independent set given by the anchor nodes can be exploited in cluster-based routing strategies such as proposed e.g. in [44].

Another important advantage of the structure created implicitly is with respect to the local time slot allocation in the TDMA based MAC scheme. Recall that the frame length is globally fixed, and the time slots in a frame together with the allocation to the respective owner are repeated in a cyclic way. Not considering nonmember active, but anchor and bridge nodes only, we only require a constant number of time slots per frame so that the allocation scheme presented in Section 6.2.2 results in an operational network. We call a network operational if every active node has chosen a non-conflicting time slot. Looking at the 2-neighborhood of an active node, it is easy to see that there are only a constant number of active nodes in this neighborhood as follows.

In the 2-neighborhood, there are at most $f(2)$ anchor nodes, however, considering an anchor node, there are bridges introduced to connect with all anchor nodes within distance 3. There are at most $f(3)$ such anchor nodes, and in order to connect a pair of anchor nodes, at most $f(1)$ bridges are introduced. It cannot happen that more bridges are present as these would know about one another and redundant bridges are not created. A bridge consists of at most 2 nodes. Overall, there are thus no more than a constant number of nodes active within distance 2 of a node. The allocation of time slots follows an online greedy approach which then also requires a constant number of time slots.

At the initial employment of the network, the connected active set is not created yet, and all nodes try to own a time slot. In this case, more time slots may be needed. As explained earlier, this is not possible and nodes then wait until they can choose a free time slot. It is straightforward to see that a node can eventually choose a time slot: if there are locally too many nodes that own a time slot, some of them are redundant for the connected active set. These nodes become passive, and therefore their time slot is released and can be chosen by another node.

6.4 Implementation and Results

The EMACs protocol, together with the implicit construction of the connected active set, was implement both in a simulator and in a testbed consisting of EYES wireless sensor node prototypes. In this section, we present some comparison based results from the simulations. We only present an excerpt of the simulations performed in the line of research of the EYES project. For more results, we refer to [58, 57, 44]. Nevertheless, the results presented in the following stand exemplary for many other scenarios simulated.

Before presenting the results, we would like to point out a technical detail concerning the encoding of the roles of active nodes that is used in our implementation. The approach taken has the advantage of using only a limited number of bits to be added to the TC section of an active node, and also suffices in practice to create and maintain the connected active set without additional control message.

In order to efficiently encode the roles of active nodes, the synchronization information of the TC section of a node includes an additional field, called *active identifier* (AID) of the same length as a node identifier. The encoding of this fields, and its meanings, are given in Table 6.2. For the AID, an additional first bit when using node identifiers is used to encode whether or not the node is a bridge. In case it is a bridge, this bit is set to '1', in all other cases to '0'.

Nodes that are active, but not needed in the connected active set (nonmember active nodes), are identified by the neighboring anchor with the lowest identifier which also helps identifying distributed bridges. A node that wants to become passive first becomes a nonmember active node to determine its possible

Description	Encoding
Anchor Node	$\text{AID}_v = id_v$
Bridge	$\text{AID}_v = (id_{u_1} \text{ XOR } id_{u_2})$, u_1, u_2 Anchors
Undecided Active	$\text{AID}_v = 0$
Nonmember Active	$\text{AID}_v = \min\{id_u\}$, u Anchor

Table 6.2: Roles encoded by the AID of an active node $v \in V$.

distributed bridge status, and then can release its time slot in a later frame.

The AID field of an anchor contains twice the same identifier. In this case, a single bit would have sufficed. However, the length of a TC section is always fixed to be the same length. Therefore, this approach actually saves this extra bit in the TC section.

When setting up and maintaining a structure like the connected active set, control messages would be needed for coordination. However, from a change in the AID of a neighboring node, all control messages otherwise needed can be inferred sufficiently. Thus, the connected active set is created and maintained without additional messages leaving the DATA section free for the data transfer of the higher layers.

6.4.1 Simulational Results

For a simulational study of the EMACs protocol, the OMNeT++ discrete event simulator [49], together with a framework for mobile, wireless networks is used. In the simulator, a physical layer with an energy model is implemented to record the send and receive energy consumption of the transceiver. Switching between the sending, receiving, and standby states of the transceiver takes time and consumes energy; this is also considered in the energy model. The data is taken from the RFM TR 1001 wireless transceiver which is also present in the actual sensor node prototypes. The values are given in Table 6.1 on page 95.

6.4.1.1 Simulation Scenarios

The simulation setup consists of randomly placing 46 nodes in a square area, and each node is given a transmission power setting that corresponds to a transmission range of 1/5 the side length. Of the 46 nodes, one node is chosen explicitely to be a sink node that collects data from the sensor network. This data is produced by 5 specific source nodes periodically, at varying data rates. The remaining 40 nodes are used for relaying the data to the sink.

The multi-hop transport of message packets is done with the help of Dynamic Source Routing (DSR, [31]), a reactive routing protocol designed for dynamic wireless networks. There are then two main types of messages that have to be delivered by the MAC scheme in the network: sensor data and control messages for the routing protocol.

The nodes are mobile according to an adjusted random way-point model with bounded random speed and waiting time as follows. After waiting for a certain time, a node determines a new way-point within the simulation area, together with a speed from a fixed interval at random. Then it moves towards this point on a straight line. When this way-point is reached, the process starts anew. This approach gives a good mix of static and mobile nodes.

For the following simulational results with respect to energy efficiency, we use the *network lifetime* as metric to evaluate the performance. This metric measures the total operational time of the entire network, that is, the amount of time until the network is no longer able to report sensed data sufficiently to the sink. In our setup, the network is no longer operational when 30% of the relay nodes have used up their energy. The 5 nodes that produce the data, and the sink node are given an unlimited energy supply.

In a simulational study of wireless, ad-hoc networks it is not very meaningful to give absolute results. Therefore, we resort to a comparative study, and use the SMAC protocol presented in Section 6.1.1 as a reference point. This protocol was also implemented and run on the same network topologies. Here as well, DSR is used to route the sensed data to the sink.

The EMACs protocol implemented in the simulation uses 16 time slots per frame, and a frame has a length of $1s$. Data packets containing sensor readings of 5 bytes are created at the 5 source nodes, and in the DATA section of each node only one data packet can be sent.

6.4.1.2 Results and Discussion

There are two basic scenarios which were simulated, a static and a mobile network. The results for various rates with which the sensor data is injected into the network are given in Figure 6.5 for the static case, and in Figure 6.6 for the dynamic case. All data is normalized to the network lifetime when using SMAC as Medium Access Control Scheme in static network topologies.

From the figures, we can see that the EMACs protocol prolongs the lifetime of the network by 30% to 55% in the static network topologies compared to

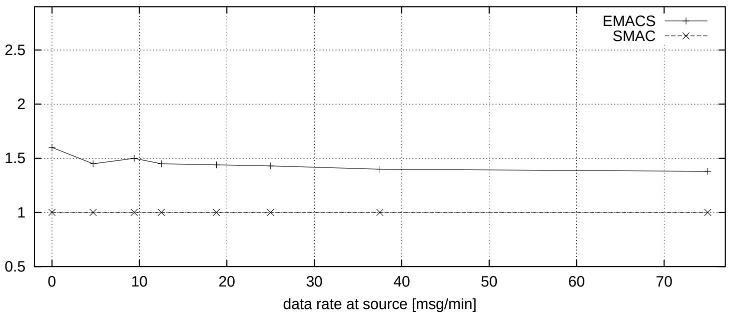


Figure 6.5: Network lifetime in static topologies (Normalized to SMAC).

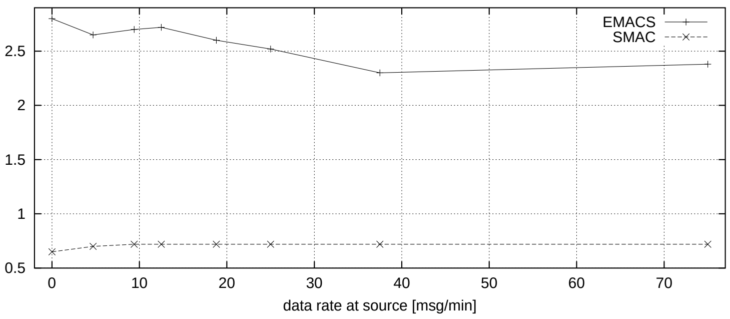


Figure 6.6: Network lifetime in mobile topologies (Normalized to SMAC in static topologies).

SMAC. The EMACs protocol clearly benefits from mobility, and it is able to extend the lifetime by a factor of more than 2 compared to SMAC in the static case.

Comparing static and mobile case against one another, the performance of SMAC degrades to 75% compared to the static case. This is due to the larger number of control messages needed to reestablish paths broken due to mobility. Additionally, nodes more frequently have to adopt to different sleep/listen periods, and these overlapping sleep schedules result in longer listening time.

Especially interesting is the fact that EMACs performs better in dynamic scenarios than in static topologies. This results mainly from the creation and maintenance of the connected active set: a node that decided to be active remains active until it runs out of energy in a static network. On the other hand, also a passive node only becomes active after an active neighbor no longer participates in the backbone. This obviously leads to unbalanced energy consumption in the network, and shows the need for the concept of *role rotation* where the burden of being active is distributed better among the nodes. Role rotation is—so to say—the introduction of some dynamic behavior into the connected active set structure in rather static topologies. In the mobile scenarios, the changes in the topology already force this role rotation.

6.5 Conclusions

In this chapter, we proposed a TDMA-based communication scheme for wireless sensor networks called EMACs. The operation of the scheme is completely based on local information, except for time-synchronization, which is done by timed, periodically transmitted control messages. The EMACs integrates medium access control and a backbone creation given by the active nodes, the Connected Active Set. This structure is used manyfold, by enabling network-wide, collision-free communication, by ensuring a feasible time slot allocation in a frame of fixed length, and by allowing passive nodes to follow a more rigorous sleeping pattern.

EMACs has good energy-efficiency due to the scheduled operations which allows for long periods with a node's radio being turned off. These periods are present in both active, and passive nodes. Due to the stringent TDMA structure in which a node is assigned a slot that cannot be used by other, neighboring nodes, bandwidth utilization and latency are not optimal. However, this is not a crucial factor in WSNs, as the envisioned applications require only a low data rate.

Taking a closer look at the Connected Active Set structure, which is a mesh-like dominating set in the communication graphs, we also see that the set of anchor nodes forms a maximal independent set in the network. This can be immediately exploited in higher layers of the network by independent set based clustering schemes.

In a simulational, comparative study, the potential of the approach with respect to energy consumption has been shown. There, the EMACs scheme significantly increases the network lifetime compared to SMAC, especially in dynamic network topologies.

6.5.1 Testbed Implementation

A testbed running EMACs was implemented on sensor node prototypes and evaluated in the EYES project.

The EYES sensor node prototype is based on the RFM TR 1001 RF transceiver and the Texas Instruments MSP430F149 microprocessor. Apart from these two devices, a 4 MBit Serial Flash memory, an RS 232 interface, and a digital potentiometer to adjust the RF output power is present [19].

On the prototype, the protocol only requires a small amount of memory. Including basic on-demand routing functionality, less than 200 bytes data memory, and 5 kbytes of program memory are used in the implementation [20]. A frame consists of 32 slots, each of $100ms$ length. A frame is then $3.2s$ long. This may seem long, but as the data rates in wireless sensor networks are expected to be low, such a long frame allows for long periods in which the radio hardware can save energy.

We close this chapter with the following statement from the EYES project documentation on the evaluation of the EMACs protocol in the testbed implementation:

> [In practice, EMACs] proves to be fault resistant and assures a very small number of collisions. [20]

Chapter 7

Conclusions

In this thesis, we discussed independent and dominating sets in graphs that model wireless communication networks. The discussion was done both from a graph-theoretic and practical angle, and also both from a central and local perspective. The research was motivated by the settings given in wireless ad hoc and sensor networks, and both theoretical results and practical approaches are obtained and presented.

First, we have discussed the possibilities of modeling wireless communication networks by Bounded Area Graphs and considered the (polynomially) bounded growth property. One of the advantages of describing a wireless communication network topology via bounded growth is the fact that we leave the geometry of the models behind. Having a geometric representation available differs greatly from the case that only adjacency information of a graph is present. The questions of recognition, reconstruction, and embedding of geometric graphs were discussed.

On polynomially bounded growth graphs, we presented a novel polynomial-time approximation schemes for the Maximum (Weight) Independent Set and Minimum Dominating Set problems based on local neighborhoods. Directly exploiting the bound on the cardinality of independent sets in neighborhoods of bounded radii, we obtain approximation schemes that require no geometric information. The algorithms of the schemes work expanding local neighborhoods of a chosen central vertex until the cardinality of an optimal solution in these neighborhoods no longer increases too much. All approximation algorithms have a runtime of $n^{O(1/\varepsilon \log 1/\varepsilon)}$, when the desired approximation guarantee is $(1+\varepsilon) > 1$.

Furthermore, we developed robust approaches, which thus always terminate and produce meaningful output, independent of the input presented to the respective algorithm. In light of the intractability of the recognition problem, this means that we can run the algorithm even without being certain of its bounded growth, and we receive either a desired solution or a certificate disproving the bounded growth property of the input graph. This feature of the algorithm is especially useful for real-world wireless communication graphs, where the clearly defined theoretical properties of such graphs may not be given due to unforseen circumstances.

Additionally to the advantage of being independent of geometric information, our local neighborhood based approach is more direct and much simpler to implement compared to existing schemes based on geometric separation. In separation based approaches, the shifting strategy requires the construction of several candidate solutions, one for each possible combination of separation parameters, and then the best one is returned. In contrast, in our approach, a partial solution inside a neighborhood is kept and we can neglect this part of the graph upon going on.

Low-resource wireless ad hoc and sensor networks are inherently local. The nodes usually lack sufficient memory and computational power to obtain and maintain a global view of the network topology. In the resulting scenario, represented by the $\mathcal{LOCAL}$ message passing model, we discussed distributed, in-network algorithms to create independent and dominating sets. In this area, maximal independent sets play an important role, both in theory and in practice. Having only local topology information, fast symmetry breaking is a non-trivial task, and a MIS captures this task in a very simple statement. In Chapter 5, we introduced a fast and deterministic algorithm that constructs a maximal independent set in graphs of bounded growth. The approach works in subsequent phases by first constructing a sparse, $O(\log \Delta)$-ruling independent set, and then making this set dense up to the point where maximality is reached. The algorithm terminates after $O(\log \Delta \log^* n)$ communication rounds and requires only messages of small, i.e. $O(\log n)$, size.

Combining the ideas of the neighborhood based PTAS for the Max-IS and Min-DS problems on polynomially bounded growth graphs with the fast MIS construction, we obtained distributed approximation schemes for these optimization problems. The approach allows us to construct $(1 + \varepsilon)$-approximate solutions distributed within the network, and without requiring information about the global topology in the nodes.

On the practical side, we presented EMACs, a communication approach for wireless sensor networks with TDMA based, scheduled transmissions. The approach implicitly creates a backbone that is based on a maximal independent set given by the anchor nodes. This backbone allows for relatively long sleeping periods of nodes, while at the same time maintaining a connected and dominating structure that keeps the overall network operational. The communication protocol is implemented and evaluated both in a simulation environment and in a real-world testbed. It significantly increases the network lifetime compared to an existing communication approach.

Though this thesis gives positive answers to the previously open problems of a PTAS for the Maximum (Weight) Independent Set and Minimum Dominating Set problems on Unit Disk Graphs without geometric representation, and a poly-logarithmic local, distributed maximal independent set construction for bounded growth graphs, there are still interesting open questions in this area. Further research in the area of graph-theoretic concepts in wireless communication networks may, e.g. , go into the following directions:

- **Wireless Communication Graphs**

  Concerning graph models for wireless communication networks, there are still many open questions on recognition and embedding. For example, it is not known whether Unit Disk Graph Recognition is in the class *NP*. An embedding of Unit or Quasi Disk Graphs with good quality is an important problem, for which little is known. Also from a practical point of view, provably good algorithms giving each node its position will improve many applications.

- **Approximation Algorithms**

  For algorithms to compute approximative solutions to optimization problems, a next step would be to look into the various variants of independent and dominating sets. For example, considering connected dominating or independent dominating sets, and trying to devise approximation schemes for these problems that do not depend on geometric information is an interesting research direction.

  Another open problem for graphs without geometric representation is the Minimum Vertex Cover problem (this problem is already solved for (Unit) Disk Graphs with a given representation). Since the Minimum Vertex Cover problem is the dual to the MAX-IS problem, for neighborhoods of bounded radius we can obtain a locally optimal partial solution to the problem. However, combining several of such local, partial solutions seems to be not that easy.

- **Distributed and Local Algorithms**

  Especially in wireless sensor networks, local algorithms are called for. On a higher level, the question of how global solutions can be approximated by a collection of local, partial solutions offers many challenges. Even for problems which are considered solved from a central perspective, the question of locality still leaves room for major improvements, and further understanding.

  In this context, a fast, i.e. poly-logarithmic, maximal independent set construction by a local algorithm is still a famous open problem on a general graph.

Generally speaking, while it is important to study graph-theoretic problems on general graphs, it is also fundamental to study these problems on graph classes that model topologies that occur in practice. In this thesis, we contributed to the study of independent and dominating sets in wireless communication networks.

Bibliography

[1] P.K. Agarwal, M. van Kreveld, and S. Suri. Label placement by maximum independent set in rectangles. *Computational Geometry*, 11(3–4), p. 209–218, 1998.

[2] A.D. Amis and R. Prakash. Load-balancing clusters in wireless ad-hoc networks. In *Proc. 3rd IEEE Symp. on Application-Specific Systems and Software Engineering Technologies*, p. 25–32, 2000.

[3] P. Assouad. Plongements Lipschitziens dans $\mathbb{R}^n$. *Bull. Soc. Math. France*, 111(4), p. 567–583, 1983.

[4] B. Awerbuch. Complexity of network synchronization. *Journal of the ACM*, 32(4), p. 802–823, 1985.

[5] B. Awerbuch, A.V. Goldberg, M. Luby, and S.A. Plotkin. Network decomposition and locality in distributed computing. In *Proceedings of the 30-th Symp. on Foundations of Computer Science (FOCS)*, p. 364–369, 1989.

[6] B.S. Baker. Approximation algorithms for NP-complete problems on planar graphs. *Journal of the ACM*, 41(1), p. 153–180, 1994.

[7] R. Boppana and M.M. Halldorsson. Approximating maximum independent sets by excluding subgraphs. *Bit*, 32, p. 180–196, 1992.

[8] H. Breu and D.G. Kirkpatrick. Unit disk graph recognition is NP-hard. *Computational Geometry. Theory and Applications*, 9(1-2), p. 3–24, 1998.

[9] X. Cheng, X. Huang, D. Li, W. Wu, and D.-Z. Du. A polynomial-time approximation scheme for the minimum-connected dominating set in ad hoc wireless networks. *Networks*, 42, p. 202–208, 2003.

[10] B. N. Clark, C. J. Colburn, and D. S. Johnson. Unit disks graphs. *Discrete Mathematics*, 86, p. 165–177, 1990.

[11] S.A. Cook. The complexity of theorem-proving procedures. In *Proc. 3rd Annual Symposium on Theory of Computing*, p. 151–158, 1971.

[12] P. Crescenzi and V. Kann (Eds.). A compendium of NP optimization problems. `http://www.nada.kth.se/~viggo/problemlist/`.

[13] B. Das and V. Bharghavan. Routing and multicasting in multihop, mobile wireless networks. In *Proc. ICC'97*, p. 376–380, 1997.

[14] S. Aaronson (Ed.). The complexity zoo. . `http://www.complexityzoo.com/`.

[15] F. Eisenbrand and F. Grandoni. On the complexity of fixed parameter clique and dominating set. *Theoretical Computer Science*, 326, p. 57–67, 2004.

[16] T. Erlebach, K. Jansen, and E. Seidel. Polynomial-time approximation schemes for geometric graphs. In *Proceedings of the 12-th ACM-SIAM symposium on discrete algorithms*, p. 671–679, Washington, DC, 2001.

[17] EYES. Energy-Efficient Sensor Networks. (IST-2001-34734), `http://www.eyes.eu.org`.

[18] EYES. Energy efficient protocols for sensor networks, 2003. EYES project deliverable 2.2.

[19] EYES. Energy efficient sensor architecture, 2003. EYES project deliverable 2.1.

[20] EYES. Prototype sensor nodes, 2003. EYES project deliverable 2.3.

[21] U. Feige. A threshold of $\ln n$ for approximation set cover. *Journal of the ACM*, 45(4), p. 634–652, 1998.

[22] J. Fiala, A.V. Fishkin, and F.V. Fomin. Off-line and on-line distance constrained labeling of graphs. In *European Symposium on Algorithms, 9th ESA '01*, p. 464–475. Springer, LNCS 2161, 2001.

[23] M.R. Garey and D.S. Johnson. *Computers and Intractability: a Guide to the Theory of NP-completeness*. W.H. Freeman and Company, 1979.

[24] M. Gerla and J. Tsai. Multicluster, mobile, multimedia radio network. *ACM/Baltzer Wireless Networks*, 1(3), p. 255–265, 1995.

[25] A. Gupta, R. Krauthgamer, and J. Lee. Bounded geometries, fractals, and low-distortion embeddings. In *Proceedings of the 44-th IEEE Symposium on Foundations of Computer Science*, p. 534–543, 2003.

[26] M.M. Halldorsson. Approximating the minimum maximal independence number. *Information Processing Letters*, 46, p. 169–172, 1993.

[27] J. Hastad. Clique is hard to approximate within $n^{1-\epsilon}$. *Acta Mathematica*, 182, p. 105–142, 1999.

[28] D.S. Hochbaum. Efficient bounds for the stable set, vertex cover and set packing problems. *Discrete Applied Mathematics*, 6, p. 243–254, 1983.

[29] D.S. Hochbaum and W. Maass. Approximation schemes for covering and packing problems. *Journal of the ACM*, 32(1), p. 130–136, 1985.

[30] H.B. Hunt III, M.V. Marathe, V. Radhakrishnan, S. S. Ravi, D.J. Rosenkrantz, and R.E. Stearns. NC-approximation schemes for NP- and PSPACE-hard problems for geometric graphs. *J. Algorithms*, 26(2), p. 238–274, 1998.

[31] D. Johnson, Y. Hu, and D. Maltz. The dynamic source routing protocol for mobile ad hoc networks. IETF Internet draft, 2003.

[32] J. Kleinberg, A. Slivkins, and T. Wexler. Triangulation and embedding using small sets of beacons. In *Proceedings of the 45-th IEEE Symposium on Foundations of Computer Science*, p. 444–453, 2004.

[33] R. Krauthgammer and J. Lee. Navigating nets: Simple algorithms for proximity search. In *Proceedings of the 15-th ACM-SIAM Symposium on Discrete Algorithms*, p. 798–804, 2004.

[34] F. Kuhn, T. Mosciboda, and R. Wattenhofer. Unit disk graph approximation. In *Proceedings of the 2nd ACM DIALM-POMC Joint Workshop on Foundations of Mobile Computing*, p. 17–23, 2004.

[35] F. Kuhn, T. Moscibroda, T. Nieberg, and R. Wattenhofer. Fast deterministic distributed maximal independent set computation on growth-bounded graphs. In *Proceedings of the 19-th International Symposium on Distributed Computing (DISC)*, p. 273–287. Springer, LNCS 3724, 2005.

[36] F. Kuhn, T. Moscibroda, T. Nieberg, and R. Wattenhofer. Local approximation schemes for ad hoc and sensor networks. In *Proceedings of the 3rd ACM DIALM-POMC Joint Workshop on Foundations of Mobile Computing*, p. 97–103, 2005.

[37] F. Kuhn, R. Wattenhofer, and A. Zollinger. Ad-hoc networks beyond unit disk graphs. In *Proceedings of the 1st ACM DIALM-POMC Joint Workshop on Foundations of Mobile Computing*, p. 69–78, 2003.

[38] N. Linial. Locality in distributed graph algorithms. *SIAM Journal on Computing*, 21, p. 193–201, 1992.

[39] M. Luby. A simple parallel algorithm for the maximal independent set problem. *SIAM Journal on Computing*, 15, p. 1036–1053, 1986.

[40] N.A. Lynch. *Distributed Algorithms.* Morgan Kaufmann Publishers, Inc., 1996.

[41] M.V. Marathe, H. Breu, H.B. Hunt III, S. S. Ravi, and D.J. Rosenkrantz. Simple heuristics for unit disk graphs. *Networks*, 25, p. 59–68, 1995.

[42] T. Moscibroda, R. O'Dell, M. Wattenhofer, and R. Wattenhofer. Virtual coordinates for ad hoc and sensor networks. In *Proceedings of the 2nd ACM DIALM-POMC Joint Workshop on Foundations of Mobile Computing*, p. 8–16, 2004.

[43] S.Y. Ni, Y.C. Tseng, Y.S. Chen, and J.P. Sheu. The broadcast storm problem in a mobile, ad-hoc network. In *Proc. MOBICOM*, p. 151–162, 1999.

[44] T. Nieberg, S. Dulman, P.J.M. Havinga, L.F.W. van Hoessel, and J. Wu. Collaborative algorithms for communication in wireless sensor networks. In T. Basten, M. Geilen, and H. De Groot, editors, *Ambient Intelligence: Impact on Embedded System Design.* Kluwer Academic Publishers, 2003.

[45] T. Nieberg, P.J.M. Havinga, and J.L. Hurink. On the advantages of cluster-based routing in wireless sensor networks. In *EYES Workshop, EWSN'05*, 2005.

[46] T. Nieberg and J.L. Hurink. Wireless communication graphs. In *Proceedings of the Intelligent Sensors, Sensor Networks and Information Processing Conference*, p. 367–372, 2004.

[47] T. Nieberg and J.L. Hurink. A PTAS for for the minimum dominating set problem in unit disk graphs. In *Proceedings of the 3rd workshop on approximation and online algorithms*, p. 296–306. Springer, LNCS 3879, 2005.

[48] T. Nieberg, J.L. Hurink, and W. Kern. A robust PTAS for maximum independent sets in unit disk graphs. In *Proceedings of the 30th workshop on graph theoretic concepts in computer science*, p. 214–221. Springer, LNCS 3353, 2004.

[49] OMNeT++. discrete event simulator. `http://www.omnetpp.org`.

[50] A. Panconesi and A. Srinivasan. Improved distributed algorithms for coloring and network decomposition problems. In *Proceedings 24-th Ann. ACM Symp. on Theory of Comp. (STOC)*, p. 581–592, 1992.

[51] D. Peleg. *Distributed Computing.* SIAM Monographs on Discrete Mathematics and Applications, Vol. 5, 2000.

[52] C.E. Perkins and E.M. Royer. Ad hoc on-demand distance vector routing. In *Proc. 2nd IEEE Workshop on Mobile Computing Systems and Applications*, p. 90–100, 1990.

[53] V. Raghavan and J. Spinrad. Robust algorithms for restricted domains. *Journal of Algorithms*, 48(1), p. 160–172, 2003.

[54] P. Schuurman and G.J. Woeginger. Approximation schemes – a tutorial, 2002. preprint.

[55] R. Sivakumar, P. Sinha, and V. Barghhavan. CEDAR: A core-extraction distributed ad-hoc routing algorithm. *IEEE Journal on Selected Areas in Communication*, 17, p. 1454–1465, 1999.

[56] P. Su and R.L. Drysdale. A comparison of sequential delaunay triangulation algorithms. In *Proceedings Symposium on Computational Geometry*, p. 61–70, 1995.

[57] L.F.W. van Hoesel, T. Nieberg, H.J. Kip, and P.J.M. Havinga. Advantages of a TDMA based, energy-efficient, self-organizing MAC protocol for WSNs. In *Proceedings of IEEE VTS Spring*, p. 1598–1602, 2004.

[58] L.F.W. van Hoesel, T. Nieberg, J. Wu, and P.J.M. Havinga. Prolonging the lifetime of wireless sensor networks by cross-layer interaction. *IEEE Wireless Communication Magazine*, 11, p. 78–86, 2004.

[59] M. Weiser. Nomadic issues in ubiquitous computing, 1996. Keynote speach at Nomadic '96.

[60] W. Ye, J. Heidemann, and D. Estrin. An energy-efficient mac protocol for wireless sensor networks. In *Proceedings of the 21-st INFOCOM*, p. 1567–1576, 2002.

About the Author

Tim Nieberg was born July 22, 1976, in Hamm, Germany. After attending school in Bad Säckingen, Berlin, Oldenburg in Germany, and Tipp City, Ohio, U.S.A., he graduated with honors in 1994 from the Tippecanoe High School in Tipp City and received his German high school diploma from the Herbartgymnasium in Oldenburg in 1996.

In 1997, he started studying Mathematics with a minor in Commerce at the University of Osnabrück, Germany. During the 2000/01 academic year, he visited the University of British Columbia, Vancouver, B.C., Canada, on a Rotary Ambassadorial Scholarship. In 2002, he graduated from the University of Osnabrück with a thesis on a tabu search approach for flow-shop and job-shop problems with intermediate buffers.

Since 2002, he is with the Discrete Mathematics and Mathematical Programming group at the University of Twente, Enschede, The Netherlands. His research interests include algorithmic graph theory, especially with a relation to communication networks, and scheduling problems with further constraints. During his time in Enschede, he also worked in the European project EYES on energy-efficient sensor networks, as well as the follow-up Dutch project Smart Surroundings.

Publications

- *A PTAS for the Minimum Dominating Set Problem in Unit Disk Graphs*, with J.L. Hurink, Proc. 3rd Workshop on Approximation and Online Algorithms (WAOA'05), Palma, Spain, October 2005. Springer LNCS 3879, p. 296–306.
- *Fast Deterministic Distributed Maximal Independent Set Computation on Growth-Bounded Graphs* with F. Kuhn, T. Moscibroda, and R. Wattenhofer, 19th International Proc. Symposium on Distributed Computing (DISC'05), Krakow, Poland, September 2005. Springer LNCS 3724, p. 273–287.
- *Local Approximation Schemes for Ad Hoc and Sensor Networks*, with F. Kuhn, T. Moscibroda, and R. Wattenhofer, Proc. of 3rd ACM Joint Workshop on Foundations of Mobile Computing (DIALM-POMC'05), Cologne, August 2005, p. 97–103.
- *On the Advantages of Clusterbased Routing in Wireless Sensor Networks*, with P.J.M. Havinga, and J.L. Hurink, EYES-Workshop, 2nd European Workshop on Wireless Sensor Networks (EWSN'05), Istanbul, Turkey, January 2005.
- *Prolonging the Lifetime of Wireless Sensor Networks by Cross-Layer Interaction*, with L.F.W. van Hoesel, J. Wu, and P.J.M. Havnga, IEEE Wireless Communications, Vol. 11(6), 2004, p. 78–86.
- *Wireless Communication Graphs*, with J.L. Hurink, Proc. DEST International Workshop on Signal Processing for Sensor Networks, ISSNIP'04, Melbourne, Australia, December 2004, p. 367–372.
- *Multipath Routing with Erasure Coding for Wireless Sensor Networks*, with J. Wu, S. Dulman, and P.J.M. Havinga, Proc. ProRisc2004, Veldhoven, The Netherlands, November 2004.
- *A Robust PTAS for Maximum Independent Sets in Unit Disk Graphs*, with J.L. Hurink, and W. Kern, Proc. 30th Workshop on Graph-Theoretic Concepts in Computer Science (WG'04), Bad Honnef, Germany, June 2004. Springer LNCS 3353, p. 214–221.
- *Local, Distributed Topology Control for Large-Scale Wireless Ad-Hoc Networks*, with J.L. Hurink, Proc. International Workshop on Wireless Ad-Hoc Networks (IWWAN'04), Oulu, Finland, June 2004.

- *Communciation in the EYES Wireless Sensor Network: Tight Integration of Networking Layers Extends Lifetime*, with L.F.W. van Hoesel, J.Wu, and P.J.M. Havinga, Proc. International Workshop on Wireless Ad hoc Networks (IWWAN'04), EU cluster day, Oulu, Finland, June 2004

- *Advantages of a TDMA Based, Energy-Efficient, Self-Organizing MAC Protocol for WSNs*, with L.F.W. van Hoesel, H.J. Kip, and P.J.M. Havinga, Proc. IEEE Semiannual Vehicular Technology Conference (VTC2004-Spring), Milan, Italy, May 2005.

- *Job-Shop Scheduling with Buffers*, with S. Heitmann, and J.L. Hurnik, Proc. 9th International Workshop on Project Management and Scheduling (PMS'2004), Nancy, France, April 2004.

- *Size-Controlled Dynamic Clustering in Mobile Wireless Sensor Networks*, with P.J.M. Havinga, and J.L. Hurink, Proc. Workshop on Computer Networks and Distributed Systems (CNDS'04), San Diego, USA, January 2004.

- *A Data Aggregation Framework for Wireless Sensor Networks*, with H. Karl, and M. Loebbers, Proc. ProRisc2003, Veldhoven, The Netherlands, November 2003.

- *Distributed Algorithms for Wireless Sensor Networks*, Proc. 2nd Cologne-Twente Workshop on Graphs and Combinatorial Optimization (CTW), Enschede, The Netherlands, May 2003. Electronic Notes in Discrete Mathematics, Vol. 13, p. 81–83.

- *Collaborative Communication Protocols for Wireless Sensor Networks*, with S. Dulman, L.F.W. van Hoesel, and P.J.M. Havinga, Proc. Workshop on European Research on Middleware and Architecture for Complex and Embedded Systems, IEEE ISADS 2003, Pisa, Italy, April 2003.

- *Trade-Off Between Traffic Overhead and Reliability in Multipath Routing for Wireless Sensor Networks*, with S. Dulman, J. Wu, and P.J.M. Havinga, Proc. WCNC Workshop 2003, New Orleans, USA, March 2003.

Chapters in Books

- *Collaborative Algorithms for Communications in Wireless Sensor Networks*, with S. Dulman, P.J.M. Havinga, L.F.W. van Hoesen, and J. Wu.

In: T. Basten, M. Geilen, H. de Groot (eds.) *Ambient Intelligence: Impact on Embedded Systems Design*, Kluwer Academic Publishers, ISBN: 1-4020-7668-1, p. 271–294.

Technical Reports

- *A PTAS for the Minimum Dominating Set Problem in Unit Disk Graphs*, with J.L. Hurink, Memorandum No. 1732, Universiteit Twente, Fac. of Mathematical Sciences, 2004.
- *Job-Shop Scheduling with Limited Capacity Buffers*, with P. Brucker, S. Heitmann, and J.L. Hurink, Osnabrücker Schriften zur Mathematik, Reihe P, No. 253, 2004. To appear in *OR Spectrum.*
- *On Cyclic Plans for Scheduling a Smart Card Personalisation System*, TR-CTIT-04-01, 2004.
- *A new PTAS for Maximum Independent Sets in Unit Disk Graphs*, with J.L. Hurnik, and W. Kern, Memorandum No. 1688, Universiteit Twente, Fac. of Mathematical Sciences, 2003.
- *Collaborative Communication Protocols for Wireless Sensor Networks*, with S. Dulman, L.F.W. van Hoesel, and P.J.M. Havinga, TR-CTIT-03-44, 2003.
- *A Data Aggregation Framework for Wireless Sensor Networks*, with H. Karl, and M. Loebbers, TKN-06-016, Technische Universität Berlin, 2003.
- *Trade-Off Between Traffic Overhead and Reliability in Multipath Routing for Wireless Sensor Networks*, with S. Dulman, J. Wu, and P.J.M. Havinga, TR-CTIT-03-08, 2003.
- *Multipath Routing for Data Dissemination in Energy Efficient Sensor Networks*, with S. Dulman, P.J.M. Havinga, and P. Hartel, TR-CTIT-02-20, 2002.