Positioning and Scheduling of Wireless Sensor Networks

—

Models, Complexity, and Scalable Algorithms

zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften

von der Fakultät für Informatik
der Universität Fridericiana zu Karlsruhe (TH)
genehmigte

Dissertation

von

Bastian Katz
aus Lübeck

Tag der mündlichen Prüfung: 29. Januar 2009
Erster Gutachter: Professor Dr. Dorothea Wagner
Zweiter Gutachter: Professor Dr. Sándor Fekete

Acknowledgments

Working as a PhD student was an exciting time full of new impressions and changes. A few lines are not enough to express my sincere thanks to all the people who contributed to this experience through their guidance and their support.

First of all, I would like to thank my advisor **Dorothea Wagner** for the freedom I could work in and the confidence she placed in my work. I am especially grateful for the opportunity of working in a group shaped by her open and respectful nature and for introducing me to many interesting people that inspired my work. My thanks also go to **Uwe D. Hanebeck**, the chairman of the Research Training Group I had the honor to be part of, for his efforts to form the cooperation and the exchange of knowledge within this group. This thanks extends to all participants and supervisors not named in person who shared this vision and contributed to my work with their critique and their experience.

I had the privilege to work within a group of colleagues at the ITI Wagner that showed interest for each other's problems and shared ideas. Most of all, I am deeply indebted to **Robert Görke**, whose ever-friendly character and willingness to help often made him the first person to turn to with any problem I had, just like the dubious pleasure of proof-reading this thesis—all errors are mine! Working at the ITI wouldn't have been 10% the fun without him. I would also like to thank **Marco Gaertler** for sharing his scientific experience and cooking skills, **Steffen Mecke** as the second "sensor network guy" in the department for many hours of valuable discussions and bright ideas, **Ignaz Rutter** and **Marcus Krug** for inviting me to their discussions, sating my interest in someone else's problems, **Reinhard Bauer** for his help when I needed a real mathematician, as well as **Michael Baur**, **Daniel Delling**, **Martin Holzer**, **Sascha Meinert** and **Martin Nöllenburg**, who made my work easier and more fun in countless situations. Special thanks go to **Lilian Beckert**, **Elke Sauer** and **Bernd Giesinger**, whose readiness to help goes far beyond what one could expect, doing the 'little things' that would turn into monsters without them.

I would also like to express my thanks to **Jie Gao** and **Leonidas Guibas** for showing their interest in my work and inviting me to work in their groups in Stony Brook and Palo Alto, and to **Sándor Fekete** for readily accepting the task of reviewing my dissertation. I thank **Christian Pich** for sharing his implementation of MDS.

Finally, I wish to thank my parents, **Hertje** and **Michael Katz**, who supported me in countless ways. This thesis is dedicated to my wife **Isabelle**, who believed in me and whose patience and encouragement has upheld me.

Drahtlose Sensornetze (DSN) stellen eine vielversprechende Technologie dar, deren Möglichkeiten und Grenzen sich in vielen Bereichen gerade erst andeuten.

Sie bestehen aus einer Vielzahl kleiner Sensorknoten, vollwertiger, wenngleich leistungsarmer Kleinstrechner, die drahtlos miteinander kommunizieren und ihre Umwelt mit Hilfe zumeist einfacher Sensorik beobachten, mitunter sogar durch Aktorik beeinflussen können. Die Entwicklung solcher Sensorknoten ist die Konsequenz immer kleiner und leistungsfähiger werdender Komponenten: Hochintegrierte Mikrocontroller, Speicher und Funkchips, Sensoren für Druck, Licht, Wärme, Chemikalien usw. lassen sich je nach Leistungsfähigkeit heute schon im Bereich weniger Kubikzentimeter, zum Teil noch deutlich darunter realisieren. In ihrer Kombination eröffnen diese Technologien neue Einsatzgebiete.

In der Vision, die die Entwicklung vorantreibt, sieht man bereits Tausende von Sensorknoten zu Stückpreisen weniger Euro, die zur Beobachtung eines Phänomens ausgebracht werden. Die Knoten vernetzen sich selbstorganisiert, verabreden die dazu nötige Arbeitsteilungen untereinander und kompensieren bei Bedarf Knotenbewegungen und -ausfälle. Sie beobachten ohne weitere Steuerung ihre Umgebung, erkunden sie vielleicht sogar, klassifizieren Veränderungen kollektiv und unterscheiden kritische Ereignisse zuverlässig durch gegenseitigen Abgleich der Messungen, um sie an den Rand des Netzes zu melden.

Die Einsatzgebiete sind vielfältig: Schon heute werden Sensornetze in der Überwachung von Wildtierbeständen und der Aufzeichnung von Klimabedingungen zum Beispiel in der Agrarwirtschaft genutzt. Sie werden zur Überwachung von Gletschern und Vulkanen eingesetzt, in der Hoffnung, in Zukunft frühzeitig vor Gefahren warnen zu können. Sensornetze überprüfen die Lagerung von Chemikalien und sollen die Stabilität von Bauwerken überwachen. Am Körper angebrachte und funkvernetzte Sensoren sollen Patienten ermöglichen, sich frei zu bewegen.

Erste Einsätze haben jedoch auch gezeigt, wie weit wir von der beschriebenen Vision noch entfernt sind. Dafür sind vor allem die beiden folgenden Eigenschaften solcher Netze verantwortlich:

- In aller Regel sind Sensorknoten batteriebetrieben; hier ist die Entwicklung längst nicht so schnell vorangeschritten wie in anderen Bereichen. Das zur Verfügung stehende Energiebudget ist daher oft der die (autarke) Lebenszeit eines Sensornetzes begrenzende Faktor.
- Die Kommunikation im drahtlosen Medium unterscheidet Sensornetze deutlich von anderen verteilten Systemen und die große Anzahl an Knoten auch von anderen Ad-hoc-Netzen. Kommunikation hängt in diesen Netzen stark von der räumlichen Verteilung ab; Knoten können nur mit nahen Knoten Informationen austauschen, gleichzeitig stören sich Übertragungen mit geringem Abstand gegenseitig. Kommunikation, vor allem über größere Strecken, kostet viel Energie und bleibt unzuverlässig.

Aus diesen Eigenschaften ergeben sich neuartige Probleme, aber auch neuartige Ansätze.

Die Geometrie solcher Netze lässt sich, sofern Knoten ihre Position kennen, nutzen, um zum Beispiel Übertragungen so zu planen, dass sie sich gegenseitig nicht stören. Auf der anderen Seite lässt sich aus der Beobachtung, welche Knoten mit welchen Knoten kommunizieren können, viel Information über die Knotenpositionen ableiten.

Diese Arbeit behandelt diese beiden Aspekte aus algorithmischer Sicht: Sie stellt mathematische Modelle vor, die die Eigenschaften drahtloser Kommunikation und eingeschränkter Sensorknoten abbilden, betrachtet die Komplexität der entstehenden Probleme und zeigt, welche Lösungen sich auch in großen Netzen realisieren lassen. Kern dieser Lösungen ist die Beschränkung auf lokale Kommunikationsmuster.

Lokalisierung großer Netze

Viele Algorithmen für Sensornetze setzen die Kenntnis der Knotenpositionen voraus. Sie ist auch die Voraussetzung für die Analyse gemessener Daten oder das Melden von Ereignissen.

Trotzdem ist nicht selbstverständlich, dass Knoten ihre eigene Position ermitteln können. Nicht immer stehen GPS oder ähnlich mächtige Hilfen zur Lokalisierung zur Verfügung. Auf der anderen Seite enthält die Struktur des Netzes wegen des starken Zusammenhangs zwischen Kommunikation und Geometrie bereits viel Information über die relative Lage der Knoten. Diese Arbeit stellt einige Ergebnisse zur Lokalisierung großer Netze gänzlich ohne sogenannte *Ankerknoten* mit bekannten Knotenpositionen vor. Die wesentlichen Ergebnisse sind:

- Beweis der NP-Schwere der Lokalisierung bei bekannten paarweisen Entfernungen *und* relativen Richtungen bei beliebig kleiner Störung der Eingabe
- Entwurf eines exakten Verfahrens zur richtungsbasierten Lokalisierung, das durch verteilte Berechnung einer Zerlegung des Netzes in starre Teilstrukturen (Laman-Partitionierung) die Kommunikation und den Berechnungsaufwand zur Identifikation und Einbettung von maximalen Teilgraphen mit eindeutiger Lokalisierung minimiert.
- Entwurf und Analyse eines hierarchischen Lokalisierungsverfahrens auf Basis der Multidimensionalen Skalierung (MDS). Dieses Verfahren vereinbart die Stärken von MDS, das sich als extrem robustes Lokalisierungsverfahren für kleine Netze bewährt hat, mit Techniken zur hierarchischen Ausdünnung großer Netze. Das resultierende Verfahren erlaubt so, eine äußerst präzise Lokalisierung auch großer Netze verteilt zu berechnen, ohne dass einzelne Knoten dabei die Struktur des gesamten Netzes kennen müssen.

Koordination von Abfragebäumen

Das vorherrschende Kommunikationsmuster in Sensornetzen besteht nach wie vor aus dem Einsammeln von Daten entlang eines Anfragebaumes zu einer Senke hin. Gerade bei so berechenbarer, aber energieaufwendiger Kommunikation lohnt sich das zeitliche Abstimmen der Übertragungen, um Energieverluste durch Kollisionen, unnützes Mithören des Funkkanals oder durch das Überlaufen von Puffern zu verhindern. Auf der anderen Seite

darf der Mehraufwand zur Abstimmung der Benutzung des Funkmediums natürlich selbst nicht ins Gewicht fallen, wenn insgesamt Energie eingespart werden soll. Wir betrachten daher die erreichbare Qualität von Protokollen zur zeitlichen Koordination von Übertragungen in diesem Szenario, die einem sehr strengen Modell genügen: Jeder Knoten darf mit jedem Nachbarknoten im Baum nur ein – in der Größe beschränktes – Paket austauschen. Damit kennt insbesondere keiner der Knoten die Gesamtstruktur des Anfragebaumes. Das Ergebnis einer solchen Absprache soll in jedem Fall eine Nutzung des Funkmediums sein, die den Transport der Daten mit minimalem Energie- und geringem Zeitaufwand ermöglicht und von keinem Knoten verlangt, mehr als eine konstante Anzahl Pakete zu puffern.

Analysiert werden verschiedene (kombinatorische) Interferenzmodelle, die Hauptergebnisse sind:

- Vereinbarkeit der Einschränkungen und der Verabredung zeitoptimaler Nutzung des Kanals ohne Interferenz
- Analyse verschiedener Protokolle zur Verabredung zeitapproximativer Nutzung des Kanals bei totaler Interferenz (keine parallelen Übertragungen) und zeitoptimaler Nutzung bei geringfügig erhöhtem Mehraufwand, sowie die Unmöglichkeit der Verabredung zeitoptimaler Nutzung unter Einschränkungen
- Protokolle zur Verabredung zeitapproximativer Nutzung des Kanals mit konstantem Approximationsfaktor bei lokaler Interferenz
- Entwurf und Analyse eines generischen Protokolls, das die vorgestellten Protokolle so robust gegen Übertragungsfehler macht, dass ein vollständiges Einsammeln der Daten garantiert werden kann, ohne die Dauer des Einsammelns um mehr als einen konstanten Faktor zu verlangsamen

Koordination allgemeiner Verbindungen

Bei der Koordination von Übertragungen in allgemeineren Szenarien bauen algorithmische Lösungen bisher auf geometrischen und graphenbasierten Interferenzmodellen auf. Diese modellieren Interferenz als binäre Relation sich gegenseitig ausschließender Übertragungen. Sie ermöglichen es somit, Probleme in Sensornetzen auf bekannte Probleme der Graphentheorie zu reduzieren. Diese Ansätze blieben aber eine Antwort auf die Frage schuldig, unter welchen Umständen und Verlusten sich die so gewonnenen Lösungen in realistischere Interferenzmodelle übertragen lassen. In Modellen, wie sie in der Simulation von Sensornetzen üblich sind, wird der Signalausbreitung und -überlagerung viel stärker Rechnung getragen und der Empfang einer Übertragung ist genau dann möglich, wenn das Nutzsignal sich deutlich genug von störenden Signalen unterscheidet. Diese Arbeit liefert die Nachweise, dass die Lösungen *bestimmter* graphenbasierter Interferenzmodelle in komplexeren Modellen zum einen immer noch korrekt sind, zum anderen selbst gierige Lösungen in solchen graphenbasierten Modellen auch nur um kleine, konstante Faktoren schlechter als optimale Lösungen in realistischen Interferenzmodellen sind.

Chapter 1

Introduction

Wireless sensor networks (WSN) have attracted an enormous attention among researchers from diverse scientific communities. In a unique way, wireless sensor networks offer a completely new technology, opening the door for new kinds of applications and services, but at the same time demand for new ideas and paradigms.

It is, however, not the development of single components that inspire the enthusiasm. Sensor network technology is nothing more than the result of the expected development in the integration of sensors, microcontrollers and radio transceivers. In its combination, or, better, in the prospect to ship devices that combine these components in large numbers at low costs and small form factors, this development breaks a barrier. It was this outlook that gave the impulse to envision networks of thousands of tiny devices—so-called *motes*—that, once deployed in an area of interest, would work as a single, omnipresent sensor, and, moreover, as a tool that could analyze complex phenomena, providing the owner with essential data and alerts. In this vision, nodes are dropped from a plane in unsafe or inaccessible territories, attached to animals, buildings, or dispersed as "Smart Dust" [KKP99]. The nodes themselves would then initiate their organization, choosing nodes and routes for communication and duty cycles, adapting to the environment, exploring it, and ultimately monitor it. As a network, nodes would answer complex queries for conditions and matters, and would watch for events, combining information from different types of spatially close sensors to distinguish between real events and sensor failures. Among the scenarios that illustrate this vision best, we have the oft-quoted forest fire, that can be discovered earlier by sensors equipped with temperature sensors and smoke detectors, but also smart storage containers that cooperatively monitor storage conditions, e. g., with regard to regulations concerning chemicals that must not be stocked in dangerous combinations.

It has not only been for the new applications that sensor networks became such an enthralling field of research. It was also the challenge to solve the puzzling problems that such a novel platform confronted many disciplines with, far beyond the design of the hardware.

Sensor networks differ from all the parallel or distributed networks and models of computation. They are assumed to grow larger, sometimes are mobile and consist of nodes of very restricted capacities. Among these restrictions, we have the following:

Limited energy budget: A typical mote runs on batteries, whose development did not advance as far as the integration of the other components. But even if the plans to let future motes "harvest" their energy using solar cells, from temperature differences or other processes, energy will remain a very scarce resource. For the vision of a sensor network that runs autonomously for months or years, operating 24/7, it is hence vital to save energy in every aspect, minimizing communication, computation and duty times.

Limited memory and computational power: Today's motes are fully operational systems consisting of a microprocessor and often both, volatile and non-volatile memory. Their capacities, however, are far from what we have nowadays even in other embedded systems. Size, price, and, most importantly, the need for low-energy components put hard constraints on the components.

These restrictions are but one reason why sensor networks are so novel to designers of algorithms and protocols. They mix with aspects that cannot only be seen as limitations, but also as potentials of sensor networks:

Geometry: Much more than any other distributed system, wireless sensor networks are geometric networks. Nodes have positions in either the plane or three-dimensional space. The data they are processing is closely related to this position and also the interconnection between the nodes highly depends on the nodes' embedding: Nodes can only communicate with other nodes directly that are spatially close.

Redundancy: The high number of low-cost motes, deployed in a possibly hostile environment, is an essential part of the vision of sensor networks. It entails a low reliability of single nodes, of which some can be substandard, and others can be destroyed or blocked. Similarly, communication between nodes can be unreliable. Nodes can lose connection due to objects moving in the way or other changes in the environment.

A wonderful example of how geometry and redundancy can be exploited to compensate limited memory and changing network topology are geographic hashing techniques that assign data not to single nodes, but to geographic positions, such that nodes close to this position are responsible for the data. This technique has been algorithmically analyzed in the context of location services, where nodes regularly send their current position to a set of rendezvous points, which other nodes can then query [ADM04, FW06]. There is a variety of such examples of so-called "geometry-aware" algorithms for sensor networks.

Another development that shows how severe the differences to known platforms really are, is the regress to application-specific cross-layer optimization. Unlike for other networks, an abstraction of network protocol stacks like the established OSI model does not seem to be suitable anymore. Competitive solutions for sensor networks seemingly need to address problems across the traditional layers. This is, to a considerable part, due to the fact that

concurrent communication is subject to much more restrictions and that optimizing the network's structure and timing often negatively affects the behavior of distributed algorithms running in this network. The downside of this trend towards cross-layer optimization and tailored protocol stacks clearly is that we are almost completely missing an agreed computational model to design algorithms for. Still, it is unclear, whether we can hope for such a model, or whether we will still see a variety of tailored protocol stacks for different purposes.

Some of the visions above have been disenchanted in the light of these conditions. Today's sensor networks are still far from these visions and do not only suffer from teething problems; some issues remained hard from both, the practical and theoretical point of view, and some prerequisites for many applications such as synchronization, positioning and reliable multi-hop routing are not understood well. On the other hand, first networks have been deployed and provide valuable service, among them systems to monitor critical conditions and raise alarm on hazardous events and such that protocol environmental parameters in agriculture or research of ecosystems. Typically, present systems have in common that they build on simple network traffic patterns, such as flooding and routing trees, do not involve complex in-network processing beyond data aggregation and often did not grow too large.

Our contribution

In this work, we will focus on two problems that are fundamental for higher-level algorithms, but have not yet been answered satisfactory, namely positioning and scheduling. Both problems are among the first to arise in newly deployed networks, and we are interested in appropriate models, complexity and, most of all, algorithms that solve these inherently global problems using low and local communication, making them applicable also to the large networks envisioned for the future.

In positioning, the question is how nodes that are tiny and have to observe spatial phenomena and communicate in a geometric network obtain information about their positions. This question is often answered by an external infrastructure such as GPS or other anchorage, but there have always been efforts to design algorithms for autonomous positioning, that fits much better into the picture of self-organization and does not rely on infrastructure that is not available in all environments. GPS, for example, has become cheap enough for integration, but does not provide accurate positions and does not work well indoors, in densely wooded forests etc. This class of *anchor-free* positioning problems has always been about the contrast between computationally hard problems and the development of heuristics that exploit the geometric information in networks that are dense enough. We will contribute to both these facets by a complexity result and the design of algorithms for positioning problems.

Scheduling is the problem to coordinate transmissions in sensor networks. Such a coordination can save time and energy in many ways: First, nodes that know when to communicate do not have to listen to the radio during idle times, second, scheduled communication prevents collisions of transmissions that make retransmissions necessary, which in turn cost time and energy. We contribute to the problem in two ways: First, we approach the question how to distributedly set up schedules for a given transport demand typical to many sensor network applications with minimum communication. In addition, we target the modelling

aspects of interference in sensor networks, following the questions how to make the complex effects of interference algorithmically tractable, without the oversimplifications that has been the basis for so much research in the past.

Organization of this work

In the following chapter, we will provide some background on sensor networks, including a short overview on the hardware and protocols that currently mark the state-of-the art. We also give a short synopsis of the mathematical concepts we will employ in the later chapters. Last, this chapter contains some standard models currently used to make sensor networks approachable for algorithmic design and reasoning. Neither of these three parts will be comprehensive, but hopefully provides a sufficient basis for the later chapters.

In Chapter 3, we will present first results on positioning problems dealing with the problem to recover node positions solely from local information, with a strong focus on the complexity and correspondence to mathematical structures. The main results are a fast and partly distributed direction-based positioning algorithm and a hardness proof for the most optimistic input we could hope for in reality.

In Chapter 4, we will approach the positioning problem in a more pragmatical way. Here, we propose a multi-level technique to hierarchically combine local solutions in a purely distributed fashion. This framework, teamed with the most promising positioning method for small networks, multidimensional scaling, outperforms existing positioning algorithms while providing admissible worst-case message and time complexities and robustness to irregular radio patterns.

In Chapter 5, we turn our attention to a scheduling problem that is very special to sensor networks: It deals with the problem to set up schedules for bulk data transport in query trees with a minimum of communication overhead to gain both, time and energy efficiency. This chapter provides ready-to-use protocols for applications gathering data at a central repository using slotted communication for different interference models, also accounting for unreliable communication.

Chapter 6 covers modelling aspects of scheduling problems in a broader sense. It compares widely used interference models extending the taxonomy of interference models by the class of local interference models, proving that locality is the feature that—asymptotically—causes the gap in the quality of common graph-based interference models and the more realistic, yet complex SINR model.

The work is concluded with Chapter 7.

Part of this work has been published in [KGW07, KW07, KW08, KMW08, KVW08]. The work presented in Chapters 5 and 6 is joint work with Steffen Mecke and Markus Völker, respectively.

Chapter 2

Basics

This chapter deals with fundamental concepts and the technological background that is assumed in the later chapters. First, this includes current sensor network technology, considering hardware, wireless communication and areas of application. Second, we give an overview on mathematical concepts and notation, and third, we will motivate and describe some standard models used in algorithmic research for wireless sensor networks.

2.1 Technology and Protocols

Sensor networks have been developed for some ten years, counting from the SmartDust project [KKP99]. There is no final answer for many parameters that describe wireless sensor networks ultimately, neither in the number of nodes nor in their power, the communication standards or in the way, sensor nodes are programmed or the applications they run. It is reasonable to assume that we will see progress in all of these fields in the coming decade. Nevertheless, we will have a short look at current technology to get a feeling of how sensor networks developed until now.

2.1.1 Current Hardware

There is a variety of existing sensor node platforms, developed in both, academia and commercial enterprises. They typically integrate the following hardware:

Sensors: The foremost task of sensor networks is to monitor conditions with some kind of sensor hardware. Typical sensors that are integrated in many sensor node platforms include temperature, light, pressure and humidity. Some also have acceleration sensors on

board, and almost all platforms can be extended by arbitrary sensor hardware that is not too energy-consuming. Nevertheless, nodes do not necessarily need to be equipped with sensors. To track objects having nodes attached, for example, it is sufficient to "sense" the moving nodes' communication. And also in other applications, nodes without sensors can still serve as relays and provide cheap network infrastructure for a small set of actively sensing nodes.

Processor and memory: Sensor motes are fully operational computation devices, and there is a variety of microcontrollers that is integrated into sensor nodes, ranging from minimal CPUs limited to clock speeds of some MHz to very powerful CPUs with clock speeds of some hundred MHz. Most processors in use support different modes of operation, including low-energy idle and sleep modes. The need for low-power states also led to the integration of an additional wake-up processor, e. g., in the current SunSPOT [Smi07]. As with processors, also memory equipage differs from platform to platform. In general, memory ranges from no more than 3kB to 512kB of volatile memory. Again, limitations are mainly due to the energy drain that comes with larger memory. Some platforms also have additional flash memory for long-time storage of data, typically of larger sizes of up to 32MB.

Communication: Obviously, wireless communication is essential to WSN, and by definition, motes are equipped with a radio subsystem to communicate with each other. The most widespread protocol stacks base on the IEEE 802.15.4 standard for low-rate wireless personal area networks, such as ZigBee or WirelessHART [GCB03, SHM+08]. They mostly operate in the 2.4GHz frequency band. Some networks with high communication demand and less restrictive power budgets also use Wi-Fi standards from the IEEE 802.11 family, operating in the same frequency band. In the future, we are likely to see more customized protocol stacks for wireless sensor networks, answering the demands of sensor networks better.

Energy supply: The idea of autonomous and wireless sensor node does imply an energy supply to be attached to each sensor node, sufficient to power the mote for a reasonable lifetime, or, preferably, as long as the node is running. Currently, the only energy source that can provide enough power for sensor nodes with an acceptable form factor are batteries or other kinds of energy reservoirs such as micro fuel cells. They typically account for the better part of sensor nodes' weight and volume and still do not suffice to power a node for long-time service. There have been many efforts to find alternative power sources and to use energy efficiently. But even with the most recent platforms and energy management, all of today's motes need to be collected and recharged (or have their batteries changed) periodically.

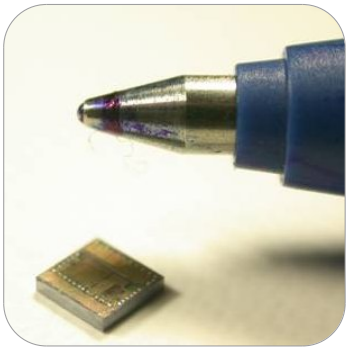
Wireless sensor networks do not necessarily have to be composed of a singly type of motes. Rather, many predicted scenarios assume inhomogeneous networks that mix different kinds of motes with different strengths. A sample of very recent sensor mote platforms is depicted in Figure 2.1. This selection is not representative for the hardware currently in use, but for the trends in hardware development: Unlike any other platform, Sun's SunSPOT builds on a Java virtual machine directly running on a 180 MHz 32 bit ARM920T proces-


www.snm.ethz.ch/Projects/SunSPOT

(a) SunSPOT


www.xbow.com/Products

(b) Imote2


www.jlhlabs.com/jhill_cs/spec/

(c) Spec mote (prototype)

Figure 2.1: *Sensor node platforms*

sor (Figure 2.1(a)). The Crossbow Imote2 currently is one of the most powerful platforms with an Intel XScale processor running with up to 416 MHz, and up to 32MB flash (Figure 2.1(b)) [NHS+08]. The Spec mote is an example of the efforts towards high integration. While both other platforms consist of standard components, not really optimizing the form factor, chips like the Spec mote integrate almost all necessary functions on a single chip (Figure 2.1(c)). Note that the SunSPOT is the only platform that is depicted with batteries, which usually account for the better part of a sensor node's volume. A much longer list of wireless sensor platforms and their features can be found at the sensor network museum[1].

2.1.2 Wireless Communication

Sensor motes are not the first to use wireless channels for communication, but the expected high number of nodes, the limited energy and the combination of missing infrastructure, spatial spread and low radio ranges make communication in wireless sensor networks unique. Generally, in wireless communication, if a node transmits a message, the signal somehow decays with the distance from the sender very irregularly, influenced by obstacles, reflections and many other factors. Another node receives the signal and can decode it, roughly spoken, if it hears the signal clear enough, i. e., if the signal is sufficiently strong and if there is not too much noise from other sources, such as third nodes transmitting. This imposes consequences that are unknown to wired networks, such as the fact that a node can reach different sets of nodes depending on the transmission power, but at the same time disturbs transmissions between more distant pairs of nodes. For a more general overview on the principles of wireless communication see [Rap96].

The latter effect also marks a new quality compared to other wireless networks, which primarily try to avoid multiple nodes sending at the same time rather than putting effort in the spacial separation of concurrent transmissions. In wireless sensor networks, the large number of nodes alone makes concurrent transmissions in many situations necessary.

[1] `http://www.snm.ethz.ch`

Wireless communication is a dominant factor in energy consumption. Not only are transmissions expensive in terms of the energy spent, but on most platforms, receiving and decoding messages also consumes energy in the same order of magnitude. Moreover, even listening to the wireless channel, waiting for a transmission, is not much cheaper, and hence, the most energy can be saved by switching the radio off completely for as much time as possible.

All communication protocols specify the rules for *medium access control (MAC)*, i. e., rules that aim at a coordination of transmissions, first to avoid too many collisions and second to achieve a good channel utilization. There is a whole zoo of MAC protocols for wireless networks (see [DEA06] for a survey), but the most important decision is between contention-based protocols and schedule-based protocols. There also are many protocols of a hybrid type, but in their pure form, these two types can be characterized as follows [LH05]:

Contention-based: Contention-based protocols allow to share a channel between many communicating entities without pre-coordination. In sensor networks, contention-based protocols usually implement some kind of CSMA/CA-based approach. CSMA, *carrier sense multiple access*, stands for the idea of "listening before talking", i. e., to wait for an idle channel prior to a transmission start. CA, *collision avoidance*, are techniques that replace the *collision detection* (CD) in wired networks, since listening to the channel cannot necessarily detect other senders interfering at the receiver. Nevertheless, CSMA-based protocols are very flexible, but can never completely avoid collisions.

Schedule-based: In schedule-based protocols, the access to the wireless channel, i. e., the allowance to transmit, is regulated by a schedule, either managed by a central instance or subject to negotiation between the nodes. This adds protocol overhead and needs some sort of synchronization, and is hence rarely used in current sensor networks. On the other hand, scheduled communication is much more energy-efficient and has successfully been used in low-energy networks [RKM+06]. We will sometimes refer to schedule-based protocols as *time division multiple access (TDMA)* protocols. This refers to the usual technique to use synchronization and slotted time for scheduling messages. Nevertheless, schedule-based protocols can also schedule the usage of different wireless channels.

Another big difference to other wireless networks is the importance of multi-hop communication. Networks grow far beyond the ranges of single nodes and messages need to be relayed by many nodes to travel through the network.

2.1.3 Applications

Monitoring goods and structures: As mentioned in the introduction, wireless motes have been used to monitor the storage of goods [KDD04], e. g., supervising that chemicals are not stored in critical combinations. In logistics, WSN can also be used to aid the management of transport processes [EBMP+05].

Sensor nodes can be used to monitor the structural integrity of buildings and bridges, ships and airplanes, e. g., detecting anomalies in the response to excitations such as vibrations from passing vehicles or wireless actuators [CFP+06, CDB+04]. Sensor networks deployed

at volcanoes and in glaciers are used to monitor and analyze the processes in order to develop early warning systems [WASW06, MPR+05]. Similarly, sensor networks can help to detect landslides [TAMW06].

People Surveillance: From the very beginning, sensor networks were associated with surveillance tasks. Typical examples are sensor networks for intrusion detection and tracking of moving targets [HVY+06]. Sensor networks have also been proposed for localizing snipers [SMAL+04]. Tracking applications can also be used in civil scenarios, e. g., to track patients in hospitals, monitoring critical parameters [SCL+05].

Monitoring flora, fauna and environment: Many academic projects uses sensor networks to monitor animals, e. g., for gathering data about the movements of wild zebras in the ZebraNet project in Kenya, bird habitats on Great Duck Island, and bats in Australia [Mar06, MPS+02, SMP+04, PEC06]. They have also been proposed for the study of animal and human social behavior, e. g., for the interaction of children in a "smart kindergarten" as in [CMY+02], other groups of humans [LGA+06] or cattle [SBH07]. Similarly, sensor networks have been deployed to monitor conditions and water use efficiency in agriculture [Bag05, MMG+08], to record meteorologic and hydrologic conditions in national parks, in forests, or in coral reefs [LCD03, BRY+04, BOP07]. In urban areas, sensor networks are used to collect data on noise pollution [SOV08].

2.2 Mathematical Foundations

This section introduces mathematical notations and concepts used in the later chapters. We assume the reader to be familiar with basic mathematical concepts. Hence, this section is not intended to comprehensively cover all mathematical aspects, but to avoid confusion about notations and to recapitulate and reference important results.

We will use $\mathbb{Z}$, $\mathbb{N}$, $\mathbb{N}_0$, $\mathbb{R}$, $\mathbb{R}_+$, $\mathbb{R}_-$ in the usual sense to denote the set of integers, positive integers, non-negative integers, real numbers, positive real numbers, or negative real numbers. For any point $x \in \mathbb{R}^d$, $\|x\|$ denotes the Euclidean norm, and the Euclidean distance between two points $x, y \in \mathbb{R}^d$ is denoted by $d(x,y) := \|x-y\|$. We will sometimes shortcut it by d_{xy}. For $x \in \mathbb{R}^d$ and $r \in \mathbb{R}_+$, we will use $B_r(x) := \{y \in \mathbb{R}^d \mid d(x,y) \leq r\}$ to denote a ball of radius r around x. To avoid confusion, we denote the zero in $\mathbb{R}^d$ by $\mathbf{0}$. The kernel of a matrix $A \in \mathbb{R}^{m \times n}$ is $\ker A := \{x \in \mathbb{R}^n : Ax = \mathbf{0}\}$

For any set S and $x \in S$, we sometimes shortcut $S - x := S \setminus \{x\}$.

2.2.1 Graphs and Networks

We also follow the usual definitions of (directed) *graphs* $G = (V, E)$ consisting of a set of *vertices* or *nodes* V and unordered (ordered) pairs of vertices E, called *edges* (*arcs*). Where convenient, we will also denote an arc in a directed graph as $\{u, v\}$, discarding orientation, or denote a connected pair in an undirected graph as (u, v), assigning an arbitrary orien-

tation. Whenever a graph is used to model connectivity in a sensor network, we will also refer to the edges as *connections*. A (directed) *path* of length l in a graph (V,E) is a sequence $v_1,\dots,v_l$ of nodes such that $\{v_{i-1},v_i\}\in E$ ($(v_{i-1},v_i)\in E$) for $1<i\leq l$. The *graph distance* between two nodes is the length of the shortest undirected path between them, denoted by $d_G(u,v)$. The *diameter* of a graph $G=(V,E)$ is defined as $\mathrm{diam}_G := \max_{u,v\in V} d_G(u,v)$. The k-hop *neighborhood* $N_G^k(v)$ denote the set of nodes u with $d_G(u,v)\leq k$. We write $N_G(v) := N_G^1(v)-v$ and omit the index if it is clear from the context. A set of nodes $V'\subset V$ is called *independent*, if for any two nodes $u,v\in V'$, $\{u,v\}\notin E$. It is called *dominating*, if for every $v\in V$ at least one node within v's one-hop neighborhood is in V', i. e., if $V'\cap N_G^1(v)\neq\emptyset$. A node's *degree* is denoted by $\delta_G(u) := |N_G(v)|$, the maximum degree of a graph $G=(V,E)$ by $\Delta_G := \max_{v\in V}\delta_G(v)$. For a graph $G=(V,E)$, *edge-* and *node-induced subgraphs* are defined as $G[V'] := (V',E[V'])$ for $V'\subseteq V$ and $G[E'] := (V[E],E')$ for $E'\subseteq E$ with $E[V'] := \{\{u,v\}\in E : u,v\in V'\}$ and $V[E'] := \bigcup_{\{u,v\}\in E'}\{u,v\}$.

For some $d>0$, an *embedding* of a set of nodes V is a function $\mathbf{p}: V\to\mathbb{R}^d$, assigning to every node a position. In $\mathbb{R}^2$, an *orientation* in the plane is a function $\mathbf{o}: V\to[0,2\pi)$. When the embedding is clear from the context, e. g., if V is a set of sensor nodes and $\mathbf{p}$ the nodes' positions, we will sometimes identify a node with its position and, for example, write $d(v,u)$ instead of $d(\mathbf{p}(v),\mathbf{p}(u))$.

A *rooted tree* is a directed graph $T=(V,E)$ with a *root* $r\in V$ such that $|E|=|V|-1$ and every node has a directed path to r. We then denote by $h_T(v) := d_T(v,r)$ the *height* of a node v, by $h_T := \max_{v\in V} h_T(v)$ the *height of* T and call the set of nodes with $h_T(\cdot)=h'$ the nodes at *level* h'. Wit respect to a rooted tree T, a node v's *descendants*, or $D_T(v)$ are the nodes that have a directed path to v in T (including v), and v's *ancestors* $A_T(v)$ are the nodes that v has a directed path to. A node v's *children* are $C_T(v) := \{u\in V : (u,v)\in E\}$, its *parent* $p_T(v)$ the single node u with $v\in C_T(u)$. For any rooted tree, we assume that any node v's children are ordered as $C_T(v)=\{c_1,\dots,c_{|C_T(v)|}\}$. With respect to this order of children, we use the terms *pre-* and *postorder* of a rooted tree's nodes as usual. For a more elaborate introduction to graph theory, see [Jun08].

We assume the reader to be familiar with network flow problems. A survey ca be found in [AMO93]. In this work, we only consider a very special case of a flow problem for *bipartite* networks (V_1,V_2,E,κ,b) where $E\subset V_1\times V_2$ and we are given *capacities* $\kappa: E\to\mathbb{N}_0$ and *demands* $b: V_1\cup V_2\to\mathbb{N}_0$. A *flow* then is a function $f: E\to\mathbb{N}_0$ such that

$$\begin{array}{ll} \forall (u,v)\in E: & 0\leq f(u,v)\leq \kappa(u,v) \\ \forall u\in V_1: & \sum_{(u,v)\in E} f(u,v)\leq b(u) \\ \forall v\in V_2: & \sum_{(u,v)\in E} f(u,v)\leq b(v) \end{array} \tag{2.1}$$

A *maximum flow* then is a flow maximizing $|f| := \sum_{e\in E} f(e)$. The problem of maximizing a flow in such a network can easily be reduced to the standard max-flow problem. With respect to some flow f, we denote the *residual demand* as

$$b_f(v) := \begin{cases} b(v)-\sum_{(v,u)\in E} f(v,u) & \text{if } v\in V_1 \\ b(v)-\sum_{(u,v)\in E} f(u,v) & \text{if } v\in V_2 \end{cases} \tag{2.2}$$

2.2.2 Algorithms & Complexity

In the field of algorithms and complexity, we follow the definitions of [GJ79]. We will refer to a *decision problem* Π as a set of possible *instances* $I \in \Pi$ classified as having answers "yes" or "no". We will assume that instances are encoded using some finite alphabet implicit to Π, and refer to the number of symbols necessary to encode an instance I as I's size, $\langle I \rangle$. An algorithm $\mathcal{A}$ solves Π if it finds the correct answer for every $I \in \Pi$.

In *optimization problems*, every instance I has feasible values $F(I) \subset \mathbb{R}$, and an algorithm $\mathcal{A}$ *solves* Π if it returns $\mathrm{OPT}(I) := \max_{x \in F(I)} x$, if in $F(I)$ a maximum exists, "infeasible" if $F(I)$ is empty, or "unbounded" if $F(I)$ is either unbounded or $\limsup(F(I)) \notin F(I)$. Every optimization problem can be translated into a decision problem extending instances to contain a parameter p_I. An Instance (I, p_I) then is a "yes" instance if and only if the answer to the optimization problem for I was some $x \geq p_I$.

For the asymptotic analysis of the runtime of algorithms, we use the common Landau notation O, Ω, Θ to denote sets of functions with some upper or lower bounded asymptotic behavior: For $f, g : \mathbb{N} \to \mathbb{R}$, $f \in O(g)$ if and only if there exist $c \in \mathbb{R}$ and $n_0 \in \mathbb{N}$ such that $f(n) \leq cg(n)$ for all $n \geq n_0$, and $f \in \Omega(g)$ if and only if there exist $c \in \mathbb{R}_+$ and $n_0 \in \mathbb{N}$ such that $f(n) \geq cg(n)$ for all $n \geq n_0$. For $g : \mathbb{N} \to \mathbb{R}$, $\Theta(g) = O(g) \cap \Omega(g)$. We say a function f is *polynomially bounded*, if for some $k \in \mathbb{N}$, $f(n) \in O(n^k)$ and we say that f *grows exponentially*, if $f(n) \in \Omega(k^n)$ for some $k > 1$.

Landau notation allows us not to be too strict in defining the model of computation algorithms run on, but unless stated otherwise, we assume some kind of RAM. To abstract even further, we say a decision problem is in class P, if for some polynomially bounded f there exists an algorithm $\mathcal{A}$ such that a deterministic Turing machine running $\mathcal{A}$ solves Π using at most $f(\langle I \rangle)$ steps. It is in class NP, if the same holds for a indeterministic Turing machine. A problem Π is said to reducible to a problem Π' in polynomial time, if there is a polynomial-time algorithm to compute for every instance $I \in \Pi$ an instance $I' \in \Pi'$ with the same answer. It is still unknown whether the obvious inclusion $\mathrm{P} \subseteq \mathrm{NP}$ is strict or not. As formalization of the believed higher complexity allowed in NP, a problem Π is said to be NP-hard if every problem in NP is reducible to Π in polynomial time. Not saying that $\mathrm{P} \neq \mathrm{NP}$, this notion of NP-hardness covers all problems that are as hard as any other problem in NP. Under the assumption that $\mathrm{P} \neq \mathrm{NP}$, proving a problem Π to be NP-hard by a polynomial-time reduction of any known NP-hard problem Π' to Π is equivalent to proving that no polynomial-time algorithm exists solving Π. On the other hand, proving a problem $\Pi \in \mathrm{P}$ to be NP-hard would prove $\mathrm{P} = \mathrm{NP}$.

Such a proof by reduction obviously presupposes at least one problem to be proven to be NP-hard by some other technique. The earliest example for such a proof is the proof that Sat, the problem to answer whether a boolean formula using only AND, OR, NOT, variables and parentheses has a variable assignment, such that the whole expression is true. Nowadays, there is a huge number of known NP-hard problems to reduce from, one of the most famous being the 3-Sat problem.

2.2.3 Probability Theory

We will use some terms from probability theory and denote the probability of some event E with $\mathbb{P}[E]$, and the expectation value of some random variable X with $\mathbb{E}[X]$. We will denote the uniform distribution on $[a,b]$ by $U(a,b)$ and the normal distribution with mean μ and variance σ^2 by $N(\mu,\sigma^2)$. Aside from these standard notations, we will refer to *Markov chains* as sequences of random variables $X_1, X_2, \ldots$ with the *Markov property* that

$$\mathbb{P}\left[X_{i+1} = x \mid X_1 = x_1, \ldots, X_n = x_n\right] = \mathbb{P}\left[X_{i+1} = x \mid X_i = x_i\right] \quad . \tag{2.3}$$

Markov chains are used to model stochastic processes where the distribution of possible states at time t only depends on the preceding state (and not on the whole history). The Markov chains we are considering have a finite state space S and fulfill the requirements to have a *stationary distribution*, i. e., probabilities $\pi : S \to [0,1]$ with

$$\pi(s) = \sum_{s' \in S} \left(\pi(s') \cdot \mathbb{P}\left[X_{i+1} = s \mid X_i = s'\right]\right) \quad . \tag{2.4}$$

We will not prove these properties explicitly.

Another result we are going to use is that for n independent random binary variables $X_1, \ldots X_n$, each having $\mathbb{P}\left[X_i = 1\right] = p_i$ for some p_i, the random variable $X = \sum_{i=1}^{n} X_i$ can be bounded using Chernoff's inequality ([HR90]) for any δ by

$$\mathbb{P}[X \leq (1+\delta)\mathbb{E}[X]] < \exp\left(-\mathbb{E}[X]\,\delta^2/2\right) \quad . \tag{2.5}$$

In the special case that $\mathbb{P}\left[X_i = 1\right] = p$ for a fixed p, i. e., when X is binomially distributed, we have $\mathbb{E}[X] = np$ and for any $k < np$, we can set $\delta = (np-k)/np$, yielding

$$\mathbb{P}[X \leq k] = \exp\left(-\frac{(np-k)^2}{2np}\right) \quad . \tag{2.6}$$

2.2.4 Linear Programming

A *linear program (LP)* is an optimization problem optimizing a linear function in $\mathbb{R}^n$ under a set of linear constraints, i. e., linear equalities and inequalities. Every LP can canonically be written as

$$\begin{aligned} \max \quad & c^T x \\ \text{s.t.} \quad & Ax \leq b \end{aligned} \tag{2.7}$$

for some $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$, and $b \in \mathbb{R}^m$. As usual, we will refer to the x_i as the *decision variables* and to the convex polyhedron given by $\{x \in \mathbb{R}^n : Ax \leq b\}$ as the *feasible region*. It was a long-standing open question, whether LPs are solvable in polynomial time. This question has been answered affirmatively with Khachiyan's ellipsoid method, which has a worst-case polynomial running time and has since been improved by interior point methods, such as the interior point projective method by Karmarkar. For more details see [Sch86].

2.2.5 Multidimensional Scaling

Classical multidimensional scaling (MDS) [Tor52] is a technique to find a good low-dimensional embedding given a matrix of metric dissimilarities or distances between a set of items. More formally, for some $n \in \mathbb{N}$ and given distances δ_{ij} for $1 \leq i, j \leq n$, MDS aims at finding positions $x_1, \ldots, x_n \in \mathbb{R}^d$ for some $d \ll n$, such that $d(x_i, x_j) \approx \delta_{ij}$, more precisely, such that $\sum_{i,j \leq n} (d(x_i, x_j) - \delta_{ij})^2$ is minimized. MDS utilizes that from

$$\delta_{ij}^2 = d(x_i, x_j)^2 = (x_i - x_j)^T (x_i - x_j) = x_i^T x_i - 2 x_i^T x_j + x_j^T x_j \ , \tag{2.8}$$

it can be shown that for $X = [x_1, \ldots, x_n]^T \in \mathbb{R}^{n \times k}$ and $XX^T = (b_{ij})$,

$$b_{ij} = -\frac{1}{2} \left(\delta_{ij}^2 - \frac{1}{n} \sum_{r=1}^{n} \delta_{rj}^2 - \frac{1}{n} \sum_{s=1}^{n} \delta_{is}^2 + \frac{1}{n^2} \sum_{r=1}^{n} \sum_{s=1}^{n} \delta_{rs}^2 \right) \ . \tag{2.9}$$

In other words, XX^T is obtained from the matrix of squared distances by *double-centering*, such that each column and each row sums to zero. More importantly, XX^T is a normal matrix and can be diagonalized using a basis of eigenvectors as $XX^T = VDV^T$. Now, we could yield n-dimensional positions $x_1, \ldots, x_n$ by setting

$$X = VD^{1/2} \ , \tag{2.10}$$

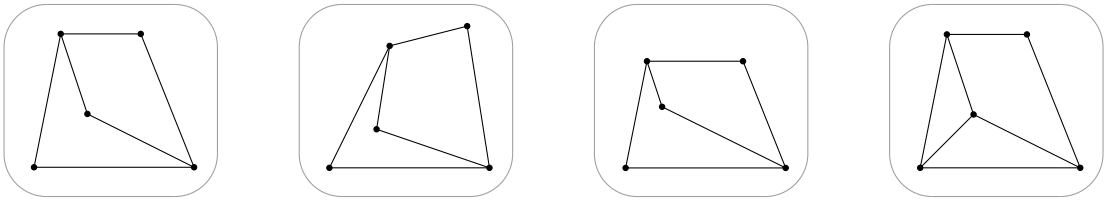
which can be approximated in $\mathbb{R}^d$ using only the d largest eigenvalues and the corresponding columns from V. For our purposes, we only need the two or three largest eigenvectors of XX^T, which can be computed by *power iteration*: A random vector $x \in \mathbb{R}^n$ is repeatedly multiplied with XX^T and normalized, converging towards the eigenvector x_{λ_1} to the largest eigenvalue λ_1. To compute the eigenvector to the second largest eigenvalue, λ_2, x is also orthogonalized to x_{λ_1} in every iteration. Given the matrix of dissimilarities, construction of XX^T takes $\Theta(n^2)$ and power iteration takes additional $O(n^2)$ per iteration step. For a more detailed description and variants of MDS see [CC01], for background on matrix computations, see [GvL96]. For all experiments, we used an implementation of classical MDS by Brandes and Pich [BP06].

2.2.6 Rigidity Theory

Rigidity theory provides us with a number of results on structures that do not allow for continuous deformation. The most prominent example is a characterization of graphs that, embedded in general position in the plane, cannot be continuously deformed without changing edge lengths. This property of so-called "bar-joint-frameworks" is also known as *generic rigidity* or, short, *rigidity* of a graph and the characterization as *Laman's counting property*:

Theorem 2.1 (Laman's counting property [Lam70]) *A graph $G = (V, E)$ is generically rigid if and only if it contains a set of edges $E' \subseteq E$ with $|E'| = 2|V| - 3$ such that for all subsets $E'' \subset E'$*

$$|E''| \leq 2|V[E'']| - 3 \ . \tag{2.11}$$



(a) A (parallely) non-rigid graph

(b) continuous deformation preserving edge lengths

(c) continuous deformation preserving edge directions

(d) A (parallely) rigid graph

Figure 2.2: *Rigidity and parallel rigidity in the plane*

It is a long-standing open problem to find a tight characterization for $\mathbb{R}^3$ or higher dimensions [Hen92]. A much less known result considers the property of an embedded graph not to allow for continuously changing positions while maintaining edge directions. This property also is called *parallel rigidity*. The characterization of parallely rigid graphs coincides with the characterization of rigid graphs in $\mathbb{R}^2$, but can be generalized for arbitrary dimensions:

Theorem 2.2 (Laman's theorem for parallel embeddings [Whi88]) *A graph $G = (V, E)$ is generically parallel rigid in $\mathbb{R}^d$, $d \geq 2$, if and only if it contains a set of edges $E' \subseteq E$ with $(d-1)|E'| = d|V| - (d+1)$ such that for all subsets $E'' \subset E$*

$$(d-1)|E''| \leq d|V[E'']| - (d+1) \ . \tag{2.12}$$

In Figure 2.2, two graphs with the same set of vertices and the same number of edges are depicted to illustrate the concepts of rigidity and parallel rigidity. Note that Theorems 2.1 and 2.2 also imply that a graph $G = (V, E)$ with a sufficient number of edges ($|E| \geq 2|V| - 3$) must either be (parallely) rigid in the plane or have a subgraph $S = (V_S, E_S) \subset G$ with $|E_S| > 2|V_S| - 3$.

Despite these analogies, there is another huge difference between standard rigidity and parallel rigidity: Given a rigid graph, embedded in the plane, the graph still has many other, completely different embeddings with the very same edge lengths, but it has only *similar parallel embeddings*: Given a graph $G = (V, E)$ and an embedding $\mathbf{p} : V \to \mathbb{R}^d$, an embedding $\mathbf{p}' : V \to \mathbb{R}^d$ is called *parallel*, if every edge $\{u, v\} \in E$ is embedded parallel or anti-parallel, i. e., if

$$\big(\mathbf{p}(u) - \mathbf{p}(v)\big)^{\perp} \cdot \big(\mathbf{p}'(u) - \mathbf{p}'(v)\big)^{\perp} = 0 \ , \tag{2.13}$$

and *similar*, if $\mathbf{p}' = c\mathbf{p} + \mathbf{x}$ for some $c \in \mathbb{R}$ and $\mathbf{x} \in \mathbb{R}^d$. For a more in-depth treatise of the different notions of rigidity, see [Whi97].

Notably, (parallely) rigid graphs in the plane can also be described as graphs with a maximum $(2, 3)$-sparse edge set [Lor79]: Given a set of nodes V, a set of edges E is (k, l)-sparse, if for every $V' \subseteq V$, V spans at most $k|V| - l$ edges $E[V']$. (k, l)-sparse edge sets form a matroid, the simplest being the standard graphic matroid. We will hence call a set of

edges *independent* if it is (k,l)-sparse and k and l are clear from the context, usually $k = 2$ and $l = 3$.

Maximum independent edge sets in (k,l)-sparse graphs can be identified in time $O(n^2)$ using the algorithm of Lee and Streinu [LS07]. This algorithm not only returns a maximum independent set of edges E' for some graph $G = (V,E)$ and parameters $k \in \mathbb{N}$, $l \leq 2k$, but also returns maximal components $V_1, \ldots, V_c$, such that $|E'(V_i)| = k|V_i| - l$, i. e., components that span a maximum number of independent edges, generalizing and improving the original *pebble game* algorithm of Jacobs and Hendrickson [JH97].

2.3 Models

Theoretical research in wireless sensor networks is still driven by the continuous search for appropriate models, most of them considering the communication of spatially distributed nodes. Many of them aim at a formal description of the circumstances under which nodes can exchange information. These efforts face a huge number of complex aspects that are imposed by wireless communication, as well as a major uncertainty of realistic assumptions of future applications and scenarios. Among these aspects, we have signal propagation and interference, synchronization, medium access, energy consumption and transmission failures. To these aspects, mobility and node failures can add. Yet, there is no all-embracing model of computation in sight that offers both, a convenient and credible abstraction of sensor networks that takes all of the above issues into account. Unfortunately, it is questionable whether a satisfactory such model exists. More likely, the trend to different models for different applications is going to persist, yielding more general abstractions for universal sensor networks and fine-grained models for specific assumptions on node deployment and network traffic patterns. The variety and diversity of models that mushroomed in the algorithmic community have its counterpart in practice, where long-standing paradigms such as the OSI Layer model are forgone in favor of cross-layer optimization and protocols tailored to particular purposes.

Nevertheless, there are some standard models that proved helpful in the past, and which we will sketch in the following.

2.3.1 Distributed Computing

Among existing models of computation, models of *distributed computing (DC)* are the most reasonable first approximation of computation in wireless sensor networks. While the concepts of DC do not model a spatially shared communication channel, they are able to cover the properties of a WSN to consist of a large number of independently working computation units, each running the same program, and connected by some arbitrary symmetric neighborhood relation. Nodes can have additional input, and the also the result of an distributed algorithm $\mathcal{A}$ is usually an individual result for every node. In the synchronous model of distributed computing, nodes are globally synchronized and perform communication rounds,

in each of which every node can first perform polynomial-time sequential computation accessing only locally available data, and second send a message to each of its neighbors.

The complexity of a distributed algorithm $\mathcal{A}$ can be measured in the number of communication rounds, which is called $\mathcal{A}$'s *time complexity*. It can also be measured in the number of messages sent during the execution of $\mathcal{A}$, which we call $\mathcal{A}$'s *message complexity*. The complexity of problems depends on the additional restrictions in different models. In the context of sensor networks, the three most relevant models of distributed computing are the following [Pel00, ZG04]:

LOCAL: The LOCAL model is the most powerful model for distributed computing. Here, message sizes are unbounded, i. e., at the end of each communication round, each node can send an arbitrary amount of data to each of its neighbors. This model is of utmost importance for the question what nodes can decide by local knowledge: In every algorithm $\mathcal{A}$ with time complexity k, the result at some node v is independent from the input and from neighborhood relations of nodes that have a distance of more than k hops. Moreover, the problems that can be solved using k rounds of communication also are exactly the problems that can be solved without any communication if every node learns the input and neighborhoods of all nodes within k hops distance. This follows from the observation that, given $\mathcal{A}$ and the input and neighborhood relation in its k-hop neighborhood, every node can emulate $\mathcal{A}$ with exactly the same outcome as by running $\mathcal{A}$. An algorithm with time complexity k in the LOCAL model is also said to be k-local.

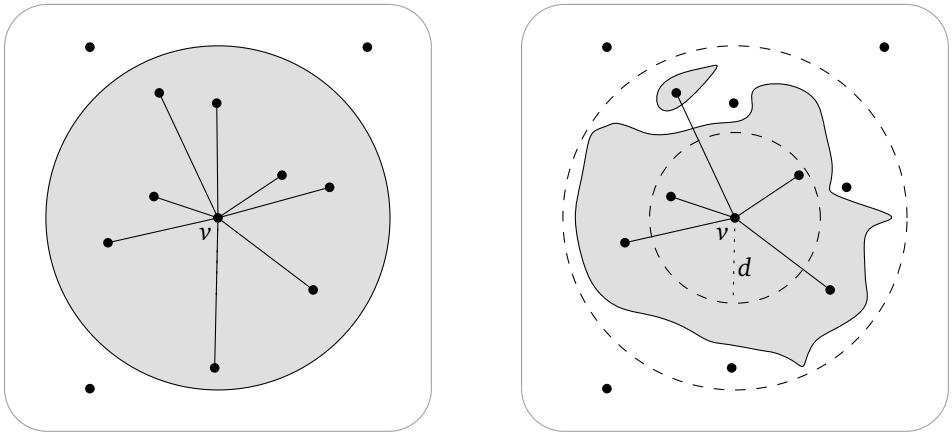
CONGEST: While providing a good foundation for impossibility results, the unbounded communication per time makes the LOCAL model unrealistic even for wired communication between the entities. In the CONGEST model, message sizes are hence restricted to $O(\log n)$, n being the number of nodes.

LOCALIZED: In a k-LOCALIZED algorithm, each node is allowed to pass messages to its neighbors in at most k rounds. While this allows for worst-case runtime of $\Theta(n)$, it in some way reflects the need for algorithms with a low communication overhead per node better than the restriction to a linear message complexity.

Adapting these models to wireless sensor networks, we will also assume a more restricted node model where nodes have limited memory. Unless stated otherwise, we assume a memory that allows storing node IDs of size $\Theta(\log n)$ of at least a constant-hop neighborhood, i. e., we assume that nodes have memory $\Omega(\Delta^{O(1)} \log n)$ as in [Krö08].

2.3.2 Radio Model, Connectivity and Interference

Spatial connectivity and interference of concurrent transmissions, being the most outstanding attributes of wireless networks, have been subject of countless modelling approaches. These phenomena, arising from the fact that nodes communicate via radio transmissions are the major difference to standard models from distributed computing and both, boon and bane. On the one hand, spacial connectivity models constrain the structure of networks and allow for tailored algorithms to operate on such networks. On the other hand, interference is a new problem restricting the communication between nodes.



(a) UDG model (b) d-QUDG model

Figure 2.3: *Communication ranges for UDG and QUDG models.*

Since in reality, connectivity and interference are composed of many hard-to-capture phenomena such as multipath fading, algorithmic research developed numerous simplifications. The common assumption behind almost all theoretical models is that every node v's signal decays with distance $d(v,\cdot)$ as $d(v,\cdot)^{-\alpha}$ for some so-called *path-loss exponent* α, empirically $2 < \alpha < 6$. Assuming further that nodes are restricted to the same maximum transmission power and identical, perfect omnidirectional antennae, the first and simplest model for connectivity assumes that there is some radius r such that each node v can communicate exactly with the nodes within $B_r(v)$. Normalizing this radius to 1, this is equivalent to the assumption that the network is embedded as a *unit disk graph* [CCJ90]:

Definition 2.3 (Unit Disk Graph) *A graph $G = (V,E)$ is a* unit disk graph (UDG)*, if there is an embedding $\mathbf{p} : V \to \mathbb{R}^2$, such that*

$$\{u,v\} \in E \Leftrightarrow d\left(\mathbf{p}(u),\mathbf{p}(v)\right) \leq 1 \tag{2.14}$$

for all $u,v \in V$. Then, $\mathbf{p}$ embeds G as a UDG.

Obviously, this is an oversimplification. Even under laboratory conditions, communication ranges do not form perfect disks. As a compromise between the powerful unit disk graph model and the irregular communication ranges that are observed even in the absence of obstacles, the unit disk graph model is generalized to the assumption that sensor networks are embedded as *quasi unit disk graphs* as introduced in [BFN01] (Figure 2.3):

Definition 2.4 (Quasi Unit Disk Graph) *A graph $G = (V,E)$ is a d-*quasi unit disk graph (d-QUDG) *for some $0 < d \leq 1$, if there is an embedding $\mathbf{p} : V \to \mathbb{R}^2$, such that*

$$\begin{array}{rcl} \{u,v\} \in E & \Rightarrow & d\left(\mathbf{p}(u),\mathbf{p}(v)\right) \leq 1 \\ \{u,v\} \notin E & \Rightarrow & d\left(\mathbf{p}(u),\mathbf{p}(v)\right) > d \end{array} \tag{2.15}$$

for all $u,v \in V$. Then, $\mathbf{p}$ embeds G as d-QUDG.

Obviously, a 1-QUDG is a UDG. It is important to note that, given a set of nodes V with positions in the plane, the corresponding UDG is well-defined, while the d-QUDG model does not make any assumption on the connectivity of nodes within distance between d and 1. In a straightforward manner, both models can be generalized to higher dimensions. The resulting models are coined *unit ball graph (UBG)* or *d-quasi unit ball graph (d-QUBG)*. The strength of these class of models is, that they allow to exploit structural properties for the design of algorithms. To name but one, in these models, each node can only have a constant number of independent neighbors, e. g., in a UDG, no node can have more than 5 independent neighbors.

As an even further relaxation, this observation led to the class of *bounded independence graphs (BIG)* of graphs where every node v's k-hop neighborhood $N_G^k(v)$ contains at most $O(k^r)$ independent nodes for some fixed r.

Connectivity models such as the UDG or d-QUDG models gave rise to a number of important results combined with the above models of distributed computing, e. g., for clustering and coloring [KMNW05]. They model the effects of spatial communication quite well, but they do not take the limits of the wireless medium into account, that concurrent transmissions can interfere with each other and make reception impossible. We call models that focus on these limits *interference models*. We will give a more rigorous definition later, but sketch some of the most prominent examples of interference models to point out the differences to connectivity models and the spread of interference models.

Not surprisingly, the first interference models that have been studied algorithmically closely relate to the disk graph models: In the *distance interference* model, nodes can communicate if and only if they are within Euclidean distance 1 and if there is no concurrent sender in Euclidean distance less than some constant $R \geq 1$ of the receiver. In a slight variation, the *protocol model* also tries to take into account that signals decay with the distance. Here, nodes u, v can communicate if and only if they have distance at most 1 and if there is no concurrent sender with distance less than $R \cdot d(u, v)$ to the receiver. There are many variants that evolved from similar considerations, some of them accounting for acknowledgments, some of them purely combinatorial, like the *hop interference* model, which we will have a closer look at in Chapter 5.

As mentioned before, capturing a transmission from a communication engineer's point of view depends mainly on the so-called signal-to-interference-plus-noise ratio (SINR) (cf. Section 2.1.2). In its most general form, the SINR model assumes a *gain matrix* $G = (g_{ij}) \in \mathbb{R}_+^{n \times n}$, n being the number of nodes in the network, as well as individual background noise levels η_v and reception thresholds β_v for each node v. A transmission from some sender s can then successfully be decoded by a node v if and only if

$$\frac{p_s g_{sv}}{\eta_v + \sum_{u \in V - s} p_u g_{uv}} \geq \beta_v \ , \tag{2.16}$$

p_u denoting transmission powers for nodes $u \in V$.

Under the assumptions that the signal decays with the distance d as $d^{-\alpha}$, and that further all nodes are completely identical and do not receive different levels of background noise—which is implicit to all of the above models—, this yields the *geometric SINR* model:

Definition 2.5 *In the* geometric SINR model*, a node v receives a transmission from a sender s if and only if*

$$\frac{p_s d(s,v)^{-\alpha}}{\eta + \sum_{u \in V - s} p_u d(u,v)^{-\alpha}} \geq \beta \quad , \tag{2.17}$$

p_u denoting transmission powers for nodes $u \in V$, i. e., $p_u = 0$ for nodes that are not transmitting.

We will see a more rigorous taxonomy of interference models in Chapter 6. For an overview of sensor network models, see, for example, [SW06].

Chapter 3

Positioning

This chapter deals with positioning problems from a theoretic point of view. Positioning here always refers to the problem of recovering node positions without any previously known positions of single "anchor" nodes[1]. We here focus on algorithmic results, i. e., complexity status and runtime of solutions for well-defined problems.

3.1 Introduction and Problem Statement

Many protocols and applications take the availability of position information to nodes as granted. Position-awareness is a prerequisite for almost all surveillance or monitoring tasks: If a sensor network raises an alarm, it should be able to provide the information, where the critical event has been observed. If it protocols conditions in ecosystems or answers queries for the situation in a monitored area, it should be able to also consider the geographic context the sensed data relates to.

But even more importantly, many protocols for large-scale sensor networks exploit geometric properties of networks where each node knows its position. The probably most prominent example is geographic routing which assume that a packet can be addressed with a target node's geographic position rather than a meaningless ID. Protocols like GPSR and all its successors then route packets greedily as long as possible, each node passing a packet to the neighbor which is closest to the packet's destination, overcoming local minima with geometric techniques [KK00]. Position-awareness is also an essential ingredient of

[1] In the literature, the terms "positioning" and "localization" are sometimes used synonymously, but also to distinct between the (independent) localization of single nodes and the task to find positions for the whole network simultaneously.

many clustering techniques and geographic hashing techniques. The latter constitutes the idea to see a spatially distributed sensor network as a continuous data storage that can be addressed with coordinates. Information is stored at geographic positions and maintained by the nodes close to this position.

There have been many different answers to the questions how to yield position-awareness in sensor networks, including GPS-equipped nodes, central infrastructure [PCB00] or manual programming of a fraction of the nodes to let them know their positions. To us, the most challenging problem remains to let nodes find out their positions by themselves without any external infrastructure. The input to this class of *anchor-free positioning* problems are observations of the single nodes. This can be the neighborhood relation only, which together with assumptions of a connectivity model carries a lot of information, but it can also be any kind of measurements, such as the direction in which a node senses a neighbor or the distance. Depending on the kind of input, positioning algorithms are typically divided into connectivity-based, direction-based or range-based algorithms. It is important to note that connectivity information comes "for free", whereas availability of distance and direction information is platform-dependent. It is often argued that among these two, distance information is cheaper, since it can be derived from the *received signal strength indicator (RSSI)*, but it has also become evident that this indicator at best provides poor estimations. There are several so-called *ranging technologies* such as ultrasound available for more accurate distance estimations [PCB00], but also direction-based localization has been studied [WSC07]. It can be supported using antenna arrays for direction-of-arrival estimation or beam-forming techniques [God97], which have successful implementations, although, to our knowledge, not in the context of wireless sensor networks.

3.1.1 Related Work

In the past, many variants of anchor-free positioning have been looked at. To determine the complexity status, usually the decision problem whether there is a realization that complies with the given input is considered, and, most times proven to be NP-hard. It is important to note that this *realization problem* is different to the problem to decide whether there is a unique embedding realizing the given input.

Table 3.1: *Complexity results of finding a valid embedding for a network*

Realization problem	*Complexity*
UDG connectivity	NP-hard [BK98]
UDG approximation, QUDG approximation	NP-hard [KMW04]
UDG and 1-hop distances	NP-hard [AGY04]
UDG and 1-hop directions	NP-hard [BGJ05]
1-hop distances and directions (relative or absolute)	trivial
1-hop directions (relative or absolute)	in P(folklore, [BGJ05])
1-hop distances	NP-hard [Sax79]
Distances and global directions, arbitrarily small errors	NP-hard [BGMS06]
Distances and local directions, arbitrarily small errors	NP-hard (Section 3.3)

Table 3.1 summarizes the complexity results for realization problems. Breu and Kirkpatrick showed that it is hard to decide whether a graph is a unit disk graph [BK98]. Recently, Kuhn et al. showed that it is also hard to approximate embeddings that witness a UDG or QUDG property [KMW04]. More precisely, the authors showed that it is hard to find an embedding as a $\sqrt{2}/3$-QUDG for a given UDG, and a similar result for the embedding of d-QUDGs as d'-QUDGs for $d' \leq d$. Aspnes et al. showed that it remains NP-hard to answer whether a graph can be embedded as UDG, if additionally edge lengths are given [AGY04], and Bruck et al. proved the same for edge directions [BGJ05]. Given edge lengths and directions, the realization problem is trivial, it remains easy if only directions are constrained (and no connectivity model is involved), but for given edge lengths, the realization problem in general graphs is hard (Saxe [Sax79]). The problem to find an embedding that realizes given directions and distances with a minimum error has been shown to be NP-hard for the case that *global* directions are given [BGMS06]. We will show in Section 3.3 that it is also hard for local directions.

The problem to decide whether a graph's embedding is uniquely defined by some constraints has been addressed for general graphs in [Hen92], considering sufficient conditions for unique embeddings and for sensor networks in [EGW$^+$04, AEG$^+$06], presenting probabilistic results for random sensor networks.

3.2 Direction-based Positioning

Unlike most range- or connectivity-based positioning problems, direction-based positioning is not a hard problem unless combined with other constraints. In direction-based positioning, the task is to recover the true node positions $\mathbf{p} : V \to \mathbb{R}^2$ and orientations $\mathbf{o} : V \to [0, 2\pi)$ solely from relative edge directions for all directed edges $(u, v) \in E$ as

$$\omega_{\mathbf{p},\mathbf{o}}(u, v) := R_{-\mathbf{o}(u)} \cdot \frac{\mathbf{p}(v) - \mathbf{p}(u)}{|\mathbf{p}(v) - \mathbf{p}(u)|} \ , \tag{3.1}$$

R_θ being the rotation matrix for rotation angle θ. However, as long as we do not consider noisy input, we can without loss of generality assume that $\mathbf{o} \equiv 0$: With a single broadcast message flooded through the network, all nodes can agree on the same global reference direction and translate relative edge directions accordingly.

Embeddings that yield the same edge directions are *parallel* in the same sense as in parallel rigidity with the only difference that edges may not be embedded anti-parallel, and with $\mathbf{o} \equiv 0$, an embedding can only be determined up to translation and scaling by edge directions.

Bruck et al. proved that it is NP-hard to decide whether a graph can be realized as UDG with given edge directions. On the other hand, finding a valid embedding for general graphs reduces to a linear program with edge lengths $\ell : E \to \mathbb{R}_+$ as variables: An embedding

realizes given directions $\omega(w,w')$, if for all cycles C the following holds

$$\sum_{\{w,w'\}\in C} \ell(w,w') \cdot \omega(w,w') = \mathbf{0} \ , \tag{3.2}$$

and all edge lengths are strictly positive. Obviously, it is sufficient to consider cycles of any cycle base only, such as the cycles induced by non-tree edges for an arbitrary spanning tree T, yielding a linear program with m variables and $2(m-n+1)$ linear constraints, not counting the non-negativity constraints. This shows how misleading hardness proofs for realization problems can be: Direction-based graph realization being hard under UDG constraints does not make direction-based graph realization harder. If a graph's (or a subgraph's) embedding is uniquely determined by the given edge directions—up to the inevitable degrees of freedom, translation and scaling—, it is also easy to answer whether this embedding is a UDG for some scaling. From the hardness proof we only learn that additional UDG constraints do in general not help when directional constraints are not sufficient to recover node positions.

We hence focus on the identification and positioning of maximal subgraphs that are uniquely determined by the given directions. As stated in Section 2.2.6, maximum rigid components can be identified in time $O(n^2)$ using the so-called pebble game. But running the pebble game requires to gather connectivity information of the whole network centrally, and we have some very intuitive techniques to find partial solutions within the network. In the following, we will first describe a protocol to distributedly partition a network into (not necessarily maximum) rigid subgraphs, called *bodies* in the following sense:

Definition 3.1 (Laman partitions) *Let $G=(V,E)$ be a simple undirected graph and $\mathcal{B}$ be a set of pairwise edge-disjoint, generically rigid subgraphs or* bodies*. We call $\mathcal{B}$ a* Laman partition *of $G(\mathcal{B}) := (V(\mathcal{B}), E(\mathcal{B}))$ as follows:*

$$V(\mathcal{B}) := \bigcup_{(V,E)\in\mathcal{B}} V \quad \textit{and} \quad E(\mathcal{B}) := \bigcup_{(V,E)\in\mathcal{B}} E \ , \tag{3.3}$$

We call $\mathcal{B}$ rigid *if $G(\mathcal{B})$ is rigid and it is* maximal *if there is no $\mathcal{B}' \subsetneq \mathcal{B}$ which is rigid.*

Note that a subset $\mathcal{B}' \subset \mathcal{B}$ of a Laman partition is also a Laman partition.

Second, we describe how to efficiently answer the question which of these bodies together form rigid components for using a slight improvement on the pebble game algorithm. This step still needs to gather information centrally, but only information of nodes that are part of more than one rigid body. Third, we show briefly how a focus on uniquely determined subgraphs can speed up positioning compared to an LP formulation.

3.2.1 Distributed Preprocessing

A very lightweight protocol for distributed identification of rigid subgraphs and positioning of nodes based on directions is successive triangulation starting at any edge and growing a set of localized nodes: Starting at an edge $\{u,v\}$, the two nodes u and v assign themselves

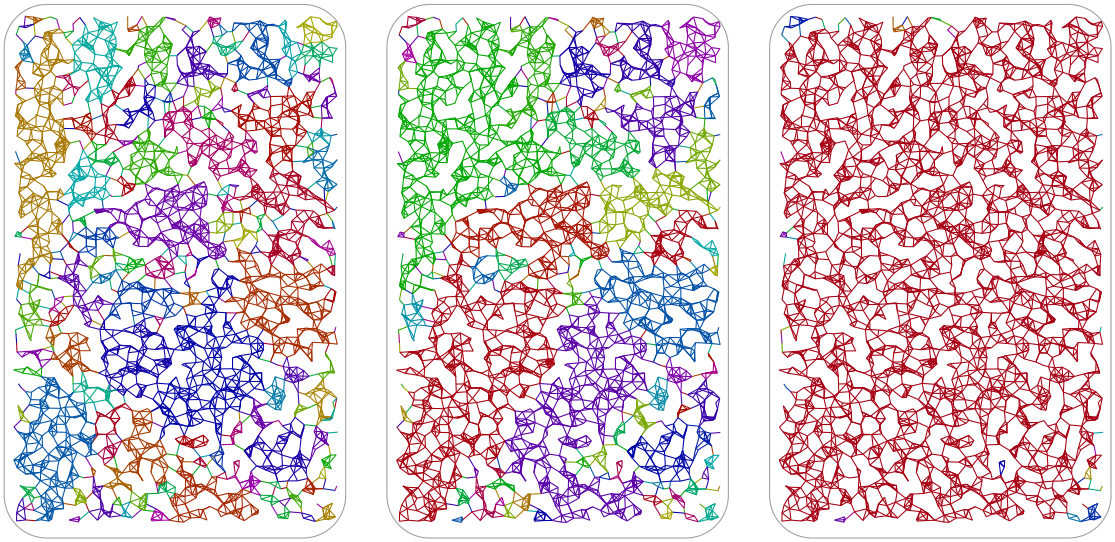
Protocol 3.1: *DistributedLamanPartitioning*(r)

set-up
every edge e picks some t_e
in round $1 \leq t \leq r$
all edges $e = \{u, v\}$ **do in parallel**
if $t = t_e$ *and* $p_e = \emptyset$ **then** $p_e \leftarrow e$, $P_u \leftarrow P_u \cup \{e\}$, $P_v \leftarrow P_v \cup \{e\}$
all nodes u **do in parallel**
$J_u \leftarrow \{e_b : \sharp\{v \in N_G(u) \mid e_b \in P_v\} \geq 2\} \setminus P_u$
all edges $e = \{u, v\}$ **do in parallel**
$J_e \leftarrow (P_u \cap J_v) \cup (P_v \cap J_u)$
if $p_e = \emptyset$ *and* $J_e \neq$ **then** $j_e \leftarrow$ edge e_b with minimum ID in J_e.
all nodes u **do in parallel**
$P_u \leftarrow P_u \cup \big(\{e_b : \sharp\{e \ni u \mid j_e = e_b\} \geq 2\}\big)$
all edges $e = \{u, v\}$ **do in parallel**
if $p_e = \emptyset$ *and* $j_e \in P_u \cap P_v$ **then** $p_e \leftarrow j_e$
if $p_e = \emptyset$ *and* $P_u \cap P_v \neq \emptyset$ **then** $p_e \leftarrow$ edge e_b with minimum ID in $P_u \cap P_v$

positions $(0,0)$ and $(0,1)$, respectively, and whenever a node has two neighbors with positions relative to that edge, it can also assign itself a position. The identified rigid subgraph then consists of all nodes with positions and edges between these nodes.

This idea can easily be extended to parallel identification of non-overlapping subgraphs running in $r \in \mathbb{N}$ rounds, as depicted in Protocol 3.1. For the description of this synchronous protocol, we will assume that edges also are entities that can store data and communicate with their incident nodes. We assume that for every edge, one of its incident nodes performs the necessary computations and communication. Within the range $[0, r]$, every edge e picks a random starting slot t_e. Then, every round, the following five steps are performed, each involving at most the exchange of information between a node and its neighbors. During execution, edges are marked with exactly one edge as representative for the body it belongs to, p_e. A node u belongs to any body that at least one incident edge belongs to, in the protocol denoted by P_u. First, in round t, any edge e with $t = t_e$ starts its own body if it has not become part of a rigid subgraph until then, i. e., it assigns its incident nodes positions $(0,0)$ and $(0,1)$, respectively. Second, if a node has two or more neighbors with positions referencing the same edge e_b, it marks itself as eligible to join e_b's body. Any edge that has one endpoint belonging to some e_b's subgraph and one endpoint marked as eligible to join the same body marks itself as a candidate to join e_b's subgraph for this round, breaking multiple choices of e_b in favor of the one with the lowest ID. If a node now has two or more incident edges marked as candidates to join some e_b's subgraph, in the last two steps, it joins them. Also, in the last step, all incident edges which were candidates to join e_b's body join it. Finally, every edge that does not belong to a body can join a body if it has two endpoints belonging to a common body. Again, ties are broken in favor of the lower ID.

This protocol ensures that after r rounds, the network is partitioned into a Laman par-



(a) distributed triangulation (b) exploiting overlapping edges (c) maximal rigid partition

Figure 3.1: *Laman partitions of a network*

tition induced by the edges' membership to rigid subgraphs: Obviously, every edge is part of a body, as it starts its own if necessary. We also ensure that a body is only extended if some node finds two edges which join the body together with it, maintaining the rigidity of bodies.

The quality of this partition, i. e., the number of bodies or the number of body-node-incidences, depends on the number of rounds r, the network's density and the way edges pick starting slots t_e. In Figure 3.1(a), such a partition is depicted for a random quasi unit disk graph of 2556 nodes and 6837 edges for $r = 250$ and cumulative probability density $F(t) = (t/r)^4$. It does not much improve for higher r. It consists of 1358 node-body incidences between 426 bodies and 638 nodes that are part of more than one rigid subgraph, which is less than ten percent incidences to be gathered at the central instance for further processing. Notably, most of the remaining bodies consist of single edges.

One can think of more sophisticated ways to partition the network distributedly. E. g., it is possible to allow that edges join multiple subgraphs and then merge these two subgraphs, which obviously together form a larger rigid subgraph. It may also be possible to identify any pair of subgraphs that overlap in at least two nodes, even if these are not adjacent, and merge these subgraphs. It is questionable whether these operations are worth the communication necessary to perform merging operations distributedly. They also do not improve the resulting partition much compared to the simple distributed triangulation: In Figure 3.1(b), a partition of the same graph is depicted that results from triangulation and merging subgraphs that cover a common edge. It ends in this example with 891 node-body incidences between 423 bodies and 285 nodes in which bodies overlap. Again, most remaining bodies consist of single edges.

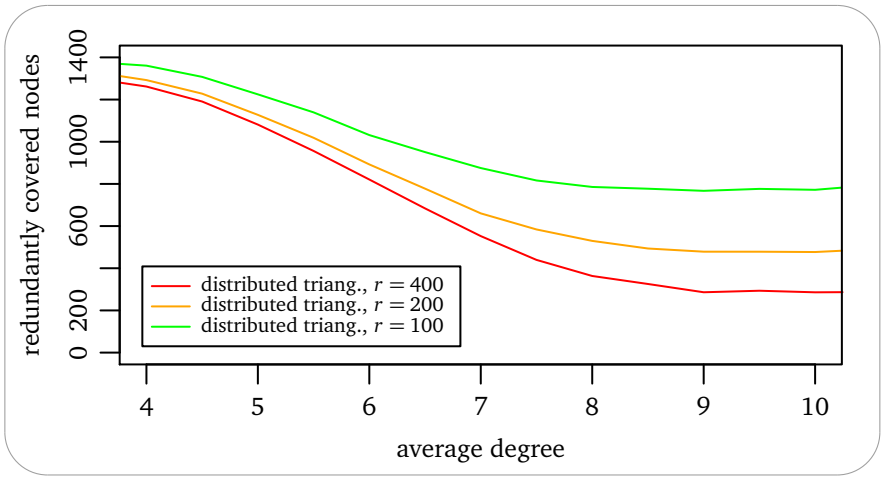


Figure 3.2: *Number of nodes that belong to different rigid subgraphs after distributed triangulation in random QUDG with 4000 nodes.*

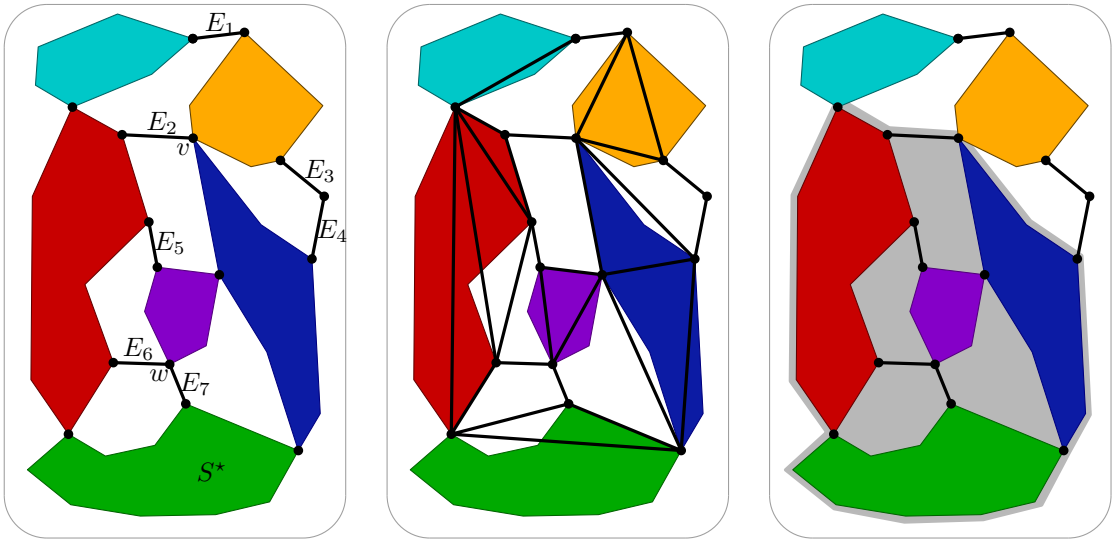
Figure 3.1(c) depicts a partition into maximum rigid subgraphs to compare the above results to. Here, the largest component covers almost the whole network; overall, in this partition only 48 rigid components remain. In Figure 3.2, the reduction of nodes that are part of more than one body after distributed triangulation is depicted. These are the only nodes whose information is needed to compute which and how the identified rigid bodies can be merged to gain maximum rigid components. Depending on the density of the network and the number of rounds, this number can be decreased to less than 10%. Note that we also do not need to gather information about all incidences to edges, but only about all incidences to rigid bodies.

3.2.2 Maximum Rigid Components

While distributed preprocessing can identify some rigid substructures, the identification of maximum rigid component requires central processing. This is a simple consequence of the fact that even in the plane, it is easy to have settings where an arbitrary number of subgraphs form a rigid and maximal Laman partition. A small such non-trivial situation is depicted in Figure 3.3(a), where we have 13 bodies—some of them single edges—, of which 8 form a rigid structure (cf. Figure 3.3(c)). We thus turn our attention to the problem of how to identify maximum rigid sets within a Laman partition $\mathcal{B}$ given *only* the node-body incidences between redundantly covered nodes and bodies as follows:

Definition 3.2 (Redundancy) *Let $\mathcal{B}$ be a Laman partition. The* redundancy *of a node $v \in V(\mathcal{B})$ is defined as*

$$\mathtt{rd}_{\mathcal{B}}(v) := |\{(V,E) \in \mathcal{B} : v \in V(\mathcal{B})\}| - 1 \ . \tag{3.4}$$



(a) 13 rigid bodies (b) virtual edges (c) maximal partition

Figure 3.3: *Rigid partition with non-trivial rigid subset*

This notion is extended to Laman partitions by

$$\mathtt{rd}(\mathcal{B}) := \sum_{v \in V(\mathcal{B})} \mathtt{rd}_{\mathcal{B}}(v) \quad . \tag{3.5}$$

We denote the redundantly covered nodes *as* $R(\mathcal{B}) := \{v \in V(\mathcal{B}) \mid \mathtt{rd}_{\mathcal{B}}(v) > 0\}$ *and the* node-body incidences *as* $\mathcal{I}_{\mathcal{B}} := \{(v, S) \in R(\mathcal{B}) \times \mathcal{B} \mid v \in V_S\}$.

In Figure 3.3(a), only redundant nodes are depicted. Only the nodes v and w have $\mathtt{rd}_{\mathcal{B}}(\cdot) = 2$, all other nodes have $\mathtt{rd}_{\mathcal{B}}(\cdot) = 1$. Given this input, one can without loss of generality assume that $V(\mathcal{B}) = R(\mathcal{B})$ and that every body $S = (V_S, E_S)$ contains exactly a minimal rigid set of $|E_S| = 2|V_S| - 3$ edges. A quite straightforward way to identify maximum rigid sets within a Laman partition $\mathcal{B}$ from $\mathcal{I}_{\mathcal{B}}$ is to replace each body with such a set of virtual edges between the redundant nodes (cf. Figure 3.3(b)). Applying the pebble game to this graph for any Laman partition $\mathcal{B}$ reveals, which bodies together form a rigid structure in time $O(n_{\mathcal{B}}^2)$ for $n_{\mathcal{B}} := |R(\mathcal{B})|$. In the following, we will show how to reduce this to $O(m_{\mathcal{B}} \log m_{\mathcal{B}} \log \Delta_G + k_{\mathcal{B}}^2)$ for $m_{\mathcal{B}} := |\mathcal{I}_{\mathcal{B}}|$ and $k_{\mathcal{B}} = |\mathcal{B}|$ using an approach that directly operates on the set of bodies.

We therefore define a notion of a surplus of edges within a graph as follows:

Definition 3.3 *Given a graph* $G = (V, E)$*, the* surplus of edges *with respect to Laman's Theorem is denoted by* $\mathtt{sp}(G) := |E| - 2|V| + 3$*. We also write* $\mathtt{sp}(\mathcal{B})$ *for some Laman partition to shortcut* $\mathtt{sp}(G(\mathcal{B}))$.

Note that a graph $G = (V, E)$ has at most $|E| - \mathtt{sp}(G)$ independent edges.

The approaches from [LS07, JH97] have in common that they manage a growing independent set of edges. Due to the matroidal character of the problem, an edge can greedily be chosen to join this set if there is no dependency to present edges. Rigid areas of the network can be identified en passant. When talking about rigid bodies, we lose some of this ease, since a subgraph can have both, edges that are independent to the edges of formerly chosen bodies as well as dependent ones. But the greedy approach still works: If we go through the bodies of a Laman partition and merge bodies as soon as there are bodies that form a larger rigid structure, we end up with a partition into maximum rigid components. With the following results, we can save the time for successively introducing edges for each body, starting with the observation that a Laman partition with sufficiently overlapping bodies must have enough edges to fulfill Laman's theorem:

Lemma 3.4 *Let $\mathcal{B}$ be a rigid partition. Then*

$$\mathtt{sp}(\mathcal{B}) = 2 \cdot \mathtt{rd}(\mathcal{B}) - 3(|\mathcal{B}| - 1) \quad . \tag{3.6}$$

Proof. As the graphs in a rigid partition have disjoint edge sets, the edges of $G(\mathcal{B})$ sum up as

$$|E(\mathcal{B})| = \sum_{(V,E)\in\mathcal{B}} |E| = \sum_{(V,E)\in\mathcal{B}} (2 \cdot |V| - 3) \quad , \tag{3.7}$$

where each node $v \in V(\mathcal{B})$ is covered $\mathtt{rd}_{\mathcal{B}}(v) + 1$ times. Thus, we have

$$|V(\mathcal{B})| = \sum_{(V,E)\in\mathcal{B}} |V| - \mathtt{rd}(\mathcal{B}) \quad , \tag{3.8}$$

and hence

$$\begin{aligned} \mathtt{sp}(\mathcal{B}) &= |E(\mathcal{B})| - 2 \cdot |V(\mathcal{B})| + 3 \\ &= \sum_{(V,E)\in\mathcal{B}} (2 \cdot |V| - 3) - 2 \cdot \sum_{(V,E)\in\mathcal{B}} |V| - 2\mathtt{rd}(\mathcal{B}) + 3 \\ &= 2\mathtt{rd}(\mathcal{B}) - 3(|\mathcal{B}| - 1) \quad . \end{aligned} \tag{3.9}$$

□

From Laman's theorem follows that a Laman partition $\mathcal{B}$ with $\mathtt{sp}(\mathcal{B}) \geq 0$ at least contains a rigid subset. Adapting the iterative scheme, we will use the following theorem to maintain a maximal rigid partition, merging bodies whenever a rigid subset is completed:

Lemma 3.5 *Let $\mathcal{B}$ be a rigid partition and $S^\star \in \mathcal{B}$ such that $\mathcal{B} - S^\star$ is maximal. Then $\mathcal{B}' \subset \mathcal{B}$ is rigid if and only if for all non-empty $\mathcal{B}'' \subseteq \mathcal{B}'$ that contain $S^\star$, the inequality $\mathtt{sp}(\mathcal{B}') \geq \mathtt{sp}(\mathcal{B}'')$ holds.*

Proof. First assume that $\mathcal{B}'$ is rigid. If there was any subset $\mathcal{B}''$ of $\mathcal{B}'$ with $\mathtt{sp}(\mathcal{B}') < \mathtt{sp}(\mathcal{B}'')$, one could not choose $|E(\mathcal{B}')| - \mathtt{sp}(\mathcal{B}')$ edges from $E(\mathcal{B}')$ without choosing more than $|E(\mathcal{B}'')| - \mathtt{sp}(\mathcal{B}'')$ edges from $E(\mathcal{B}'')$. Therefore, any $2|V(\mathcal{B})| - 3$ edges from $G(\mathcal{B})$ cannot be independent.

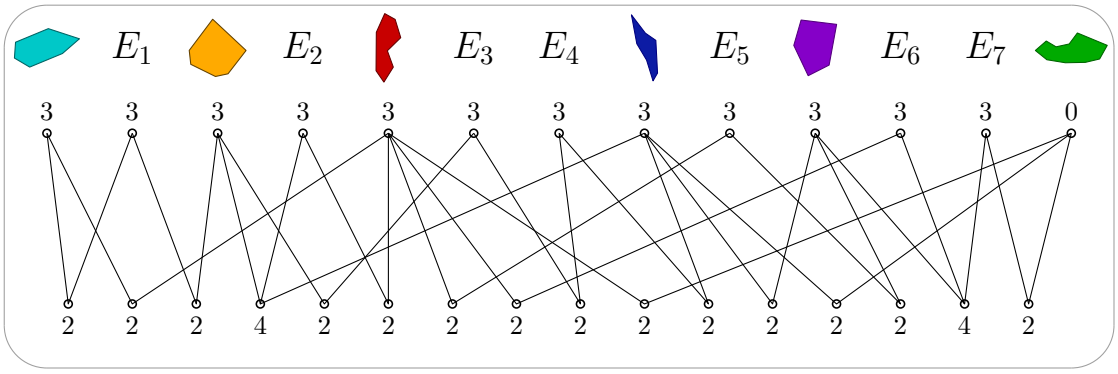


Figure 3.4: *Intersection network for the Laman partition from Figure 3.3, bodies at the top, redundant nodes at the bottom. All edges have capacity 2, bodies and nodes are annotated with their demand.*

If on the other hand for all $\mathcal{B}'' \subset \mathcal{B}'$ with $S^\star \in \mathcal{B}''$ the inequality $\mathtt{sp}(\mathcal{B}') \geq \mathtt{sp}(\mathcal{B}'')$ holds, then we know that $\mathtt{sp}(\mathcal{B}') \geq 0$, as it holds for all bodies, i. e., $\mathtt{sp}(S) = 0$ for all $S \in \mathcal{B}$. Suppose that $\mathcal{B}$ is not rigid. According to Laman's theorem, there must be a rigid subgraph $G' = (V', E') \subsetneq G(\mathcal{B}')$ with $|E'| > 2|V(E')| - 3$. This graph G' spans over at least two graphs in $\mathcal{B}$ which also form a rigid graph with at least one dependent edge. All those non-trivial rigid subsets $\mathcal{B}''$ include $S^\star$; thus their union $\mathcal{B}^\cup$ forms the unique maximal rigid subgraph $G(\mathcal{B}^\cup)$. But then, we are able to choose $|E(\mathcal{B}')| - \mathtt{sp}(\mathcal{B}')$ edges from $E(\mathcal{B}')$ even if we restrict ourselves to take only a set of independent edges from $E(\mathcal{B}^\cup)$ where we only have to leave out $\mathtt{sp}(\mathcal{B}^\cup) \leq \mathtt{sp}(\mathcal{B}')$ edges. These $2|V(\mathcal{B}')| - 3$ edges are either independent, so that $\mathcal{B}'$ must be rigid, or there still is a subgraph with $G' = (V', E') \subsetneq G(\mathcal{B}')$ with $|E'| > 2|V(E')| - 3$ which is not covered by $\mathcal{B}^\cup$. Both cases are inconsistent with either the assumptions or the definition of $\mathcal{B}^\cup$. □

We present an efficient algorithm to detect subsets with this property by a reduction to a maximum-flow problem.

Definition 3.6 *Let $\mathcal{B}$ be a Laman partition and $S^\star \in \mathcal{B}$ such that $\mathcal{B} - S^\star$ is maximal. The bipartite* intersection network *$\mathcal{N}(\mathcal{B}, S^\star) = (R(\mathcal{B}), \mathcal{B}, \mathcal{I}_\mathcal{B}, \kappa, b)$ is given by*

$$\begin{aligned} \kappa &\equiv 2 \\ b(v) &= 2 \cdot \mathtt{rd}_\mathcal{B}(v) \\ b(S) &= \begin{cases} 3 & : S \neq S^\star \\ 0 & : S = S^\star \end{cases} \end{aligned} \tag{3.10}$$

The intersection network of the Laman partition from Figure 3.3 is depicted in Figure 3.4. Note that $\mathcal{B} - S^\star$ does not contain any rigid subsets, i. e., is maximal. In Figure 3.5, a maximum flow for this network is depicted and the maximum rigid subset is highlighted. A closer look reveals that this (and any other) maximum flow f satisfies the demand of all bodies in the rigid subset, whereas for every subgraph S that cannot be merged with

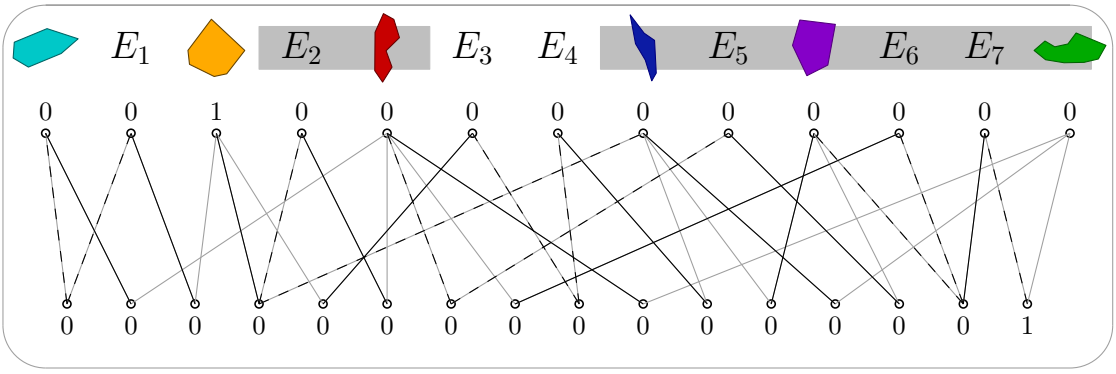


Figure 3.5: *Maximum flow in the intersection network from Figure 3.4. Solid black lines denote a flow of* 2*, dashed lines a flow of* 1*. A maximum closed and saturated set of bodies is highlighted.*

other bodies to form a new rigid subgraph, there is a maximum flow f that does leave some demand $b_f(S)$. We will formalize this observation with the following lemmas. To ease notation, we will in the following extend any flow f to arbitrary pairs $(v,S) \in R(\mathcal{B}) \times \mathcal{B}$ by setting $f(v,S) := 0$ for $(v,S) \notin \mathcal{I}_\mathcal{B}$.

Definition 3.7 *Let $\mathcal{B}$ be a Laman partition, $S^\star \in \mathcal{B}$ such that $\mathcal{B} - S^\star$ is a maximal Laman partition. Let f be any flow in $\mathcal{N}(\mathcal{B}, S^\star)$. The* support *of $\mathcal{B}'$, $V_f(\mathcal{B}')$ is defined as the set of nodes in $R(\mathcal{B})$ having a non-zero flow to $\mathcal{B}'$, i. e.,*

$$V_f\left(\mathcal{B}'\right) := \left\{ v \in R(\mathcal{B}) \mid \sum_{S \in \mathcal{B}'} f(v,S) > 0 \right\} \quad , \tag{3.11}$$

the remaining nodes by $V_{\not f}\left(\mathcal{B}'\right)$ as

$$V_{\not f}\left(\mathcal{B}'\right) := \left\{ v \in R(\mathcal{B}) \mid \sum_{S \in \mathcal{B}'} f(v,S) = 0 \right\} \quad . \tag{3.12}$$

A set $\mathcal{B}' \subset \mathcal{B}$ is called saturated *if $\sum_{v \in R(\mathcal{B})} f(v,S) = b(S)$ for all $S \in \mathcal{B}'$ and* closed *if for every v and every pair $S, S' \ni v$*

$$S \in \mathcal{B}' \wedge S' \notin \mathcal{B}' \Rightarrow f(v,S) = 0 \vee f(v,S') = 2 \quad . \tag{3.13}$$

The (minimal) closure of any $\mathcal{B}' \subset \mathcal{B}$ is denoted by $[\mathcal{B}']$. Analogously, the closure of a set of nodes $V' \subset V(\mathcal{B})$ is defined as $[V'] := [\{S \in \mathcal{B} \mid \exists (v,S) \in \mathcal{I}_\mathcal{B} : f(v,S) < 2\}]$.

From the above definitions, it is clear that the union of closed sets is also closed, and the union of saturated sets is also saturated. We are interested in closed and saturated sets, which are saturated by any maximum flow f. They are is inclusion-maximal if and only if they contain all closed and saturated sets, and hence also have maximum cardinality. In Figure 3.5, such a maximum set $\mathcal{B}'$ is marked. It is easy to identify by removing first all

bodies that are not saturated, and then iteratively all bodies which violate Equation 3.13. If there is no nontrivial closed and saturated set of bodies, this process would end with $\{S^\star\}$. In the example, a smaller closed and saturated set is $\mathcal{B}' - E_7$.

The following two lemmas ensure that for a maximum flow first, any saturated and closed set is rigid, and second, as long as a non-trivial rigid set is contained, there has to be a non-trivial saturated and closed set.

Lemma 3.8 *Let $\mathcal{N}(\mathcal{B}, S^\star)$ be a rigidity network and f any valid flow. For any set $\mathcal{B}' \subset \mathcal{B}$ the following properties hold:*

1. *The flow to subgraphs $\mathcal{B}'$ is*

$$\sum_{v \in R(\mathcal{B}), S \in \mathcal{B}'} f(v,S) \geq 2\mathtt{rd}_{\mathcal{B}'}\left(V_f\left(\mathcal{B}'\right)\right) - b_f\left(V_f\left(\mathcal{B}'\right)\right) \quad . \tag{3.14}$$

   This equation is tight if and only if $\mathcal{B}'$ is closed under f.

2. *The flow to subgraphs $\mathcal{B}'$ is*

$$\sum_{v \in R(\mathcal{B}), S \in \mathcal{B}'} f(v,S) \leq \begin{cases} 3|\mathcal{B}'| & \text{if } S^\star \notin \mathcal{B}' \\ 3(|\mathcal{B}'|-1) & \text{if } S^\star \in \mathcal{B}' \end{cases} \tag{3.15}$$

   This equation is tight if and only if $\mathcal{B}'$ is saturated under f.

3. *The graph $S^\star$ is contained in any closed and saturated set $\mathcal{B}'$.*

4. *For any $\mathcal{B}' \ni S^\star$, we have*

$$\mathtt{sp}\left(\mathcal{B}'\right) \leq b_f\left(V_f\left(\mathcal{B}'\right)\right) + 2\mathtt{rd}_{\mathcal{B}'}\left(V_f\left(\mathcal{B}'\right)\right) \quad . \tag{3.16}$$

   This equation is tight if $\mathcal{B}'$ is closed and saturated and strict otherwise.

5. *Any closed and saturated set $\mathcal{B}'$ is rigid.*

Proof. We prove these properties one at a time:

1. First, the flow for arcs (v,S) for $v \in V_f\left(\mathcal{B}'\right)$ and $S \notin \mathcal{B}'$ is $f(v,S) \leq 2$, and this equation is tight if and only if $\mathcal{B}'$ is closed. Hence,

$$\begin{aligned} \sum_{v \in R(\mathcal{B}), S \in \mathcal{B}'} f(v,S) &\geq \sum_{v \in V_f(\mathcal{B}'), S \in \mathcal{B}} f(v,S) - 2\underbrace{\left|\left\{(v,S) \in \mathcal{I}_{\mathcal{B}} : v \in V_f(\mathcal{B}'), S \notin \mathcal{B}'\right\}\right|}_{=\mathtt{rd}_{\mathrm{B}}\left(V_f(\mathcal{B}')\right) - \mathtt{rd}_{\mathrm{B}'}\left(V_f(\mathcal{B}')\right)} \\ &= 2\mathtt{rd}_{\mathcal{B}}\left(V_f\left(\mathcal{B}'\right)\right) - b_f(V_f\left(\mathcal{B}'\right)) - 2\mathtt{rd}_{\mathcal{B}}\left(V_f\left(\mathcal{B}'\right)\right) + 2\mathtt{rd}_{\mathcal{B}'}\left(V_f\left(\mathcal{B}'\right)\right) \\ &= 2\mathtt{rd}_{\mathcal{B}'}\left(V_f\left(\mathcal{B}'\right)\right) - b_f\left(V_f\left(\mathcal{B}'\right)\right) \quad , \end{aligned} \tag{3.17}$$

   and again, this equation is tight if and only if $\mathcal{B}'$ is closed under f.

2. Follows directly from the definition of b.

3. Let $\mathcal{B}'$ be a closed and saturated set. Hence, applying Equation 3.15 as

$$\sum_{v \in R(\mathcal{B}), S \in \mathcal{B}'} f(v,S) \geq 3(|\mathcal{B}'|-1) \ , \tag{3.18}$$

we have from Lemma 3.4

$$\begin{aligned} \mathtt{sp}\left(\mathcal{B}'\right) &= 2\mathtt{rd}\left(\mathcal{B}'\right) - 3(|\mathcal{B}'|-1) \\ &= 2\mathtt{rd}_{\mathcal{B}'}\left(V_f\left(\mathcal{B}'\right)\right) + 2\mathtt{rd}_{\mathcal{B}'}\left(V_{\not f}\left(\mathcal{B}'\right)\right) - 3(|\mathcal{B}'|-1) \\ &\overset{3.14}{=} \textstyle\sum_{v \in R(\mathcal{B}), S \in \mathcal{B}'} f(v,S) + b_f\left(V_f\left(\mathcal{B}'\right)\right) + 2\mathtt{rd}_{\mathcal{B}'}\left(V_{\not f}\left(\mathcal{B}'\right)\right) - 3(|\mathcal{B}'|-1) \\ &\overset{3.15}{\geq} b_f\left(V_f\left(\mathcal{B}'\right)\right) + 2\mathtt{rd}_{\mathcal{B}'}\left(V_{\not f}\left(\mathcal{B}'\right)\right) \ . \end{aligned} \tag{3.19}$$

Hence, $\mathtt{sp}(\mathcal{B}') \geq 0$ and $\mathcal{B}'$ at least contains a rigid subset, which then must contain $S^\star$.

4. If $S^\star \in \mathcal{B}'$ Equation 3.19 becomes

$$\begin{aligned} \mathtt{sp}\left(\mathcal{B}'\right) &= 2\mathtt{rd}\left(\mathcal{B}'\right) - 3(|\mathcal{B}'|-1) \\ &= 2\mathtt{rd}_{\mathcal{B}'}\left(V_f\left(\mathcal{B}'\right)\right) + 2\mathtt{rd}_{\mathcal{B}'}\left(V_{\not f}\left(\mathcal{B}'\right)\right) - 3(|\mathcal{B}'|-1) \\ &\overset{3.14}{\leq} \sum_{(v,S): S \in \mathcal{B}'} f(v,S) + b_f\left(V_f\left(\mathcal{B}'\right)\right) + 2\mathtt{rd}_{\mathcal{B}'}\left(V_{\not f}\left(\mathcal{B}'\right)\right) - 3(|\mathcal{B}'|-1) \\ &\overset{3.15}{\leq} b_f\left(V_f\left(\mathcal{B}'\right)\right) + 2\mathtt{rd}_{\mathcal{B}'}\left(V_{\not f}\left(\mathcal{B}'\right)\right) \ . \end{aligned} \tag{3.20}$$

Since both, Equation 3.14 and 3.15 are tight for a closed and saturated $\mathcal{B}'$, this also holds for Equation 3.20.

5. Let $\mathcal{B}' \subset \mathcal{B}$ be closed and saturated under f. With Lemma 3.5, it is sufficient to show that for all $\mathcal{B}'' \subset \mathcal{B}'$ with $S^\star \in \mathcal{B}''$ the inequality $\mathtt{sp}(\mathcal{B}') \geq \mathtt{sp}(\mathcal{B}'')$ holds. We get this from

$$\begin{aligned} \mathtt{sp}\left(\mathcal{B}'\right) &= b_f\left(V_f\left(\mathcal{B}'\right)\right) + 2\mathtt{rd}_{\mathcal{B}'}\left(V_{\not f}\left(\mathcal{B}'\right)\right) \\ &= b_f(V_f\left(\mathcal{B}''\right)) + b_f(V_f\left(\mathcal{B}'\right) \cap V_{\not f}\left(\mathcal{B}''\right)) + 2\mathtt{rd}_{\mathcal{B}'}\left(V_{\not f}\left(\mathcal{B}'\right)\right) \\ &\geq b_f(V_f\left(\mathcal{B}''\right)) + 2\mathtt{rd}_{\mathcal{B}''}\left(V_f\left(\mathcal{B}'\right) \cap V_{\not f}\left(\mathcal{B}''\right)\right) + 2\mathtt{rd}_{\mathcal{B}''}\left(V_{\not f}\left(\mathcal{B}'\right)\right) \\ &= b_f\left(V_f\left(\mathcal{B}''\right)\right) + 2\mathtt{rd}_{\mathcal{B}''}\left(V_{\not f}\left(\mathcal{B}''\right)\right) \\ &\geq \mathtt{sp}\left(\mathcal{B}''\right) \ , \end{aligned} \tag{3.21}$$

using $\mathtt{rd}_{\mathcal{B}'}(v) \geq \mathtt{rd}_{\mathcal{B}''}(v)$ for all v and $b_f(v)) \geq 2\mathtt{rd}_{\mathcal{B}''}(v)$ for $v \in V_{\not f}(\mathcal{B})$.

□

Lemma 3.9 *Let $\mathcal{B}$ be a Laman partition, $S^\star$ such that $\mathcal{B} - S^\star$ is maximal. If $\mathcal{B}' \subset \mathcal{B}$ is an inclusion-maximal rigid subset, then for any maximum flow in $\mathcal{N}(\mathcal{B}, S^\star)$, $\mathcal{B}'$ is saturated and closed.*

Algorithm 3.2: *MergeRigidComponents*($\mathcal{I}_\mathcal{B}$)

```
    B^I ← ∅
    while B ≠ ∅ do
        choose S* from B
        B ← B − S*
A       while ∃S ∈ B^I : |V(S*) ∩ V(S)| > 1 do
            B^I ← B^I − S
            S* ← G({S, S*})
B       f ← maximum flow in B(B^I ∪ {S*}, S*)
        B' ← inclusion-maximal closed and saturated set with respect to f
        if |B'| > 1 then S* ← G(B')
        B^I ← B^I \ B'
        B^I ← B^I ∪ {S*}
    return B
```

Proof. Let $\mathcal{B}'$ be an inclusion-maximal non-trivial rigid subset of $\mathcal{B}$. As all rigid subsets overlap in $S^\star$, $\mathcal{B}'$ is well-defined as the union of all rigid subsets of $\mathcal{B}$. Suppose, $\mathcal{B}'$ was not closed and saturated with respect to a maximum flow f. Since it is rigid, we have

$$b_f\left(V_f\left(\mathcal{B}'\right)\right) + 2\mathtt{rd}_{\mathcal{B}'}\left(V_{\not f}\left(\mathcal{B}'\right)\right) > \mathtt{sp}\left(\mathcal{B}'\right) \geq 0 \ . \tag{3.22}$$

Hence, there is a node $v \in V(\mathcal{B}')$ with $b_f(v) > 0$. Let V_x denote all these nodes and $\mathcal{B}'' := [V_x]$. As f is maximal, the closure $\mathcal{B}''$ is saturated, closed, non-trivial and, by Lemma 3.8, we also know that $\mathcal{B}'' \subset \mathcal{B}'$. We now have by the choice of $\mathcal{B}''$

$$b_f(V(\mathcal{B})) = b_f(V_f\left(\mathcal{B}''\right)) + 2\mathtt{rd}_{\mathcal{B}''}\left(V_{\not f}\left(\mathcal{B}''\right)\right) \overset{3.16}{=} \mathtt{sp}\left(\mathcal{B}''\right) \ , \tag{3.23}$$

but

$$\mathtt{sp}\left(\mathcal{B}''\right) \overset{Lem.\,3.5}{\leq} \mathtt{sp}\left(\mathcal{B}'\right) \overset{3.16}{<} b_f(V_f\left(\mathcal{B}'\right)) + 2\mathtt{rd}_{\mathcal{B}'}\left(V_{\not f}\left(\mathcal{B}'\right)\right) \leq b_f(V(\mathcal{B})) \ , \tag{3.24}$$

which yields a contradiction. □

Together, these two lemmas are the foundation for our algorithm that finds maximum rigid components starting with an arbitrary Laman partition $\mathcal{B}$. It is depicted in Algorithm 3.2: One by one, subgraphs are added to some set $\mathcal{B}^I$, and every time a new subgraph is added, it is repeatedly tested whether the new subgraph can be merged with any of the previously added subgraphs. Then, a maximum flow in the incidence network is used to identify larger sets of subgraphs that can be merged.

Theorem 3.10 *Algorithm 3.2 returns the unique Laman partition into maximum rigid subgraphs. It can be implemented to run in* $O(m_\mathcal{B} \log m_\mathcal{B} \log \Delta + k_\mathcal{B}^2)$.

Proof. First, the algorithm maintains the invariant that $\mathcal{B}^I$ is independent: Before adding a graph $S^\star$ is added to $\mathcal{B}^I$, we find a maximum rigid subset of $\mathcal{B}^I \cup \{S^\star\}$, remove the involved graphs from $\mathcal{B}^I$ and add the graph formed by them to $\mathcal{B}^I$. For this to hold (and thus for the correctness of the algorithm), we do not need the steps marked with 'A', which will play their role in the runtime analysis.

Second, Algorithm 3.2 can be implemented to run in $O(m_\mathcal{B} \log m_\mathcal{B} \log \Delta + k_\mathcal{B}^2)$ as follows: Without loss of generality, we assume that node-body incidences in $\mathcal{I}_\mathcal{B}$ are pairs of integers in $[1, n_\mathcal{B}] \times [1, k_\mathcal{B}]$ for $n_\mathcal{B} := |R(\mathcal{B})| < m_\mathcal{B}$. If the nodes in $R(\mathcal{B})$ and $\mathcal{B}$ have any total order, conversion can be done in $O(m_\mathcal{B} \log m_\mathcal{B})$. Now, each subgraph S is annotated with b, initially 0, and two lists of arcs, one for *active* arcs with respect to the current $\mathcal{B}^I \cup S^\star$, i. e., arcs from nodes that currently have $b(v) > 0$ and one for arcs from other nodes. Initially, all arcs to some subgraph are put in the list of inactive arcs. Each $v \in R(\mathcal{B})$ is also annotated with $b(v)$, initially 0, and an *ordered* list of arcs to bodies in $\mathcal{B}^I \cup S^\star$, initially also empty. This set-up can be done in linear time. Now, when some $S^\star$ is chosen, we first update this network. All arcs to $S^\star$ are put in the list of arcs of the respective node $v \in R(\mathcal{B})$. It is moved to $S^\star$'s list of active arcs only if v's list was not empty. In this case, also $b(v)$ is increased by 2. If v's list contains a single arc (v, S), then this arc is moved from the list of inactive arcs to the list of active arcs at S. These steps take amortized $O(m_\mathcal{B} \log \Delta)$ steps. Now, we can detect whether there is some $S \in \mathcal{B}^I$ with $|V(S^\star) \cap V(S)| > 1$ in time $O(k)$ enumerating and marking the neighbors of all nodes v with $(v, S^\star)$ in $S^\star$'s list of active arcs. Each of these arcs is part of at least one subgraph $S \neq S^\star$, and it is impossible to mark more than k subgraphs without marking one twice. In this case, this subgraph is merged with $S^\star$ as described below. The test for a rigid subset of size 2 in $\mathcal{B}' \cup \{S^\star\}$ can at most be performed $k - 1$ times succeeding (reducing the number of subgraphs) and at most k times fail (once for every $S^\star$), incurring a total cost of $O(k^2)$.

To merge two subgraphs, $S^\star$ and S', we process the arcs of the subgraph with less arcs in its lists, without loss of generality S'. This way, each arc can at most be processed $\log m_\mathcal{B}$ times. We put these arcs into the other subgraph's respective list, changing the tuple describing the arc accordingly. Each arc (v, S') also has to be deleted and reinserted to v's (ordered) list as $(v, S^\star)$. Together, these operations take amortized time $O(m_\mathcal{B} \log m_\mathcal{B} \log \Delta)$. They also allow us to identify arcs that occur twice. Such arcs are discarded.

For further analysis, we observe that we only have $O(k)$ active arcs in our network in part 'B': Less than $3/2|\mathcal{B}^I| - 3/2$ of them can have more than two arcs to graphs in $\mathcal{B}^I$, as every such node v has $\mathtt{rd}_{\mathcal{B}^I}(v) > 0$ and $\mathcal{B}^I$ is independent, and no k arcs to $S^\star$ can be active without any $S \in \mathcal{B}^I$ sharing two or more nodes with $S^\star$.

After flow maximization, which we analyze last, we can identify a maximum closed and saturated set of bodies by removing all non-saturated bodies in the network and repeatedly shrink this set to a closed set. This can be done in $O(k)$ time. Then, we successively merge bodies within that set with $S^\star$. Naively implemented, this could lead to a complexity of $\Theta(k^2)$ per solved maximum flow problem, but we can re-use flows as follows: Whenever we merged $S^\star$ with some other subgraph S, we reset the flow for arcs to S to zero. This can at most affect 3 arcs, which can be identified in constant time if we keep track of the nodes that have positive flow to bodies $S \in \mathcal{B}^I$. All other flows remain unchanged. Merging larger

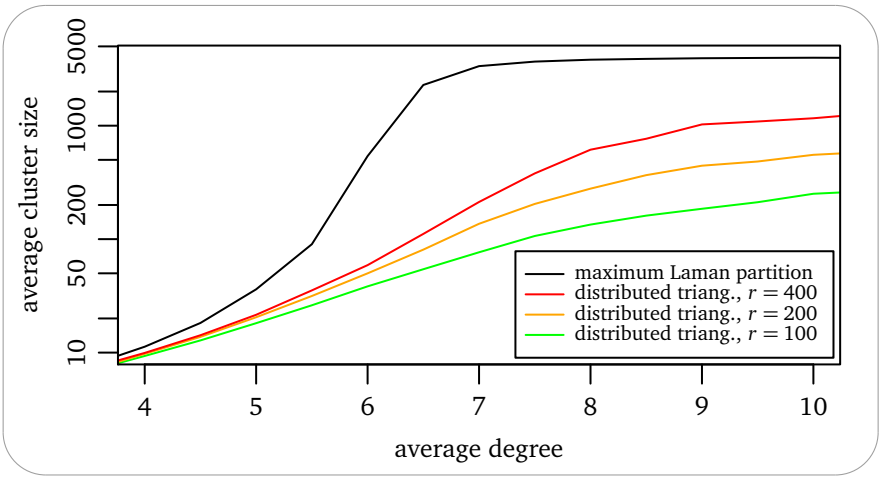


Figure 3.6: *Size of the largest body a node belongs to, averaged over all nodes.*

sets of bodies with $S^\star$ can be done iteratively. This way, the flow to all graphs in $\mathcal{B}^I$ does not decrease (S is removed when merged with $S^\star$) and as every subgraph can take at most a flow of 3, we can have at most $3k_\mathcal{B}$ successful increasing steps and $k_\mathcal{B}$ failing steps, each with a time-complexity in $O(k_\mathcal{B})$. □

In Figure 3.6, the average body size of the maximum Laman partition is compared to the Laman partitions produced by distributed triangulation for different numbers of rounds. The body size depicted is the average size of the largest body a node belongs to, averaged over all nodes to evaluate the knowledge of an average node about its position in the network. It shows the quality of both, distributed triangulation and central identification of maximum components: Distributedly, nodes can gain reasonable cluster sizes as soon as possible, but central processing significantly adds to this information. The noticeable increase of average body size for the maximum partition for degrees 6 to 7 marks the critical density for the giant component phenomenon, which also exists for rigid components. Beyond this density, in infinite random geometric graphs, there exists a so-called giant component covering an infinite number of nodes. This effect has also been observed in [EGW+04].

3.2.3 Positioning

The detection of maximum rigid components is not only of interest for knowing for which part of the network the LP provides us with dependable positions, but it also makes the whole LP-based approach obsolete. A closer look at the LP formulation reveals that valid edge lengths can also be described as $\ell \in \ker A \cap \mathbb{R}^2_+$ for some matrix A. For (parallely) rigid graphs that can be embedded with the given edge directions, we know that $\ker A$ is one-dimensional and hence, either x or $-x$ contains only positive edge lengths for any $x \in \ker A \setminus \{\mathbf{0}\}$. Hence, whenever we identify a set of k' rigid subgraphs, we only have to solve a homogeneous linear equation system with k' variables and $\Theta(k')$ equations. In the

worst-case, we have to merge all subgraphs at the same time, yielding a runtime of $\Theta(k_B^3)$. This is an upper bound since

$$\sum_{i=0}^{t} k_i^3 \leq \left(\sum_{i=0}^{t} (k_i - 1) + 1 \right)^3 , \tag{3.25}$$

where the left hand side is an upper bound for the complexity if we can solve t subproblems with $k_1, \ldots, k_t \geq 2$ bodies, each decreasing the number of remaining bodies by $(k_i - 1)$, whereas the right hand side corresponds to the complexity of solving the whole linear equation system—up to constant factors. Although in worst-case, computation of positions takes time $\Theta(k_B^3)$, this decreases to $\Theta(k)$ for solving $\Theta(k)$ problems involving a constantly bounded number of components to be merged. Not surprisingly, in our experiments only the latter case occurred, usually for very small bounds, making the additional costs for layout calculation negligible.

3.3 Hardness of Noisy Measurements

Many realization problems have been proven to be NP-hard, even disregarding potential errors in the input, which are simply unavoidable with real sensor nodes. Sadly, in the face of erroneous measurements, even the simplest realization problem with distances and directions as input becomes NP-hard in the presence of arbitrarily small errors. This has been proven for the case that directions are measured globally by Basu et al. [BGMS06]. We here add the proof for local measurements:

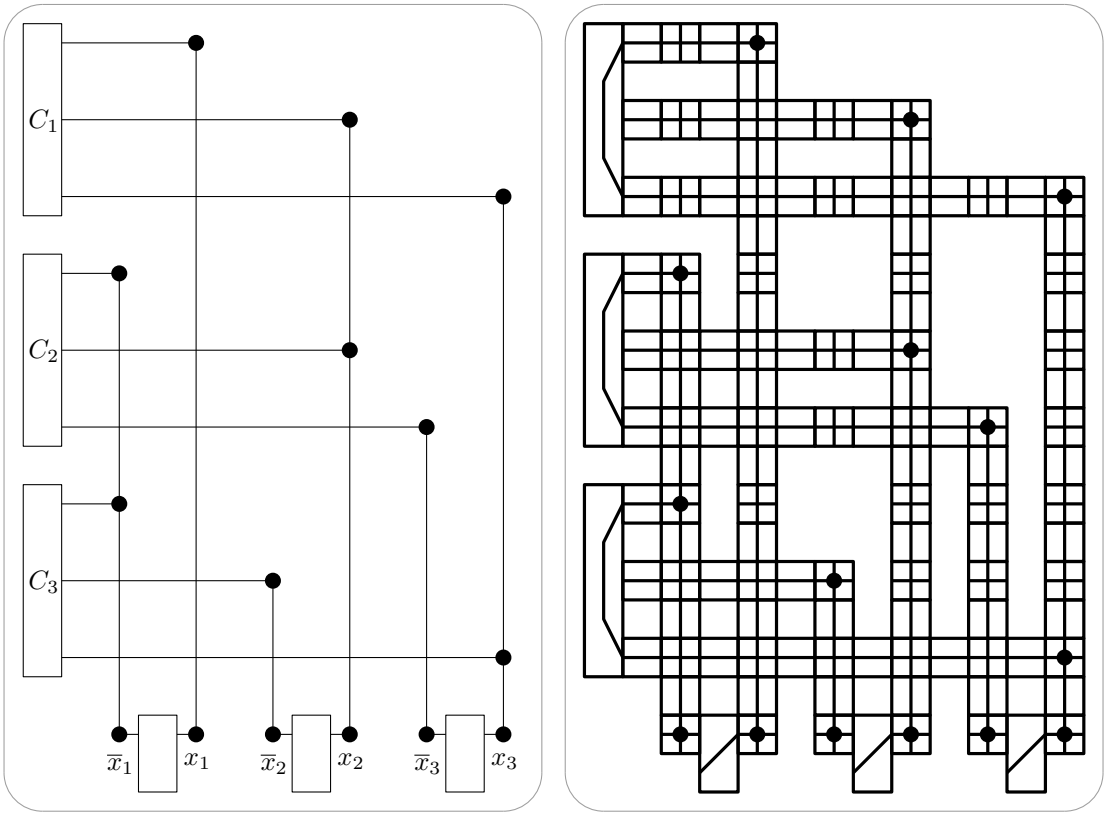
Problem 3.11 (Error-Realization) *Given a graph $G = (V, E)$, edge lengths $d : E \to \mathbb{R}_+$, relative edge directions $\omega : E \to [0, 2\pi)^2$ and small $\epsilon, \delta > 0$, is there an embedding $\mathbf{p} : V \to \mathbb{R}^2$ and an orientation $\mathbf{o} : V \to [0, 2\pi)$, such that for all u, v with $\{u, v\} \in E$*

$$\frac{d_{\mathbf{p}}(u, v)}{d(u, v)} \in [1 - \epsilon, 1 + \epsilon] \quad \textit{and} \quad \omega_{\mathbf{p},\mathbf{o}}(u, v) - \omega(u, v) \in [-\delta, \delta] \mod 2\pi \ ?$$

In the following, we prove that it is NP-hard to find an embedding and an orientation of the nodes such that measured distances and angles do not differ from the ones induced by these two by more than given, arbitrarily small factors and angles, respectively.

Theorem 3.12 *Error-Realization is NP-hard even for fixed, arbitrarily small error bounds ϵ, δ.*

Proof. We prove this theorem by a reduction from 3-Sat. Given an instance of 3-Sat, we draw the corresponding instance canonically as shown in Figure 3.7 with the building blocks, i. e., variables, wires, crossings connectors and clauses. From this drawing, we derive an input to the Error-Realization as follows.



(a) An instance of 3-Sat as schematic drawing

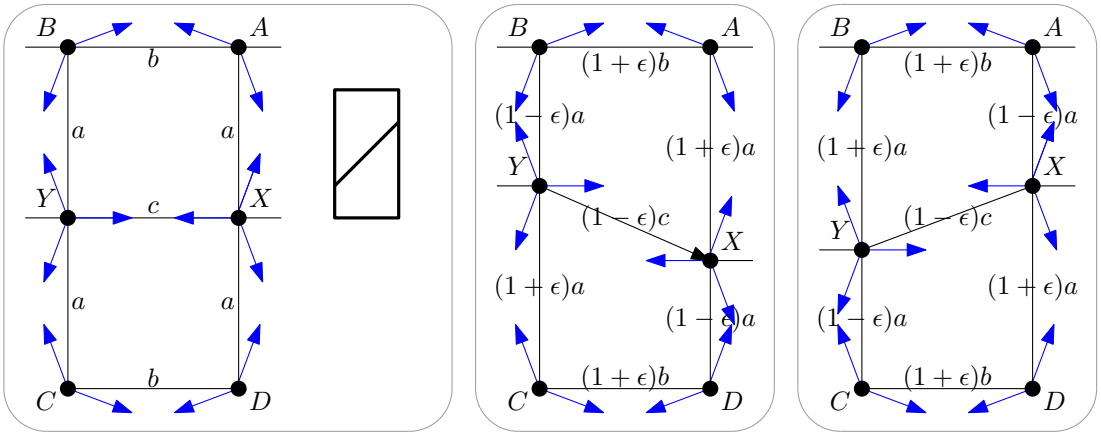
(b) An instance of 3-Sat built from basic building blocks

Figure 3.7: *Drawings of the 3-Sat instance* $(x_1 \vee x_2 \vee x_3) \wedge (\overline{x}_1 \vee x_2 \vee \overline{x}_3) \wedge (\overline{x}_1 \vee \overline{x}_2 \vee x_3)$ *as graph (a) and in a grid-like fashion with building blocks, i. e., variables, horizontal and vertical wires, crossings, connections and clauses (b)*

First, we observe how to design an input to the Error-Realization problem that forces fixed angles in any valid embedding. Let a graph G have a cycle of nodes we want to be realized as a polygon with prescribed angles. To this extent, we choose the input directions ω to point all outward (or all inward) by an angle of δ with respect to an arbitrary embedding and orientation of G that realizes the prescribed angles (see Figure 3.8(a)). Now, every valid embedding of G that differs from ω by no more than δ for any edge-node incidence, embeds the cycle as a polygon with the prescribed angles.

Based on this argument, we construct variables by building a rectangle of six nodes, four nodes, A to D in the corners and one in each long edge's center, X and Y as in Figure 3.8(a). We fix input directions for this cycle of six nodes to all point outward by δ and introduce an additional inner edge YX. We also assign lengths to each edge, namely $a, b, c \in \mathbb{R}_+$, such that

$$a = c \cdot \frac{(1-\epsilon)\sin(\delta)}{2\epsilon} \quad \text{and} \quad b = c \cdot \frac{(1-\epsilon)\cos(\delta)}{(1+\epsilon)} \ . \tag{3.26}$$



(a) input (b) "false" realization (c) "true" realization

Figure 3.8: *Variable for 3-Sat reduction*

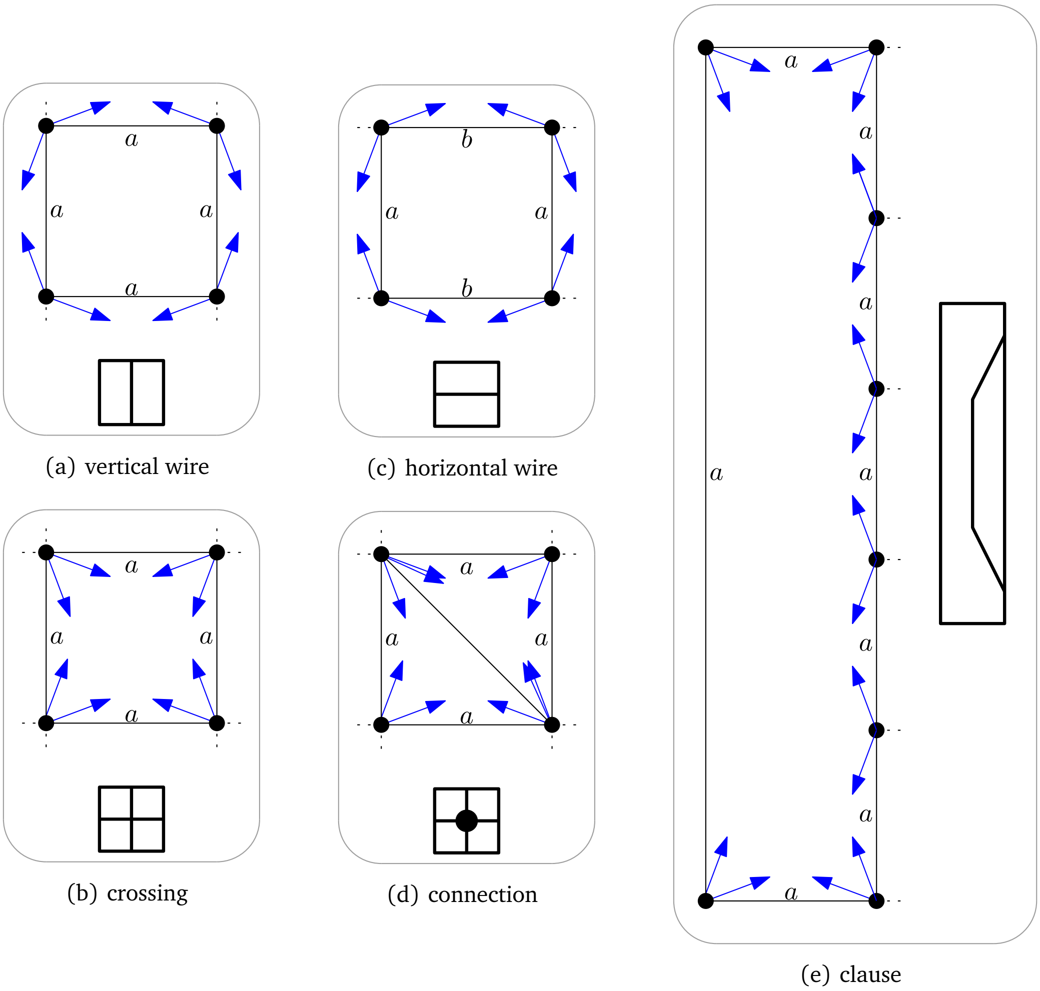
With these values, even $(1+\epsilon)b$ being slightly smaller than $(1-\epsilon)c$, YX cannot be realized to be parallel to BA and CD. With the input directions fixed as in Figure 3.8(a), it can deviate by at most δ in either direction, which, by the choice of b and c is just enough. In the resulting drawings, the edges with input length a are all maximally stretched or compressed, leaving no freedom but to choose between one of the two embeddings depicted in Figure 3.8(b) and 3.8(c). We will call the embedding where AX has an edge length of $(1+\epsilon)a$ and BY has length $(1-\epsilon)$ a *false* assignment, and the other one, where these two lengths are swapped a *true* assignment. Wiring, connecting and crossing is comparably easy. The gadgets are shown in Figures 3.9(a) to 3.9(d). Essentially, they are all rectangles with fixed angles, which therefore must have the same lengths for opposite sides. Variables and wires have all input directions pointing outward, while in crossings and connectors, input directions point inward by δ. These two kinds of gadgets are put together alternately (cf. Figure 3.7(b)). Note that all rows have heights $\approx a$, but the width of columns is $\approx a$ only for columns corresponding to literals, whilst it is necessary for the other columns containing horizontal links to have width $\approx b$ like variables.

The last gadget is a clause, shown in Figure 3.9(e). If we choose x such that

$$(1-\epsilon)x = (3(1+\epsilon)+2(1-\epsilon))\,a \Leftrightarrow x = a\frac{5+\epsilon}{1-\epsilon}\ , \tag{3.27}$$

a clause can only be drawn in a way, such that at most two edges on the left have lengths $(1-\epsilon)a$, i. e., if at least one connected literal is *true*, while more are always possible.

On the other hand, it is easy to see how a valid assignment can be transformed into a valid embedding: Rows that correspond to an occurrence of a literal have the respective height, the other rows have height $(1+\epsilon)a$. Columns that correspond to a literal again have the width induced by the literal's value, other columns have width $(1+\epsilon)b$. □



(a) vertical wire
(c) horizontal wire
(b) crossing
(d) connection
(e) clause

Figure 3.9: *Gadgets for 3-Sat reduction*

Chapter 4

Filtration-based Positioning

This chapter deals with positioning from a more practical point of view. We develop a multi-level-framework, that, instantiated with a proper technique for the positioning of small networks bridges between good, but complex, positioning algorithms for small-scale networks and the demand for a scalable and distributed algorithm for large networks with irregular shapes. A particular instantiation, MDS-MAP(F), is evaluated in challenging scenarios and compared to state-of-the-art competitors.

4.1 Problem Statement and Related Work

Hardness proofs are of undisputed theoretical interest, but they contrast with the multitude of algorithms that have been developed for positioning so far. We are only aware of a single polynomial-time approximation algorithm, producing an $\Omega(\log^{2.5} n\sqrt{\log\log n})$-QUDG embedding for a given UDG [MOWW04]. Apart from that, there exist a lot of positioning algorithms that are heuristics, following very different approaches, leading to very different achievements. Again, we are only considering anchor-free positioning, leaving out a zoo of results for anchor-based positioning.

First, the positioning problem has been tackled directly using some more complex optimization techniques, such as semidefinite programming (SDP) for range-based positioning [BY04, DPG01], assuming comparably high quality and number of measurements. For connectivity-based positioning, MDS proved to perform very well on small-scale networks, also named MDS-MAP [SRZF03]. There have also been efforts to adapt MDS to estimate anisotropy in wireless propagation [JZ04]. The usual way to use these techniques in the context of larger networks is by patching local solutions together, sometimes combined

with local refinement. This has been evaluated for MDS as MDS-MAP(P) (for "patch") and MDS-MAP(P,R) (for "patch and refine") [SR04] and also in a distributed fashion as MDS-MAP(D) [SRF06]. However, these techniques fail for larger networks in which the errors introduced by patching operations sum up. All of the aforementioned techniques are typically evaluated in networks with less that 250 nodes.

Many approaches for large-scale networks employ a heuristic to find a good initial solution to be improved by local optimization, typically some kind of mass-spring relaxation. The most important feature of such an initial solution is to be *folding-free*, i. e., not to have different parts of the networks folded over each other. Examples are AFL [PBDT03] and EIGEN [GK04], both working completely distributed. The former initializes the position of 5 cleverly chosen nodes to form the center and the corners of a square and positions the rest of the nodes according to their distances to these virtual anchors. The latter implements a distributed variant of spectral embedding based on Eigenvalues of an adjacency-based matrix. Both approaches are very sensitive to the shape of the network and fail if it is not deployed in a convex area. This issue has more recently been addressed by Lederer et al. [LWG08], who successfully employed high-level topology information for the positioning of convex-shape networks: In a first step, boundary nodes and nodes on the medial axis are identified, then landmark nodes on the boundary are chosen according to the local distance between boundary and the medial axis. For those landmark nodes, the Delaunay complex is computed, which then can be used to embed landmark nodes folding-free, to use local optimization and to place the remaining nodes between the landmarks.

In the following, we will propose a distributed multi-level framework for the positioning of nodes, replacing the patching techniques mentioned above. This framework, instantiated with MDS to position local neighborhoods, is able to position large-scale networks with higher accuracy than previous approaches, robust to irregular communication ranges, and can also be applied if the sensors are deployed in a field that does not have a clear boundary or is three-dimensional.

4.2 Generic Framework

Rather than being overly distributed, the general idea of hierarchical positioning is to aggregate the information from local-range positioning. To this extend, local solutions are not patched together, but provide estimations for relative positions between a sample of the nodes, which are then taken as input of a recursive process. This approach is based on the technique of graph filtration and multi-scale layout, which has quite a tradition in graph drawing (see, e. g. [GGK04]). Graph filtration here denotes the process to successively restrict a graph to a fraction of nodes, while adding edges between nodes in increasing distance.

Definition 4.1 (Graph filtration) *Given a graph $G = (V, E)$, a* filtration *is a sequence $G = G_0, G_1, \ldots, G_h$ with $G_i = (V_i, E_i)$, such that $V = V_0 \supset V_1 \supset \cdots \supset V_h = \{\hat{v}\}$.*

Algorithm 4.1: *GenericMultiLevelPositioning*$(G = (V, E), \lambda)$

forall $v \in V$ **do**
  $\mathbf{p}_v \leftarrow$ *SolveCentrally*$(G[N_G^k(v)], \lambda)$
  if $N_G^k(v)) = V$ **then return** $\mathbf{p}_v$
select a subset $V' \subset V$ such that for every $u \notin V'$, $N_G^1(u) \cap V' \neq \emptyset$
select $E' \subset \{(u, v) \mid d_G(u, v) \leq k\}$
forall $\{u, v\} \in E$ **do** $\lambda'(u, v) \leftarrow \lambda_{\mathbf{p}_v}(u, v)$
$\mathbf{p}' \leftarrow$ *GenericMultiLevelPositioning*$(G' = (V', E'), \lambda')$
forall $u \in V$ **do**
  pick some $v \in N_G^1(u) \cap V'$
  $\mathbf{p}(u) \leftarrow \mathbf{p}'(v) \oplus \mathbf{p}_v(u)$
return p

In graph drawing, this technique is used to simplify the input and the layout is computed in a top-down fashion: Given a layout for the nodes in V_i, the nodes from $V_{i-1} \setminus V_i$ are placed according to some good guess based on the positions of V_i. Then, the layout is improved by minimizing some stress function, before nodes from $V_{i-2} \setminus V_{i-1}$ are re-inserted, and so on. This approach has been used for sensor network positioning, albeit not for connectivity-based localization, but for input consisting of both, noisy distances and directions [EEF+06]. For the initial placement, distances within the network are deduced from dead reckoning, a technique whose quality also decreases with the network size.

In contrast, we perform most of the optimization in a bottom-up manner: We let all nodes localize some small neighborhood, and select a sample of nodes to be active in the next level of filtration. In this level, close selected nodes are connected. They derive relative position information from local solutions. Thereby, with progressing filtration, nodes position sparser samples of neighborhoods with increasing range.

A more formal description of this approach is given in Algorithm 4.1 in a sequential manner: Given a connected Graph $G = (V, E)$ together with some kind of information on relative positions, λ, first, each node's k-hop neighborhood is positioned using a centralized algorithm. Here, λ can be any kind of information regarding a connected pair of nodes that is invariant under isometries of the underlying space, e. g., distance, relative direction a relative sector, or combinations, subject to discretization or noise. Second, a subset of the nodes $V' \subset V$ is chosen. Among pairs that mutually lie within k-hop neighborhood, edges E' are chosen from

$$E' \subset \left\{ \{u, v\} \in \binom{V'}{2} \mid u \in N_G^k(v) \right\} ,$$

and information on relative positions λ' is derived from local solutions. We will examine conditions on the selection of V' and E' later in this chapter. This scheme is recursively applied to $G' = (V', E')$ with the support of λ'. As soon as at some level, all active nodes are within some node's k-hop neighborhood, this node's positioning is returned. Given positions for nodes on any level, V', the rest of the nodes, $V \setminus V'$ are positioned combining the known

Algorithm 4.2: *DistributedHierarchicalPositioning*$(v, N_G(v), \lambda_{N_G(v)})$

```
ℓ ← 0
N_0 ← N_G(v), λ_0 ← λ_{N_G(v)}
active ← true
while active do
    exchange information with nodes in N_ℓ^k
    p_{ℓ,v} ← SolveCentrally(G[N_ℓ^k])
    if N_ℓ^k covers all nodes then
        choose a leader among nodes with this property and set its position to 0
    active ← NodeFiltration()
    exchange information with nodes in N_ℓ^k
    if active then
        select N_{ℓ+1} among active nodes in N_ℓ^k
        store routes to N_{ℓ+1} in N_ℓ^k
        forall u ∈ N_{ℓ+1} do derive λ_{ℓ+1}(u) from p_{ℓ,v} and p_{ℓ,u}
        ℓ ← ℓ + 1
wait for some u ∈ N_ℓ to send a position
send positions to all nodes in ⋃_{i=0}^{ℓ-1} N_i
```

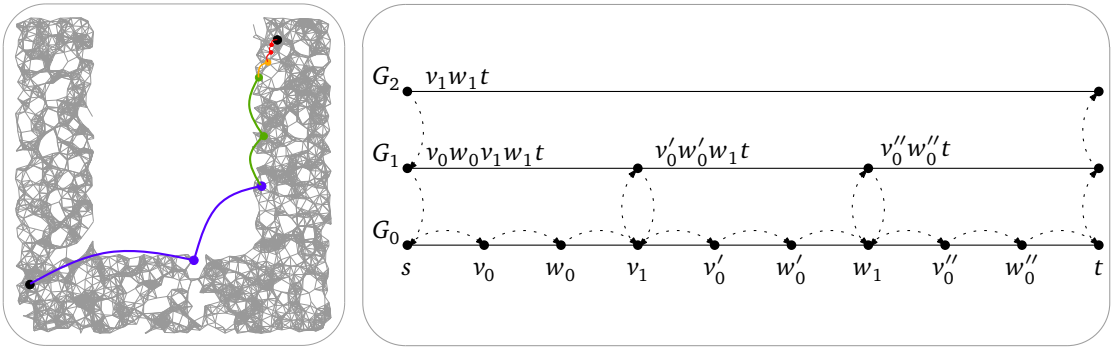
position of the closest node in V' and the respective local solution.

This framework leaves certain degrees of freedom open: First, nodes can position their neighborhoods using any suitable algorithm. This choice can depend on the kind of input λ available to the nodes as well as on the nodes' capabilities. Second, we have the choice of the filtration step, i. e., the choice of V' and E' with respect to the parameter k. The basic idea is to perform filtration steps in a way that the size of local problems does not increase with progressing filtration, and that second the recursion depth is logarithmic in the number of nodes.

4.2.1 Distributed Implementation

Algorithm 4.1 can completely be implemented in a distributed way if the filtration step can be performed distributedly; an outline is given in Algorithm 4.2. Starting with a setup, where every node v has position information $\lambda_{N_G(v)}$ for its one-hop neighbors $N_G(v)$, each node broadcasts its information to its k-hop neighborhood. Every node now computes positions for its k-hop neighborhood. If some node's neighborhood covers all remaining nodes, these nodes choose a leader and set its position to $\mathbf{0}$. Otherwise, nodes perform the filtration step, i. e., use some local protocol to decide which nodes remain active.

Again, we allow the exchange of information in the k-hop neighborhood to let active nodes know each other. If a node is chosen to be active on the next level, it selects its neighbors for the next level among active nodes in the current k-hop neighborhood. It also derives relative position information $\lambda_{\ell+1}(u)$ for each neighbor u on the next level from its local solution as well as a path in the current k-hop neighborhood to get there. Both, the



(a) Routing information in a one-hop transmission in G_4

(b) Insertion of waypoints in a transmission in G_2

Figure 4.1: *Routing in higher levels*

selection of next-level neighbors and extraction of $\lambda_{\ell+1}(\cdot)$ must be symmetric, i. e., have the same result for both nodes involved.

This procedure necessarily ends with one node being positioned at $\mathbf{0}$. To spread position information, the following is sufficient: As soon as a node knows a position, it sends packets assigning positions to its neighbors on all levels in which the node was active, based on its own position and the local solutions.

Note that on each level, each node has to communicate only a constant number of times with its k-hop neighbors on the respective level. Routing these packets also can exploit the filtration: To route a message over one hop in level ℓ, it is sufficient to have $(k-1)\ell+1$ IDs enclosed to route the packet in the underlying network: To send a message from some node $s \in V_\ell$ to a node $t \in N_\ell(s)$, the path in G_ℓ is $s -_0 t$, denoting a hop in level $(\ell - i)$ by $-_i$. Now, s replaces the first hop of the path adding the at most $k-1$ intermediate nodes, e. g., for $k = 3$ some v_1 and w_1, yielding $s -_1 v_1 -_1 w_1 -_1 t$. Again, the first hop in level $\ell - 1$ is replaced by the at most $k-1$ relay nodes on the next lower level, e. g., $s -_2 v_2 -_2 w_2 -_2 v_1 -_1 w_1 -_1 t$, and so on.

Repeatedly applying this scheme, we start a message knowing the first k hops in G_0, then the next $k-1$ hops in G_1, $k-1$ hops in G_2 and so on. Everytime the packet is passed to the next node, we can remove the respective "waypoint" from the list contained in the packet, and whenever a node $u \in V_\ell \setminus V_{\ell+1}$ is reached for some ℓ, the next hop in the packet is on level ℓ and u can fill in the partial routing information on the lower levels using its stored routes for its neighborhoods on these levels. The development of such a routing information is depicted in Figure 4.1. In Figure 4.1(a), the waypoints included in a packet travelling one hop in G_4 from the black node in the upper right to the black node in the lower right corner: Two hops in G_0 (red), one in G_1 (orange), two in G_2 (green), and two in G_3 (blue). In Figure 4.1(b) it is depicted how relaying nodes in G_1 fill in missing waypoints en route.

4.3 Filtration Techniques

The choice of the filtration technique is crucial, since consistency and the burden of single nodes heavily depend on the way nodes are chosen to be active and connected on each level:

Connectivity and consistency: The most important property that has to be guaranteed under all circumstances by the filtration technique is that the overlay graph G_ℓ remains connected on all levels (given the underlying network was connected). If this property was violated at some point, recursion would stop with more than one node assigning itself a position and propagating positions down the hierarchy, leading to inconsistent positions.

Size of local problems: The main idea of positioning local neighborhoods is to run complex positioning algorithms on small instances only, such that single nodes are able to perform them. Thus, it is important that even though in higher levels nodes become connected to further and further nodes, k-hop neighborhoods do not become too large.

Height of filtration hierarchy: Some nodes have to solve local problems on all levels, and also the other nodes may have to relay packets for communication on all levels. Hence, it is important to bound the height of the filtration hierarchy, i. e., the number of filtration steps.

In the following, we first propose a geometric filtration technique that guarantees connectivity and, given sufficient quality of local solutions, constant bounds on the size of local problems as well as a logarithmic height of the filtration hierarchy. Additionally, we propose a combinatorial technique, that can only guarantee a consistent assignment of positions, but empirically fulfills the above requirements much better than the geometric filtration technique.

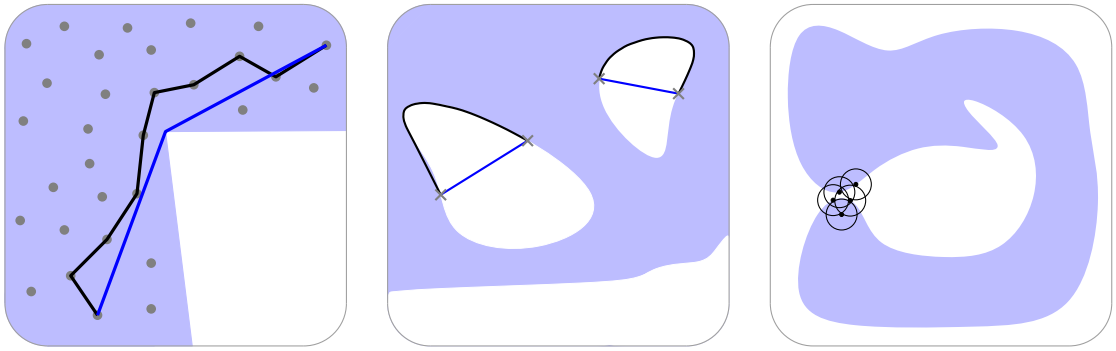
4.3.1 Geometric Filtration

In geometric filtration, the conditions determining which nodes to choose for the next filtration step rely on the local solutions. In the following, we show that geometric filtration yields consistent positions and derive bounds under the assumption that local solutions are perfect reconstructions. It is based on the geometric intuition of active nodes dominating an increasing range.

Definition 4.2 *Let $G = (V, E)$ be a graph in $\mathbb{R}^d$. A set of nodes $V' \subset V$ is r-$d^\star_G$-independent if for any two nodes $u, v \in V'$, $d^\star_G(u, v) > r$, where $d^\star_G(u, v)$ denotes the* shortest path distance *between u and v in G,*

$$d^\star_G(u, v) := \min\left\{ \sum_{\{a,b\}\in P} d(a, b) \mid P \text{ connects } u \text{ and } v \right\} \tag{4.1}$$

It also makes use of the following observations: If nodes are deployed densely enough in some area $A \subset \mathbb{R}^d$, shortest pathes in the network are only by a constant factor larger than shortest path distances in A (cf. Figure 4.2(a)). Also deployment regions can be described using a measure of independence: Given some radii $r_1, r_2 \in \mathbb{R}_+$, how many points can be



(a) shortest path in a dense graph (black) compared to shortest path in deployment area (blue)

(b) region A with $s_A = 2$

(c) A well-formed region with higher span

Figure 4.2: *Ingredients of geometric filtration*

chosen that have pairwise distance greater than r_1 in A, but are all covered by some ball of radius r_2 in $\mathbb{R}^d$? For $A = \mathbb{R}^d$ this obviously can be bounded by $c \cdot (r_2/r_1)^d$ for some small constant c, and this is also possible for many regions that are not arbitrarily complex. One example are regions where shortest paths have a length of at most some *span* s_A times the Euclidean distance of the endpoints. Then this constant is at most s_A^d times the constant for $\mathbb{R}^d$ (cf. Figure 4.2(b)). This also applies to convex regions A and $s_A = 1$. But also for other regions, in which some shortest pathes are much longer than Euclidean distances, this constant can remain low, as in Figure 4.2(c): Here, for every r_1 and r_2, we can at most choose twice the number of points with pairwise distance r_1 in any disk of radius r_2.

Putting these two observations together, we describe a graph G by its $d_G^\star$-independence as follows:

Definition 4.3 *Let $G = (V, E)$ be a graph in $\mathbb{R}^d$. G has $d_G^\star$-independence γ if for all $0 < r_1 < r_2$, all $x \in \mathbb{R}^d$ and all r_1-$d_G^\star$-independent $V' \subset V$, the following holds:*

$$\left|V' \cap B\left(x, r_2\right)\right| \leq \gamma g_d(r_1/r_2) \ ,$$

where $g_d(x)$ denotes the cardinality of a maximum set of points with pairwise distance x in the d-dimensional unit ball.

In geometric filtration as defined in the following, this $d_G^\star$-independence is, together with a parameter $\hat{q}$, which we will set to $-\log q$ for the application of q-QUDG, the main parameter determining the size of local problems:

Definition 4.4 *(Geometric Filtration) Given some $\hat{q} \in \mathbb{N}_0$, in a* geometric filtration*, $V_{\ell+1}$ is an inclusion-maximal subset of V_ℓ such that for every $v \in V_{\ell+1}$ either*

1. *there is some $u \in N_\ell$ with $d(u, v) > 2^{\ell-\hat{q}}$ or*
2. *there is no node $u \in N_{\ell-1}(v) \cap V_{\ell+1}$ with $d(u, v) < 2^{\ell-2\hat{q}-3}$.*

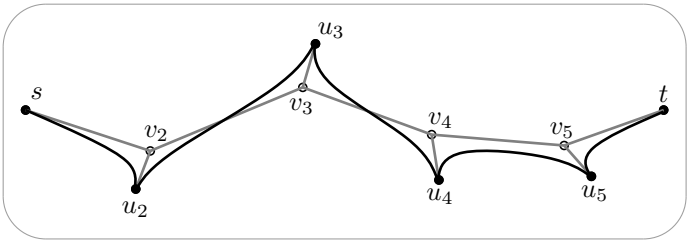


Figure 4.3: *Connectivity after filtration step*

For convenience, we set $N_i := N_0$ for $i < 0$. $E_{\ell+1}$ then is defined as

$$E_{\ell+1} = \left\{ \{u,v\} \in \binom{V_{\ell+1}}{2} : u \in N_\ell^3(v) \leq 3 \text{ and } d(u,v) < 2^{\ell-\hat{q}+1} \text{ or } \{u,v\} \in E_\ell \right\} . \quad (4.2)$$

In other words, in geometric filtration, the decision which nodes to choose and how to connect chosen nodes is done based on three radii, all doubling with every filtration step: The basic idea is to pick an inclusion-maximal set that has pairwise distance of at least $2^{\ell-2\hat{q}-3}$, which causes increasing spatial separation of active nodes, and connect chosen nodes that have distance up to $2^{\ell-\hat{q}+1}$, causing connectivity in an increasing range. For the following, it will become important to admit pairs of nodes with distance less than $2^{\ell-2\hat{q}-3}$ that have not been connected in the preceding filtration graph $G_{\ell-1}$. In addition, we have to make sure to choose all nodes that are incident to edges too long to be a result of the previous steps to remain active and to keep these connections, because otherwise the graph could be separated.

Observation 4.5 *Let G be any graph, $G = G_0, G_1, \ldots, G_h$ a geometric filtration for some $\hat{q} \in \mathbb{N}_0$. If G is connected, then every G_i is connected for $0 \leq i \leq h$.*

Proof. We prove that $G_{\ell+1}$ is connected given that G_ℓ is connected: Let $s, t \in V_{\ell+1}$ and $s = v_1 - \cdots - v_p = t$ be a path in G_ℓ. Without loss of generality, we can assume that no $v_i \notin V_{\ell+1}$ for $1 < i < p$, since otherwise, if there is some such $v_i \in V_{\ell+1}$, it is sufficient to show that s and v_i, and v_i and t are connected in $G_{\ell+1}$. For the same reason, we can assume that $d(v_i, v_{i+1}) < 2^{\ell-\hat{q}}$ for all $1 \leq i < p$, since otherwise, we have $v_i, v_{i+1} \in V_{\ell+1}$ and $\{v_i, v_{i+1}\} \in E_{\ell+1}$ making it sufficient to prove that s is connected to v_i and v_{i+1} is connected to t.

Now, for every $1 < i < p$, there is some $u_i \in V_{\ell+1}$ such that $\{v_i, u_i\} \in E_\ell$ and $d(v_i, u_i) < 2^{\ell-2\hat{q}-3}$. Hence, the pairs $\{s, u_2\}, \{u_{p-1}, t\}$ and $\{u_i, u_{i+1}\}$ for $1 < i < p-1$ have at most a distance $2^{\ell-\hat{q}} + 2 \cdot 2^{\ell-2\hat{q}-3} < 2^{\ell-\hat{q}+1}$ and are within mutual 3-hop neighborhood in G_ℓ (cf. Figure 4.3), i. e., are connected in $G_{\ell+1}$. □

To prove the spatial separation and connectivity is a little more cumbersome.

Lemma 4.6 *Let G be a graph, embedded as q-QUBG for some q, and $G = G_0, G_1, \ldots, G_h$ a geometric filtration for some $\hat{q} \in \mathbb{N}_0$ with $q > 2^{-\hat{q}}$. Then*

$$\max_{\{u,v\} \in E_\ell} d(u,v) \leq \max(1, 2^{\ell - \hat{q}}) \ , \tag{4.3}$$

for any $0 \leq \ell \leq h$ any two nodes $u, v \in V_\ell$ with $d_G^\star(u,v) \leq 2^{\ell - 2\hat{q} - 2}$ are connected and for all $0 < \ell \leq h$, V_ℓ is $2^{\ell - 2\hat{q} - 4}$-$d_G^\star$-independent.

Proof. Every edge in some E_ℓ either existed in G and has length at most 1 or the edge has been created on some level $\ell' \leq \ell$ and has length at most $2^{\ell - \hat{q}}$.

For the second and third part of the claim, we look more closely at the way nodes are suppressed from being included into some $V_{\ell+1}$ by close nodes due to property 2 of Definition 4.4: Whenever a node $v \in V_\ell$ is not included in $V_{\ell+1}$, there is some node $u \in N_{\ell-1}(v) \cap V_{\ell+1}$ with $d(u,v) < 2^{\ell - 2\hat{q} - 3}$ such that $v \in V_{\ell+1}$. In this case, we call u the replacement of v in level $\ell + 1$, denoted by $\mathtt{rp}_{\ell+1}(v)$. We extend this notation by defining $\mathtt{rp}_\ell(v) := v$ if $v \in V_\ell$ and $\mathtt{rp}_{\ell+1}(v) := \mathtt{rp}_{\ell+1}(\mathtt{rp}_\ell(v))$ if $v \notin V_\ell$. With this notation, we first get that for every node v

$$d(v, \mathtt{rp}_\ell(v)) \leq \sum_{i=1}^{\ell} d(\mathtt{rp}_{i-1}(u), \mathtt{rp}_i(u)) \leq \sum_{i=1}^{\ell} 2^{(i-1) - 2\hat{q} - 3} \leq 2^{\ell - 2\hat{q} - 3} \ . \tag{4.4}$$

For all ℓ and all $\{u,v\} \in E$, we additionally know that $\mathtt{rp}_\ell(v) \in N_\ell(\mathtt{rp}_\ell(u))$, since for ℓ with $d(u,v) > 2^{\ell - \hat{q}}$, $\mathtt{rp}_{\ell+1}(u) = u$ and $\mathtt{rp}_{\ell+1}(v) = v$, and for ℓ with $d(u,v) \leq 2^{\ell - \hat{q}}$, it follows inductively from the fact that $\mathtt{rp}_{\ell+1}(u) - \mathtt{rp}_\ell(u) - \mathtt{rp}_\ell(v) - \mathtt{rp}_{\ell+1}(v)$ is a path in G_ℓ and

$$\begin{aligned} d\left(\mathtt{rp}_{\ell+1}(u), \mathtt{rp}_{\ell+1}(v)\right) &\leq d\left(\mathtt{rp}_{\ell+1}(u), u\right) + d(u,v) + d\left(v, \mathtt{rp}_{\ell+1}(v)\right) \\ &\leq 2^{\ell - 2\hat{q} - 3} + 2^{\ell - \hat{q}} + 2^{\ell - 2\hat{q} - 3} \\ &< 2^{\ell - \hat{q} + 1} \ . \end{aligned} \tag{4.5}$$

If now any two nodes $u, v \in V$ have distance $d_G^\star(u,v) < 2^{\hat{s}}$ for some $\hat{s} \geq 1$, let $u = v_0 - v_1 - \cdots - v_k = v$ be a path of minimum Euclidean length between u and v. Since G is embedded as q-QUBG, this path cannot have more than $2^{\hat{s} + \hat{q} + 1}$ hops, because in any path of minimum euclidean length, two nodes v_i, v_{i+2} have distance at least $2^{-\hat{q}}$.

On the other hand, we have $d\left(v_i, v_{i+1}\right) < 1$, and we can show inductively that for $\ell > \hat{q}$ and every $s \leq 2^{\ell - \hat{q} - 1}$, we have that

$$\left\{\{\mathtt{rp}_{\ell+1}(v_i), \mathtt{rp}_\ell(v_{i+s})\}, \{\mathtt{rp}_\ell(v_i), \mathtt{rp}_\ell(v_{i+s})\}, \{\mathtt{rp}_\ell(v_i), \mathtt{rp}_{\ell+1}(v_{i+s})\}, \right\} \subset E_\ell \ . \tag{4.6}$$

We start with the observation that for any $u \in V$, $\mathtt{rp}_{\hat{q}+2}(u) \in N_{\hat{q}}(\mathtt{rp}_{\hat{q}}(u))$, since

$$d\left(\mathtt{rp}_{\hat{q}}(u), \mathtt{rp}_{\hat{q}+2}(u)\right) \leq 2^{(\hat{q}+2) - 2\hat{q} - 3} = 2^{-\hat{q} - 1} < q/2 \ . \tag{4.7}$$

Since $\mathtt{rp}_{\hat{q}+2}(v_i) - \mathtt{rp}_{\hat{q}}(v_i) - \mathtt{rp}_{\hat{q}}(v_{i+1}) - \mathtt{rp}_{\hat{q}+1}(v_{i+1})$ forms a path in $G_{\hat{q}}$, and with

$$d\left(\mathtt{rp}_{\hat{q}+2}(v_i), \mathtt{rp}_{\hat{q}+1}(v_{i+1})\right) \leq 2 \ , \tag{4.8}$$

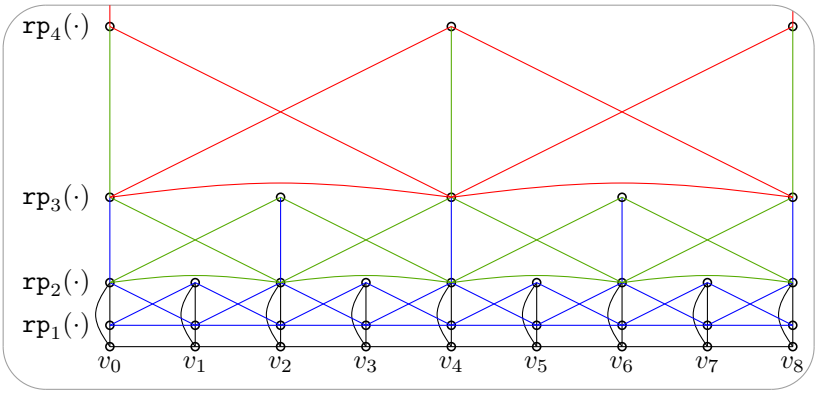


Figure 4.4: *Schematic guaranteed connectivity of replacements in different levels of filtration*

$\mathtt{rp}_{\hat{q}+2}(v_i)$ and $\mathtt{rp}_{\hat{q}+1}(v_{i+1})$ become neighbors in $G_{\hat{q}+1}$, as well as $\mathtt{rp}_{\hat{q}+1}(v_i)$ and $\mathtt{rp}_{\hat{q}+2}(v_{i+1})$ by symmetric arguments. This is the basis for $\ell = \hat{q}+1$ and $s = 1$. Then, if for $\ell = \hat{q}+j$ and every $s \leq 2^{j-1}$, Inclusion 4.6 holds, then for $s' \leq 2^j$ and $s'' := \lfloor s'/2 \rfloor$, $\mathtt{rp}_{\ell+2}(v_i) \in N_{\ell+1}(\mathtt{rp}_{\ell+1}(v_{i+s'}))$ since $\mathtt{rp}_{\ell+2}(v_i) - \mathtt{rp}_{\ell+1}(v_i) - \mathtt{rp}_{\ell}(v_{i+s''}) - \mathtt{rp}_{\ell+1}(v_{i+s'})$ forms a path in G_ℓ and we have similarly as above

$$d\left(\mathtt{rp}_{\ell+2}\left(v_i\right), \mathtt{rp}_{\ell+1}\left(v_{i+s'}\right)\right) \leq 2^{\ell-2\hat{q}-1} + 2^{\ell-\hat{q}} + 2^{\ell-2\hat{q}-2} < 2^{\ell-\hat{q}+1} \ . \tag{4.9}$$

Again, by symmetric arguments, we get $\mathtt{rp}_{\ell+2}(v_{i+s'}) \in N_{\ell+1}(\mathtt{rp}_{\ell+1}(v_i))$ and by similar arguments $\mathtt{rp}_{\ell+1}(v_i) \in N_{\ell+1}(\mathtt{rp}_{\ell+1}(v_{i+s'}))$, proving Equation 4.6 for all $\ell > \hat{q}$. These connections are schematically depicted in Figure 4.4 for $\hat{q} = 0$, i. e., for UDG. Depicted are only the most relevant edges, black those in G, blue, green and red the edges in G_1, G_2 and G_3.

Hence, $\mathtt{rp}_\ell(v) \in N_\ell(\mathtt{rp}_\ell(u))$ for $\ell \geq \hat{s} + 2\hat{q} + 2$. This proves that in G_ℓ, any two nodes u, v with $d^\star_G(u,v) \leq 2^{\ell-2\hat{q}-2}$ are connected, and this in turn proves that for any $\ell \geq \hat{q}$ any two nodes $u, v \in V_{\ell-2}$ with $d^\star_G(u,v) \leq 2^{\ell-2\hat{q}-4}$ are neighbors in $G_{\ell-2}$, which allows at most one of them to be in V_ℓ. □

Observation 4.7 *Let $G = (V,E)$ be a graph with $d^\star$-independence γ, embedded in $\mathbb{R}^d$ as q-QUBG. Let γ_0 denote the maximum number of nodes per unit ball, i. e., $\gamma_0 := \max_{x\in\mathbb{R}^d} |\{v \in V : d(x,v) \leq 1\}|$. Let $G = G_0, \ldots, G_h$ be a geometric filtration of G for some $\hat{q} \in \mathbb{N}_0$ with $\hat{q} = \lceil -\log q \rceil$. Then,*

1. *$h \in O\left(\log(\mathrm{diam}_G/q)\right)$*
2. *Neighborhood sizes are bounded by $\left|N^3_\ell(v)\right| \in O\left(\gamma_0 + \gamma/q^d\right)$ for all $\ell \geq 0$ and $v \in V_\ell$.*
3. *on each level, every node receives and sends at most $O\left(\gamma_0^2 + (\gamma/q^d)^2\right)$ messages.*

Proof. The first statement directly follows from Lemma 4.6: Since edges in G have length at most one, we have

$$\max_{u,v\in V} d^\star_G(u,v) \leq \mathrm{diam}_G \ , \tag{4.10}$$

and hence, for ℓ with $2^{\ell-2\hat{q}-2} \geq \text{diam}_G$, i. e., for $\ell = \text{ld}(\text{diam}_G) + 2\hat{q} + 2$ at the latest, G_ℓ would be a clique.

The second claim follows from the fact that in the graphs G_0 to $G_{\hat{q}}$, no edge exists or is introduced with length more than 1, meaning that the 3-hop neighborhood covers an area of size bounded by a constant. For $\ell > \hat{q}$, any nodes $u, v \in V_\ell$ with $u \in N_\ell^3(v)$ have distance at most $d(u,v) \leq 3 \cdot 2^{\ell-\hat{q}}$, and $N_\ell^3(v)$ is $2^{\ell-2\hat{q}-4}$-$d_G^\star$-independent, such that $|N_\ell^3(v)|$ contains at most $\gamma g_d(3 \cdot 2^{\hat{q}+4})$ nodes.

Similarly, we can show that any node $w \in V$ can only relay messages being sent on level ℓ, i. e., from some $v \in V_\ell$ to some $u \in N_\ell^3(v)$, if both, v and u have Euclidean distance less than $2^{\ell-\hat{q}+2}$ to w: On the route from w to u, the message reaches the first $w_1 \in V_1$ in at most 2 hops in G_0. Both hops must have lengths less than $2^{-\hat{q}}$ by the definition of V_1. After w_1, the first node $w_2 \in V_2$ is at most 2 hops away in G_1, having lengths less than $2^{1-\hat{q}}$ and so on. Overall, this gives

$$d(w,u) \leq \sum_{i=0}^{\ell} 2 \cdot 2^{i-\hat{q}} \leq 2^{\ell-\hat{q}+2} \quad . \tag{4.11}$$

By symmetric arguments, we get the same bound for $d(v,w)$. However, the number of nodes from V_ℓ that are located within that range is bounded by $\gamma \cdot g_d(2^{\hat{q}+6}) \in O(\gamma/q^d)$ for $\ell > \hat{q}$ and by $O(\gamma_0)$ otherwise. Since every pair of nodes in V_ℓ exchanges at most a constant number of messages, the claimed bound holds. □

For geometric filtration as defined above, nodes in G_ℓ need to find a maximal independent set of nodes, restricted to nodes that do not have a neighbor that is chosen due to a long edge, and restricted to edges $e \in E_{\ell-1}$ with length less than $2^{\ell-\hat{q}}$. One way to do this, distributedly and not increasing the message count, is to exchange IDs and let nodes decide when either one of its neighbors signals that it joined the set or until all nodes with higher IDs signal they will not be in the set. In the nomenclature of distributed computing, this is a localized algorithm, but with a linear worst case runtime. On the other hand, all of the above bounds do not change if we compute a filtration such that condition 2 of Definition 4.4 can be violated, but still guaranteeing that in expectation, for every $v \in V_{\ell+1}$, the number of neighbors violating the condition, i. e., $|\{u \in N_{\ell-1}(v) \cap V_{\ell+1} : d(u,v)\}|$ is bound by some fixed constant c. This can, for example, be achieved using random numbers in a range with size $\Theta(\gamma_0 + \gamma/q^d)$ instead of IDs for the above approach, simply by accepting some close and connected nodes joining $V_{\ell+1}$ in the same round.

4.3.2 Combinatorial Filtration

Geometric filtration guarantees desired properties for relevant scenarios, especially in terms of the size of positioning problems to solve centrally at each node. However, we pay with comparably high hidden constants for its bounds on neighborhood sizes and filtration height, which empirically stay far below these bounds for much simpler, combinatorial filtration techniques. We will define a purely combinatorial filtration technique, which we will use for evaluation of filtration-based MDS to be defined in Section 4.4. The definition slightly differs for different values of k to ensure connectivity:

Definition 4.8 (Combinatorial filtration, $k \geq 3$) *In a combinatorial filtration for $k = 3$, $V_{\ell+1}$ is a dominating set in G_ℓ, and for any $v \in V_{\ell+1}$ we set $N_{\ell+1}(v) := N_\ell^k(v) \cap V_{\ell+1}$.*

Definition 4.9 (Combinatorial filtration, $k = 2$) *In a combinatorial filtration for $k = 2$, $V_{\ell+1}$ is a set of nodes such that for every edge $\{u, v\} \in E_\ell$, there one node $w \in V_{\ell+1} \cap N_\ell^1(u) \cap N_\ell^1(v)$, and for any $v \in V_{\ell+1}$ we set $N_{\ell+1}(v) := N_\ell^2(v) \cap V_{\ell+1}$.*

Observation 4.10 (Connectivity of combinatorial filtration) *Let G be a connected graph and $G = G_0, \ldots, G_h$ be a combinatorial filtration for $k \geq 2$. Then, every G_i is connected.*

Proof. The proof is basically the same as for Observation 4.5: We again pick any two nodes $s, t \in V_{\ell+1}$ and a path $s = v_0 - v_1 - \cdots - v_p = t$ in G_ℓ, without loss of generality, we assume again $v_1, \ldots, v_{p-1} \notin V_{\ell+1}$. Now, for $k = 2$, there must be nodes $u_1, \ldots, u_p \in V_\ell$, not necessarily pairwise different, such that $u_i \in N_\ell(v_{i-1}) \cap N_\ell(v_i)$ for $1 \leq i \leq p$. Hence, in $G_{\ell+1}$, s is connected to u_1, every u_i to u_{i+1} for $1 \leq i < p$, and u_p to t. For $k = 3$, we have nodes $u_1, \ldots, u_{p-1} \in V_\ell$, such that $u_i \in N_\ell(v_i)$ for $1 \leq i < p$. Hence, in $G_{\ell+1}$, s is connected to u_1, every u_i to u_{i+1} for $1 \leq i < p-1$, and u_{p-1} to t. □

Combinatorial filtration does not rely on the outcome of local positioning. Hence, not all nodes in some V_ℓ have to run the positioning algorithm on their local neighborhood. Only those nodes that are selected for $V_{\ell+1}$ have to position their neighborhoods. Running the positioning algorithm only on selected nodes reduces the number of invocations of the local positioning algorithm by one for each node, such that most nodes never have to run it at all. Again, such a filtration with small V_i can be computed distributedly in a constant number of rounds.

In Figure 4.5, a exemplary combinatorial filtration is shown for $k = 3$.

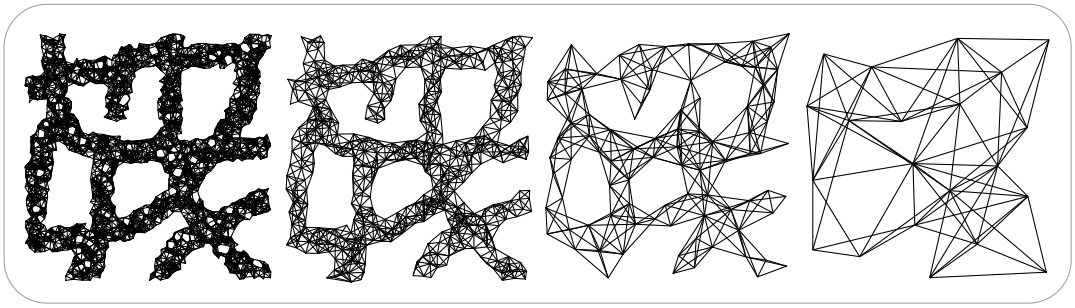
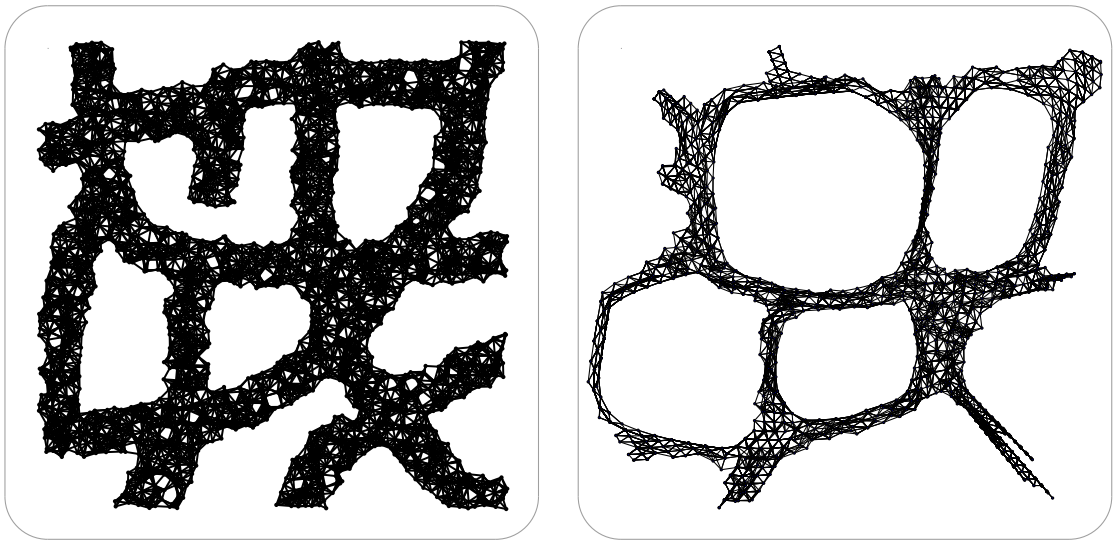


Figure 4.5: *An exemplary filtration for $k = 3$ for a network with 2500 nodes. Shown are graphs G_0, G_1, G_2, G_3.*

4.4 MDS-MAP(F): Filtration-based MDS

Since most hardware platforms provide at most poor distance estimations, we focus on the case where we can rely on connectivity information only or have at most noisy distance measurements available.

Multidimensional Scaling (MDS) has proven to produce good localizations in this scenario for small-scale solutions, but for large-scale networks, it not only becomes impractical due to its runtime and the necessity of determining all-pair shortest path distances as input. It also completely fails to recover network structures if the deployment area is not convex.

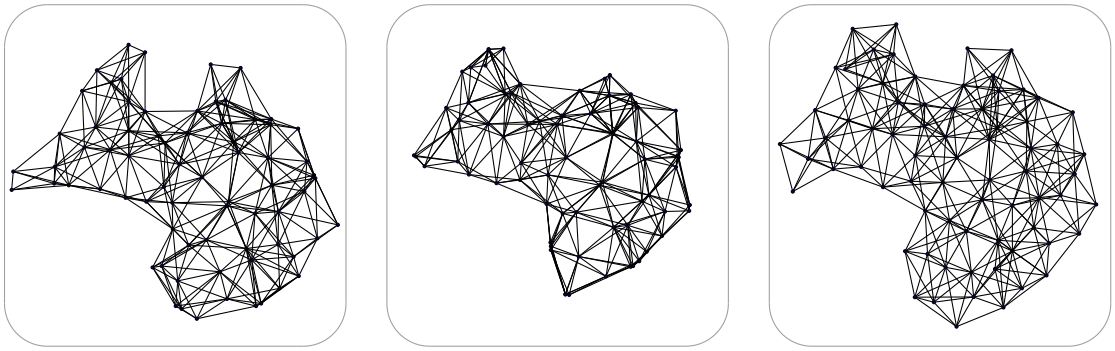


(a) Ground truth (b) MDS result

Figure 4.6: *MDS-based positioning of 2800 nodes spanning a network in a non-convex area; input is connectivity only*

In Figure 4.6, a quite typical result of MDS-based positioning is shown. It is plain to see how the localization of the whole network is distorted since shortest path lengths do not reflect Euclidean distances very well. Some parts of the network are even folded, such that local optimization could not correct the positioning either. Figure 4.7 in contrast shows, how the 3-hop neighborhood of a node is recovered much more accurately. Moreover, since the MDS result recovers the original positions well enough, local optimization can considerably lessen the artifacts of MDS. Yet, positions are not perfectly recovered since there are no edge lengths available.

As mentioned in the introduction, the quality of small-scale MDS positioning has been utilized and extended to more complex networks within distributed frameworks that either patch local solutions together, or by dead reckoning along paths to estimate relative directions of endpoints. MDS also very naturally fits in the framework of distributed filtration-based positioning. On the first level, i. e., in G_0, where we are only given the neighborhood



(a) Ground truth (b) MDS result (c) Refined MDS result

Figure 4.7: *MDS-based positioning of a node's 3-hop neighborhood*

relation and possibly distance estimations, we use MDS to find good positions for local neighborhoods. This provides us with distance estimations for pairs of nodes in E_1, but we are not restricted to take only distances as input for higher levels and apply MDS again. Instead, we are in the position that nodes that become neighbors on the next level know positions for overlapping neighborhoods. This allows to deduct the best-fit transformation between any two local solutions which overlap in at least $d+1$ nodes as λ_1. This not only carries information on distance of pairs in E_1, but also allows to do positioning of neighborhoods on the next level by placing nodes at initial positions: Transformations are combined along paths of at most 3 hops to place nodes; then this solution is refined using simple force-directed optimization without any more MDS solving involved. This way, MDS is only applied for positioning of local neighborhoods on the very first level. Following the conventions, we will call this instantiation of filtration-based positioning MDS-MAP(F).

Best-fit affine transformation

When two nodes $u, v \in V_\ell$ become neighbors in $G_{\ell+1}$, they both have positions for overlapping neighborhoods in G_ℓ, $\mathbf{p}_{\ell,u} : N_\ell^k(u) \to \mathbb{R}^d$ and $\mathbf{p}_{\ell,v} : N_\ell^k(v) \to \mathbb{R}^d$. We assume for all such embeddings that $\mathbf{p}_{\ell,u}(u) = \mathbf{0}$.

Now, k-hop neighborhoods of nodes not further than k hops apart overlap at least in

$$\min\{k+1, |V_\ell|\}$$

nodes. Hence, for $k \geq d$, and as long as no single node's k-hop neighborhood covers all nodes, any pair of nodes u, v to become neighbors in some $G_{\ell+1}$ can exchange enough positions for nodes to find an affine transformation to align the two neighborhoods. In general, nodes can even exchange a larger set of nodes $V_\ell^{uv} \subset N_\ell^k(v) \cap N_\ell^k(u)$ both nodes have positions for. It is sufficient if one of the two nodes, say v, sends positions of a small set of nodes that have no more than k hops to u in $G[N_\ell^3(v)]$. With at least $d+1$ positions $\mathbf{p}_{\ell,v}(w_1), \ldots, \mathbf{p}_{\ell,v}(w_j)$ and $\mathbf{p}_{\ell,u}(w_1), \ldots, \mathbf{p}_{\ell,u}(w_j)$ known to u, it can compute the unique

affine transformation matrix $T_\ell(u,v)$ such that

$$\sum_{i=1}^{j} d\left(\mathbf{p}_{\ell,v}(w_j), T_\ell(u,v)\cdot\mathbf{p}_{\ell,u}(w_j)\right)^2 \tag{4.12}$$

is minimized (see [TCPK01]). In addition, if two neighborhoods overlap in more than $d+1$ nodes, the transformation can be annotated with the average error of transformation on this set of nodes as a measure of quality.

Initial Layout

Knowing the nodes within the k-hop neighborhood as well as transformations for all edges in $E_\ell[N_\ell^3(v)]$, $\ell > 0$, each active node v can simply assign initial positions to every node $u \in N_\ell^3(v)$ by multiplying transformation matrices along any path $v = v_0 - \cdots - v_p = u$ in G_ℓ as

$$\mathbf{p}^0_{\ell,v}(u) := \prod_{i=1}^{p} T_\ell(v_{i-1}, v_i)\cdot\mathbf{0}\ . \tag{4.13}$$

It turned out to be beneficial for sparse networks to initialize positions along a shortest path tree, using a function of the number of overlaps and the average error as distance function. This prevents choosing bad paths in terms of transformation quality if good paths are available. In any case, the computation of shortest paths are restricted to the k-hop neighborhood that is known to the central node. Hence, it does not require any more communication and is computationally inexpensive.

Stress minimization

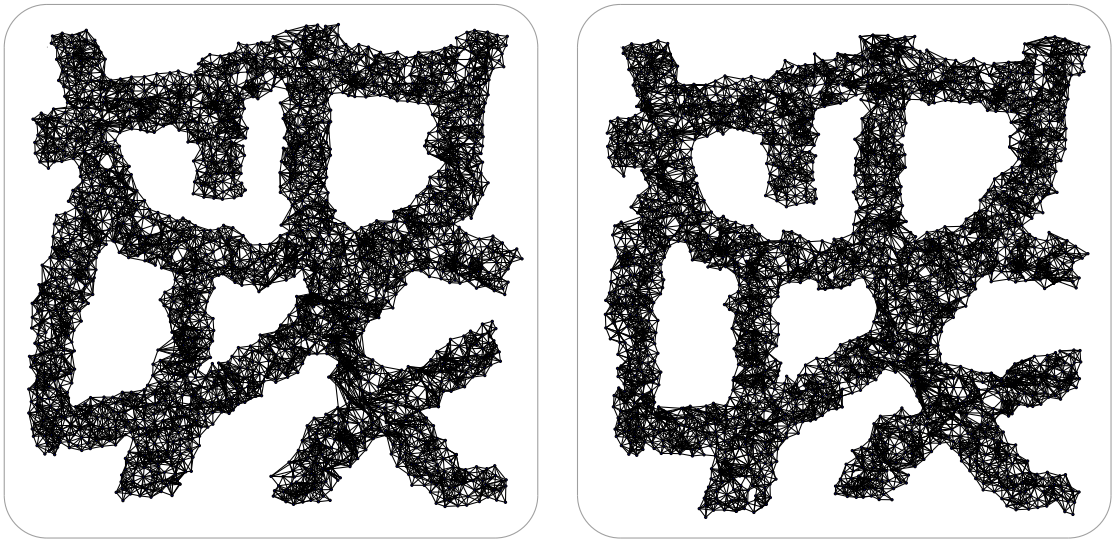
After a node v has computed initial positions for nodes in its k-hop neighborhood, it refines its local map using a simple force-directed optimization, shifting every node by

$$\mathbf{p}^{i+1}_{\ell,v} := \frac{c}{\delta(u)}\cdot\sum_{w\in N_\ell(u)\cap N_\ell^3(v)}\left(\left(|\mathbf{p}^i_{\ell,v}(u)-\mathbf{p}^i_{\ell,v}(w)| - d(u,w)\right)\cdot\frac{\mathbf{p}^i_{\ell,v}(u)-\mathbf{p}^i_{\ell,v}(w)}{|\mathbf{p}^i_{\ell,v}(u)-\mathbf{p}^i_{\ell,v}(w)|}\right)\ , \tag{4.14}$$

which usually converges fast given the quality of the initial layout and the small problem size.

Supporting nodes

The filtration techniques proposed in Section 4.3 aimed at filtrations that maintain connectivity to guarantee consistency. For local optimization, connectivity alone is not enough. If a graph allows for continuous deformations without altering edge lengths, there is no unique local or global optimum to converge to. Hence, in each filtration step we construct two graphs: In addition to $G_{\ell+1}$ as before, we also construct a graph $G^\star_{\ell+1}$ with an extended set of nodes $V^\star_{\ell+1}$ with $V_\ell \supset V^\star_{\ell+1} \supset V_{\ell+1}$, $E^\star_{\ell+1}$ defined analogously to $E_{\ell+1}$. While nodes in each



(a) Result of MDS-MAP(F) for $k = 2$

(b) Result of MDS-MAP(F) for $k = 3$

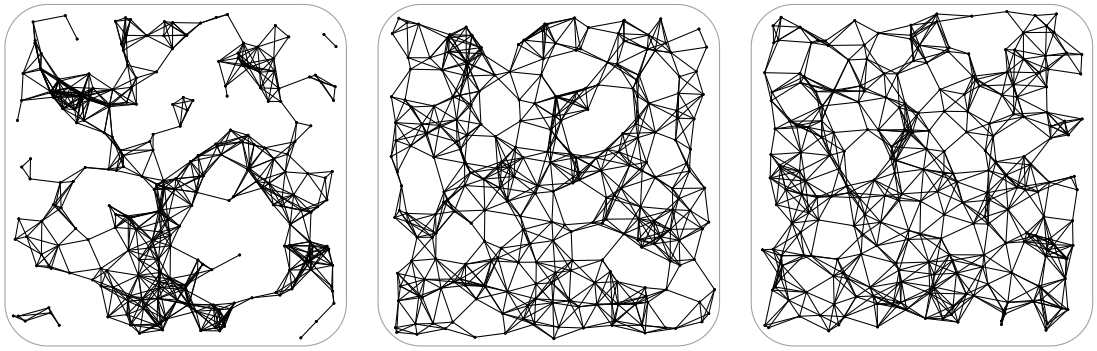
Figure 4.8: *Exemplary results of MDS-MAP(F) for* $k = 2, 3$

step may only be chosen as before, local solutions (and hence relative positions) can be computed from neighborhoods in $G_\ell^\star$. Although even choosing $V_{\ell+1}^\star = V_\ell$ would not change the results from Observation 4.7, it is sufficient to ensure that nodes must be chosen unless at least two neighbors are chosen for $k \geq 3$ (Definition 4.8), or unless for every incident edge $\{u, v\}$, at least two nodes from $N_\ell(u) \cap N_\ell(v)$ are chosen for $k = 2$ (Definition 4.9). We omit a proof, but it is straightforward to see that this ensures that all graphs $G_1, \ldots, G_h$ are rigid. In Figure 4.8, two exemplary results of MDS-MAP(F) are depicted for $k = 2$ (Figure 4.8(a)) and $k = 3$ (Figure 4.8(b)). Both variants recover the network quite faithfully, performing almost equally well in this scenario, not surprisingly with slight advantages for $k = 3$.

4.5 Application and Evaluation

4.5.1 Set-up

There is no agreed set of scenarios or quality indices to evaluate positioning algorithms with. Quite usually, evaluation covers a number of network *topologies*, here referring to the region that nodes are deployed in. For each topology, nodes are placed randomly and connected according to a connectivity model. For the positions of the nodes, two options are most wide-spread: Placing nodes uniformly at random in the deployment region or placing nodes on a grid and shift nodes by some random offset. The former is obviously the more suggesting option: This is what one would expect under a random deployment.



(a) uniform (b) GwP, shift $\sim U(0, g)$ (c) GwP, shift $\sim N(0, g^2)$

Figure 4.9: *250 nodes deployed uniformly or on a grid with perturbation in a square region, connected as UDG with an average degree of* 8

Furthermore, this is easy to implement and parameter-free. The latter process, also called *grid with perturbation (GwP)* model needs further specification of the perturbation process and entails several implementation issues: How to adjust the grid-size to gain a specific number of nodes to fall in the interior of the deployment region, how to deal with nodes that would be shifted out of the region? The reason why the grid with perturbation model is chosen more often is, that it deploys nodes more evenly. Roughly speaking, uniform deployment leads to regions with higher density and higher average degrees and to regions with lower density and lower average degree. In Figure 4.9(a), 250 nodes are deployed uniformly at random in a square region and connected as a UDG with average degree 8. If there is something like a critical density for a positioning algorithm, or a coherence of density and the quality of localization, then usually, the denser regions do not help as much as the sparser regions harm. Hence, for the same quality of positioning, uniformly deployed nodes usually need a higher average degree. This not only lets the respective results look worse, it also somehow clouds the density of the problematic regions. Where applicable, we evaluate both cases, uniform deployment and deployment for the GwP model. In the GwP model, we shift nodes in a random direction by a random offset, uniformly distributed in $[0, pg]$, g being the grid length, p a perturbation parameter. Note that this does not mean that node positions are uniformly distributed in the disk or ball of radius pg around the grid points. In the literature, many other implementations can be found, including a normally distributed shift (which also must not be mistaken for a normal distribution around the grid point) as in [BGJ05] or a uniform distribution in the grid cells as in [WGM06]. These differences are—for proper parameters—negligible compared to uniform deployment: In Figures 4.9(b) and 4.9(c), results of GwP models for uniform and normal shift are compared for the same average degree of 8. Both do not suffer from large, high-degree clusters and the giant holes caused by uniform deployment.

Mostly, positioning algorithms are evaluated for the UDG model, i. e., the radio range is adjusted for a specific degree. To evaluate the robustness, we also generate random QUDGs for a given point set with a specific *expected* average degree with the following process with

two parameters, the *relative radio range* $r_{\text{rel}} \in (0,1]$ and a $q_{\text{rel}} \in (0,1]$. For a fixed radio range r, we assign a random individual radio range $r(u) \sim r \cdot U(r_{\text{rel}},1)$ to every node u. Two nodes u,v now are connected if $d(u,v) \leq q_{\text{rel}} \cdot \min\{r(u),r(v)\}$. They are not connected if $d(u,v) > \cdot \min\{r(u),r(v)\}$. In any other case, a coin is tossed. This obviously results in a $r_{\text{rel}}q_{\text{rel}}$-QUDG, and for $r_{\text{rel}} = 1$, it would be a straightforward implementation of q-QUDGs, but it also allows to model systematic errors due to the placement of nodes or different hardware quality using a smaller r_{rel}. Unless stated otherwise, we draw random q-QUDGs setting $r_{\text{rel}} = q_{\text{rel}} = \sqrt{q}$.

4.5.2 Evaluation Criteria

There is a wide range of quality measurements used to evaluate the quality of positioning. Anchor-based positioning algorithms can be rated by the average distance of the positions returned from the ground truth, but results from anchor-free approaches can be rotated and flipped. Therefore, different notions of *stress* of solutions have been introduced comparing the ground truth distances d_{ij} to the distances in the reconstruction, $\hat{d}_{ij}$. Examples are the *global energy ratio* as defined by Priyantha et al. [PBDT03],

$$\text{GER} := \frac{2}{n(n-1)} \sqrt{\sum_{1 \leq i < j \leq n} \left(\frac{d_{ij} - \hat{d}_{ij}}{d_{ij}} \right)^2}, \tag{4.15}$$

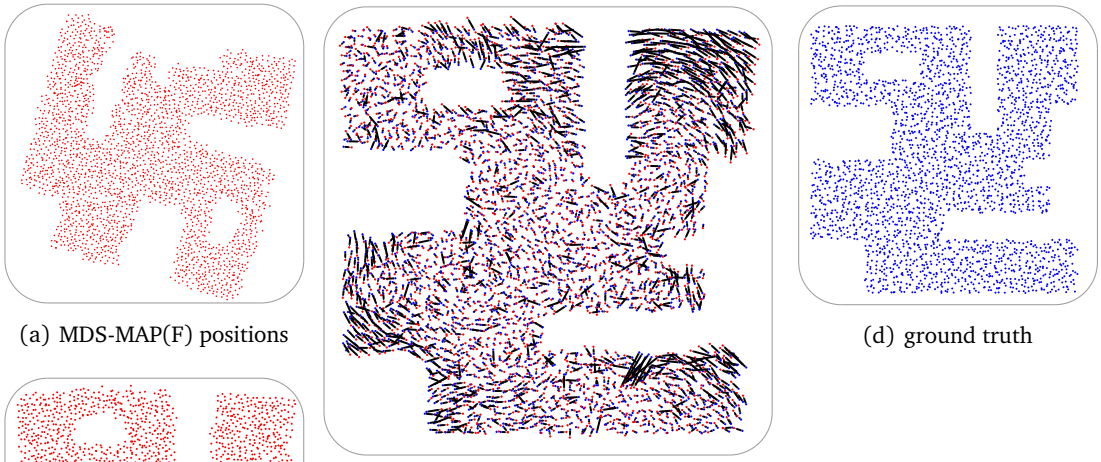
which the authors mistakenly claim to be the root-mean-square normalized error of pairwise distances, the *global stress RMS (GSR),* [KW07] correcting this error to

$$\text{GSR} := \sqrt{\frac{2}{n(n-1)} \sum_{1 \leq i < j \leq n} \left(\frac{d_{ij} - \hat{d}_{ij}}{d_{ij}} \right)^2}, \tag{4.16}$$

or the *frobenius metric (FROB)* [EEF+06] as

$$\text{FROB} := \sqrt{\frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} (d_{ij} - \hat{d})^2}, \tag{4.17}$$

which does not normalize errors and is hence sensitive to the size of the deployment area. These—as well as other, similar measures—produce numbers which are hard to read and can hide systematic errors, such as different scales. Especially for connectivity-based positioning, such errors are common, since the average distance between connected nodes depends on the connectivity model. Hence, unless otherwise stated, we will measure the quality of a reconstruction by the average distance of the nodes to their ground truth after a best-fit affine transformation. This comparison closely resembles the impression of visual inspection as well as the result of a-posteriori anchorage. Results can be scaled either to the radio range R or in percent of the size of the network, i. e., $1\% := \max_{0<i,j\leq n} d_{ij}/100$. Unless stated otherwise, every test is performed at least 250 times unless a confidence of less than $0.02R$ was attained earlier for a confidence level of 95%.



(a) MDS-MAP(F) positions

(b) aligned positions

(c) misplacement vectors

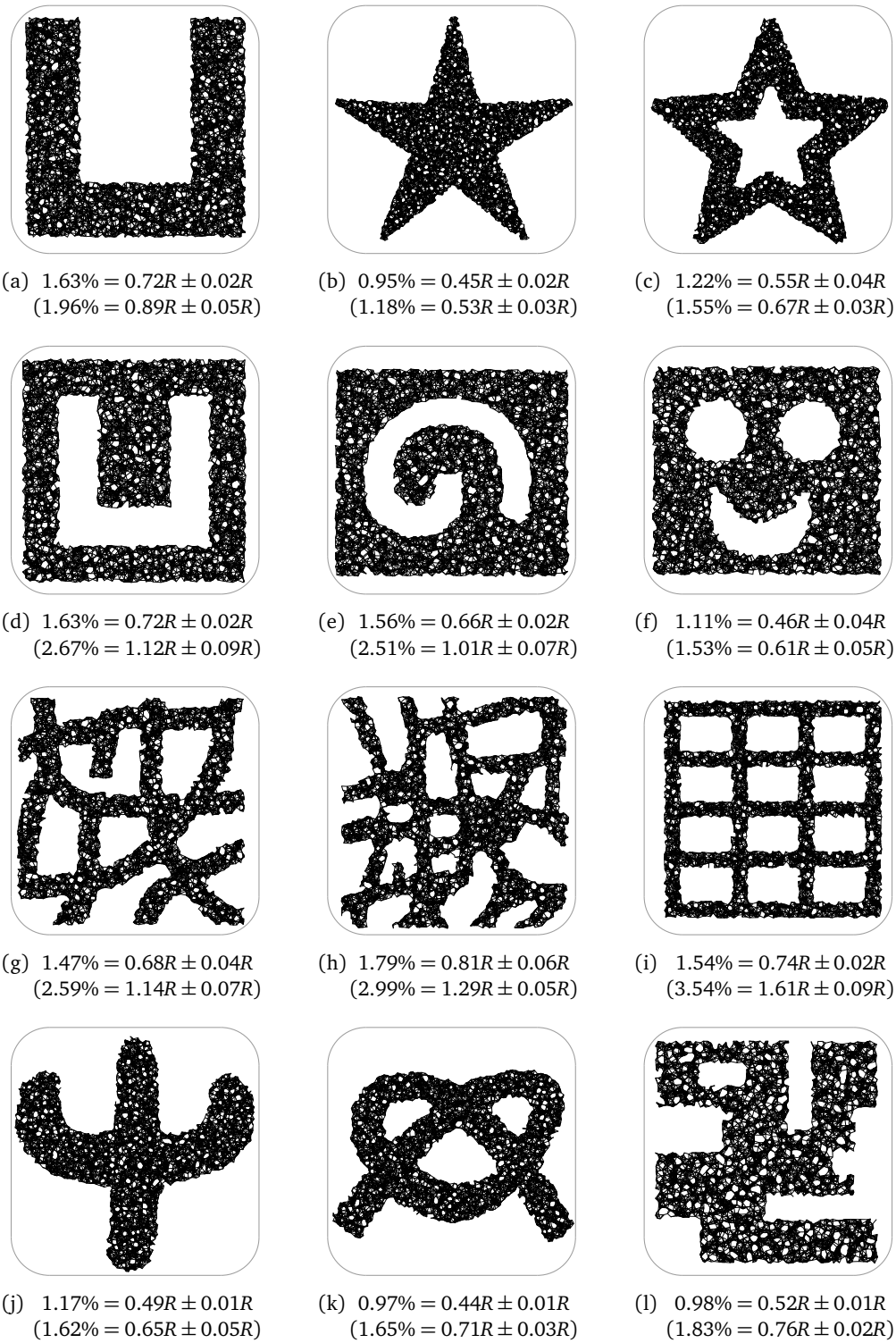
(d) ground truth

Figure 4.10: *Quality of MDS-MAP(F) reconstruction*

4.5.3 Results

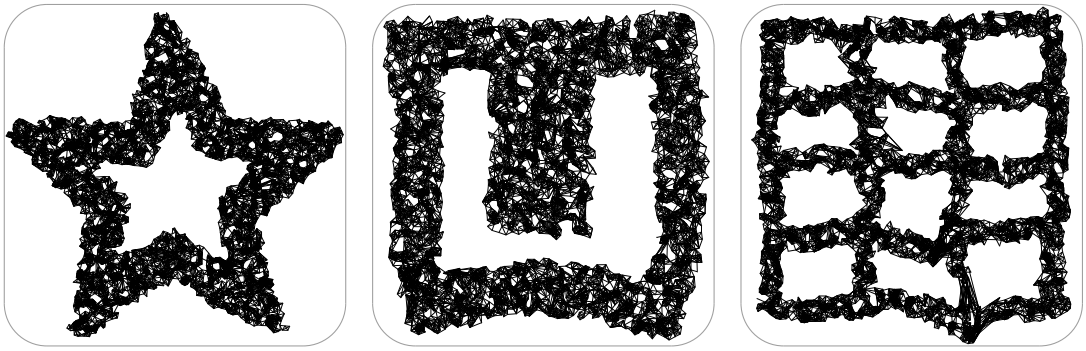
We evaluated MDS-MAP(F) on a variety of topologies we found in the literature (and some more), as shown in Figure 4.11 [EEF+06, KFPF06, LWG08]. Depicted are exemplary deployments of 4000 nodes on a grid with perturbation. Nodes are connected as 2/3-QUDG with an expected average degree of 10. The topologies are annotated with the average node distance between ground truth and reconstruction. In parentheses, results for uniformly deployed nodes with an average degree of 12 are given. Confidence intervals are given for a confidence level of 95%. In all scenarios, the degree was sufficient to position nodes with an average error of less than one radio range for nodes placed on a grid with perturbation or 2% of the maximum Euclidean distance between the two extremal nodes. For uniform deployment, errors are higher, even in spite of the higher average degree, especially for harder instances with more "bottlenecks", but still provide very reasonable positions. Three exemplary results are depicted in Figure 4.12. In the result for the most challenging scenario, Figure 4.11(i), it becomes apparent that in some areas, neighborhoods on the lowest level could not be positioned perfectly, but the big picture is recovered very accurately.

We also compare MDS-MAP(F) to the very recent approach of Lederer et al., which currently is the only algorithm for connectivity-based positioning of networks with a complex shape we are aware of. Comparison is done for the deployment described in [LWG08, WGM06], which is slightly less challenging than the deployment in the other tests we per-



(a) $1.63\% = 0.72R \pm 0.02R$ ($1.96\% = 0.89R \pm 0.05R$)
(b) $0.95\% = 0.45R \pm 0.02R$ ($1.18\% = 0.53R \pm 0.03R$)
(c) $1.22\% = 0.55R \pm 0.04R$ ($1.55\% = 0.67R \pm 0.03R$)
(d) $1.63\% = 0.72R \pm 0.02R$ ($2.67\% = 1.12R \pm 0.09R$)
(e) $1.56\% = 0.66R \pm 0.02R$ ($2.51\% = 1.01R \pm 0.07R$)
(f) $1.11\% = 0.46R \pm 0.04R$ ($1.53\% = 0.61R \pm 0.05R$)
(g) $1.47\% = 0.68R \pm 0.04R$ ($2.59\% = 1.14R \pm 0.07R$)
(h) $1.79\% = 0.81R \pm 0.06R$ ($2.99\% = 1.29R \pm 0.05R$)
(i) $1.54\% = 0.74R \pm 0.02R$ ($3.54\% = 1.61R \pm 0.09R$)
(j) $1.17\% = 0.49R \pm 0.01R$ ($1.62\% = 0.65R \pm 0.05R$)
(k) $0.97\% = 0.44R \pm 0.01R$ ($1.65\% = 0.71R \pm 0.03R$)
(l) $0.98\% = 0.52R \pm 0.01R$ ($1.83\% = 0.76R \pm 0.02R$)

Figure 4.11: *Topologies for MDS-MAP(F) evaluation. Positioning accuracy for graphs with 4000 nodes, deployed as GwP (uniformly) as* 2/3*-QUDG with expected average degree* 10 (12)*.*



(a) result for Fig. 4.11(c) (b) result for Fig. 4.11(d) (c) result for Fig. 4.11(i)

Figure 4.12: *Exemplary MDS-MAP(F) results*

formed. Hence, it is not surprising that we were able to attain good results even for average degrees as low as 6.9. We performed tests for the topologies, node counts and average degrees given in [LWG08][1]. We also compare our results with plain MDS. The results are given in Table 4.1: In all cases, MDS-MAP(F) significantly improves on [LWG08], and, as expected, returns dramatically better results than plain MDS. Not surprisingly, results for best-fit alignment are by a factor of roughly 2 better than results using only three random anchors.

Effect of network and model parameters

To evaluate the effect of the parameters fixed for the above simulations, we also tested varying the average degree, and the parameters for deployment and connectivity. The results are depicted in Figures 4.13 to 4.15. As stated above, the most relevant network parameter was the average degree (cf. Fig. 4.13): A degree of $7 - 8$ was sufficient for positioning of nodes deployed on a grid with perturbation; for degrees higher than 10, quality increases

[1] In [LWG08], the authors use three random nodes among a set of *landmarks*, which is a byproduct of their algorithm, to align their results with the ground truth. To produce comparable results, we also align the our results using only three random nodes with a minimum distance of $2.5R$ among all nodes.

Table 4.1: *Comparison to [LWG08] and plain MDS. All MDS-MAP(F) and MDS results are confident within* $\pm 0.05R$ *or better.*

topology (Fig.)	4.11(d)	4.11(f)	4.11(k)	4.11(e)	4.11(j)	4.11(c)
nodes	2161	2782	2993	2910	1692	2171
avg. degree	10.4	9.5	9.1	9.5	6.9	10.0
Lederer et al. [LWG08][1]	$1.88R$	$0.91R$	$0.95R$	$1.11R$	$2.39R$	$2.16R$
MDS-MAP(F)[1]	$0.80R$	$0.61R$	$0.60R$	$0.83R$	$1.17R$	$0.63R$
MDS-MAP(F)	$0.43R$	$0.33R$	$0.35R$	$0.47R$	$0.62R$	$0.35R$
MDS	$4.35R$	$1.07R$	$1.12R$	$5.97R$	$1.94R$	$1.87R$

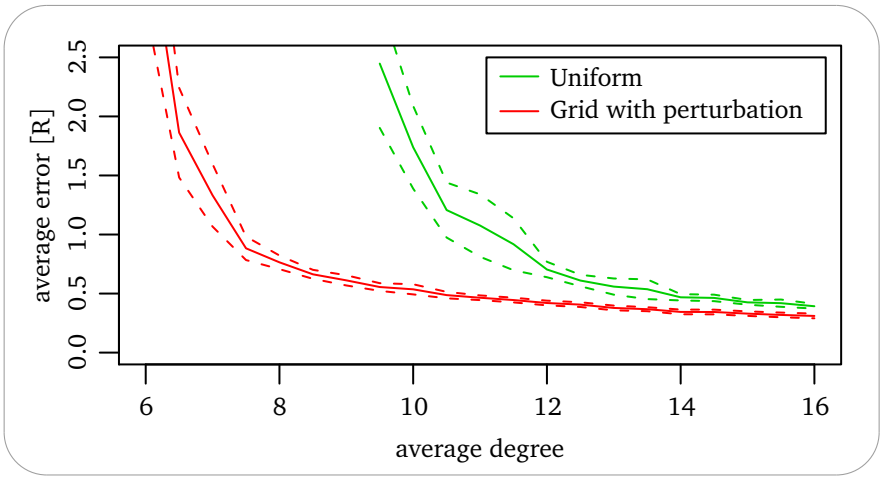


Figure 4.13: *Quality of reconstruction for varying average degree in random networks of* 4000 *nodes, deployed uniformly or on a grid with perturbation in the region from Figure 4.11(c). Dashed lines mark confidence intervals.*

only minimally with more connectivity. For uniform deployment, a degree of $10 - 12$ was necessary to attain reasonable results. Also, the deployment has a considerable impact on the quality (cf. Fig. 4.14): Only for comparably strong perturbation, uniform deployments and deployments on a grid became comparable. The connectivity model in contrast influenced the reconstruction remarkably weak: We varied both connectivity parameters q_{rel} and r_{rel} between 1 and 0.4, i. e., constructed random graphs between UDGs and 0.16-QUDGs (cf.

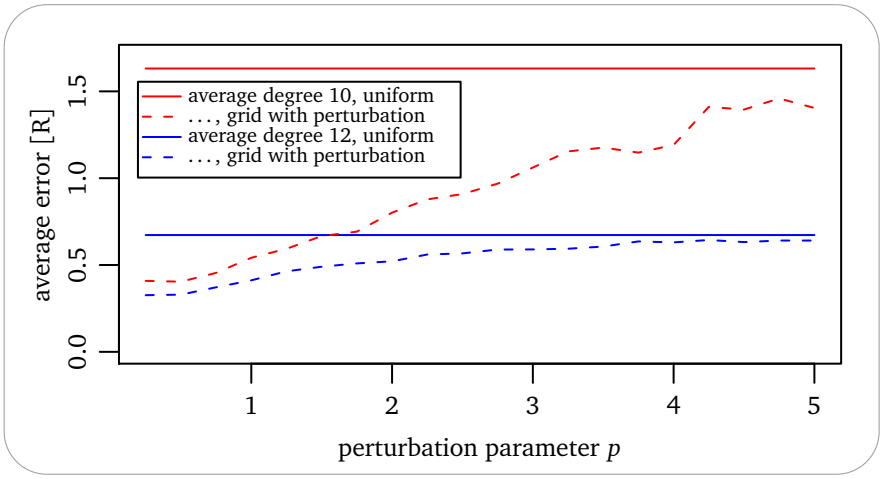


Figure 4.14: *Quality of reconstruction for uniform and GwP deployment for varying perturbation parameter p in random networks of* 4000 *nodes, deployed in the region from Figure 4.11(c) with average degrees* 10 *and* 12*.*

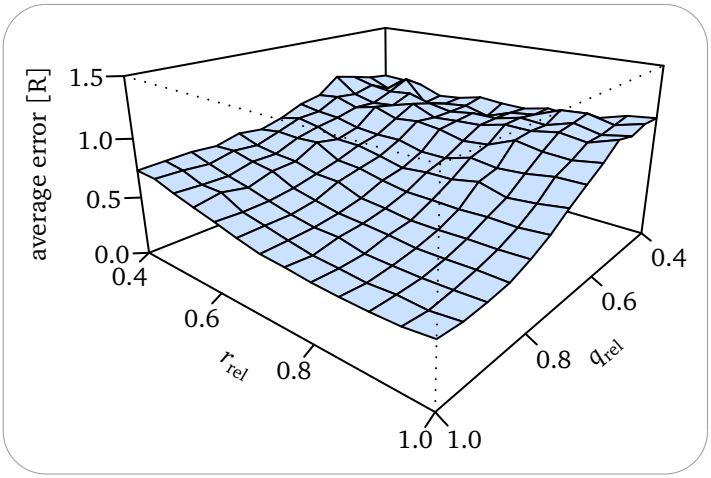


Figure 4.15: *Quality of reconstruction for varying QUDG generation parameters in random networks of* 4000 *nodes, deployed on a grid with perturbation in the region from Figure 4.11(c) with average degrees* 10*.*

Figure 4.15). All graphs could be recovered with comparably low error. We also evaluated networks of sizes with up to 128000 nodes. With increasing network size, not even the *absolute* misplacement worsened, e. g., even for 128000 nodes, the average absolute error did not exceed $0.65R$ for Figure 4.11(c). Consequently, the *relative* misplacement went down to almost zero for large networks. The number of levels in the filtration varied between 5–6 for networks with 1000 nodes and 7–8 for 128000 nodes. This is what one would expect, since for 2D deployment, the diameter only increases by roughly a factor of 10–12.

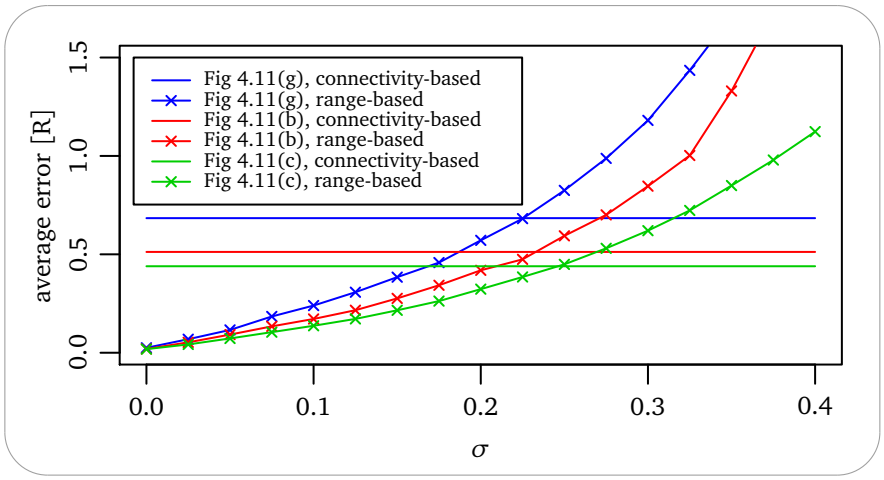


Figure 4.16: *Comparison of range-based and connectivity-based MDS-MAP(F). For a given σ and distance $d(u,v)$, the input is distributed $\sim d(u,v)N(1,\sigma^2)$*

(a) ground truth (b) misplacement vectors (c) reconstruction

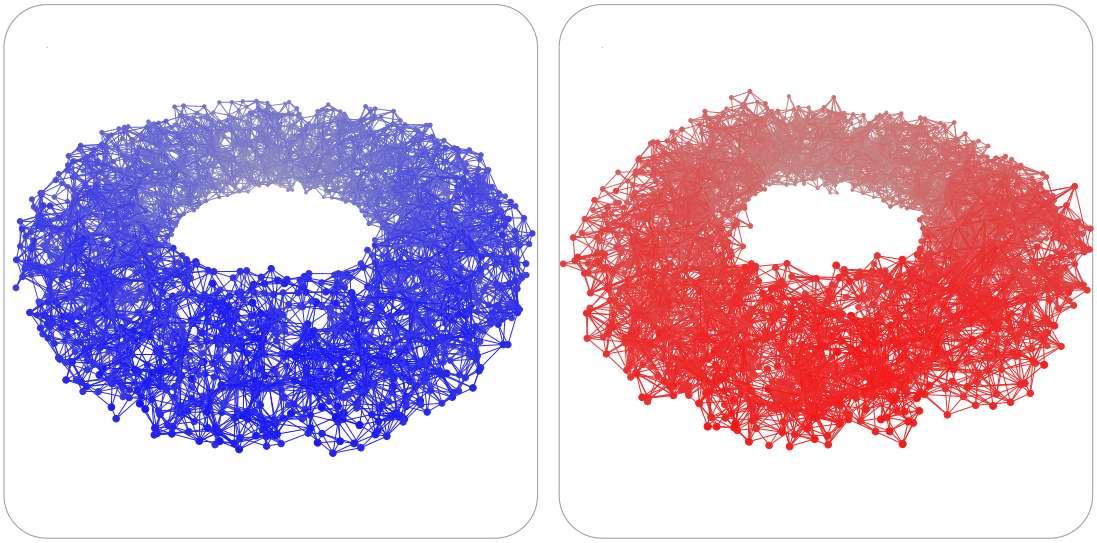
Figure 4.17: *Positioning of a network without a clear boundary*

Benefit of distance measurements

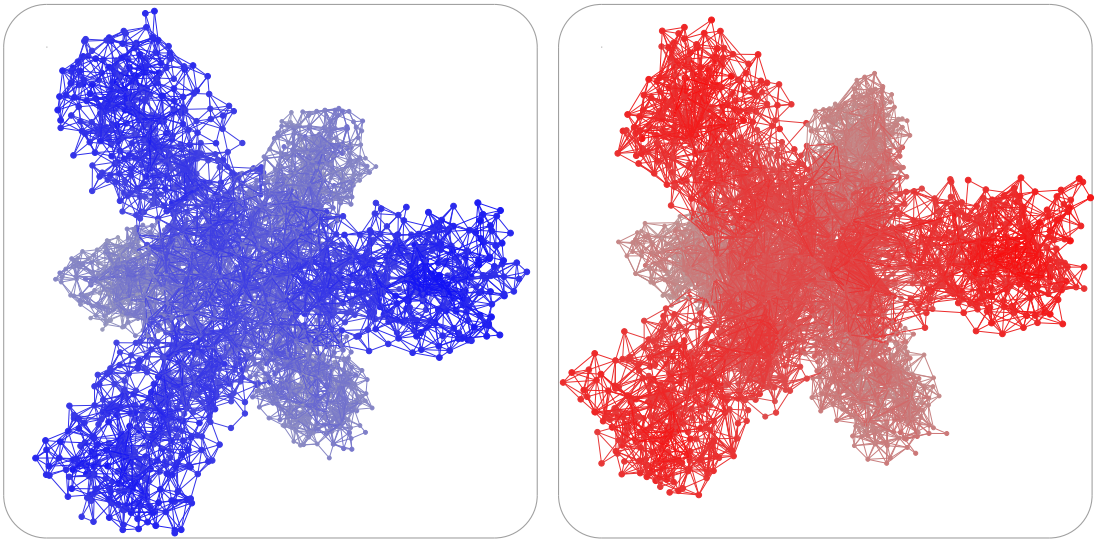
Although not necessary, MDS-MAP(F) can also be fed with distance measurements instead of connectivity information only. Figure 4.16 compares the results for connectivity-based MDS-MAP(F) and the range-based variant. For range-based positioning, the input is normally distributed around the real distance with increasing variance σ. Apparently, distance estimations do only facilitate positioning if the variance of errors is less than some 20%. For higher errors, the results became drastically worse.

Non-uniform deployment and 3D positioning

An important feature of MDS-MAP(F) is the independence of boundary recognition. This provides new opportunities in two cases where boundary recognition is problematic: First, in the case that there is no clear boundary, e. g., if the node density decreases smoothly instead of a sharp cut. Kröller et al. showed that boundary recognition is possible even in this scenario, redlining areas of non-sufficient density [KFPF06]. Nevertheless, currently no positioning algorithm is implemented that can actually handle these scenarios. MDS-MAP(F) in contrast also returns positions in this situation, typically with good results for dense regions and decreasing quality towards sparse regions. An exemplary positioning is depicted in Figure 4.17. An even larger improvement is the possibility to run MDS-MAP(F) on three-dimensional networks. Here, boundary recognition is a largely unsolved problem, and we are not aware of any algorithm that allows for connectivity-based positioning for scenarios as complex as the ones depicted in Figure 4.18.



(a) ground truth (b) MDS-MAP(F) result



(c) ground truth (d) MDS-MAP(F) result

Figure 4.18: *3D positioning:* 2800 *nodes deployed on a grid with perturbation and connected in a torus (top) and in a threedimensional cross (bottom), both with average degree* 10*. Results positioned nodes with average error of less than* 1*R.*

Chapter 5

Coordination of Data-Harvesting Trees

This chapter deals with the problem of setting up energy and time efficient TDMA-schedules to gather bulk data from the data field along a given tree at a central repository. It is motivated by the question whether it is possible for nodes with constant-size memory to agree on valid and time-optimal schedules by only exchanging one single message to each of its neighbors in the tree. These restrictions reflect very natural demands: The most sophisticated schedules for optimum channel utilization are not going to help us, if it is impossible to efficiently gather and spread the information to set it up! It turned out that this question is much more puzzling than it appears, even if schedules are not allowed to contain any concurrent transmissions. This model of *total interference* is the simplest, but at the same time the most energy-efficient and most cautious interference model.

This chapter contains protocols and analyses for two different interference models, the aforementioned model of total interference as well as the k-hop interference model as well as the case that interference can be ignored. It covers algorithms along the tradeoff between energy- and time-optimality and an approach to make all of the proposed algorithms robust to transmission failures in a very strong sense.

5.1 Introduction

Most of today's WSNs are designed to perform a very simple task compared to the vision of SmartDust. Instead of in-network processing, many applications still rely on central instances to gather and evaluate relevant data. Hence, the dominating communication pattern we find in wireless sensor networks is data gathering and/or aggregation along so-called query trees towards a network sink. What seems to be discouraging at a first glance,

can as well be interpreted as the first area of application where WSNs can provide valuable service. Easy deployment without the need for infrastructure and a working mechanism to gather sensed data, sporadically or regularly, in response to queries, or triggered by the detection of an event, provides a valuable tool for monitoring tasks of all kinds. In many of these applications, nodes have to collect bulk data at a central repository. One example are data-archiving applications that periodically sample snapshots at high rates, storing or analyzing them centrally outside the network. Another class are so-called data-harvesting scenarios in which nodes store measured data until from time to time a user requires access to the data stored in a large number of nodes. In low-power networks, these applications demand highly optimized communication management in order to keep the network operable for as long as possible. We thus consider sensor networks where sensor nodes are distributed over a geographic area and measure values in regular time intervals as in [TW07a]. At certain times, the stored data must be routed through the network and collected at a central location, the *sink*. Since the radio communication dominates the energy consumption, minimizing the communication costs is crucial to maximize the lifetime of the sensor network.

There is a number of ways to reduce the power consumption due to communication in this scenario: First by reducing the amount of data to be transmitted, second by improving the routing topology and third by avoiding all unnecessary power consumption of the radio. Although these three issues are not fully independent, we believe that it makes sense to analyze them separately.

Data reduction: Data reduction is an application-specific problem that is largely orthogonal to the other two problems. Data can be compressed using clever representations or sometimes using small-scale in-network processing, but at the end of the day, there is some amount of data that has to be transmitted to the base station. However, we assume there is no further way of data-aggregation, i. e., compression of the data from different sources at intermediate nodes.

Routing topology: From a purely theoretic point of view, the most promising approach to minimize energy when collecting data in a wireless, multi-hop sensor network seems to be the choice of the routing topology, i. e., a specification how many packets to transmit over each link. This can obviously be modelled as a network flow having all the nodes with data packets as sources and the sink as the single target, which would allow for the optimization of different objectives. Maintaining complicated routing topologies on the other hand is a complex task that would take a lot of communication, time and energy and is uncommon in real-world networks for a good reason. Even optimizing routing trees, which would also be an interesting task in this scenario, is more of academic nature: Trees typically used for data aggregation and harvesting tasks are built using very lightweight protocols. Therefore, we will assume a given routing tree.

Avoidance of unnecessary power consumption: A third factor in energy consumption is the energy actually spent to route a packet to the sink. It is lower bounded by the path the packet takes in the network, but the effective cost of routing a packet is mostly much higher due to collisions, packet drops due to limited buffer sizes and problems that arise from a

lack of coordination, namely *idle listening*, i. e., nodes listening to an idle channel, *overhearing*, i. e., nodes receiving and decoding messages they are not meant to, and *overemitting*, i. e., transmission of packets when the receiver is not ready [DEA06]. Hence, orchestrating transmissions in data gathering is a realistic way to save energy.

We will thus focus on the problem of avoiding all unnecessary power consumption by the radio for a fixed routing tree provided by some arbitrary protocol.

As argued in Section 2.1.2, schedule-based protocols outperform contention-based protocols both in terms of energy- and time-consumption. They do not suffer from energy drain due to idle listening and do not have to waste energy and time for retransmitting packets lost due to collisions. Moreover, with standard CSMA-protocols, there is no mechanism to take each node's limited memory into account. Nodes with two or more children will typically receive packets faster than they can route them towards the sink. This leads to packet drops—and again the need for retransmissions, and can only be prevented using some kind of flow control with additional protocol overhead.

While scheduled media access provides a perfect orchestration of communication when data is routed towards the sink, the drawback obviously is the increased protocol overhead for schedule set-up. Hence, we will analyze protocols to set up schedules with a minimum of set-up communication. This analysis is conducted for different interference models. Additionally, a drawback of fixed schedules is, that they do not leave room for spontaneous retransmissions if packets are lost due to link failures. We will address this issue with the analysis of a generic approach to make schedules robust to transmission failures at the cost of reasonable small additional memory and time consumption.

5.1.1 Related Work

There is a plethora of algorithms for finding topologies (or routing information) for the data gathering problem in sensor networks. There are several goals for optimization, for example throughput, latency, reliability, security and energy consumption, the last one being the most important in sensor networks. Almost all of these different approaches construct one or sometimes several routing *trees*.

There has been previous work on minimizing the *time* for data gathering. In [BGK+06] a problem similar to ours is studied, a 4-approximation algorithm is given and NP-hardness of the problem shown. A problem with variable release times is studied in [BKMSS06]. Unlike this previous work, however, we focus on distributed algorithms and take set-up cost into account. Also, we do not concentrate so much on time optimality but on *energy* optimality.

One of the very few contention-based MAC protocols that take advantage of the tree topologies present in data-archiving systems is [LKR04]. The wake-up times of nodes are staggered on paths towards the sink, which reduces latency. Measures are employed to reduce interference among packets travelling along the same paths. Additionally, special "More-to-Send" packets are used to further synchronize wake-up times and thus increase throughput. However, this protocol is designed for very low data rates. It is not energy-optimal and there are no special mechanisms to reduce congestion and ensure fairness. Flexible Power Scheduling (FPS) is described in [HDB04]. In FPS parents are responsible

for assigning time slots to their children. FPS reduces contention but does not guarantee collision-free communication. Therefore, an underlying MAC layer is still required. Fairness among children in different branches is not ensured. MPS (Multi-Flow Power Scheduling) and HPS (Hybrid Power Scheduling) are enhancements on FPS introduced in [YAGS06]. MPS is closely related to the k-hop interference protocol in Section 5.5, but performance is only evaluated experimentally and there is no theoretical analysis of the protocols. Also, the interference model is not described explicitly.

The authors of [TW07b] address the problem of congestion, fairness and robustness during the transport of high volumes of sampled data. They use the total interference model (cf. Section 5.4). Their approach is based on a slot distribution scheme. Nodes that have no more packets to send can pass their slots back to their parent. Every second slot that a node receives from its children is passed on to its parent. This aims at distributing slots more fairly: Nodes with high loads get more slots. In [TW07c] a further refinement of slot distribution strategies are developed. But even with these refinements, the channel usage is still fairly low and decreases with growing network size. In the same paper, there is another scheme (similar to ours in Section 5.4) which, however, leads to unlimited buffer size and the control message overhead is not analyzed.

DOZER ([BvRW07]) is an approach that tries to solve the problems of medium access, tree construction and scheduling together. The authors employ a local TDMA scheme which reduces requirements on clock synchronization but does not ensure fairness. Collisions are reduced by letting schedules of interfering node pairs "drift apart" through randomization. This approach is designed for scenarios with very low data rates but causes problems when there are higher volumes of data to deliver. In contrast to these approaches, we focus on a detailed theoretical analysis of throughput and protocol overhead. Our approaches are also designed for arbitrarily large networks and high data load.

5.2 Problem Statement

The problem addressed in the remainder of this chapter can be formalized as follows: We assume that we are running a sensor network consisting of a set of nodes V, one node r serving as a sink that is connected to some infrastructure. Each of the other nodes has an individual number $\pi(v)$ of own data packets to transmit to the sink. Within the network, a spanning tree T, rooted at the sink r is provided by some standard protocol, i. e., every node knows its parent and a list of its children. Throughout the following sections, we will assume that every node (except the sink, for which we assume $\pi(r) = 0$) has at least one data packet. We will shortcut the packet count

$$\pi\left(V'\right) := \sum_{v \in V'} \pi(v) \quad \text{and} \quad N := \pi(V) \tag{5.1}$$

We will also write $\tau(u) := h_T(u)\pi(u)$ for the number of transmissions induced by any node's packets and shortcut

$$\tau(V') := \sum_{v \in V'} \tau(v) \quad . \tag{5.2}$$

Now, we consider data gathering protocols that divide into the following two phases:

Set-up phase: In order to set up a schedule, nodes can exchange packets with their direct neighbors in the tree, i. e., with their parent and their children. We do not assume a specific medium access to be used during this stage.

Collecting phase: With the parameters established in the set-up phase, every single data packet must be routed to the sink using slotted media access only. We assume synchronization by any external protocol.

Following the considerations from above, we restrict set-up communication and node capabilities to the following protocol requirements:

Definition 5.1 (Constant Communication Overhead) *A data gathering protocol meets the requirement of* constant communication overhead (CCO) *if*

1. *During the set-up phase, every node sends at most one single packet of size $O(\log N)$ bits to each of its neighbors in the tree T.*
2. *During the collection phase, no protocol information is exchanged.*
3. *During the collection phase, no node has to buffer more than a constant number of data packets from other nodes.*

Within the bounds of CCO, we are only considering deterministic protocols that are energy-optimal, i. e., protocols that are provably collision-free and that do not involve idle listening. As a secondary criterion, we optimize the collection time, i. e., the time until all packets arrive at the sink. While energy-optimality depends on the routing tree only, time-optimality is also a question of the interference model telling whether or not two transmissions can be performed concurrently.

Definition 5.2 (Time optimality) *For a given interference model, a protocol is* time optimal *if the collection time is minimal among all possible schedules, without any restriction to set-up communication or computation. This minimum is denoted by $|S_{OPT}|$. A protocol is f-time-approximative if the collection time exceeds the minimum by less than a factor of f.*

We can narrow the structure of protocols complying with the limits of CCO by the observation, that every compliant protocol can be seen as first an aggregation towards the sink and second a flooding of schedule parameters:

Observation 5.3 *Every protocol that complies with the requirements of CCO and collects all data can be set up using one convergecast and one broadcast along the tree T, i. e., the set-up must always be performed as follows:*

1. *Leaf nodes send a packet to their parents. Each inner node waits until it received packets from all its children before sending a packet to its parent (convergecast).*

2. *When the sink received packets from all its children, it sends a packet to all children. Each other node waits until it received a packet from its parent before sending packets to its children (broadcast).*

Proof. It is a necessary condition for all protocols that the sink knows the number of packets after the set-up phase. This implies that every node has to receive a packet from each of its children before sending a packet to its parent. But assume that any protocol, there arises the situation in which a node u decides to transmit to any of its children before it receives a message from its parent. Obviously, the outcome of this protocol does not change if u delays this message until it received the message from its parent. □

5.3 Scheduling in Absence of Interference

The first situation we consider data gathering in is the absence of interference between concurrent transmissions. This most optimistic model not only sets natural lower bounds on the limits of any other interference model, but could also be used for hybrid MAC access: Setting up a TDMA-schedule for transmission times can be used for flow control, i. e., to ensure that no node has to receive more data per time unit then it can transmit. Within larger time slots than in pure TDMA-schedules, each transmissions can then be performed using a CSMA-based medium access.

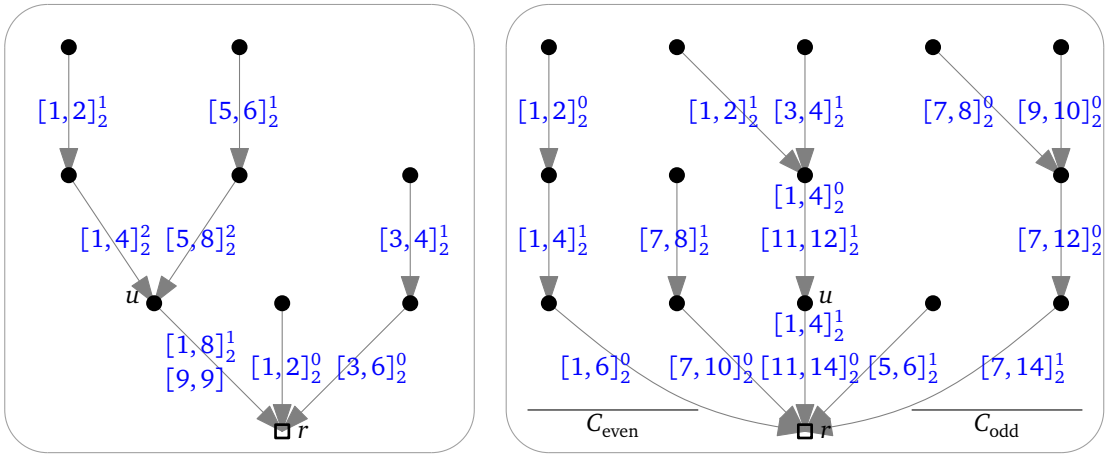
The only lower bounds that hold even in the absence of interference and thus in any interference model are the following:

Observation 5.4 *In any interference model that does not allow a node to be part of more than one transmission at a time, optimal collection time is*

$$|S_{OPT}| = \max\left\{N, \max_{c \in V-r}\left\{2\pi_T(v) - \pi(v)\right\}\right\} \quad . \tag{5.3}$$

Proof. A lower bound of N trivially follows from the fact that the sink needs one time slot for the reception of each packet. A lower bound of $\max_{c \in V-r}\left\{2\pi_T(v) - \pi(v)\right\}$ follows from the fact that every node v except the sink has to receive all packets from its descendants and transmit them to its parent—except for its own packets which need no reception. We combine the proof that this bound is tight with the proof of the following proposition. □

Indeed, it is possible to agree on a time-optimal schedule within the bounds of constant communication overhead. For convenience, we define for $a, b, d \in \mathbb{N}_0$ and $m \in \mathbb{N}$ $[a,b]_m^d := \{i \in [a,b] : i = d \mod m\}$, e. g., the set of odd numbers between 3 and 10 can be denoted as $[3,10]_2^1$.



(a) One single node, u, has more packets to relay than all other nodes have to send. Collection time is $9 = 2\pi_T(u) - \pi(u) > N$

(b) No single node has $\pi_T(\cdot) > N/2$. The sink's children form two groups for transmissions to sink in odd or even time slot, one child u uses the last even and the first odd slots.

Figure 5.1: *Optimal slot assignments in the absence of interference*

Proposition 5.5 *There is a time-optimal protocol meeting the requirements of CCO in the absence of interference.*

Proof. To agree on a time-optimal schedule, it is sufficient to let every node u learn the number of packets to receive from each of its children c, $\pi_T(c)$ as well as the number of packets that originate at each child, $\pi(c)$, during the convergecast. Now, the sink has to distinguish the two cases according to Observation 5.4. If one of the sink's neighbors is a bottleneck, i. e., if $N \leq 2\pi_T(u) - \pi(u)$ for some child u of the sink, the sink can safely assign every other child a part of the time interval $[1, 2(\pi_T(u) - \pi(u))]$ to transmit in slots with even numbers. These children themselves assign parts of this interval to their children, such that they send in slots with odd numbers and so on. For u, the situation is slightly different. It assigns its children parts of the above interval, such that they send in slots with even numbers and may itself send to the sink in slots with odd numbers within that interval and in all the slots in the interval $[2(\pi_T(u) - \pi(u)) + 1, 2\pi_T(u) - \pi(u)]$. By construction, every node but u and the sink only receives in odd and transmits in even slots or vice versa, and u only breaks this pattern for the last $\pi(u)$ transmissions which do not overlap with any other transmissions. This situation is depicted in Figure 5.1(a).

If, on the other hand the time to complete is bounded by the number of packets, i. e., if $N > 2\pi_T(u) - \pi(u)$ for any child $u \in C_T(r)$, it is sufficient to assure that in every slot there is some node transmitting to the sink. Here, we can exploit that in this case, $\pi_T(u) < N/2$ for all neighbors of the sink. It is thus possible to pick any of the sink's children u and arrange the other children into two groups, C_{even} and C_{odd} such that $\sum_{c \in C_{\text{even/odd}}} \pi_T(c) \leq N/2$.

Now, nodes in C_{even} are allowed to send in slots $[1, 2\sum_{c \in C_1} \pi_T(c)]_2^0$, nodes in C_{odd} send in slots $[N - 2\sum_{c \in C_2} \pi_T(c) + 1, N]_2^1$ leaving enough slots for u that pairwise have distance

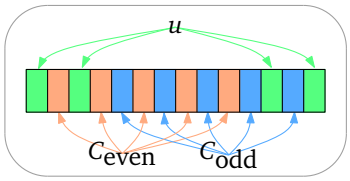


Figure 5.2: *Slot distribution for the sink's children*

at least 2 as depicted in Figure 5.2. Again, every node can divide the intervals to send in among its children to send in, swapping odd and even, and leaving some slots at the end unassigned. Since in both cases every node is assigned at most two intervals to send in either even, odd, or all slots of these intervals, packets of $O(\log N)$ bits are sufficient. The resulting schedule is depicted in Figure 5.1(b). □

5.4 Scheduling for Total Interference

Going to the other extreme, we turn our attention to the model of total interference, in which no two nodes are allowed to transmit concurrently.

Definition 5.6 (Total interference) *Under* total interference*, a transmission between a pair of nodes is successful if and only if there is no concurrent transmission by any node in the whole network.*

Total interference is the most pessimistic interference model, but at the same time a reasonable choice for energy-efficient scheduling: Only in the absolute absence of concurrent transmissions, lower bounds for energy consumption can be met in the strongest sense: Since apart from background noise there is no interference to compensate for, even the power level for transmissions can be chosen minimally. Additionally, solutions for this most restrictive model are always applicable under any other model.

5.4.1 Lower Bounds and Infeasibility

Observation 5.7 *Under total interference, optimal time to complete is*

$$|S_{OPT}| = \sum_{v \in V} \tau(v) \quad . \tag{5.4}$$

Proof. Follows trivially from the definition of $\tau(v)$. □

Apparently, it is much easier to find a time optimal schedule for total interference than for the absence of interference, disregarding the restrictions of CCO. To name just one way to set up such a schedule, we can number the packets and hand one packet after another

down to the sink. Unfortunately, such a schedule cannot be set up within the restrictions of CCO.

Proposition 5.8 *There is no time-optimal protocol meeting the requirements of CCO for total interference in which each packet is immediately passed to the sink once it has started from its originating node.*

Proof. During the set-up phase, the sink has only received $k\delta_T(r)\log N$ bits for some constant k. For every packet arriving at r, the height of the originating node can be determined by the time since the arrival of the preceding packet. As r is only allowed to be awake when packets arrive, it must know the times of arrival in advance. This implicates knowing how many packets to receive from each height. However, even if we regard trees as equivalent that have the same number of packets at any height, i. e., for

$$T \sim T' :\Leftrightarrow \sum_{\{v:h_T(v)=h\}} \pi(v) = \sum_{\{v:h_{T'}(v)=h\}} \pi(v) \text{ for all } h \geq 0 \ ,$$

there is an exponential number of equivalence classes of trees even restricted to those with a maximum degree of 2, which can be written as sequences of positive integers with sum N, such that the first number is at most 2 and each number is at most twice its predecessor. Two trees from different equivalence classes demand for different behavior of r, but r can distinguish at most

$$e^{2k\log N} = N^{2k}$$

different situations after set-up, which is polynomial. □

There seems to be plenty of room for other solutions, but the problem remains that in any protocol, there is an exponential number of possible subtrees which is indistinguishable from an exponential number of structurally different subtrees. All these subtrees have to behave identically, being idle, performing an "inner" transmission or performing a transmission by the subtree's root to its parent at the same time.

We thus conjecture that there is no way to agree on optimal schedules at all.

Conjecture 5.9 (Infeasability of CCO for total interference) *There is no energy- and time-optimal protocol meeting the requirements of CCO for total interference.*

For total interference, we will thus show how good protocols complying with the restrictions of CCO can perform and how practicable relaxations of the CCO requirements allow for better performance.

5.4.2 Protocols with Constant Communication Overhead

Within the restrictions of CCO, set-up messages can contain at most a constant number of counters for numbers up to N. They can be used for different approximations with respect to the height of the tree:

Proposition 5.10 *For any constant $d \in \mathbb{N}$, there is a protocol for total interference that meets the requirements of CCO and has time approximation factor* $\max(1, h_T/d)$.

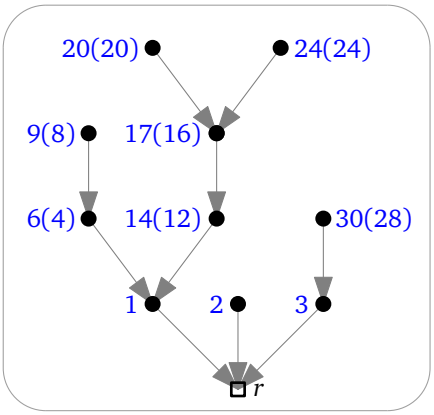


Figure 5.3: *Start slots and virtual start slots (in parentheses) for $d = 2$*

Proof. We will show that passing a constant number of counters during the convergecast and broadcast is sufficient to gather all packets from nodes v with $h_T(v) < d$ in optimal time and all other packets in time h_T per packet, yielding the claimed approximation. To this extent, every node u learns how many packets to relay from the $d-1$ higher levels $h_T(u)+1$ to $h_T(u)+d-1$ and how many packets to relay from levels d and higher together. Denoting u's descendants in levels i to j as $D_i^j(u)$ and shortcutting $D_i(u) := D_i^i(u)$, after the convergecast, every node u knows $\pi(D_{h_T(u)+i}(u))$ for $i = 0,\dots,d-1$ and $\pi(D^\infty_{h_T(u)+d}(u))$. With one additional counter it is possible to let the sink learn the height of the tree.

The sink can now determine time slots t_1 to t_d as $t_1 = 1$ and $t_i = t_{i-1}+(i-1)\pi(D_{i-1}(r))$. Also, it is obvious how to use the broadcast to let every node u know its height $h_T(u)$ and how many packets in its subtree come from levels $h_T(u)$ to $d-1$ (if $h_T(u) \le d-1$) and how many packets come from levels d and higher, i. e., every node u learns during broadcast $\pi(D_i(u))$ for $i = h_T(u),\dots,d-1$ and $\pi(D_d^\infty(u))$. The idea now is to start packets in level $1 \le i < d$ in slots $t_i + ki$, $0 \le k < \pi(D_i(r))$ and pass them directly to the sink. To this extend, every node u with $h_T(u) < d$ learns times $t_i(u)$ when the first packet in its subtree starts in level i: The sink assigns to each of its children $c_j \in C(r)$ the slots $t_i(c_j) := t_i + i\cdot\sum_{j'<j}\pi(D_i(c_{j'}))$, which analogously assign the starting slots to their children. Knowing these times, every node v knows when to start its own packets, namely in time slots $t_{h_T(v)} + kh_T(v)$, $0 \le k < \pi(v)$, and when to receive packets to relay started at a level $h_T(v) < i < d$, namely in slots $t_i + (i - h_T - 1) + ki$, $0 \le k < D_i(v)$.

The idea for packets from levels d and above is starting them "virtually" at times t_d+kh_T, $0 \le k < D_d^\infty(r)$ from height h_T in the following sense: If a packet originating at some node u with $h_T(u) \ge d$ has the virtual start time $t_d + kh_T$ for some k, the time slots $t_d + kh_T$ to $t_d + kh_T + h_T - h_T(u)$ are unused, and the packet really starts at time $t_d + kh_T + h_T - h_T(u)$. This way, it is sufficient to number packets from levels d and higher in preorder and let every node u know the lowest number in its subtree, $p(u)$, which is easy given that every node u knows $\pi(D_d^\infty(u))$. Then, every node knows the virtual starting slots for all these nodes, namely $t_d + kh_T$ to $t_d + kh_T$ for $p(u) - 1 \le k < (p(u) + \pi(D_d^\infty(u)) - 1$. It can use the

first $\pi(u)$ for own packets if $h_T(u) \geq d$, and has to relay packets from the the following. □

We will in the following refer to this method to gather packets from nodes with $h_T(\cdot) < d$ in optimal time and other packets in time h_T per packet as *d-LevelOrderCollect*. An example for $d = 2$ is depicted in Figure 5.3. Every node is annotated with the slot in which it releases its single packet, for nodes with height $h_T(\cdot) \geq d$ also the virtual start slot is given. All packets are handed down to the sink directly. The following corollary will be useful for k-hop interference to be discussed later in this chapter.

Corollary 5.11 *There is a time-optimal protocol for total interference under the restrictions of CCO for trees with heights bounded by a constant.*

It is possible to show an even better approximation for uniform packet counts:

Proposition 5.12 *If all nodes have the same number of packets, i. e., $\pi \equiv c$, d-LevelOrderCollect has a time approximation factor of* $\max\{1, 2\sqrt{n/d}\}$.

Proof. Without loss of generality, we assume $\pi \equiv 1$. Now fix some d. If for some tree T, $h_T < \sqrt{nd}$ or $h_T < d$, the claim follows from Proposition 5.10. We thus consider only the case that $h_T > \sqrt{nd}$ and $h_T > d$.

The worst-case scenario is a path of length h_T plus some additional nodes at height d, i. e., $N - h$ children of the $(d-1)$th node on the path. *d-LevelOrderCollect* here needs time

$$|S_{d\text{-}LOC}| = (d-1) \cdot d/2 + (n-d+1)h_T < nh_T - (d-1) \cdot (h_T - d/2) < nh_T \tag{5.5}$$

to complete collection. The optimal time is

$$|S_{\mathrm{OPT}}| = h_T(h_T+1)/2 + (n-h_T)d \ . \tag{5.6}$$

Hence, the approximation ratio is

$$\frac{|S_{d\text{-}LOC}|}{|S_{\mathrm{OPT}}|} < \frac{nh_T}{h_T(h_T+1)/2 + (n-h_T)d} < \frac{nh_T}{h_T^2/2} \ . \tag{5.7}$$

And we get with $h_T > \sqrt{nd}$

$$\frac{|S_{d\text{-}LOC}|}{|S_{\mathrm{OPT}}|} \leq \frac{nh_T}{h_T\sqrt{nd}/2} = 2\sqrt{n/d} \ . \tag{5.8}$$

□

For the standard scenario where nodes are deployed sufficiently dense and more or less uniformly, this very simple protocol is only a constant factor away from being optimal if all nodes within some range of the sink have to deliver a similar number of packets to the sink:

Proposition 5.13 *For query trees with $\Theta(\ell)$ packets in every level (and hence a height in $\Theta(\sqrt{n})$), d-LevelOrderCollect is $O(1)$-time-approximative for any d.*

Proof. Let $\pi^- : \min_{1\le\ell\le h_T} \pi\left(D_\ell(r)\right)$ and $\pi^+ : \max_{1\le\ell\le h_T} \pi\left(D_\ell(r)\right)$. Then, $|S_{\mathrm{OPT}}|$ is bounded by

$$|S_{\mathrm{OPT}}| \geq \pi^- \sum_{\ell=1}^{h_T} \ell^2 \geq \pi^- h_T^3/3 \ .$$

d-LevelOrderCollect in turn uses (even for $d = 1$)

$$|S_{d\text{-}LOC}| \leq \pi^+ h_T \sum_{\ell=1}^{h_T} \ell \leq \pi^+ h_T \left(h_T^2 + h_T\right)/2 \leq 2\pi^+ h_T^3 \ ,$$

i. e., is $6\pi^+/\pi^-$-time-approximative. □

Until now, every node's knowledge of the tree has been restricted to the knowledge of its parent and children. However, many protocols establishing a routing tree let nodes also know their distance to the sink, i. e., their height. This can be used to use a constant number of counters more cleverly:

Proposition 5.14 *If every node v knows its height $h_T(v)$, for every constant $d \in \mathbb{N}$ and $m > 1$ there is a protocol meeting the requirements of CCO that has time-approximation factor* $\max(m^{1/d}, h_T/m)$.

Proof. We set $a := m^{1/d}$. With $d+1$ counters, it is possible to let each node v know the numbers $\pi(D_0^{\lfloor a^i \rfloor}(v))$ for $i = 1,\dots,d$ and $\pi(D(v))$. Now, using the same idea as in *d-LevelOrderCollect*, all packets from the levels $\lceil a^{i-1} \rceil$ to $\lfloor a^i \rfloor$ are accounted for with a virtual height of $\lfloor a^i \rfloor$. This way, a packets from a node u with $h_T \leq \lfloor a^d \rfloor = m$ accounts for a number of slots that exceeds the optimum by at most a factor of $\lfloor a^i \rfloor / \lceil a^{i+1} \rceil \leq a = m^{1/d}$. Packets from levels higher than m have, as in *d-LevelOrderCollect*, virtual height h_T, accounting for at most h_T/m times the optimum time. □

In the case that the height of tree is known, this can be used to set $m = h_T$, yielding a $h_T^{1/d}$-time-approximation.

5.4.3 Time-Optimality with Increased Packet Size

Relaxing the requirement of not sending any routing information during the collection phase, there is a way to achieve time-optimality with $\mathrm{ld}(h_T)$ additional bits per packet.

Theorem 5.15 *There is a protocol that completes collection in optimal time and meets the requirements 1 and 3 of CCO, but in which every node v sends $O(|\pi_T(v)| + |D(v)| \log h_T)$ bits of routing information in the collection stage.*

Proof. We will show how to set up and run a schedule, in which the nodes start their packets in pre-order, which then are routed directly to the sink. During set-up, all a node has to learn is the slot for its first transmission as well as its height, which can obviously be achieved using a constant number of counters: During the convergecast, every node u learns the

number of packets in each child c's subtree as well as the number of transmissions necessary to bring them to c, i. e., $\pi(D(c))$ and $\tau(D(c)) - h_T(c) \cdot \pi(D(c))$. This is trivial if c is a leaf, where $\pi(D(c)) = \pi(c)$ and $\tau(D(c)) - h_T(c) \cdot \pi(D(c)) = 0$. Every inner node v simply sets $\pi(D(v)) = \sum_{c \in C(v)} \pi(D(c)) + \pi(v)$ and

$$\tau(D(v)) - h_T(v) \cdot \pi(D(v)) = \sum_{c \in C(v)} \left(\tau(D(c)) - h_T(c) \cdot \pi(D(c)) + \sum_{c \in C(v)} \pi(D(c)) \right) . \quad (5.9)$$

Then, during the broadcast, every node v learns its height—and hence the $\tau(D(c))$ for all its children—and its starting slot: The sink assigns itself the imaginary slot 1 and every node u with a first transmission in slot t_u now assigns its first child $c_1$the starting slots $t_u + \pi(u)h_T(u)$, its second child the starting slot $t_u + \pi(u)h_T(u) + \tau(D(c_1))$ and so on. This provides enough information to start packets timely, but nodes do not know when to receive and relay packets. All knowledge we use in the following is, that if a node u has a child, it has to relay its first packet $h_T(u)$ time slots after it released its last own packet. With this information, we extend data packets during collection phase with the information of how long nodes relaying this packet have to wait for the next packet to relay—this way we ensure that at any time, all nodes that have to relay the next packet know its arrival in advance. This is true when the first packet starts, since it is always a packet from the sink's first neighbor to the sink. Whenever a packet starts at some node v having more packets, the node simply adds the information that the next packet will follow $h_T(v)$ slots later. If the node starting the packet has no more packets to send, but at least one child, it can announce the next packet to follow in $h_T(v) + 1$ slots. The interesting case is that a leaf sent its last packet. In this case, the leaf starting the packet cannot announce the next packet. It hence flags the packet to have an unknown next-packet delay. Every node v receiving a packet without a next-packet delay for at most the $(|C(v)| - 1)$th time changes it to $h_T(v) + 1$. In this case, v is the first node on the packet's way, for which the packet is not the last one to relay: The next packet to relay will come from one of v's children. Using a constant number of bits to encode an unknown packet delay and a repeated packet delay (i. e., a more-to-send flag), only the last packet of each node has to carry $O(\log h_T)$ bits routing information on its way. □

5.5 Scheduling for Hop Interference

In this section, we consider a slightly less pessimistic model of interference, which allows for concurrent transmissions unless they are close in terms of graph distance.

Definition 5.16 (k-hop interference) *Under k-*hop interference*, a transmission between a pair of nodes (u, v) is successful if and only if there is no concurrent transmission by any node within v's k-hop neighborhood $N_G^k(v)$.*

This model is a reasonable, yet a cautious approximation of interference in dense networks, in which Euclidean distance and hop-distance closely correlate. Quite typically, routing trees for data gathering or aggregation in sensor networks are set up using some kind of request flooding, i. e., the routes between a node and the sink are shortest paths (in graph distance). This is not only a very lightweight, robust protocol, but also guarantees that data travels on short routes, which reduces the risk of packet loss. In such trees, a transmission (u, v) is always successful if u is the only active sender among all nodes with $|h_T(v) - h_T(u)| \leq k$. We call this property of a rooted tree *k-layer bounded interference*:

Definition 5.17 (k-layer bounded interference) *A tree has* k-layer bounded interference *if a transmission $(u, p_T(u))$ is successful if there is no concurrent transmission by any node v with $|h_T(v) - h_T(p_T(u))| \leq k$.*

5.5.1 k-LS

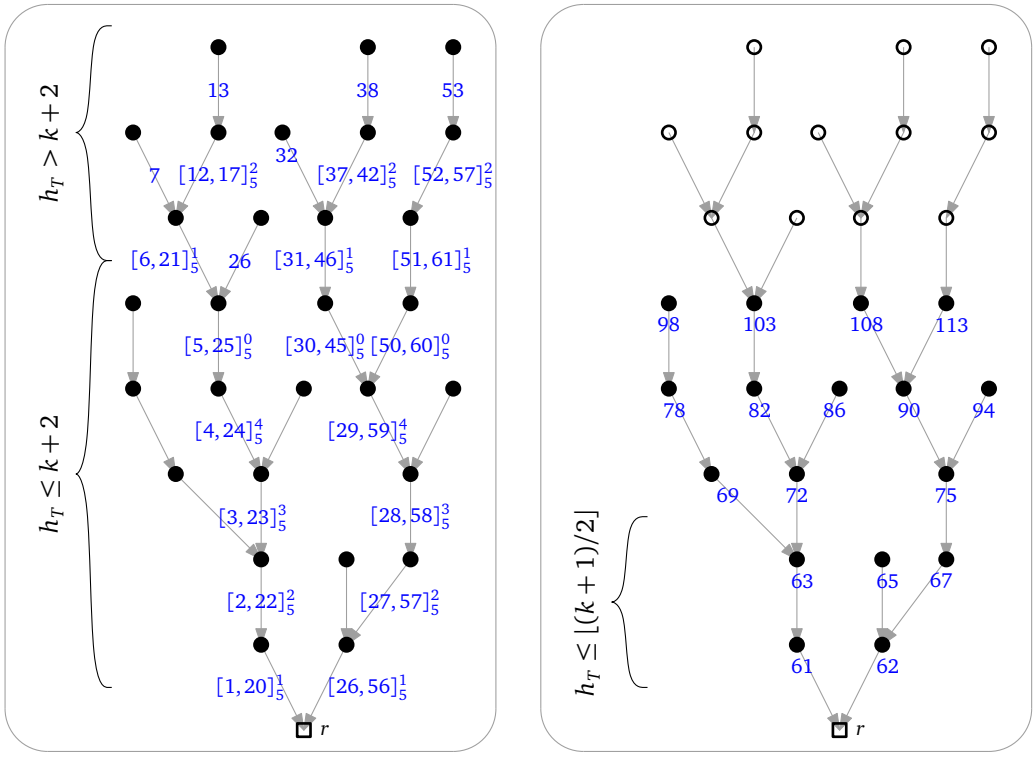
Proposition 5.18 *In the k-hop interference model, there is a protocol meeting the requirements of CCO for trees with k-layer interference that is time approximative within a factor of $k + 2/\lfloor (k+1)/2 \rfloor \leq 4$.*

Proof. The key idea of k-LS is to set up two collection phases that are performed successively, similar to *d-LevelOrderCollect*, but in reversed order: In the first phase data from notes with height $h_T(\cdot) > k + 2$ is "pipelined" towards the sink ending in a state where no node v having a height of more than $k + 2$ has any packets left. In more detail, the sink's neighbors pass packets to the sink every $k + 2$ slots starting with the first slot and sending one after another. Whenever a node transmits a packet, it receives a packet in the next slot from one of its children as long as its children have more packets. Every node v passes $\pi(D^{\infty}_{k+3}(v))$ packets, i. e., as many packets as there are in its subtree in height $k + 3$ and above. In the second phase, data is collected from the remaining nodes according to Proposition 5.10: Setting $d := k + 2$, it is possible to gather all remaining packets in time $\tau(D^{k+2}_1(r))$ (which was optimal for total interference).

During the convergecast, each node v learns how many packets to relay from nodes with heights $h_T(v), \ldots, h_T(v) + k + 2$ and from more distant levels. Obviously, a message size of $(k + 3)\mathrm{ld}N$ bits is sufficient to achieve this. In the broadcast phase, each node v learns its height $h(v)$ and a starting slot $t_s(v)$ to be described later. Given its height, a node knows how many packets to relay that do originate in nodes $u \notin D^{k+2}_1(r)$, i. e., $\pi\left(D^{\infty}_{k+3}(r)\right)$. The starting slot now is assigned as follows: The sink assigns itself the (imaginary) starting slot $t_s(r) = 0$, and along with the height, every node v assigns start slots

$$t_s(c_i) := t_s(v) + 1 + (k + 2) \cdot \sum_{0<j<i} \pi\left(D^{\infty}_{k+3}(c_j)\right) \tag{5.10}$$

to each of its children c_i. In the pipelining phase, each node v now transmits in slots $t_s(v) + i(k + 2)$, $i = 0, \ldots, \pi(D^{\infty}_{k+3}(v))$. It receives a packet in the following slot the first $\sum_{c \in C(v)} \pi(D^{\infty}_{k+3}(c))$ times. This process is depicted in Figure 5.4(a) for $k = 3$ and $\pi \equiv 1$. This part of the collection phase completes after $(k + 2)\pi(D^{\infty}_{k+3}(r)) + 1$ slots. Nevertheless,



(a) active slots for pipeling phase—this phase completes in $(k+2)\pi(D^{\infty}_{k+3}(r)) + 1 = 61$ slots.

(b) release times in second phase—this phase completes in $\tau(D_1^{k+2}(r)) = 57$ slots in slot $60 + 57 = 117$

Figure 5.4: *The two phases of k-LS for* $k = 3$

for the analysis, we can ignore the last slot, in which a last packet is sent from a node with height $k+3$, since this transmission can overlap with the first transmission of the second part from a neighbor of the sink and the sink. After completion, no node with height more than $k+2$ has any packets left and each node v in $D_1^{k+2}(r)$ has exactly $\pi(v)$ packets as in the beginning.

For the second phase of collection, nodes in $D_1^{k+2}(r)$ additionally receive the starting slots according to Proposition 5.10 (with $d = k+2$) and the number of packets that are pipelined in the first phase, $\pi(D^{\infty}_{k+3}(r))$ during set-up. With a delay of $(k+2)\pi(D^{\infty}_{k+3}(r))$ slots, nodes in $D_1^{k+2}(r)$ now are collected at the sink in $\tau(D_1^{k+2}(r))$ slots. Starting slots for this phase are depicted in Figure 5.4(b)

Again, since there is no idle listening, this collection scheme is energy optimal. To show an approximation factor of $(k+2)/\lfloor (k+1)/2 \rfloor$, we observe that for $k' := \lfloor (k+1)/2 \rfloor$, no two nodes in $D_1^{k'}(r)$ can transmit at the same time to their parent in any valid schedule, because for any $u, v \in D_1^{k'}(r)$, $d_G\left(u, p_T(v)\right) \leq k$. Thus, in an optimal schedule, each packet originating at a node with height of more than k' accounts for at least those k' slots where it is the only one transmitted by a node with height of k' or less in any valid schedule.

Similarly, every packet originating at a node $v \in D_1^{k'}(r)$ accounts for at least $h_T(v)$ slots. We hence get a lower bound for the completion of an optimal schedule of

$$|S_{\text{OPT}}| \geq \sum_{v \in V} \pi(v) \cdot \min\left(h_T(v), \lfloor (k+1)/2 \rfloor\right) \quad . \tag{5.11}$$

The collection phase of *k-LS* in turn uses $k+2$ slots for every node with height more than $k+2$ in the pipelining part and $h_T(v)$ slots for every other node. This can be written as

$$|S_{k\text{-}LS}| = \sum_{v \in V} \pi(v) \cdot \min\left(h_T(v), k+2\right) \quad , \tag{5.12}$$

which directly proves the approximation of $(k+2)/\lfloor (k+1)/2 \rfloor \leq 4$. □

Applying very similar arguments as above, we can observe that *k-LS* completes collection in optimal time if the first $\lfloor (k+3)/2 \rfloor$ levels of t form a single path. Second, if T is a hop-shortest path tree in the sink's $\lfloor (k+3)/2 \rfloor$-hop neighborhood, *k-LS* is time approximative not only compared to an optimal schedule of T, but for among all schedules of all spanning trees:

Corollary 5.19 *The schedule produced by k-LS is optimal for the given tree if $D_0^{\lfloor (k+3)/2 \rfloor}(r)$ forms a path in T and time approximative within a factor of $\frac{k+2}{\lfloor (k+1)/2 \rfloor}$ among all schedules of all spanning trees of G if $h_T(v) = d_G(v,r)$ for all v with $d_G(v,r) \leq \lfloor (k+1)/2 \rfloor$.*

Proof. In the case that $D_0^{\lfloor (k+3)/2 \rfloor}(r)$ forms a path, no two nodes from $D_1^{k+2}(r)$ can transmit at the same time. Hence, an optimal schedule takes time at least

$$|S_{\text{OPT}}| \geq \sum_{v \in V} \pi(v) \cdot \min\left(h_T(v), k+2\right) = |S_{k\text{-}LS}| \quad . \tag{5.13}$$

In the case that $h_T(v) = d_G(v,r)$ for all v with $d_G(v,r) \leq \lfloor (k+1)/2 \rfloor$, these nodes are gathered in optimal time by *k-LS*. All other nodes account for at most $k+2$ slots in *k-LS*, but have distance at least $\lfloor (k+1)/2 \rfloor$ to r in any spanning tree, which directly proves the approximation. □

5.5.2 k_o/k_i-hop Interference

k-hop interference can be considered an oversimplification of interference even in dense networks, and we relied on the assumption that concurrent senders within a receiver's k-hop neighborhood jams reception when proving the time approximation. However, the results can very naturally be generalized to a more realistic model quantifying (combinatorial) locality bounds for interference, k_o and k_i, similarly as QUDG generalize UDG. Here, k_o denotes a hop-distance around any receiver, outside of which it is always harmless to transmit concurrently and k_i denotes a hop-distance containing only nodes whose concurrent transmission positively will make reception impossible. Nodes at a hop-distance between k_i and k_o may or may not interfere with reception.

Definition 5.20 (k_o/k_i-hop interference) *In the k_o/k_i-hop interference model, a transmission between a pair of nodes (u,v) is successful if there is no concurrent transmission by any node within v's k_o-hop neighborhood $N_G^{k_o}(v)$. It is not successful if there is a concurrent transmission by any node within v's k_i-hop neighborhood $N_G^{k_i}(v)$.*

In a straightforward manner, the above results can be generalized to this model, e. g., proving a time approximation of $k_o/\lfloor (k_i+1)/2 \rfloor$.

5.6 Making Schedules Robust

An extremely important issue especially for slotted communication is the robustness to transmission failures. While more or less spontaneous contention-based medium access allows for retransmissions, schedule-based protocols need to take transmission failures into account during set-up. Additionally, in order to set up transmission schedules that eventually deliver all data packets to the sink, retransmissions cannot be scheduled on a "per transmission"-base. To ensure that with high probability every single transmission is successfull, nodes would have to budget for at least a number of retransmissions that is logarithmic in the number of packets in the network. As we will see, there is a tradeoff between the number of transmission attempts allowed per transmission and the buffer size of nodes:

We will thus focus on slightly less restricted nodes that are able to buffer a number of data packets that can be logarithmic in the number N of packets in the whole network. Under this slightly relaxed node model, we will analyze a generic approach that basically allows us to make all of the protocols proposed in Sections 5.3, 5.4 and 5.5 robust to transmission failures. This analysis aims at a delivery of all packets with high probability, i. e., with probability higher than $1 - 1/N$.

5.6.1 Preliminaries

We make the following assumptions, which are obviously necessary for the setup of a protocol with guaranteed delivery.

1. Successful transmissions are acknowledged using some kind of `ACK` packet.
2. For every link $(u, p_T(u)) \in T$, a lower bound $\alpha(u)$ on the reception probability (including the feedback) is known.
3. Reception probabilities are independent for every transmission.

For the generic approach, we assume that we are able to agree on a non robust schedule S with the following invariant:

Invariant 5.21 *At any time, the number of packets received by any node must not exceed the number of packets sent by more than one. The number of packets sent must not exceed the number of packets received by more than one unless there are no more packets to receive.*

This invariant basically states that from a node's point of view (ignoring idle slots), the schedule starts with either the transmission or the reception of a packet, followed by an alternating sequence of transmissions and receptions, and ends with $\pi(u)$ transmissions. This is the case in all of the protocols defined above. With respect to a protocol $\mathcal{P}$, we denote by $\delta_{\mathcal{P}}(u,t)$ the *tangled reception degree* of a node v at time t, i. e., the number of nodes that u receives data packets before *and* after t. We then denote by $\delta_{\mathcal{P}}(u)$ the maximum value of $\delta_{\mathcal{P}}(u,t)$ for any t and by $\Delta_{\mathcal{P}}$ the maximum value of $\delta_{\mathcal{P}}(u)$ for any u. As an example, the tangled reception degree of *d-LevelOrderCollect* for $d = 1$ is $\Delta_{\mathcal{P}} = 1$. Here, reception of packets from different children does not interleave. For the other protocols, all childs can repeatedly become active in their natural order. In this case, we have the maximum tangled reception degree of $\Delta_{\mathcal{P}} = \Delta_T - 1$.

5.6.2 Generic description

Roughly spoken, robust gathering can be realized using a combination of scheduled retransmissions that *on average* are sufficient not to fall far behind schedule, sufficient buffer sizes, and some reserved extra time to deplete buffers in the end. More precisely, the protocol we propose, *RobustGathering* is defined by two parameters, the aforementioned γ and a buffer size parameter β, such that each node u is able to buffer $\beta \cdot (\delta_{\mathcal{P}}(u) + 1)$ packets. Then, the schedule is set up as before with two slight differences: First, the schedule budgets slightly more than $\pi(u)$ packets per node u, namely $\pi(u) + \beta \cdot (\delta_{\mathcal{P}}(u) + 2)$. Second, the schedule allocates some $t(u) \geq \lceil \gamma / \alpha(u) \rceil$ successive time slots for every scheduled transmission for some $\gamma \geq 2$. If, for example, reception probability is higher than $1/2$ for every link, i. e., $\alpha(u) \geq 1/2$ for all u, one very simple way to realize this would be to reserve $\lceil 2\gamma \rceil$ time slots per transmission. Providing solutions that take individual reception probabilities into account, however, depends on the interference and the respective protocol. We will discuss this issue after a description of the generic protocol. In the following, we will call the $t(u)$ time slots reserved per transmission as a *transmission round*.

With the schedule set up, the buffers and multiple time slots per transmission are used as follows:

- Buffers are initially empty. When a packet is added to a buffer, it is unmarked or "fresh". They will later be marked as "pending".
- At the begin of each transmission round, the sending node marks one of the unmarked packets in its buffer as pending. If there is no unmarked packet, it first adds one of its own packets to the buffer.
- During a transmission round, the sending node can make up to $t(u)$ transmission attempts, but may only transmit packets marked as pending. It removes successfully transmitted packets from its buffer. The listening node adds received packets to its buffer. If a packet is received for a second time (which can happen due to lost `ACK`s), the second reception is ignored, but acknowledged.

As stated above, there is a tradeoff between the parameter β describing the buffer size and the parameter γ describing the allocation of time slots per transmission.

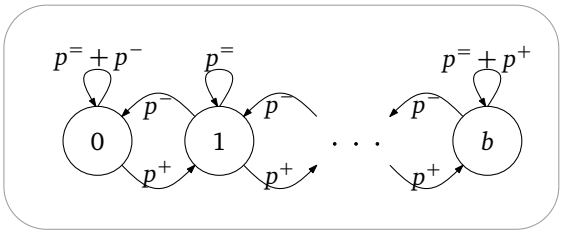


Figure 5.5: *Modeling buffer utilization as Markov process*

Theorem 5.22 *If $\beta\gamma \geq 3\mathrm{ld}N$, with high probability, RGP delivers all packets.*

Before we prove this claim, we have to prove the following lemma:

Lemma 5.23 *If $\beta\gamma \geq 3\mathrm{ld}N$, with high probability, no buffer of any node u ever contains more than β packets marked as pending and more than $\beta\delta_{\mathcal{P}}(u)$ unmarked packets.*

Proof (Lemma 5.23). First, we observe that in a transmission round, the number of nodes marked as pending at a sender u can increase by at most one. This happens if all $t(u)$ transmission attempts fail. It decreases by at least one if two or more transmission attempts are successful. The probability p^+ of the former case is

$$p^+ = (1-\alpha(u))^{t(u)} \leq (1-\alpha(u))^{\gamma/\alpha(u)} \ . \tag{5.14}$$

The probability of a decreasing number of marked nodes is

$$p^- = 1-p^+-t(u)\cdot\alpha(u)\cdot(1-\alpha(u))^{t(u)-1} \geq 1-(1-\alpha(u))^{\gamma/\alpha(u)}-\gamma\cdot(1-\alpha(u))^{\gamma/\alpha(u)-1} \ , \tag{5.15}$$

unless there were no pending packets at the begin of the transmission phase. In this case, it is zero.

Modeling the buffer utilization as a finite Markov chain as depicted in Figure 5.5, we get a probability of less than $(p^+/p^-)^b$ to be in a state where the buffer contains more than b pending packets by steady state analysis. The probability to reach such a state in any node in any transmission phase is thus less than

$$\mathbb{P}\left[\text{buffer overflow}\right] \leq 1-\left(1-\left(p^+/p^-\right)^b\right)^{N^2} \ . \tag{5.16}$$

From Equations 5.14 and 5.15, we get the following (see claim A.1 in the appendix)

$$\frac{p^+}{p^-} \leq \frac{(1-\alpha(u))^{\gamma/\alpha(u)}}{1-(1-\alpha(u))^{\gamma/\alpha(u)}-\gamma\cdot(1-\alpha(u))^{\gamma/\alpha(u)-1}} \leq e^{-\gamma/\sqrt{2}} \ , \tag{5.17}$$

and for $\beta\gamma \geq 3\mathrm{ld}N \geq 3\sqrt{2}\ln N$, we have

$$\mathbb{P}\left[\text{buffer overflow}\right] \leq 1-\left(1-e^{-b\gamma/\sqrt{2}}\right)^{N^2} \leq 1-\left(1-1/N^3\right)^{N^2} < 1/N \ . \tag{5.18}$$

Second, the number of unmarked packets in a node u's buffer can never exceed $\beta\delta_{\mathcal{P}}(u)$: Looking at a node u, we focus on the number of packets in its children's buffers marked as pending for transmission and the number of unmarked packets in u's own buffer. During a transmission round of any child c to u and the subsequent transmission phase[1] from u to $p(u)$, there is only one situation in which this sum increases, namely if all transmissions from the child to u fail *and* if u has no unmarked packet to mark as pending. But with high probability, the number of pending packets in any node's buffer stays below b. Hence, we can show that the sum (of unmarked packets in u's buffer and all pending packet in its children's buffers) never exceeds $\beta\delta_{\mathcal{P}}(t,u)$ in the following sense: If the sum increases, there are no unmarked packets in u's buffer. Additionally, we do only have to consider children that had transmission phases in the past, since only those nodes can have pending packets, *and* that will have transmissions in the future, because from u's point of view, only pending packets of those nodes can lead to buffer utilization. This exactly yields the bound of $\beta\delta_{\mathcal{P}}(t,u)$ for the sum of packets pending for transmission to u and unmarked packets in u's buffer if at time t this sum increases. Hence, a buffer size of $\beta\delta_{\mathcal{P}}(u)$ for unmarked plus β for unmarked packets is sufficient with high probability. □

We are now able to prove the main theorem:

Proof (Theorem 5.22). Before the additional $\beta\cdot(\delta_{\mathcal{P}}(u)+2)$ transmission phases start, every node u had enough transmission phases to shift all its own packets to the buffer. From Lemma 5.23 we know that at any time, the number of unmarked packets with high probability buffer is below $\beta\delta_{\mathcal{P}}(u)$. After the first $\beta\delta_{\mathcal{P}}(u)$ of the additional transmission phases, u's buffer hence contains only marked packets, and again by Lemma 5.23 at most β of them. Now, since all remaining packets are marked as pending, all of the following time slots can be used for transmission attempts. It remains to show that with high probability, each node can deplete these less than β packets in the last 2β transmission phases.

We can upper bound the probability that for a single node u, of these $2\beta\cdot\lceil\gamma/\alpha(u)\rceil$ transmission attempts less that β are successful by *Chernoff's inequality* as (with $\gamma' := \lceil\gamma/\alpha(u)\rceil\alpha(u)\geq\gamma\geq 2$)

$$\begin{aligned}\mathbb{P}\left[\text{single node failure}\right] &\leq \exp\left(-\frac{\left(2\beta\gamma'-\beta\right)^2}{4\beta\gamma'}\right)\\ &= \exp\left(-\frac{\beta\left(2\gamma'-1\right)^2}{4\gamma'}\right)\\ &\leq \exp\left(-\beta\left(\gamma'-1\right)\right)\\ &\leq \exp\left(-\beta\gamma/2\right)\leq\exp\left(-3\mathrm{ld}N/2\right)\\ &\leq \exp\left(-2\ln N\right)\leq 1/N^2\ .\end{aligned} \tag{5.19}$$

This again bounds the probability that any node has any packets left when the schedule ends by $1-\left(1-1/N^2\right)^{|V|}<1/N$. □

[1] u may be idle for some time, but the next active phase always is a transmission phase

Retransmissions of failed transmissions are inevitable. To send and relay $\pi(D(u))$ packets, a node needs in expectation $\pi(D(u))/\alpha(u)$ transmission attempts. The same holds for our protocol, but nevertheless, energy is wasted in our protocol when a node has no pending packet to send in a transmission phase. In this case, at least one (dummy) packet has to be sent in order to stop the parent from listening for the rest of the transmission phase. However, this can only happen, if there is no packet to mark as pending at the beginning of the transmission phase. Since with high probability no node has any packet left at the end of the schedule, every node u can run into this situation at most $\beta \cdot (\delta_{\mathcal{P}}(u) + 2)$ times.

Time, however, is wasted first due to the additional transmission phases and due to the length of transmission phases, that reserve by a factor γ more time slots than necessary. A quite natural choice for the parameters would thus be $\gamma = 2$ and $\beta = 2\ln N$, yielding logarithmically sized buffers and roughly double the minimum time to completion. Unfortunately, the parameter β needs to be known in advance to nodes, and in cases, in which β cannot be set to an upper bound to $2\ln N$, it might be necessary to "slow" down the schedule by adjusting the parameter γ, which the sink can set accordingly after the collection phase.

Adapting this approach to the proposed schedules can incur additional costs. A good adaption is comparably easy if there is some reasonable lower bound on the reception probability, i. e., if there is some small constant bounding $\max_{u\in V} \alpha(u)/\min_{u\in V} \alpha(u)$. Both values can easily be aggregated at the sink, which then broadcasts a $t \equiv \lceil \gamma/\min_{u\in V} \alpha(u) \rceil$ to all nodes. We consider this case to be much more relevant than arbitrarily small reception probabilities, but at least for total interference, individual transmission phase lengths can be realized treating nodes as paths of length $\lceil \gamma/\alpha \rceil$ (or $\lceil 1/\alpha \rceil$ if γ is fixed and broadcast by the sink). In this case, heights in the tree change accordingly.

For the time-optimal protocol from Section 5.4.3, robustness can incur higher costs: Since packets contain additional routing data, missing packets can sometimes only be compensated for by idle listening when waiting for the next packet.

Chapter 6

Local Link Scheduling

This chapter deals with the more general problems of scheduling links in different interference models. More precisely, we consider interference models that are provably correct with respect to the most realistic geometric model used in algorithmic research, the geometric SINR model and introduce a new classification of interference models based on geometric locality. We show that this parameter accounts for the gap between the quality of optimal schedules in correct graph-based models and those in the geometric SINR model.

6.1 Introduction and Problem Statement

Agreeing on good schedules for a given set of transmissions in wireless networks is not only a question of good and practical, i. e., local and distributed scheduling algorithms. The correctness of any scheduling algorithm's output relies on the underlying interference model. Choosing an interference model is thus crucial for any kind of scheduling protocol in sensor networks. Both, interference models and scheduling problems have been studied thoroughly in the "tradition" of sensor networks. In the simulation and design of sensor networks, complex interference models incorporating sophisticated signal fading models and antenna characteristics, developed over the years, proved helpful. In the algorithmic community, however, the need of a clear, preferably combinatorial and geometric, interference model led to a focus on graph-based interference models. These models all have in common that they are local in the sense that mutual exclusion between transmissions only "connects" transmissions that are close to each other. The simple combinatorial character of these models naturally translates scheduling problems to coloring problems in graphs. Moreover, the geometric properties of these graphs allow for tailored coloring protocols. Despite their

simplicity, the downside of these models clearly is that they could neither be proven to be correct nor good in real sensor networks. They cannot model interference from far away nodes summing up and jamming communication, nor can they model that if in reality any pair out of three transmissions can successfully be performed simultaneously, this does not necessarily mean that all three transmissions can be performed simultaneously.

Algorithmic research considering a class of models that renders signal propagation much more realistically did to the best of our knowledge not yet lead to local algorithms. Please recall that in SINR models, successful or sufficiently probable reception is assumed if at a receiver, the respective sender's signal strength outperforms the sum of all interfering signals plus the background noise by a hardware dependent constant. The geometric SINR models closely cover the main features of sophisticated fading models such as the two-ray-ground model without losing too much of the simplicity needed for algorithmic results.

In the following, we introduce the concept of locality and correctness of interference models. We prove fundamental limitations of all models that are local in a very straightforward sense and correct with respect to the geometric SINR model. We show under which conditions well known concepts such as graph coloring *can* be used to approximate scheduling problems and a generalization that improves the quality of easy-to-implement scheduling algorithms. We believe that the introduced models open a door to more realistic, yet viable solutions not only for scheduling, but for many protocols that rely on local, dependable communication.

A *scheduling problem* in a wireless network is a set $\mathcal{Q}$ of communication requests, each request (s,r) consisting of a sender s and a receiver r, both from some set $\mathcal{V}$ of nodes. A schedule then is a sequence $T_1, T_2, \ldots, T_k$ of sets of transmissions of the form (s,r,p) for some $(s,r) \in \mathcal{Q}$ and some power assignment $p \in \mathbb{R}_+$, such that for every $(s,r) \in \mathcal{Q}$, there is a transmission (s,r,p) in one of the T_i, and every T_i is valid with respect to an interference model. We refer to the problem of finding a schedule of minimum length as Schedule, and to the problem of finding a maximum number of transmissions that can be scheduled to a single slot as OneShotSchedule as in [GOW07]. We will also denote the maximum link length occurring in a schedule request $\mathcal{Q}$ by $\ell(\mathcal{Q}) := \max_{(s,r)\in\mathcal{Q}} d_{sr}$. If the scheduling problem is combined with the problem of assigning transmission powers, usually powers must be chosen from some power range $p = [p_{\min}, p_{\max}]$. In the following, we will focus on the problem of finding schedules for a fixed power p and thus also write (s,r) to denote a transmission (s,r,p).

6.1.1 Related Work

Interference of concurrent communication has been subject of countless publications. We have seen a short introduction to this field in Section 2.3: Most of the algorithmic models model interference as a binary relation on transmissions, among them the unit disk graph (UDG) with distance or hop interference or the protocol model. In SINR models, successful reception depends on the ratio between the received signal strength on the one side and the interference from concurrent transmissions plus the background noise on the other side [Rap96, GK00]. They differ in whether they assume signal strength decay to be a function of the distance (geometric SINR) or allow an arbitrary gain matrix. In the geometric

SINR model, Gupta and Kumar analyzed the capacity of ad-hoc networks and proved an upper bound on the throughput of $\Theta(1/\sqrt{n})$ for networks of n nodes [GK00]. Until now, the effects of the SINR models to algorithm design raise interesting questions [MWW06].

Scheduling of link transmissions has been addressed in many interference models and, in most cases, proven to be NP-hard. Among others are proofs for scheduling in graph-based models [HMR+98, KMR00], in the abstract SINR model in [BVY04], and, recently, in the geometric SINR model for fixed power assignment by Goussevskaia et al. [GOW07]. The joint problem of scheduling and power assignment is still open in the geometric SINR model [LvRW08]. A variety of graph-based scheduling algorithms has been proposed and analyzed [HS88, KMPS04, MW05]. It is however argued in various works that graph-based scheduling is inferior to scheduling designed for the SINR model [BR03, GH01]. Among the early publications addressing scheduling in geometric SINR models, Moscibroda and Wattenhofer show that uniform or linear power assignments in worst-case scenarios need exponentially longer schedules for a strongly connected set of links [MW06] than more sophisticated assignments. Moscibroda et al. also propose a scheduling algorithm for arbitrary power assignment in [MOW07] that outperforms previous heuristics by an exponential factor. In [GOW07], Goussevskaia et al. propose an approximation algorithm for link scheduling and the problem of finding a maximum number of links that can transmit concurrently in the geometric SINR model under the fixed power assumption. The latter three works introduced many of the techniques applied in the following under the practically more relevant assumptions that nodes do not feature arbitrarily high transmission powers and cannot rely on a global instance to compute a schedule, but are restricted to a local view. Locality has, to our knowledge, only been looked at in a combinatorial sense [Lin92, NS95, KMW06].

6.2 A Taxonomy of Interference Models

A *deterministic* interference model $\mathcal{M}$ can be interpreted as a property telling for a fixed set of nodes $\mathcal{V}$ whether a set of transmissions T between nodes in $\mathcal{V}$ can be carried out simultaneously for given transmission powers. More formally, let $\mathcal{T} := \left\{(u, v, p) \in \mathcal{V}^2 \times \mathbb{R}_+ \mid u \neq v\right\}$ be the set of all possible transmissions and transmission powers. Then, a model $\mathcal{M} \subseteq \mathcal{P}(\mathcal{T})$ contains all sets of transmissions which are valid. We further assume that less concurrent transmissions cannot cause a transmission to fail, i. e., that for all $T' \subseteq T \subseteq \mathcal{T}$,

$$T \in \mathcal{M} \Rightarrow T' \in \mathcal{M} \ , \tag{6.1}$$

which holds for all models which are currently used and most likely for all models which are meaningful. One should note that the restriction to deterministic models alone already is a giant step away from reality and the probabilistic models typically employed by communication theorists. But still, even deterministic models are not understood well. Such models can rely on various kinds of additional input and assumptions of radio propagation, antenna characteristics and so on. In higher layer protocol design, however, there is a need

to "model away" the complexity of most of these unrulable phenomena. An aspect that has not received much attention yet is how different approaches to model interference relate to each other, or, in other words: If I choose a simpler model, are my algorithms or schedules still correct with respect to a more realistic one or are they just suboptimal? Do optimal solutions in a simple model approximate optimal solutions in a more complex model? In the following, we will call an interference model $\mathcal{M}$ *conservative* with respect to another model $\mathcal{M}'$ if $\mathcal{M} \subset \mathcal{M}'$.

6.2.1 Graph-based Interference Models / SINR

Most analytical research on scheduling problems has been done in some kind of *graph-based* interference model according to the following definition from [MWW06].

Definition 6.1 (Graph-based model) *A* graph-based model $\mathcal{M}$ *can be defined by two directed graphs, one connectivity graph* $D_C = (\mathcal{V}, A_C)$ *restricting possible transmissions and one interference graph* $D_I = (A_C, A_I)$ *connecting conflicting transmissions, such that* $T \in \mathcal{M}$ *if and only if* $T \subset A_C$ *and* $T^2 \cap A_I = \emptyset$.

Usually, a simpler model consisting of two graphs $G_C = (\mathcal{V}, E_C)$ and $G_I = (\mathcal{V}, E_I)$ is used, in which a set of transmissions is valid, if for every sender, the intended receiver is a neighbor in G_C and no receiver of a distinct transmission is connected in G_I. Sometimes, the connectivity graph and interference graph are defined implicitly, i. e., as the result of a geometric setting. Graph-based models all have in common that they claim that a set of transmissions whose transmissions can *pairwise* be carried out at the same time, collectively may be scheduled into one single time slot. This is unrealistic in general, and the models fail to formulate the assumptions under which they guarantee not to produce schedules that do not comply with more realistic models. On the other hand, in the single-power case, graph-based models reduce scheduling problems to well-known coloring problems.

As opposed to the oversimplification of graph-based interference models, the models capturing the findings of signal propagation and reception best are the *signal-to-noise-plus-interference* (SINR) models. Their main paradigm is that a transmission is (almost) always successful, if the sender's signal strength *at the receiver* is significantly stronger than the sum of all interfering signals, including other sender's signals and (individual) background noise. Thus, in its most general form, an SINR model is defined by a *gain* matrix (g_{uv}) denoting the signal fading between nodes u and v, on the background noise η_v at each of the nodes and the (individual) ratio β_v a node v needs for proper reception. Here, a set of transmissions is valid, i. e. $T \subset \mathcal{M}$, if and only if for all $t = (s, r, p_s) \in T$

$$\frac{p_s g_{sr}}{\eta_r + \sum_{(u,v,p_u) \in T \setminus \{t\}} p_u g_{ur}} \geq \beta_r \quad . \tag{6.2}$$

6.2.2 Geometric Interference Models

The first property of interference that has never been described explicitly in the literature is that of being geometric in the following sense:

Definition 6.2 (Geometric model) *In a* geometric model*, $\mathcal{M}$ is defined for $\mathcal{V} = \mathbb{R}^2$ such that $\mathcal{M}$ is invariant under all isometries.*

Generally speaking, geometric interference models are incapable of modeling individual characteristics of nodes, but are restricted to those of geometric settings. This does not mean that a geometric model has to be parameter-free, but for geometric SINR models, this definition implies a much simpler structure:

Theorem 6.3 *Every geometric SINR model can also be defined equivalently such that all η_v and all β_v are independent of the respective position v and all g_{uv} can be expressed as $g_{uv} := f(d(u,v))$ for a $f : \mathbb{R} \to \mathbb{R}$.*

Proof. Let $\mathcal{M}$ be a geometric SINR model defined by (g_{ji}), (η_i) and (β_i). We get an equivalent model for (g'_{ji}), (η'_i), and (β_i), in which all $\eta'_i = \eta'$ are equivalent by setting $g'_{ji} = g_{ji}\eta'/\eta_i$. Now, for any d, take two pairs u_1, v_1 and u_2, v_2. We know that $\{(u_i, v_i, p)\} \in \mathcal{M}$ if and only if $p \geq \eta'\beta'_{v_i}/g_{u_i v_i}$. Since there is an isometry mapping u_1 to u_2 and v_1 to v_2, and since $\mathcal{M}$ is geometric, transmissions (u_1, v_1, p) are valid, for exactly the same values of p as (u_2, v_2, p). Thus, $p \geq \eta'\beta'_{v_1}/g'_{u_1 v_1}$ if and only if $p \geq \eta'\beta'_{v_2}/g'_{u_2 v_2}$ and thus $\beta'_{v_1}/g'_{u_1 v_1} = beta'_{v_2}/g'_{u_2 v_2}$. I. e., all pairs of nodes with distance d have the same ratio of g'_{ji} and β_i and by fixing some β' and setting $g''_{ji} = g'_{ji}\beta'/\beta_i$, we get a representation of the claimed form. □

The class of *geometric SINR models (SINR$_G$)* is a quite straightforward application of the above definition. Individual characteristics such as the background noise and the necessary SINR ratio are replaced by common constants η and β and the gain g_{uv} is replaced by a function of the distance, usually $K d_{uv}^{-\alpha}$ for a so-called *path-loss exponent* α and some constant K. Currently, the SINR$_G$ models widely agreed are the best models to reason about in the algorithmics of sensor networks. Thus, we will focus on local models that are conservative with respect to this class of models.

6.2.3 Local Interference Models

We introduced the concept of local algorithms in Section 2.3: A distributed algorithm is said to be k-local, if the outcome for every node only depends on nodes which are inside a k-hop-neighborhood. Unfortunately, this concept is too restrictive to allow for any local scheduling algorithms in a geometric SINR model with nodes that do not feature arbitrarily high transmission powers, but are limited to some maximum power. Even if we define a node's neighborhood as the set of nodes the node can communicate with when no other communication takes place at the same time, in an SINR$_G$ model $\mathcal{M}_G$, it might be impossible to arrange a schedule at all. If we denote the maximum possible link length of an interference model $\mathcal{M}$ by $\ell(\mathcal{M}) := \limsup_{\{(u,v,p)\} \in \mathcal{M}} d_{uv}$, we get $\ell(\mathcal{M}_G) = \sqrt[\alpha]{Kp/(\beta\eta)}$, since for nodes with higher distance even in the absence of concurrent transmissions sending at maximum power does not result in a received signal strength of $\beta\eta$, which is necessary due to the background noise alone. In Figure 6.1, such a situation is depicted: Out of the two sender/receiver pairs in the transmission request, only the pairs themselves have a distance less than $\ell(\mathcal{M}_G)$, and thus, there is no communication possible between the different pairs,

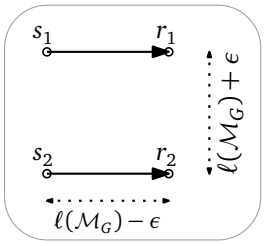


Figure 6.1: *Links with distance $\ell(\mathcal{M}_G)+\epsilon$ may need communication*

which, however, have to agree not to transmit at the same time, since both of them cannot compensate for the interference caused by the other. We will thus in the next section look at a weaker, geometric definition of locality and its consequences for scheduling problems.

Definition 6.4 (Local model) *A* ρ-local model *is a geometric model $\mathcal{M}$ with the additional constraint that $T \in \mathcal{M}$ if for every $t = (s, r, p) \in T$*

$$T(s,\rho) := \{(s', r', p') \in T \mid d(s, s') \leq \rho\} \in \mathcal{M} \ . \tag{6.3}$$

In other words, an interference model is *local*, if for a set of transmissions T, it is sufficient that for every sender in T the transmissions in its ρ-neighborhood comply with the model to make T valid. Models of this kind not only allow to tell that a set of transmissions will be successful by only locally looking at the transmissions, but they are also essential for the design of local algorithms. They can be seen as a rule for every node that can only observe nearby nodes, either during a setup phase or, more importantly maintaining a dynamic link transmission schedule. The geometric graph-based models mentioned above quite naturally have this property, but SINR_G models do not, which proved to be one of the main obstacles when tackling scheduling problems in these models. This holds for existing centralized approximation algorithms which try to break the interwoven dependencies into independent subproblems as in [GOW07], and it inherently does so in distributed settings – how could nodes come up with a provably valid schedule with local communication, when the validity of a schedule cannot be judged locally?

6.3 Bounds of Local Interference Models

Local interference models on the other hand seem to be incorrect by design: They are blind for interference that arises from nodes that are far away, and thus cannot factor what these nodes are doing. From this time on, let $\mathcal{M}_G = (K, \eta, \beta, \alpha)$ be a standard SINR_G model. We start with an observation which illustrates the first limitations of local reasoning about interference implicates. It is a generalization of the considerations above.

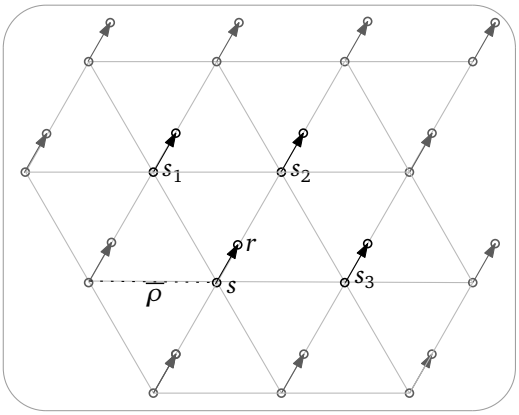


Figure 6.2: *Lower bounding interference in ρ-local models*

Observation 6.5 *Let $\mathcal{M}_L$ be a $\mathcal{M}_G$-conservative, ρ-local interference model. The following two inequalities hold:*

$$\ell(\mathcal{M}_L) < \frac{\rho}{1+\sqrt[\alpha]{\beta}} \quad \textit{and} \quad \ell(\mathcal{M}_L) < \ell(\mathcal{M}_G) \cdot \left(1+\frac{3Kp}{\eta\rho^\alpha}\right)^{-1/\alpha} \tag{6.4}$$

Proof. Let $T \in \mathcal{M}_L$ be any set of transmissions that is accepted by the local model and $t = (s,r)$ any transmission in T. Let $\ell_t := d_{sr}$. By the subset acceptance property (6.1) and the fact that $\mathcal{M}_L$ is geometric (and thus invariant under isometries of the plane), $\mathcal{M}_L$ would accept any set of transmissions T in which all senders have pairwise distances of more than ρ and all transmissions from T have length ℓ_t.

Now, consider a set of transmissions T, as depicted in Figure 6.2, where senders are placed on a triangular grid with edge length $\overline{\rho} := \rho + \epsilon$, i. e., which complies with the considerations above. First, if we assume that $\ell_t \geq \rho/(1+\sqrt[\alpha]{\beta})$, the interference of the sender s_2 alone would interfere with the reception of the transmission t to the limit,

$$\lim_{\epsilon\to 0} \frac{Kp\overline{\rho}^{-\alpha}(1+\sqrt[\alpha]{\beta})^\alpha}{Kp\overline{\rho}^{-\alpha}(1-1/(1+\sqrt[\alpha]{\beta}))^{-\alpha}} \leq \lim_{\epsilon\to 0} \frac{(1+\sqrt[\alpha]{\beta})^\alpha}{(\sqrt[\alpha]{\beta}/(1+\sqrt[\alpha]{\beta}))^{-\alpha}} = \beta \ , \tag{6.5}$$

and together with the additional interference caused by other senders, reception would become impossible, contradicting with the choice of t.

Second, as we now know that $\ell_t < \rho/2$, we get that the interference of the senders s_1, s_2 and s_3 at the receiver is at least $3Kp\rho^{-\alpha}$. Thus, since $T \in \mathcal{M}_G$,

$$\frac{Kp\ell_t^{-\alpha}}{3Kp\rho^{-\alpha}+\eta} \geq \beta \Leftrightarrow \ell_t \leq \left(\frac{\beta\eta}{Kp}+\frac{3\beta}{\rho^\alpha}\right)^{-1/\alpha} = \ell(\mathcal{M}_G)\cdot\left(1+\frac{3Kp}{\eta\rho^\alpha}\right)^{-1/\alpha} \ ,$$

which concludes the proof. □

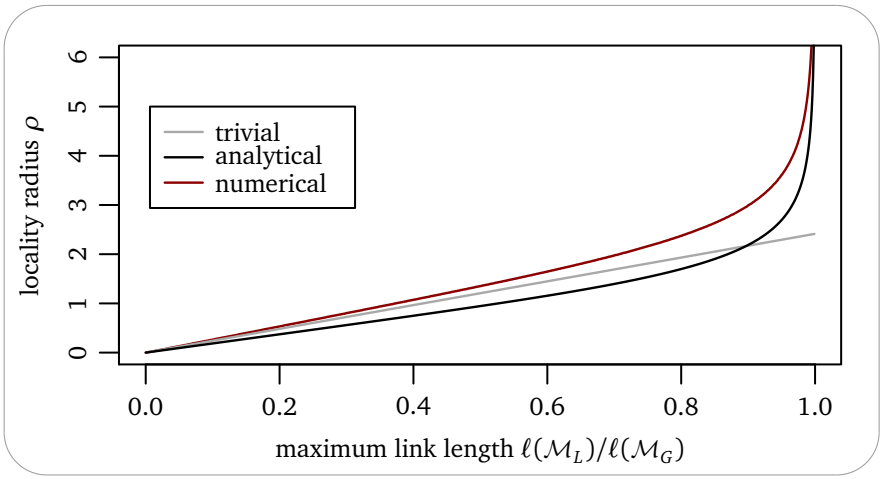


Figure 6.3: *Lower bounds on ρ ($\alpha = \beta = 4.0$)*

Note that this bound is by no means tight, but it shows how severe the restrictions are that one can only overcome by globally solving schedule problems: To allow for longer links, especially of lengths close to $\ell(\mathcal{M}_G)$, the radius ρ has to be chosen accordingly. We can derive better bounds by calculating interferences more accurately than above by summing up interference for more senders on the same triangular grid. Figure 6.3 shows an exemplary tradeoff between the maximum link length needed, $\ell(\mathcal{M}_L)$ and the resulting analytical and numerical lower bounds on the radius ρ for $\alpha = 4$, $\beta = 4$ ($\approx$ 6dB) and η, p, K normalized to $\ell(\mathcal{M}_G) = 1$. It shows that in the case that we do not assume that nodes can communicate with nodes outside their transmission radius, e. g., by the assumption that the node density is sufficiently high, no link length longer than 40% of the maximum link length can safely be scheduled in realistic scenarios.

The second observation we can make about local interference models regards the case that nodes cannot send with arbitrarily low power:

Observation 6.6 *Let $\mathcal{M}_L$ be a $\mathcal{M}_G$-conservative ρ-local interference model for a $\rho < \infty$. Even for requests with $\ell(\mathcal{Q}) \leq \ell(\mathcal{M}_L)$), optimal solutions to* Schedule *and* OneShotSchedule *in $\mathcal{M}_L$ can be arbitrarily worse than in $\mathcal{M}_G$.*

For the sake of brevity, we will only give a sketch of the proof here. We look at a request of a ring of n transmissions as depicted in Figure 6.4 with sufficiently small transmission lengths $\ell(g)$ plus one transmission $t^\star$ of length $\ell(n)$ in the middle. It is easy to see that in $\mathcal{M}_G$, it is admissible to schedule all transmissions but $t^\star$ to the same slot (and $t^\star$ to a second). In $\mathcal{M}_G$, assigning a slot to $t^\star$ and to the rest of transmissions is independent. Thus, at no time more than a constant number of the n outer transmissions can be carried out, allowing for concurrent transmission of $t^\star$.

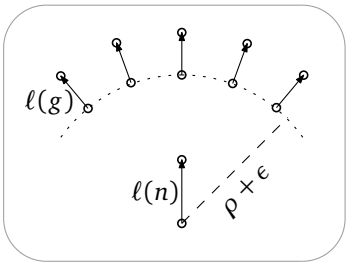


Figure 6.4: *Ring of transmissions*

6.4 Ω(1)-Sender Models

In every meaningful local model, the acceptance of a (local) set of transmissions must follow this consideration: Given the rules for local acceptance of a set of transmissions – is it guaranteed that if all nodes obey these local rules, no node possibly has to accept more interference from outside the ρ-neighborhood than allowed, given the amount of interference arising from local transmissions? A quite straightforward implementation of this concept is the following: For some function $\mu : \mathbb{R}_+ \to \mathbb{R}_+$, which serves as an upper bound for interference from far away nodes, a set of transmissions T is licit if for every transmission (s, r) a *local signal-to-noise-plus-interference* condition holds:

$$\frac{Kpd_{sr}^{-\alpha}}{\sum_{(\hat{s},\hat{r})\in T(s,\rho)} Kpd_{\hat{s},r}^{-\alpha} + \eta + \mu(d_{sr})} \geq \beta \ , \tag{6.6}$$

and if it is guaranteed that a transmission (s, r) cannot receive more interference than $\mu(d_{sr})$ from senders further away than ρ from s. One way to guarantee the latter is to prohibit that close senders are transmitting concurrently and thus, to limit the density of active senders:

Definition 6.7 (Ω(1)-sender model) *In the* Ω(1)-sender model $\mathcal{M} = (\rho, c, \mu)$*, a set of transmissions T is valid if and only if for every $(s, r) \in T$ equation (6.6) holds, and any two senders in T have distance at least c.*

Clearly, such a model is ρ-local if $c \leq \rho$, but quite obviously not $\mathcal{M}_G$-conservative for an arbitrary μ. However, for certain values of ρ, c, and μ, the resulting model (ρ, c, μ) is $\mathcal{M}_G$-conservative and local:

Lemma 6.8 *Let $\mathcal{M}_L = (\rho, c, \mu)$ be an Ω(1)-sender model. $\mathcal{M}_L$ is conservative with respect to $\mathcal{M}_G$ if* [1]

$$\mu(\ell) \geq \frac{\sqrt{12}Kp\pi\zeta(\rho^2/c^2 + 2\rho/c)}{(\rho - \ell)^\alpha} =: \mu_1(\ell) \tag{6.7}$$

Proof. Let (s, r) be some sender/receiver pair with $d_{sr} = \ell$. We divide the plane into annuli A_k with center s and radii $k\rho$ and $(k + 1)\rho$ for $k \in \mathbb{N}$. The maximum number of senders

[1] for the Riemannian ζ-function and $\zeta := \zeta(\alpha - 1)$, a constant $1 < \zeta < 2$ for $\alpha \geq 3$

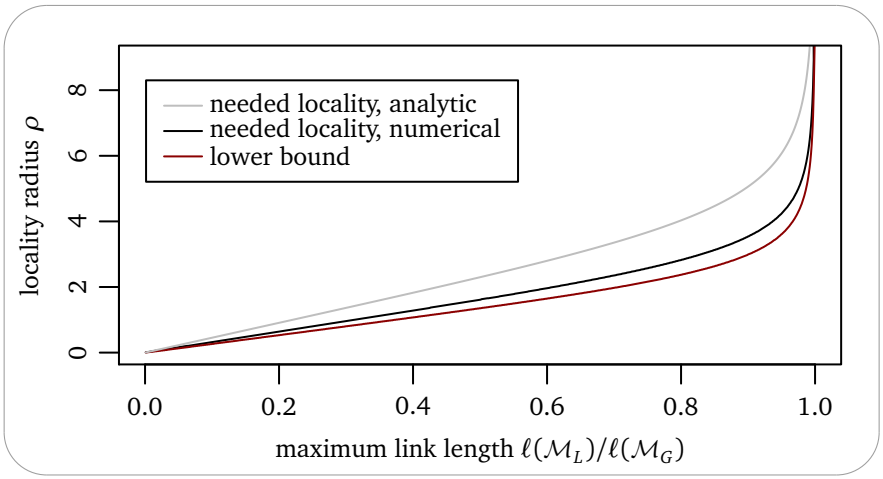


Figure 6.5: *Minimum locality as function of maximum link length, analytical and numerical bounds compared to lower bound*

lying within the kth annulus is the maximum number of disks of radius c within an annulus with radii $k\rho - c$ and $(k+1)\rho + c$. Since senders in $\Omega(1)$-sender models form a Minkowski arrangement, which cannot exceed a density of $\frac{2\pi}{\sqrt{3}} \approx 3.638$ [Tot65], we get that the number of senders in A_k is at most

$$N_k := \left\lfloor \frac{2\pi}{\sqrt{3}} \cdot \frac{\pi\left(((k+1)\rho + c)^2 - (k\rho - c)^2\right)}{\pi c^2} \right\rfloor \leq \underbrace{k\sqrt{12}\pi\left(\rho^2/c^2 + 2\rho/c\right)}_{:=N^*},$$

The interference received from any of the senders in A_k can be bounded by

$$I_k \quad := \quad Kp(k\rho - \ell)^{-\alpha} \leq k^{-\alpha} \underbrace{Kp\left(\rho - \ell\right)^{-\alpha}}_{=:I^*},$$

and the total interference received from any sender can then be bounded by $\sum_{k=1}^{\infty} N_k I_k \leq N^* I^* \sum_{k=1}^{\infty} k^{-\alpha+1} = N^* I^* \zeta$. □

Let (ρ, c) denote a shortcut for the $\mathcal{M}_G$-conservative model (ρ, c, μ_1). Obviously, the bound μ_1 can very straightforward be replaced by a better numerical bound. Figure 6.5 shows how these bounds compare to each other and to the lower bound from the last section. Note that all bounds are correct, and, given the SINR parameters, easy to calculate. We will use the closed-form result for further analysis and the improved bounds for simulation.

With the approximation above, we still have the choice of first the maximum possible link length and second, the balance of the locality factor ρ and the exclusion radius c.

Corollary 6.9 *Let $\ell = u \cdot \ell(\mathcal{M}_G)$ for some $0 < u < 1$ be a link length and $a \geq 1$, the $\mathcal{M}_G$-*

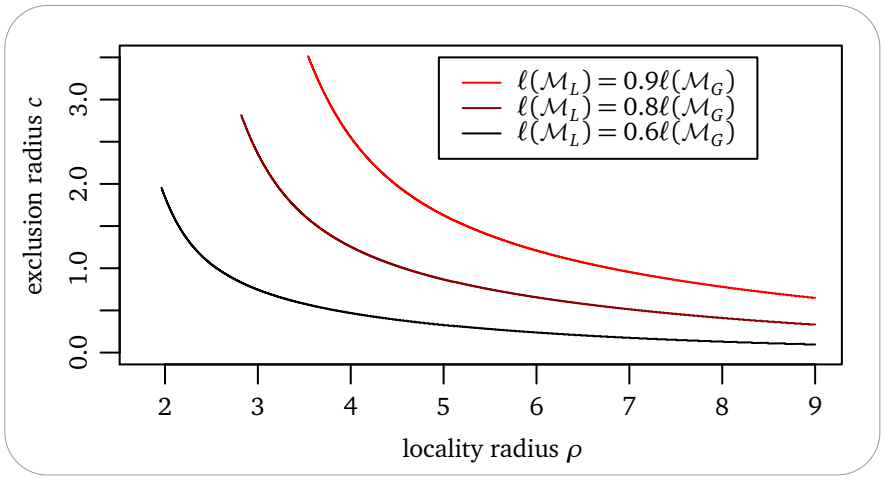


Figure 6.6: *Tradeoff between the minimum pairwise sender distance and the locality radius for fixed* $\ell(\mathcal{M}_L)/\ell(\mathcal{M}_G)$

conservative $\Omega(1)$*-sender model* $\mathcal{M}_L = (\rho, \rho/a)$ *with*

$$\rho = \left(1 + \sqrt[\alpha]{\frac{\sqrt{12}(a^2+2a)\pi\beta\zeta}{1-u^\alpha}}\right) \cdot \ell \tag{6.8}$$

has $\ell(\mathcal{M}_L) = \ell$*. It is graph-based for* $a = 1$*, yielding* (ρ, ρ)*.*

Proof. According to Lemma 6.8, $\mathcal{M}_L$ is $\mathcal{M}_G$-conservative. It hence remains to show that $\ell(\mathcal{M}_L) = \ell$. First, by $\ell = u\sqrt[\alpha]{Kp/\beta\eta}$, we observe that

$$\rho - \ell = \sqrt[\alpha]{\frac{\sqrt{12}(a^2+2a)\pi\beta\zeta}{1-u^\alpha}} \cdot \ell = \sqrt[\alpha]{\frac{\sqrt{12}(a^2+2a)\pi Kp\zeta}{\eta(u^{-\alpha}-1)}} \tag{6.9}$$

and therefore by the definition of u

$$\ell(\mathcal{M}_L) = \sqrt[\alpha]{\frac{Kp}{\beta\eta + \sqrt{12}(a^2+2a)\pi Kp\beta\zeta\,(\rho-\ell)^{-\alpha}}} = \sqrt[\alpha]{\frac{Kp}{\beta\eta u^{-\alpha}}} = \ell \tag{6.10}$$

Obviously, it is graph-based for $a = 1$. □

This corollary in a way justifies the work that has been done on scheduling problems in graph-based models as it provides a very simple graph-based model that is provably correct with respect to the geometric SINR models (and, at the same time shows the price for reducing an SINR_G model to a graph-based model). Figure 6.6 shows this tradeoff between the maximum schedulable link length and the respective ρ for different ratios. As argued in Section 6.2.3, no local model can allow for "good" solutions in the single-power setting

in the sense that the scheduling problems can be approximated within a constant factor in any such model if links can be arbitrarily short. We will show that $\mathcal{M}_G$-conservative $\Omega(1)$-sender models (ρ,ρ) are only by constant and comparably small factors worse than any local model in two dimensions – first the locality needed to allow for a given maximum link length and second the quality of optimal solutions to the scheduling problems.

Lemma 6.10 *Let $\mathcal{M}_L = (\rho,\rho)$ be the $\mathcal{M}_G$-conservative $\Omega(1)$-sender model according to Corollary 6.9 for some ℓ. Then for any ρ'-local model $\mathcal{M}'_L$ which is $\mathcal{M}_G$-conservative,*

$$\frac{\rho}{\rho'} \leq \frac{1}{\sqrt[\alpha]{3\beta}} + 2\sqrt{3}\pi\zeta \tag{6.11}$$

Proof. From (6.4) and with $u := \ell/\ell(\mathcal{M}_G)$, we get that

$$\rho' \;=\; \sqrt[\alpha]{3\beta}\left(\frac{1}{\ell(\mathcal{M}_L)^\alpha} - \frac{1}{\ell(\mathcal{M}_G)^\alpha}\right)^{-1/\alpha} = \ell\,\sqrt[\alpha]{3\beta}\,(1-u^\alpha)^{-1/\alpha} \tag{6.12}$$

and, from Corollary 6.9 and $u < 1$,

$$\rho = \left(1 + \sqrt[\alpha]{\frac{6\sqrt{3}\pi\beta\zeta}{1-u^\alpha}}\right)\cdot\ell \;<\; (1-u^\alpha)^{-1/\alpha}\cdot\left(1+\sqrt[\alpha]{6\sqrt{3}\pi\beta\zeta}\right)\cdot\ell\;, \tag{6.13}$$

which directly implicates the claimed approximation. □

This bound only depends on α and β, and is thus constant for a fixed SINR_G model. For a given set of SINR_G parameters, this approximation ratio can be improved using the non-closed-form lower bounds for local models and upper bounds for the c-distant sender model. E. g., for the exemplary values used throughout this paper, the best bounds guarantee a ratio of less than $5/4$ for arbitrary ℓ.

Lemma 6.11 *Let $\mathcal{M}_L = (\rho,\rho)$ be the $\mathcal{M}_G$-conservative $\Omega(1)$-sender model according to Lemma 6.8 and Corollary 6.9. Let $\mathcal{Q}$ be a schedule request with $\ell(\mathcal{Q}) < \ell(\mathcal{M}_L)$. Optimal solutions to Scheduling and One-Shot-Schedule in $\mathcal{M}_L$ are only by a constant factor worse than optimal solutions in any other ρ-local $\mathcal{M}_G$-conservative model.*

Proof. First we show that any ρ-local model $\mathcal{M}'_L$ with $\ell(\mathcal{M}'_L) \geq \ell$ cannot accept any set of transmissions T such that for any transmission (s,r,p), $T(s,\rho)$ contains more than $h := 4^\alpha 6\sqrt{3}\zeta$ transmissions. To this extent, let T be a set of h transmissions such that $T(s,\rho) = T$ for some $(s,r) \in T$. We add a transmission (s',r') to T with $d_{s,s'} = 2\rho$, pointing towards s (cf Figure 6.7). Note that if T is valid in $\mathcal{M}'_L$, then $T' = T \cup \{(s',r')\}$ must be valid, too. But if all transmissions in T' are carried out simultaneously, the SINR-level at r' is below

$$\frac{Kp\ell^{-\alpha}}{hKp(2\rho)^{-\alpha}+\eta} \;=\; \frac{\ell^{-\alpha}}{h4^{-\alpha}\frac{\ell^\alpha(1-u^\alpha)}{6\sqrt{3}\pi\beta\zeta} + \frac{\eta}{Kp}} = \frac{\ell^{-\alpha}}{\frac{\ell^{-\alpha}(1-u^\alpha)}{\beta} + \frac{\eta}{Kp}} = \beta\;.$$

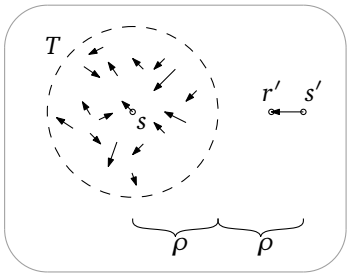


Figure 6.7: *Upper bounding number of transmissions in a ρ-radius*

Now take any schedule request $\mathcal{Q}$. Let A be the square with side $\rho/\sqrt{2}$ that contains the most senders in $\mathcal{Q}$. Let m denote this number. In every ρ-local model at most h of the senders in A can transmit concurrently, leading to a schedule length of at least $\lceil m/h \rceil$. In $\mathcal{M}_L$, in turn, we can construct a schedule of length $16m$ by the same construction as in [GOW07]: We now take a grid of grid-length ρ, 4-color the grid cells, and cyclically choose a color and pick an unscheduled sender from each cell with that color. In each of the cells, we have at most $4m$ transmissions, and the schedule hence at most lenght $16m$, which guarantees a $16h$-approximative schedule compared to an optimal solution in any ρ-local model. Similar arguments lead to a $16h$-approximation of One-Shot-Schedule: Take an optimal solution T in any ρ-local model and picture a 4-colored grid with grid-length ρ. Focus only on grid-squares that contain a sender of T. Each of the squares contains at most $4m$ transmissions in T. Now pick the color of the most non-empty squares and pick one transmission of each square. This set of transmissions contains at least $\lceil |T|/16m \rceil$ transmissions that can all be carried out concurrently in $\mathcal{M}_L$. We get a slightly worse approximation for greedy scheduling, where we will only look at the Scheduling problem: In the very same grid as above, if one cell does not contain an active sender in a slot, then for two possible reasons: First, because all senders have been scheduled to earlier slots, and second, because of some active sender in one of the adjacent cells. Thus, greedy scheduling uses at most $36m$ slots, which is $36h$-approximative. □

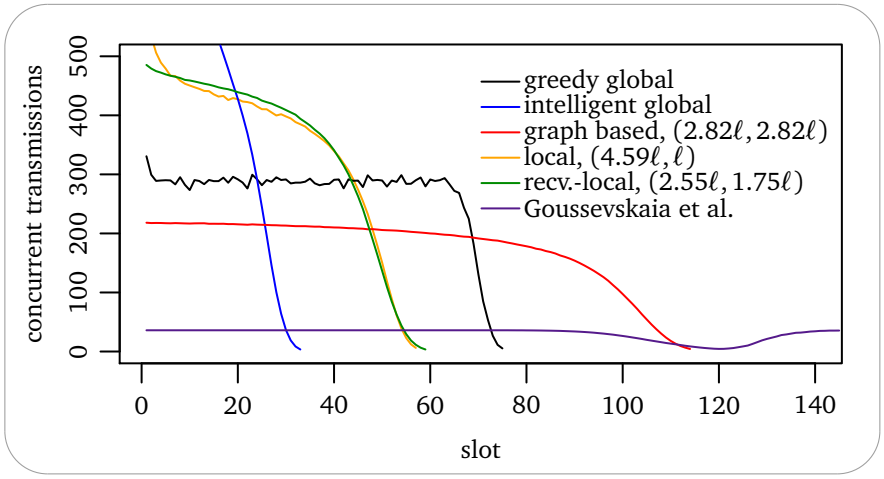
6.4.1 Application

We implemented a very basic scheduling algorithm for Ω(1)-sender models which greedily assigns slots to senders in a random order: Each sender is assigned to the first allowed slot according to the respective model. This is not only a very simple centralized approach, but also a reasonable distributed scheduling algorithm. Given that the node density is sufficient for nodes to have their ρ-neighborhood some constant number of hops away, nodes can draw random numbers and decide on their slot after all neighbors with lower numbers did so only by local communication. This approach is also suited to schedule online requests. We compare the results to three different global scheduling algorithms. First, we select nodes in random order and add them to the first slot allowed by the plain SINR_G model, i. e., schedule greedily and global. Second, we compare to the algorithm given from Goussevskaia et al. [GOW07]. Please note that this algorithm is not designed to produce good

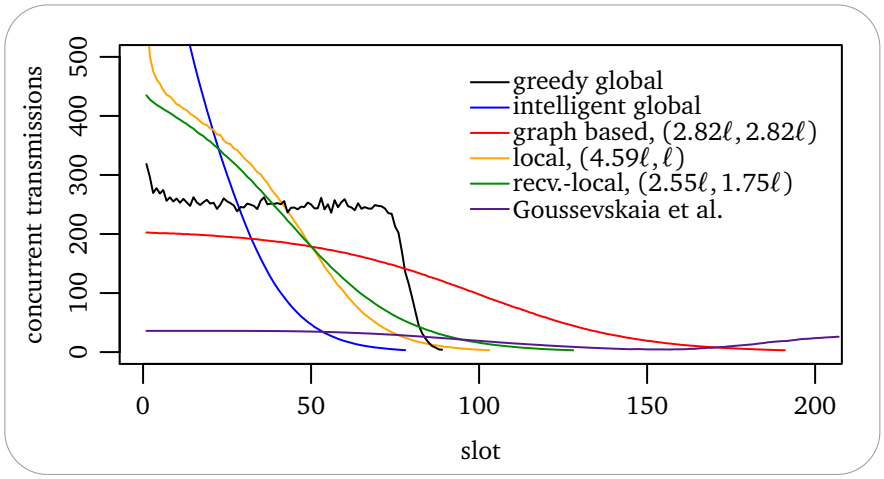
Table 6.1: *Comparison of schedules in different interference models for 20000 random links*

Algorithm	locality $[\ell(\mathcal{M}_G)]$	excl. rad. $[\ell(\mathcal{M}_G)]$	random links length	max. util.	avg. util.
Greedy scheduling in $\mathcal{M}_G$	∞	0	71.91	482.45	278.43
Intelligent scheduling in $\mathcal{M}_G$	∞	0	34.15	2726.00	586.11
Goussevskaia et al. [GOW07]	∞	0	605.03	206.01	33.06
Greedy scheduling in ...					
... (ρ,ρ), min. ρ (graph-based)	2.82	2.82	117.38	224.31	170.45
... $(\rho,\ell(\mathcal{M}_G))$, min. ρ	4.59	1.00	58.63	626.59	341.31
... $(\rho,\ell(\mathcal{M}_G))$, incr. ρ	6.00	1.00	47.89	629.41	417.86
... recv.-local (ρ,r), min. ρ	2.55	1.75	60.88	487.68	328.79
... recv.-local $(\rho,\ell(\mathcal{M}_G))$, min. ρ	3.82	1.00	54.84	675.81	364.91
... recv.-local $(\rho,\ell(\mathcal{M}_G))$, incr. ρ	6.00	1.00	46.48	646.45	430.50

schedules, but only as a proof of approximability. It is thus not surprising that it returns comparably poor results. Third, since solving the Schedule problem optimally is hard, and solving the corresponding mixed-integer linear problem only works for a very small number of transmissions, we compare to a heuristic, which produced near-optimal results for small instances of random transmission requests. We fill the slots one after another, at any time adding the transmission which causes the least drop of the minimum signal-to-noise-plus-interference ratio for all transmissions earlier added to that slot. We ran all of the above algorithms on schedule requests with at most 80% of the maximum link length in the SINR_G model. Instances were random sets of 20000 transmissions and random unit disk graphs with 5000 nodes on a 50x50 square unit area, i. e., some 10000 edges leading to some 20000 transmissions to schedule links symmetrically. For the $\Omega(1)$-sender models, we compare three configurations. First, the graph-based (ρ,ρ) model with the minimum ρ we can prove correctness for, $\rho = 2.82\ell(\mathcal{M}_G)$, second a $(\rho,\ell(M_G))$ model with minimum ρ in the same sense, which is local, but not graph-based. Third, we increase the locality radius ρ beyond what is necessary to prove correctness. Additionally, we compare to the three corresponding variants of the $\Omega(1)$-sender models, where the locality radius ρ is centered at the receiver, where the interference occurs. We call this receiver-centered locality. Not that in this case, a minimal choice of locality and exclusion radius means that the exclusion disk (centered at the sender) must completely be covered by the locality disk centered at the receiver. This class of models is never graph-based, but allows for better bounds and schedules. Table 6.1 shows values for random links, averaged over 250 runs, a selection is also plotted in Figure6.8 together with results from scheduling UDG links. Not surprisingly, the more far-seeing global algorithm performs best among all compared schemes and the global algorithm from [GOW07] by far worst. Among the greedy scheduling algorithms, the global view did not give an advantage in general. Greedy scheduling in the SINR_G model only outperformed the graph-based variant (whose big advantage is its simplicity). Increasing locality a little more or switching to the receiver-centered locality, the local models even led to better results since the exclusion radius prevented scheduling of close links and receiver-centered locality reflects the nature of interference better.



(a) 20000 random links



(b) 20000 links in a random UDG

Figure 6.8: *Slot utilization of slots for different scheduling algorithms*

Chapter 7

Conclusion

This thesis addressed two problems arising in an early stage of self-organization of wireless sensor networks: autonomous positioning and medium access control. We approached these problems with a focus on the impact of the model that is used to describe the behavior of the wireless channel and on the algorithmic aspects, namely complexity and the design of scalable algorithms, in which a single node's burden does not or only reasonably increase with the network's size.

For positioning, we showed how direction-based positioning can be interpreted as the problem of finding a maximal partition of the network into pieces which can be unambiguously reconstructed. Dividing this problem into the distributed identification of rigid subgraphs and the identification of maximal rigid groups of these subgraphs, the data that needs to be collected to finally solve this problem at a central node can be reduced drastically. For the central part, we propose a tailored algorithm for the identification of rigid structures in sets of rigid subgraphs that overlap in common nodes, improving on standard algorithms for this task. Unfortunately, we were also able to prove that positioning based on local information is hard in the case that nodes know distances and directions of their neighbors up to arbitrarily small errors. We hence turned our attention to a more heuristic approach of positioning, asking for a hierarchical version of common schemes that patch local solutions together. We developed a distributed multi-level scheme for anchor-free positioning, in which the load in terms of messages sent and local computation increases only logarithmically with the diameter of the network. It is based on the idea of graph filtration, which has been used for graph drawing earlier. Together with multidimensional scaling as the underlying technique for positioning local neighborhoods, this approach allows for accurate and robust positioning of large-scale networks in complex deployment areas even if only connectivity information is available. This combination, MDS-MAP(F), has been evaluated on networks of up to 128000 nodes and outperforms existing approaches in terms of positioning quality. It also is the first technique that demonstrates positioning of large-scale

sensor networks in 3D. Yet, we will have to wait for real sensor networks to grow to leave the field of simulations that are based on connectivity models which try to anticipate the characteristics of wireless connectivity.

In the area of coordinating medium access, we first addressed the problem how nodes can agree on a schedule for a query tree in order to gather large amounts of data with a minimum of set-up communication. Gathering of information currently is the most common kind of traffic pattern in sensor networks, and scheduling such trees in-network can greatly reduce the energy wasted in collecting the data. We proposed solutions to set up schedules for different interference models, ranging from optimal solutions in the absence of interference and time-approximative solutions for hop-interference to the conjecture that it is impossible to agree on a time-optimal schedule under the restrictions not to use the channel concurrently at all. However, we were only able to prove this conjecture in a special case. Also, we described generically how to make schedules robust to transmission failures with logarithmic buffer sizes and constant time-approximation or vice versa.

In Chapter 6, we addressed the question for the influence of interference models on the scheduling problems in a wider scope: We analyzed the relationship between graph-based interference models widely used to model interference problems and the more realistic geometric SINR model. We introduced the notion of local interference models and showed that any local model, which only accepts transmissions that are valid with respect to a geometric SINR model, can be replaced by a graph-based model without losing more than a constant factor in optimal solutions of scheduling problems.

Appendix A

Proofs

Proof of Equation 5.17

Claim A.1 *Let $\gamma \geq 2$ and $0 < \alpha \leq 1$. Then*

$$f_\gamma(\alpha) := \frac{1-(1-\alpha)^{\gamma/\alpha} - \gamma \cdot (1-\alpha)^{\gamma/\alpha - 1}}{(1-\alpha)^{\gamma/\alpha}} \geq \exp(\gamma/\sqrt{2}) \ . \tag{A.1}$$

Proof. First, we observe that for any fixed γ

$$f_\gamma(\alpha) = \underbrace{(1-\alpha)^{-\gamma/\alpha}}_{g_\gamma(\alpha)} - 1 - \underbrace{\gamma/(1-\alpha)}_{h_\gamma(\alpha)} \tag{A.2}$$

is strictly monotone increasing for $0 < \alpha < 1$. This follows from $g'_\gamma(\alpha) > h'_\gamma(\alpha)$, or, in more detail, from

$$\left((1-\alpha)^{-\alpha}\right)' > \left((1-\alpha)^{-1}\right)' > 1 \tag{A.3}$$

for $0 < a < 1$ and from $(f(x)^\gamma)' = \gamma f'(x)^{\gamma-1} \geq \gamma f'(x) = (\gamma f(x))'$ for $\gamma \geq 2$ where $f'(x) > 0$ and differentiable. It is thus sufficient to show that $\lim_{\alpha\to 0} f_\gamma(\alpha) \geq e^{\gamma/\sqrt{2}}$ for all $\gamma \geq 2$. We start with the observation that

$$\lim_{\alpha\to 0} g_\gamma(\alpha) = \lim_{\alpha\to 0}(1-\alpha)^{-\gamma/\alpha} = \lim_{n\to\infty}\left(1+\frac{1}{n-1}\right)^{-\gamma n} \geq e^\gamma \ . \tag{A.4}$$

We thus get the result for $\gamma = 2$ as

$$\lim_{\alpha\to 0} f_2(\alpha) = e^2 - 1 - 2 > e^{\sqrt{2}} \ . \tag{A.5}$$

and conclude showing that $\lim_{\alpha\to 0} f_\gamma(\alpha) - e^{\gamma/\sqrt{2}}$ is strictly increasing for $\gamma \geq 2$:

$$\left(e^\gamma - 1 - \gamma - e^{\gamma/\sqrt{2}}\right)\frac{d}{d\gamma} = \gamma e^{\gamma/\sqrt{2}} \cdot \left(e^{\sqrt{2}} - \frac{1}{\sqrt{2}}\right) - 1 \geq 2e^{\sqrt{2}} \cdot \left(e^{\sqrt{2}} - \frac{1}{\sqrt{2}}\right) - 1 > 0 \quad \text{(A.6)}$$

□

[ADM04] Ittai Abraham, Danny Dolev, and Dahlia Malkhi. LLS: a Locality Aware Location Service for Mobile Ad Networks. In Stefano Basagni and Cynthia A. Phillips, editors, *Proceedings of the DIALM-POMC Joint Workshop on Foundations of Mobile Computing (DIALM-POMC 2004)*, pages 75–84. ACM Press, October 2004.

[AEG+06] James Aspnes, Tolga Eren, David K. Goldenberg, A. Stephen Morse, Walter Whiteley, Yang Richard Yang, Brian D. O. Anderson, and Peter N. Belhumeur. A Theory of Network Localization. *IEEE Transaction on Mobile Computing*, 5(12):1663–1678, December 2006.

[AGY04] James Aspnes, David K. Goldenberg, and Yang Richard Yang. On the Computational Complexity of Sensor Network Localization. In *Proceedings of the 1st International Workshop on Algorithmic Aspects of Wireless Sensor Networks (ALGOSENSORS'04)*, pages 32–44. Springer-Verlag, July 2004.

[AMO93] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.

[Bag05] Aline Baggio. Wireless Sensor Networks in Precision Agriculture. In *Proceedings of the 2005 ACM Workshop on Real-World Wireless Sensor Networks (REALWSN'05)*, 2005.

[BFN01] Lali Barrière, Pièrre Fraigniaud, and Lata Narayanan. Robust Position-based Routing in Wireless Ad Hoc Networks with Unstable Transmission Ranges. In *Proceedings of the 5th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (Dial-M'01)*, pages 19–29. ACM Press, 2001.

[BGJ05] Jehoshua Bruck, Jie Gao, and Anxiao Jiang. Localization and Routing in Sensor Networks by Local Angle Information. In *Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC'05)*, pages 181–192. ACM Press, 2005.

[BGK+06] Jean-Claude Bermond, Jérôme Galtier, Ralf Klasing, Nelson Morales, and Stéphane Pérénnes. Hardness and Approximation of Gathering in Static Radio Networks. *Parallel Processing Letters*, 16(2):165–184, 2006.

[BGMS06] Amitabh Basu, Jie Gao, Joseph S. B. Mitchell, and Girishkumar Sabhnani. Distributed Localization Using Noisy Distance and Angle Information. In *Proceedings of the 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC'06)*, pages 262–273. ACM Press, 2006.

[BK98] Heinz Breu and David G. Kirkpatrick. Unit Disk Graph Recognition is NP-Hard. *Computational Geometry: Theory and Applications*, 9(1–2):3–24, 1998.

[BKMSS06] V. Bonifaci, P. Korteweg, A. Marchetti-Spaccamela, and L. Stougie. An Approximation Algorithm for the Wireless Gathering Problem. In *Proceeding of the 10th Scandinavian Workshop on Algorithm Theory (SWAT'06)*. Springer-Verlag, 2006.

[BOP07] Matt Bromage, Katia Obraczka, and Donald Potts. SEA-LABS: A Wireless Sensor Network for Sustained Monitoring of Coral Reefs. In *Proceedings of the 6th International IFIP-TC6 Networking Conference (NETWORKING 2007)*, pages 1132–1135, 2007.

[BP06] Ulrik Brandes and Christian Pich. Eigensolver Methods for Progressive Multidimensional Scaling. In *Proceedings of the 14th International Symposium on Graph Drawing (GD'06)*, 2006.

[BR03] A. Behzad and I. Rubin. On the Performance of Graph-based Scheduling Algorithms for Packet Radio Networks. In *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM'03)*, 2003.

[BRY+04] Maxim Batalin, Mohammed Rahimi, Yan Yu, Duo Liu, Aman Kansal, Gaurav Sukhatme, W. J. Kaiser, Mark Hansen, Gregory Pottie, Mani B. Srivastava, and D. Estrin. Call and Response: Experiments in Sampling the Environment. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SENSYS'04)*, pages 25–38. ACM Press, 2004.

[BvRW07] Nicolas Burri, Pascal von Rickenbach, and Roger Wattenhofer. DOZER: Ultra-Low Power Data Gathering in Sensor Networks. In *Sixth International Symposium on Information Processing in Sensor Networks (IPSN '07)*, pages 450–459. ACM Press, 2007.

[BVY04] P. Björklund, P. Värbrand, and D. Yuan. A Column Generation Method for Spatial TDMA Scheduling in Ad hoc Networks. *Ad Hoc Networks*, 2(4):4005–418, 2004.

[BY04] Pratik Biswas and Yinyu Ye. Semidefinite Programming for Ad hoc Wireless Sensor Network Localization. In *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks (IPSN'04)*, pages 46–54. ACM Press, 2004.

[CC01] Trevor F. Cox and Michael A. A. Cox. *Multidimensional Scaling*. Chapman & Hall/CRC, 2001.

[CCJ90] Brent N. Clark, Charles J. Colbourn, and David S. Johnson. Unit disk graphs. *Discrete Mathematics*, 86:165–177, 1990.

[CDB+04] Krishna Kant Chintalapudi, Karthik Dantu, Sandeep Babel, Ramesh Govindan, Gaurav Sukhatme, and John Caffrey. A Sensor-Actuator Network for Damage Detection in Civil Structures. In *Proceedings of the 2nd International Conference*

on Embedded Networked Sensor Systems (SENSYS'04), pages 323–323. ACM, 2004.

[CFP+06] Krishna Chintalapudi, Tat Fu, Jeongyeup Paek, Nupur Kothari, Sumit Rangwala, John Caffrey, Ramesh Govindan, Erik Johnson, and Sami Masri. Monitoring Civil Structures with a Wireless Sensor Network. *IEEE Internet Computing*, 10, 2006.

[CMY+02] Alvin Chen, Richard. R. Muntz, S. Yuen, Ivo Locher, Sung Park, and Mani B. Srivastava. A Support Infrastructure for the Smart Kindergarten. *IEEE Pervasive Computing*, 1(2):49–57, 2002.

[DEA06] Ilker Demirkol, Cem Ersoy, and Fatih Alagöz. MAC Protocols for Wireless Sensor Networks: A Survey. *IEEE Communications Magazine*, 44(4):115–121, 2006.

[DPG01] L. Doherty, K. S. J. Pister, and L. E. Ghaoui. Convex position estimation in wireless sensor networks. In *Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'01)*, pages 1655–1633, April 2001. Anchorage, AK.

[EBMP+05] L. Evers, M. J. J. Bijl, M. Marin-Perianu, R. S. Marin-Perianu, and P. J. M. Havinga. Wireless Sensor Networks and Beyond: A Case Study on Transport and Logistics. Technical report, University of Twente, 2005.

[EEF+06] Cesim Erten, Alon Efrat, David Forrester, Anand Iyer, and Stephen G. Kobourov. Force-Directed Approaches to Sensor Localization. In *Proceedings of the 8th Workshop on Algorithm Engineering and Experiments (ALENEX'06)*, 2006.

[EGW+04] Tolga Eren, David K. Goldenberg, Walter Whiteley, Yang Richard Yang, A. Stephen Morse, Brian D. O. Anderson, and Peter N. Belhumeur. Rigidity, Computation, and Randomization in Network Localization. In *Proceedings of the 23th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'04)*, April 2004.

[FW06] Roland Flury and Roger Wattenhofer. MLS: An Efficient Location Service for Mobile Ad Hoc Networks. In *7th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC), Florence, Italy*, May 2006.

[GCB03] Jose A. Gutierrez, Edgar H. Callaway, and Raymond Barrett. *IEEE 802.15.4 Low-Rate Wireless Personal Area Networks: Enabling Wireless Sensor Networks*. IEEE Standards Office, 2003.

[GGK04] P. Gajer, M.T. Goodrich, and Stephen G. Kobourov. A fast multi-dimensional algorithm for drawing large graphs. *Computational Geometry: Theory and Applications*, 29:3–18, 2004.

[GH01] J. Grönkvist and A. Hansson. Comparison Between Graph-Based and Interference-Based STDMA Scheduling. In *Proceedings of the 2nd ACM International Symposium on Mobile Ad Hoc Networking & Computing (MOBIHOC)*, pages 255–258, 2001.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.

[GK00] Piyush Gupta and P. R. Kumar. The Capacity of Wireless Networks. *IEEE Transactions on Information Theory*, 46(2):388–404, 2000.

[GK04] Craig Gotsman and Yehoda Koren. Distributed Graph Layout for Sensor Networks. In *Proceedings of the 12th International Symposium on Graph Drawing (GD'04)*. Springer-Verlag, 2004.

[God97] L. C. Godara. Application of antenna arrays to mobile communications II: Beam-forming and direction-of-arrival considerations. *Proceedings of the IEEE*, 85:1195–1245, 1997.

[GOW07] Olga Goussevskaia, Yvonne Anne Oswald, and Roger Wattenhofer. Complexity in Geometric SINR. In *Proceedings of the 8th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC'07)*, pages 100–109. ACM Press, 2007.

[GvL96] G. H. Golub and C. F. van Loan. *Matrix computations*. Johns Hopkins University Press, 1996.

[HDB04] B. Hohlt, L. Doherty, and E. A. Brewer. Flexible power scheduling for sensor networks. In *Third International Symposium on Information Processing in Sensor Networks (IPSN 2004)*, pages 205–214. IEEE Computer Society, Washington, DC, USA, 2004.

[Hen92] Bruce Hendrickson. Conditions for Unique Graph Realizations. *SIAM Journal of Computing*, 21(1):65–84, 1992.

[HMR+98] H. Hunt, M. Marathe, V. Radhakrishnan, S. Ravi, D. Rosenkrantz, and R. Stearns. NC-Approximation Schemes for NP- and PSPACE-Hard Problems for Geometric Graphs. *Journal of Algorithms*, 26, 1998.

[HR90] Torben Hagerup and Christine Rüb. A guided tour to Chernoff bounds. *Information Processing Letters*, 33:305–308, 1990.

[HS88] B. Hajek and G. Sasaki. Link Scheduling in Polynomial Time. *IEEE Transactions on Information Theory*, 34(5):910–917, 1988.

[HVY+06] Tian He, Pascal Vicaire, Ting Yan, Liqian Luo, Lin Gu, Gang Zhou, Radu Stoleru, Qing Cao, John A Stankovic, and Tarek F. Abdelzaher. Achieving Real-Time Target Tracking Using Wireless Sensor Networks. In *IEEE Real Time Technology and Applications Symposium*, pages 37–48, 2006.

[JH97] D. J. Jacobs and B. Hendrickson. An algorithm for two-dimensional rigidity percolation: the pebble game. *Journal of Computational Physics*, 137:346–365, 1997.

[Jun08] Dieter Jungnickel. *Graphs, Networks and Algorithms*. Springer-Verlag, 3 edition, 2008.

[JZ04] Xiang Ji and Hongyuan Zha. Sensor Positioning in Wireless Ad-hoc Sensor Networks Using Multidimensional Scaling. In *Proceedings of the 23th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'04)*, April 2004. Hong Long, China.

[KDD04] Uwe Kubach, Christian Decker, and Ken Douglas. Collaborative control and coordination of hazardous chemicals. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SENSYS'04)*, 2004.

[KFPF06] Alexander Kröller, Sándor P. Fekete, Dennis Pfisterer, and Stefan Fischer. Deterministic boundary recognition and topology extraction for large sensor networks. In *Proceedings of the 17th Annual ACM–SIAM Symposium on Discrete Algorithms (SODA'06)*, pages 1000–1009, 2006.

[KGW07] Bastian Katz, Marco Gaertler, and Dorothea Wagner. Maximum Rigid Components as Means for Direction-Based Localization in Sensor Networks. In Jan van Leeuwen et al., editor, *Proceedings of the 33rd International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM'07)*, volume 4362, pages 330–341. Springer-Verlag, 2007.

[KK00] Brad Karp and H. T. Kung. GPSR: Greedy Perimeter Stateless Routing for Wireless Networks. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MOBICOM'00)*, pages 243–254, 2000.

[KKP99] Joe M. Kahn, Randy H. Katz, and Kristofer S. J. Pister. Next century challenges: mobile networking for "Smart Dust". In *MobiCom '99: Proceedings of the 5th annual ACM/IEEE International Conference on Mobile Computing and Networking*, pages 271–278. ACM, 1999.

[KMNW05] Fabian Kuhn, Thomas Moscibroda, Tim Nieberg, and Roger Wattenhofer. Fast Deterministic Distributed Maximal Independent Set Computation on Growth-Bounded Graphs. In *In Proceedings of the 19th Conference on Distributed Computing (DISC'05)*, 2005.

[KMPS04] V. S. A. Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan. End-to-end packet scheduling in wireless ad-hoc networks. In *Proceedings of the 15th annual ACM-SIAM symposium on Discrete Algorithms (SODA'04)*, pages 1021–1030, 2004.

[KMR00] S. O. Krumke, M. Marathe, and S. S. Ravi. Models and approximation algorithms for channel assignment in radio networks. *Wireless Networks*, 6:575–584, 2000.

[KMW04] Fabian Kuhn, Thomas Moscibroda, and Roger Wattenhofer. Unit Disk Graph Approximation. In *Proceedings of the 8th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (Dial-M'04)*, 2004.

[KMW06] Fabian Kuhn, Thomas Moscibroda, and Roger Wattenhofer. The Price of Being Near-Sighted. In *Proceedings of the 17th Annual ACM–SIAM Symposium on Discrete Algorithms (SODA'06)*, 2006.

[KMW08] Bastian Katz, Steffen Mecke, and Dorothea Wagner. Efficient Scheduling of Data-Harvesting Trees. In *Proceedings of the 4th International Workshop on Algorithmic Aspects of Wireless Sensor Networks*, LNCS. Springer-Verlag, 2008. to appear.

[Krö08] Alexander Kröller. *Algorithms for Topology-Aware Sensor Networks*. PhD thesis, Braunschweig Institute of Technology, 2008.

[KVW08] Bastian Katz, Markus Völker, and Dorothea Wagner. Link Scheduling in Local Interference Models. In *Proceedings of the 4th International Workshop on Algorithmic Aspects of Wireless Sensor Networks*, LNCS. Springer-Verlag, 2008. to appear.

[KW07] Bastian Katz and Dorothea Wagner. Multi-scale Anchor-free Distributed Positioning in Sensor Networks. In *Proceedings of the International Workshop on Theoretical and Algorithmic Aspects of Sensor and Ad-hoc Networks (WTASA'07)*, 2007.

[KW08] Bastian Katz and Dorothea Wagner. Multi-scale Anchor-free Distributed Positioning in Sensor Networks. In S. K. Makki, X.-Y. Li, N. Pissinou, S. Makki, M. Karimi, and K. Makki, editors, *Sensor and Ad-Hoc Networks, Theoretical and Algorithmic Aspects*, volume 7 of *LNEE*. Springer-Verlag, 2008.

[Lam70] G. Laman. On graphs and rigidity of plane skeletal structures. *Journal of Engineering Mathematics*, 4:331–340, 1970.

[LCD03] Jessica D. Lundquist, Daniel R. Cayan, and Michael D. Dettinger. Meteorology and hydrology in Yosemite national park: A sensor network application. In *Proceedings of the 2nd International Symposium on Information Processing in Sensor Networks (IPSN'03)*, 2003.

[LGA+06] Mathew Laibowitz, Jonathan Gips, Ryan Aylward, Alex Pentland, and Joseph A. Paradiso. A sensor network for social dynamics. In *IPSN '06: Proceedings of the 5th international conference on Information processing in sensor networks*, pages 483–491, New York, NY, USA, 2006. ACM Press.

[LH05] Koen Langendoen and Gertjan Halkes. Energy-Efficient Medium Access Control. In R. Zurawski, editor, *Embedded Systems Handbook*. CRC Press, 2005.

[Lin92] N. Linial. Locality in Distributed Graph Algorithms. *SIAM Journal on Computing*, 21:193–201, 1992.

[LKR04] G. Lu, B. Krishnamachari, and C. S. Raghavendra. An adaptive energy-efficient and low-latency mac for data gathering in wireless sensor networks. In *Proceedings of the 18th International Parallel and Distributed Processing Symposium (IPDPS'04)*. IEEE Computer Society, Washington, DC, USA, 2004.

[Lor79] M. Loréa. On matroidal families. *Discrete Mathematics*, 28:103–106, 1979.

[LS07] Audrey Lee and Ileana Streinu. Pebble Game Algorithms and Sparse Graphs. *Discrete Mathematics*, 308(8):1425–1437, 2007.

[LvRW08] Thomas Locher, Pascal von Rickenbach, and Roger Wattenhofer. Sensor Networks Continue to Puzzle: Selected Open Problems. In *Proceedings of the 9th International Conference on Distributed Computing and Networking (ICDCN'08)*, 2008.

[LWG08] Sol Lederer, Yue Wang, and Jie Gao. Connectivity-based Localization of Large Scale Sensor Networks with Complex Shape. In *Proceedings of the 27th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'08)*, 2008.

[Mar06] Margaret Martonosi. Embedded systems in the wild: Zebranet software, hardware, and deployment experiences. In *LCTES '06: Proceedings of the 2006 ACM SIGPLAN/SIGBED conference on Language, compilers, and tool support for embedded systems*. ACM Press, 2006.

[MMG+08] John McCulloch, Paul McCarthy, Siddeswara Mayura Guru, Wei Peng, Daniel Hugo, and Andrew Terhorst. Wireless sensor network deployment for water use efficiency in irrigation. In *REALWSN '08: Proceedings of the workshop on Real-world wireless sensor networks*, 2008.

[MOW07] Thomas Moscibroda, Yvonne Anne Oswald, and Roger Wattenhofer. How Optimal are Wireless Scheduling Protocols? In *Proceedings of the 26th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'07)*, 2007.

[MOWW04] Thomas Moscibroda, Regina O'Dell, Mirjam Wattenhofer, and Roger Wattenhofer. Virtual Coordinates for Ad hoc and Sensor Networks. In *Proceedings of the DIALM-POMC Joint Workshop on Foundations of Mobile Computing (DIALM-POMC'04)*, 2004.

[MPR+05] K. Martinez, P. Padhy, A. Riddoch, H. L. R Ong, and J. K. Hart. Glacial Environment Monitoring using Sensor Networks. In *Proceedings of the 2005 ACM Workshop on Real-World Wireless Sensor Networks (REALWSN'05)*. ACM Press, 2005.

[MPS+02] Alan Mainwaring, Joseph Polastre, Robert Szewczyk, David Culler, and John Anderson. Wireless Sensor Networks for Habitat Monitoring. In *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Application (WSNA'02)*, 2002.

[MW05] Thomas Moscibroda and Roger Wattenhofer. Coloring unstructured radio networks. In *Proc. of the 17th Symposium on Parallel Algorithms and Architectures (SPAA)*, 2005.

[MW06] Thomas Moscibroda and Roger Wattenhofer. The Complexity of Connectivity in Wireless Networks. In *Proceedings of the 25th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'06)*, 2006.

[MWW06] Thomas Moscibroda, Roger Wattenhofer, and Yves Weber. Protocol design beyond graph-based models. In *Proceedings of the 5th Workshop on Hot Topics in Networks (HotNets)*, 2006.

[NHS+08] L. Nachman, J. Huang, J. Shahabdeen, R. Adler, and R. Kling. IMOTE2: Serious Computation at the Edge. In *Proceedings of the International Wireless Communications and Mobile Computing Conference (IWCMC'08)*, 2008.

[NS95] M. Naor and L. Stockmeyer. What can be computed locally? *SIAM Journal on Computing*, 24(6):1259–1277, 1995.

[PBDT03] Nissanka B. Priyantha, Hari Balakrishnan, Erik Demaine, and Seth Teller. Anchor-Free Distributed Localization in Sensor Networks. In *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems (SENSYS'03)*, pages 340–341, 2003.

[PCB00] Nissanka B. Priyantha, Anit Chakraborty, and Hari Balakrishnan. The Cricket Location-Support System. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MOBICOM'00)*, pages 32–43, 2000.

[PEC06] G.J. Pendock, L. Evans, and G. Coulson. Wireless sensor nodes for monitoring bats. In *Proceedings of the ACM Workshop on Real-World Wireless Sensor Networks (REALWSN 2006)*, 2006.

[Pel00] David Peleg. *Distributed Computing: A Locality-sensitive Approach*. Society for Industrial and Applied Mathematics, Philadelphia, USA, 2000.

[Rap96] T. Rappaport. *Wireless Communications: Principles and Practices*. Prentice Hall, 1996.

[RKM+06] Jason Redi, Steve Kolek, Keith Manning, Craig Partridge, Regina Rosales-Hain, Ram Ramanathan, and Isidro Castineyra. JAVeLEN – An ultra-low energy ad hoc wireless network. *Ad Hoc Networks*, 2006.

[Sax79] James B. Saxe. Embeddability of weighted graphs in k-space is strongly NP-hard. In *Proceedings of the 17th Allerton Conference in Communications, Control and Computing*, pages 480–489, 1979.

[SBH07] D. L. Swain and G. J. Bishop-Hurley. Using contact logging devices to explore animal affiliations: quantifying cow-calf interactions. *Applied Animal Behaviour Science*, 102:1–11, 2007.

[Sch86] Alexander Schrijver. *Theory of Linear and Integer Programming*. John Wiley and Sons, New York, 1986.

[SCL+05] Victor Shnayder, Bor-Rong Chen, Konrad Lorincz, Thaddeus R. F. Fulford Jones, and Matt Welsh. Sensor networks for medical care. In *Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems (SENSYS'05)*. ACM Press, 2005.

[SHM+08] Jianping Song, Song Han, Al Mok, Deji Chen, Mike Lucas, Mark Nixon, and Wally Pratt. WirelessHART: Applying Wireless Technology in Real-Time Industrial Process Control. *Real-Time and Embedded Technology and Applications Symposium*, 0:377–386, 2008.

[SMAL+04] Gyula Simon, Miklós Maróti, Ákos Lédeczi, György Balogh, Branislav Kusy, András Nádas, Gábor Pap, János Sallai, and Ken Frampton. Sensor Network-Based Countersniper System. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SENSYS'04)*, pages 1–12, New York, NY, USA, 2004. ACM.

[Smi07] Randall B. Smith. SPOTWorld and the Sun SPOT. In *IPSN '07: Proceedings of the 6th international conference on Information processing in sensor networks*, New York, NY, USA, 2007. ACM.

[SMP+04] Robert Szewczyk, Alan Mainwaring, Joseph Polastre, John Anderson, and David Culler. An analysis of large scale habitat monitoring application. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SENSYS'04)*, 2004.

[SOV08] Silvia Santini, Benedikt Ostermaier, and Andrea Vitaletti. First experiences using wireless sensor networks for noise pollution monitoring. In *Proceedings of the 3rd ACM Workshop on Real-World Wireless Sensor Networks (REALWSN'08)*, 2008.

[SR04] Yi Shang and Wheeler Ruml. Improved MDS-Based Localization. In *Proceedings of the 23th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'04)*, 2004.

[SRF06] Yi Shang, Wheeler Ruml, and Markus P.J. Fromherz. Positioning using local maps. *Ad Hoc Networks*, 4(2):240–253, 2006.

[SRZF03] Yi Shang, Wheeler Ruml, Ying Zhang, and Markus P. J. Fromherz. Localization from Mere Connectivity. In *Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC'03)*. ACM Press, 2003.

[SW06] Stefan Schmid and Roger Wattenhofer. Algorithmic Models for Sensor Networks. In *20th IEEE International Parallel and Distributed Processing Symposium (IPDPS '06)*, 2006.

[TAMW06] Andreas Terzis, Annalingam Anandarajah, Kevin Moore, and I-Jeng Wang. Slip surface localization in wireless sensor networks for landslide prediction. In *IPSN '06: Proceedings of the 5th international conference on Information processing in sensor networks*, pages 109–116, New York, NY, USA, 2006. ACM.

[TCPK01] Attila Tanács, Gábor Czédli, Kálmán Palágyi, and Attila Kuba. Affine matching of two sets of points in arbitrary dimensions. *Acta Cybernetica*, 15(1):101–106, 2001.

[Tor52] Warren S. Torgerson. Multidimensional Scaling: I. Theory and Method. *Psychometrika*, 17(4):401–419, 1952.

[Tot65] L. Fejes Toth. Minkowskian distribution of discs. *Proceedings of the AMS*, 16(5):999–1004, 1965.

[TW07a] Volker Turau and Christoph Weyer. Long-term Reliable Data Gathering Using Wireless Sensor Networks. In *Fourth International Conference on Networked Sensing Systems 2007*, 2007.

[TW07b] Volker Turau and Christoph Weyer. Scheduling Transmission of Bulk Data in Sensor Networks using a Dynamic TDMA Protocol. In *Proc. Int. Workshop on Data Intensive Sensor Networks 2007 (DISN'07)*, 2007.

[TW07c] Volker Turau and Christoph Weyer. TDMA Schemes for Tree-Routing in Data Intensive Wireless Sensor Networks. In *Proceedings of the First International Workshop on Protocols and Algorithms for Reliable and Data Intensive Sensor Networks (PARIS)*, pages 1–6. IEEE Computer Society, Washington, DC, USA, 2007.

[WASW06] Geoffrey Werner-Allen, Patrick Swieskowski, and Matt Welsh. Real-Time Volcanic Earthquake Localization. In *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems (SENSYS'06)*, pages 357–358, 2006.

[WGM06] Yue Wang, Jie Gao, and Joseph S. B. Mitchell. Boundary Recognition in Sensor Networks by Topological Methods. In *Proceedings of the 12th Annual International Conference on Mobile Computing and Networking (MOBICOM'06)*, 2006.

[Whi88] W. Whiteley. The union of matroids and the rigidity of frameworks. *SIAM Journal Discrete Mathematics*, 1(2):237–255, 1988.

[Whi97] W. Whiteley. Matroids from discrete geometry. In J. Bonin, J. Oxley, and B. Servatius, editors, *Matroid Theory*, AMS Contemporary Mathematics, pages 171–313. AMS, 1997.

[WSC07] Sheng-Shih Wang, Kuei-Ping Shih, and Chih-Yung Chang. Distributed direction-based localization in wireless sensor networks. *Computer Communications*, 30(6):1424–1439, 2007.

[YAGS06] Yong Yao, S. M. Nazrul Alam, Johannes Gehrke, and Sergio D. Servetto. Network Scheduling for Data Archiving Applications in Sensor Networks. In *Proceedings of the 3rd International Workshop on Data Management for Sensor Networks*, 2006.

[ZG04] Feng Zhao and Leonidas J. Guibas. *Wireless Sensor Networks: An Information Processing Approach*. Morgan Kaufmann Publishers Inc., 2004.

List of Figures

List of Tables

List of Algorithms and Protocols

Bastian Katz
born June 5, 1980 in Lübeck, Germany

Current status

since 02/06 **Research and teaching assistant**, chair *Algorithmics I*
Universität Fridericiana zu Karlsruhe (TH)

Education

02/06–01/09 **PhD student** in Informatics, member of the DFG research training group 1194: *Self-organizing Sensor-Actuator Networks*
Universität Fridericiana zu Karlsruhe (TH)
Advisors: Prof. Dr. D. Wagner, Prof. Dr. U. D. Hanebeck

10/00–01/06 **Diplom** (German M.Sc.) in Informatics
Universität Fridericiana zu Karlsruhe (TH)
Thesis: *Richtungsbasierte Lokalisierung in Sensornetzen*

08/91–07/00 **Abitur** (German A-level equivalents)
Katharineum zu Lübeck

Studies abroad

10/07 **Visiting academic** at the Department of Computer Science
Stanford University, Palo Alto
Advisor: Prof. Dr. L. Guibas

10/07–11/07 **Visiting academic** at the Department of Computer Science
State Univerity of New York, Stony Brook
Advisor: Prof. Dr. J. Gao

11/02–02/03 **Visiting academic** at the Computer Science Laboratory (RSISE)
Australian National University, Canberra
Advisor: Prof. Dr. R. Gore