Graph Connectivity with Respect to Wireless Ad-Hoc Sensor Networks

Inaugural-Dissertation

zur Erlangung des Doktorgrades
der Mathematisch-Naturwissenschaftlichen Fakultät
der Heinrich-Heine-Universität Düsseldorf

vorgelegt von

Stefan Hoffmann

aus Mönchengladbach

Düsseldorf, Mai 2016

aus dem Institut für Informatik
der Heinrich-Heine-Universität Düsseldorf

Gedruckt mit der Genehmigung der
Mathematisch-Naturwissenschaftlichen Fakultät der
Heinrich-Heine-Universität Düsseldorf

Referent: Prof. Dr. Egon Wanke

Korreferent: Prof. Dr. Jörg Rothe

Tag der mündlichen Prüfung: 15. Juli 2016

Erklärung

Ich versichere an Eides Statt, dass die Dissertation von mir selbständig und ohne unzulässige fremde Hilfe unter Beachtung der "Grundsätze zur Sicherung guter wissenschaftlicher Praxis an der Heinrich-Heine-Universität Düsseldorf" erstellt worden ist.

Desweiteren erkläre ich, dass ich eine Dissertation in der vorliegenden oder in ähnlicher Form noch bei keiner anderen Institution eingereicht habe.

Teile dieser Arbeit wurden bereits in folgenden Schriften veröffentlicht:

- Die in den Proceedings der IPSN 2016 Konferenz publizierte Schrift [47] enthält die Theoreme 10, 15, die zugehörigen Lemmata und Definitionen sowie die Ergebnisse des Abschnitts "Greedy Algorithm for LCWA"
- Die Resultate des Kapitels 4 sind als technischer Bericht erschienen. [45]

In thematischem Zusammenhang mit dieser Dissertation sind ebenfalls folgende Schriften entstanden: [36, 42–44, 46, 48]

Eine detaillierte Erklärung zu meiner Beteiligung an diesen Schriften befindet sich am Ende dieser Arbeit in dem Abschnitt "Contributions to Papers with Multiple Authors".

Düsseldorf, 22. Mai 2016 Stefan Hoffmann

Zusammenfassung

Diese Arbeit beschäftigt sich mit Zusammenhangsproblemen, die aus der Betrachtung von Funknetzwerken resultieren. Es wird u.a. die Frage beantwortet wie eine Nachricht an alle Nachbarn eines Knotens v verteilt werden kann, unter der Annahme, dass v nicht zur Erfüllung dieser Aufgabe beiträgt. Dieses Problem tritt z.B. bei der Reparatur von Routingwegen oder der gemeinsamen Entscheidungsfindung auf und wird durch Angabe des sog. k-HBF Protokolls mit Parameter $k \in \mathbb{N}$ gelöst. Es wird bewiesen, dass das Protokoll erfolgreich die Nachricht verteilt, wenn $k \geq 2d-1$ gilt und v d-lokal zusammenhängend ist, d.h. dass die d-Hop Nachbarschaft von v einen zusammenhängenden Graphen induziert. Es wird auch gezeigt, dass k-HBF bzgl. der verwendeten Knoten optimal ist: Jeder beteiligte Knoten muss von jedem Protokoll verwendet werden, das die Verteilung garantiert und auf die gleichen Topologieinformationen beschränkt ist. Empirische Auswertungen zeigen zudem hohe Erfolgsraten für kleine Werte von k. Außerdem wird untersucht wie sich die kleinste Zahl d, so dass ein gegebener Knoten d-lokal zusammenhängend ist, in Linearzeit bestimmen lässt und wie dieses Verfahren verteilt implementiert werden kann. Ausgehend von dieser Verbindung zum lokalen Zusammenhang werden anschließend graphentheoretische Probleme analysiert, mit Hilfe derer diese Eigenschaft durch Topologiekontrolle beeinflusst werden kann. Besondere Beachtung erhält dabei die Frage wie viele zusätzliche Kanten eingefügt werden müssen, damit die 1-Hop Nachbarschaft jedes Knotens einen zusammenhängenden Teilgraphen induziert, wobei sich zwei Varianten dieses Augmentierungsproblems ergeben: In der *starken* Version werden die Nachbarschaften im erweiterten Graphen betrachtet, in der *schwachen* Version die Nachbarschaften im ursprünglichen Graphen. Es wird gezeigt, dass beide Varianten NP-vollständig sind und wie die schwache Version mit einem Faktor von $1 + ln(\Delta)$ approximiert werden kann, wobei Δ den maximalen Knotengrad bezeichnet. Ebenfalls NP-vollständig sind die Frage nach dem größten zusammenhängenden induzierten Teilgraphen, in dem alle Knoten 1-lokal zusammenhängend sind und die Frage, wie viele Kanten entfernt werden können ohne diese Eigenschaft zu verletzen - letzteres wieder in zwei Varianten. Abschließend werden Funknetzwerke mit zwei Sendestärken betrachtet. Die Annahme jeder Knoten könne durch Erhöhung der Sendeleistung zusätzliche Nachbarn erreichen, wirft die Frage auf, wie viele Knoten mit höherer Leistung senden müssen, damit ein zusammenhängendes Netzwerk entsteht. Für dieses NP-vollständige Problem wird ein parametrisierter Approximationsalgorithmus vorgestellt, der eine schrittweise Erhöhung der Approximationsgüte auf Kosten zusätzlicher Rechenleistung ermöglicht.

Abstract

This thesis deals with problems that are related to graph connectivity and emerge from wireless ad-hoc multi-hop networks. The so-called NEIGHBORHOOD BROADCAST problem describes the task of distributing a message across the neighborhood of a node v under the assumption that v will not contribute to the solution. This routing problem occurs during route repair and collaborative decision making and is solved by presenting the k-HBF protocol for a positive integer k. It is proven that the protocol is successful, if $k \geq 2d-1$ and v is d-locally connected, meaning that the d-hop neighborhood of v induces a connected subgraph. It is also shown that the set of nodes participating in the execution is optimal under all protocols that are restricted to the same topology information, i.e. that every participating node also has to participate in every protocol that guarantees delivery. Simulations based on commonly used wireless network graph models demonstrate a high success rate for low values of k. The question of how to determine, in linear time, the minimum integer d such that a vertex in a given graph is d-locally connected is also answered and it is discussed how this algorithm can be implemented in a distributed environment. Due to the relationship to local connectivity, several graph theoretic problems related to topology control are considered to investigate possibilities for increasing local connectivity during network design. Special attention is given to graph augmentation in this context, i.e. the question of how many edges have to be added to a given graph in order to make the 1-hop neighborhood of every vertex induce a connected subgraph, a problem that is split into two different versions: The *strong* augmentation problem considers the neighborhoods of the augmented graph, while the *weak* augmentation problem is concerned with the original neighborhoods of the given graph. It is shown that the corresponding decision problems for both versions are NP-complete and an algorithm is developed and analyzed that approximates the weak augmentation problem within a factor of $1 + ln(\Delta)$, where Δ denotes the maximum vertex degree of the graph. It is also shown that computing the maximum connected induced subgraph, in which every vertex is 1-locally connected, is NP-complete and that the same result holds for deleting a maximum number of edges while preserving the 1-locally connected property. Finally, wireless networks with two distinct power levels are considered. Assuming that each node can increase transmission power to reach an additional set of neighbors yields the question of how many nodes have to use increased power to achieve connectivity of the network. This thesis presents a family of approximation algorithms for this NP-complete problem that allows gradually incrementing approximation quality by providing increased computational power.

Table of Contents

Introduction

This thesis deals with questions that arise from self-organizing, wireless computer networks, so called *wireless sensor networks (WSN)* or *mobile ad-hoc networks (MANET)*. One of the main characteristics of these networks is that they consist of many autonomous devices, so-called *nodes*, that need to collaborate to perform a certain task. For this purpose they exchange information with each other via radio transmissions. The communication structure of such a network is commonly modeled as a graph and therefore this topic is closely related to graph theory. This thesis focuses on the complexity and approximability of problems related to the graph theoretic concept of connectivity that emerge from these wireless network. Additionally, the obtained theoretic results are applied to computer networks by developing distributed protocols for certain problems that occur in practical deployments.

The motivation and possible applications for the individual topics as well as references to related work are given throughout the thesis in the beginning of the respective chapters and sections, which are organized as follows:

Chapter 1 is concerned with definitions, notations and algorithms from graph theory, complexity theory and wireless networking as these basics are used several times in the remainder of this work.

Chapter 2 introduces the so-called Neighborhood Broadcast, the task of distributing a message among the neighbors of a given node under the assumption that this particular node will not contribute to the solution. While this problem implicitly occurs in literature, it has not yet been studied on its own to a sufficient extent. This thesis provides a practical solution including thorough theoretical and experimental analysis. One of the results that is discovered during the analysis is that the possibility to perform a Neighborhood Broadcast depends on a graph property called *local connectivity*, which describes the connectivity of the subgraphs induced by the neighborhoods of the nodes in the network.

Chapter 3 therefore examines topology control approaches with respect to local connectivity, most of which aim for increasing local connectivity by altering the communication structure of the network. A second type of topology control that preserves existing local connectivity while decreasing necessary maintenance by removing redundant communication links is also considered in this part.

Chapter 4 analyses networks in which the nodes can choose between two transmission power levels and examines the most basic topology control problem in this scenario: The question of which nodes have to use the higher transmission power in order to obtain a connected communication graph.

Chapter 5 then concludes the thesis and discusses open problems and possible extensions that are suitable for future work.

1 Basic Terminology

This chapter provides an overview of basic definitions and notations that are used throughout the thesis. The reader is assumed to have some familiarity with graph theory, complexity theory and approximation algorithms as well as a basic understanding of computer networks.

1.1 Graph Theory

A pair (V, E) is an *undirected graph* with *vertex set* V and *edge set* E, if V is a finite set and $E \subseteq \{\{u, v\} \mid u, v \in V, u \neq v\}$. A pair (V, E) is a *directed graph* with *vertex set* V and *edge set* E, if V is a finite set and $E \subseteq V \times V$.

For an undirected graph $G = (V, E)$, an edge $e \in E$ is *incident* to a vertex $v \in V$, if $v \in e$ and two vertices $u, v \in V$ are *adjacent*, if $\{u, v\} \in E$. The *degree* of v, denoted by $\Delta(v)$, is the number of edges incident to v, a vertex $v \in V$ is called *isolated*, if $\Delta(v) = 0$ and the maximum vertex degree of G is defined as $\Delta(G) := max\{\Delta(u) \mid u \in V\}$.

A *path of length* $k \in \mathbb{N}$ *between two vertices* v_1 *and* v_{k+1} in a directed (undirected) graph $G = (V, E)$ is a sequence of vertices $v_1, \ldots, v_{k+1}$ such that $\forall i \in \{1, \ldots, k\} : (v_i, v_{i+1}) \in E$ $(\{v_i, v_{i+1}\} \in E)$. The length of a path p is denoted by $L(p)$. A path p is called *simple*, if all vertices on p are pairwise distinct.

A graph $G' = (V', E')$ is a *subgraph* of a graph $G = (V, E)$, if $V' \subseteq V$ and $E' \subseteq E$. For a subset of vertices $U \subseteq V$ of a directed (undirected) graph $G = (V, E)$ the graph $G|_U = (U, E|_U)$, $E|_U := \{(u, v) \in E \mid u, v \in U\}$ $(E|_U := \{\{u, v\} \in E \mid u, v \in U\})$ is called the *subgraph of* G *induced by* U.

Graph $G = (V, E)$ is *connected*, if there is a path between every pair of vertices. A *connected component* of G is a maximal induced subgraph of G that is connected. The set of vertex sets of the connected components is denoted by $CC(G)$. A vertex $v \in V$ is a called a *separation vertex*, if $G|_{V \setminus \{v\}}$ consists of more connected components than G. A subset $U \subseteq V$ of vertices is connected, if $G|_U$ is connected. For a positive integer $k \in \mathbb{N}$, G is k*-connected* if and only if $V \setminus U$ is connected for all vertex sets $U \subseteq V$ with $|U| \leq k - 1$.

Theorem 1. *An undirected graph* $G = (V, E)$ *is* k*-connected for a positive integer* $k \in \mathbb{N}$ *if and only if there are* k *internally vertex disjoint paths between every pair of vertices. [73]*

A path $u_1, \ldots, u_k$, $k \geq 3$ in an undirected graph $G = (V, E)$ is a *cycle* if $\{u_k, u_1\} \in E$. An undirected graph without cycles is a *forest* and a connected forest is a *tree*.

A directed graph $G = (V, E)$ is *strongly connected*, if there is a path between every pair of vertices. A *strongly connected component* of G is a maximal induced subgraph of G that is strongly connected.

Let $G = (V, E)$ be a connected graph. The *distance* $d_G(u, v)$ between two vertices $u, v \in V$ is the smallest integer d for which there is a path of length d between u and v. The set $N_G^d(v) := \{u \in V \mid d_G(u, v) = d\}$ is called the *d-hop neighborhood* of a vertex $v \in V$ and $N_G^d[v] := \cup_{i=1}^d N_G^i(v)$ is the set of vertices with distance at most d to v. If the graph G is obvious from the context, the notations $d(u, v)$, $N^d(v)$ and $N^d[v]$ are also used instead of $d_G(u, v)$, $N_G^d(v)$ and $N_G^d[v]$.

The *diameter* of a graph $G = (V, E)$, denoted by $\varnothing(G)$, is the largest positive integer such that there is a pair of vertices $u, v \in V$ with $d(u, v) = \varnothing(G)$.

Let $\overline{E} := (V \times V) \setminus E$ denote the *complementary edges* of a directed graph (V, E) and analogously for undirected graphs.

Unit Disk Graphs

A special class of graphs that can be embedded into the euclidean plane such that there is an edge between two vertices if and only if the euclidean distance between their positions is at most 1 is called unit disk graphs.

Let $\|x\|_2 := \sqrt{x_1^2 + x_2^2}$ denote the *euclidean norm* of $x = (x_1, x_2) \in \mathbb{R}^2$.

Definition 1. *An undirected graph $G = (V, E)$ is a* unit disk graph (UDG) *if there is a* UDG embedding $\rho : V \to \mathbb{R}^2$ *such that:*

1. $\forall u, v \in V$: $\quad (u \neq v) \quad \Rightarrow \quad (\rho(u) \neq \rho(v))$
2. $\forall u, v \in V$: $\quad (u \neq v) \quad \Rightarrow \quad (\quad \{u, v\} \in E \Leftrightarrow \|\rho(u) - \rho(v)\|_2 \leq 1 \quad)$

Obviously every UDG embedding ρ for a UDG $G = (V, E)$ can be scaled to construct an embedding ρ_r for an arbitrary radius $r \in \mathbb{R}, r > 0$, such that $\forall v, u \in V \qquad \|\rho(u) - \rho(v)\|_2 \leq 1 \quad \Leftrightarrow \quad \|\rho_r(u) - \rho_r(v)\|_2 \leq r$.

There are undirected graphs that are not unit disk graphs, the smallest one with respect to the number of vertices being the complete bipartite graph $K_{2,3}$ as mentioned in [31]. This particular graph not having a unit disk graph embedding will be an important observation for an algorithm developed in a later chapter, therefore this section presents a simple proof for this statement.

Definition 2. *For a finite set of points $P \subset \mathbb{R}^2$ and a positive radius $r \in \mathbb{R}$ define the corresponding unit disk graph $UDG(P) := (P, E_P)$, where* $E_P := \{\{x, y\} \mid x, y \in P \quad \wedge \quad x \neq y \quad \wedge \quad \|x - y\|_2 \leq r\}$.

Definition 3. *For $m \in \mathbb{R}^2$ define the* unit disk with center m *as* $B(m) := \{x \in \mathbb{R}^2 \mid \|x - m\|_2 \leq 1\}$.

Lemma 1. *The complete bipartite graph $K_{2,3} := (V, E)$, $V := \{v_1, v_2, u_1, u_2, u_3\}$, $E := \{\{v_i, u_j\} \mid 1 \leq i \leq 2 \;\wedge\; 1 \leq j \leq 3\}$ is not a unit disk graph.*

Proof. Assume that there is a UDG embedding $\rho : V \to \mathbb{R}^2$ for the graph $K_{2,3}$. Then $\{v_1, v_2\} \notin E$ implies that $\|\rho(v_1) - \rho(v_2)\|_2 > 1$. Without loss of generality let $\rho(v_1) := (0, 0)$ and $\rho(v_2) := (1 + \epsilon, 0)$ for some $\epsilon > 0$.

Since $\{u_i, v_j\} \in E$ for all $1 \leq i \leq 3$ and $1 \leq j \leq 2$, it follows that, for all $u \in \{u_1, u_2, u_3\}$, $\rho(u) \in L_\epsilon := B((0, 0)) \cap B((1 + \epsilon, 0))$. Furthermore $\|\rho(u_i) - \rho(u_j)\|_2 > 1$ for all $i \neq j$, because $\{u_i, u_j\} \notin E$. But $L_\epsilon \subset L_0$ for all $\epsilon > 0$ and there are no three points $p_1, p_2, p_3 \in L_0$ such that $\|p_i - p_j\|_2 > 1$ for all $i \neq j$: Let $L_- := \{(x, y) \in L_0 \mid y < 0\}$ and $L_+ := \{(x, y) \in L_0 \mid y \geq 0\}$, see Figure 1. It obviously holds that, for all $p \in L_-$, $L_- \subset B(p)$ and analogously $L_+ \subset B(p)$ for all $p \in L_+$. Therefore $L_0 \setminus (B(p_1) \cup B(p_2)) = (L_+ \cup L_-) \setminus (B(p_1) \cup B(p_2)) = \emptyset$ for all $p_1 \in L_+$ and $p_2 \in L_-$, which means that there is no third point $p_3 \in L_0$ that has euclidean distance greater 1 to both p_1 and p_2. $\square$

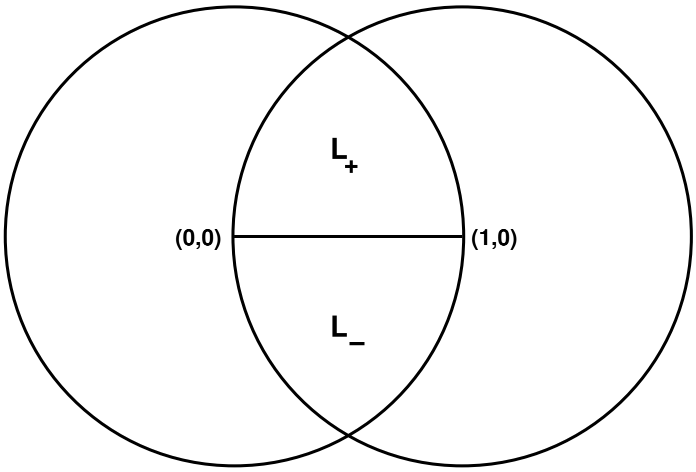

Fig. 1. Intersection of two unit disks with centers $(0, 0)$ and $(1, 0)$

Corollary 1. *If an undirected graph $G = (V, E)$ contains a set of vertices $U \subseteq V$ such that $G|_U$ is isomorphic to $K_{2,3}$, then G is not a unit disk graph.*

Proof. Any UDG embedding $\rho : V \to \mathbb{R}^2$ for G would yield a UDG embedding $\rho|_U$ for $G|_U = K_{2,3}$, which contradicts Lemma 1. □

An extension of the unit disk graph model, the d-quasi unit disk graph for a real number $0 \leq d \leq 1$, has been introduced in [63] to provide a less idealized graph model.

Definition 4. *Let $G = (V, E)$ be an undirected graph and $d \in \mathbb{R}$ be a real number, $0 \leq d \leq 1$. G is a d-*quasi unit disk graph*, if there is an embedding $\rho : V \to \mathbb{R}^2$ such that*

1. $\forall u, v \in V, u \neq v: \quad \|\rho(u) - \rho(v)\|_2 \leq d \quad \Rightarrow \quad \{u, v\} \in E$
2. $\forall u, v \in V, u \neq v: \quad \|\rho(u) - \rho(v)\|_2 > 1 \quad \Rightarrow \quad \{u, v\} \notin E$

An extension of the basic concept used in Lemma 1 suggests the following relation between d-quasi unit disk graphs and the complete bipartite graph $K_{2,4}$, which is not formally proven at this point and therefore formulated as a conjecture.

Conjecture 1. The complete bipartite graph $K_{2,4} := (V, E)$, $V := \{v_1, v_2, u_1, u_2, u_3, u_4\}$, $E := \{\{v_i, u_j\} \mid 1 \leq i \leq 2 \ \wedge \ 1 \leq j \leq 4\}$ is not a d-quasi unit disk graph for any $d \geq \sqrt{3} - 1$.

1.2 Complexity Theory

This section contains some NP-complete problems that are used in the following chapters to prove the NP-completeness of several problems regarding connectivity in graphs.

For the basic definitions and concepts of complexity theory, especially the theory of Turing machines, decision problems and NP-completeness, the reader is referred to suitable literature such as [35].

Satisfiability

Let $X = \{x_1, \ldots, x_n\}$ be a set of *boolean variables*. A *truth assignment* for X is a function $T : X \to \{0, 1\}$. If $T(x_i) = 1$, then variable x_i is *true* under T; if $T(x_i) = 0$, then variable x_i is *false* under T. For every variable $x_i \in X$ there are two *literals*, the *positive literal* x_i and the *negative literal* $\overline{x_i}$. The positive literal x_i is true under T if and only if variable x_i is true under T; negative literal $\overline{x_i}$ is true under T if and only if variable x_i is false under T.

A *clause* c over X is a set of literals over X. It represents the disjunction of literals which is *satisfied by a truth assignment* T if and only if at least one of the literals in c is true under T. A set $\mathcal{C}$ of clauses over X is *satisfiable* if and only if there is a truth assignment T that satisfies all clauses of $\mathcal{C}$.

Satisfiability	
Given:	A set of variables X and a set of clauses $\mathcal{C}$ over X.
Question:	Is there a satisfying truth assignment for $\mathcal{C}$?

Theorem 2. Satisfiability *is NP-complete. [24]*

Connected Sat

A restriction to the Satisfiability problem, so-called Connected Sat, is now introduced as it is closely related to graph connectivity.

Definition 5. *Let X be a set of boolean variables and $\mathcal{C}$ be a set of clauses over X. The* clause variable graph $G_{(X,\mathcal{C})}$ *consists of a vertex x for every variable $x \in X$ and a vertex c for every clause $c \in \mathcal{C}$. There is an edge $\{x, c\}$ between* variable vertex x *and* clause vertex c *if and only if c contains literal x or $\overline{x}$.*

Connected Sat	
Given:	A set of variables X and a set of clauses $\mathcal{C}$ over X, such that the clause variable graph $G_{(X,\mathcal{C})}$ is connected.
Question:	Is there a satisfying truth assignment for $\mathcal{C}$?

Lemma 2. Connected Sat *is NP-complete.*

Proof. Obviously Connected Sat is in NP. The NP-hardness is shown by the following polynomial time reduction from Satisfiability that transforms an instance $(X, \mathcal{C})$ into another, logically equivalent, instance $(X', \mathcal{C}')$ by successively adding new variables and clauses to merge connected components of the clause variable graph with each other.

Let G_1 and G_2 be two connected components in $G_{(X,\mathcal{C})}$ and c_1 (c_2) an arbitrary clause vertex in G_1 (G_2). Add a new variable h and replace c_1 (c_2) with the two clauses $c_1 \cup \{h\}$ and $c_1 \cup \{\overline{h}\}$ ($c_2 \cup \{h\}$ and $c_2 \cup \{\overline{h}\}$). Applying this replacement iteratively until the clause variable graph in connected yields an instance $(X', \mathcal{C}')$ that is satisfiable if and only if $(X, \mathcal{C})$ is satisfiable: If $(X, \mathcal{C})$ is satisfiable, then $(X', \mathcal{C}')$ is satisfiable, because every clause in $\mathcal{C}'$ is a superset of a clause in $\mathcal{C}$. On the other hand, if $(X', \mathcal{C}')$ is satisfiable, then $(X, \mathcal{C})$ is also satisfiable, because for every truth assignment T' of X' and every clause $c \in \mathcal{C}$ there is a clause $c' \in \mathcal{C}'$ such that $c \subseteq c'$ and none of the literals in $c' \setminus c$ is true under T'. □

Directed Hamiltonian Path

Definition 6. *Let $G = (V, E)$ be a directed graph and $n := |V|$. A path $p := v_1, \dots, v_n$ of length $n - 1$ is called a* Hamiltonian path*, if all vertices on p are pairwise distinct, i.e. $v_i \neq v_j$ for all $i, j \in \{1, \dots, n\}, i \neq j$.*

Directed Hamiltonian Path (DHP)

Given:	A directed graph $G = (V, E)$
Question:	Does G contain a Hamiltonian path?

Theorem 3. Directed Hamiltonian Path *is NP-complete. [35]*

1.3 Wireless Sensor Networks

A *wireless sensor network (WSN)* consists of a large number of small devices that are deployed across a geographic area to monitor certain aspects of the environment. Typical examples include monitoring of temperature or radiation for disaster detection and warning as well as humidity for agricultural purposes and (ultra-)sonic vibrations or radio waves for object tracking and surveillance. These small devices, the so-called *sensor nodes*, are able to communicate with each other through a wireless communication channel. As a result of their size the resources of the sensor nodes are strongly limited in terms of computational power, storage space and available energy. This also leads to a very limited range of the radio transmitters which therefore only allow direct communication between sensor nodes that are positioned sufficiently close to each other.

The communication structure of a WSN is typically modeled as either an undirected or a directed graph, in which the vertices represent the sensor nodes and the edges correspond to viable radio links. From a technical point of view, modeling a WSN as a directed graph is closer to reality, because physical radio links are not necessarily symmetric [15, 61]. However, most researchers use undirected graphs instead, which is well justified by the observation that low-level communication protocols for wireless transmissions currently in use, such as IEEE 802.11 [49] or Bluetooth, require symmetry for reliable data transmission via acknowledgements.

For the remainder of this thesis a sensor network is modeled as an undirected graph $G = (V, E)$, unless otherwise noted. The terms *(sensor) node* and *vertex* are used interchangeably with *node* referring to a physical instance of a sensor node, while *vertex* refers to the mathematical counterpart within the graph model. In conformity with common practice it is also assumed that the nodes have unique identifiers, modeled as a bijection $id : V \to \{1, \ldots, |V|\}$.

All communication and data transmission between non-adjacent nodes has to be carried out by *multi-hop routing*, a technique that requires multiple sensor nodes to work together in order to successfully transmit messages through the network. To achieve this goal the intermediate nodes on a *routing path* relay the message to reach further nodes that are outside the communication range of the node the message originated from. The question of which node should forward a message such that the destination nodes are eventually reached is subject to a *routing protocol*.

The general area of routing tasks is subdivided into several more specific problems:

- *Unicast* describes the task of transmitting a message from a single source $s \in V$ to a single target $t \in V$.
- *Multicast* requires that a message from a single source $s \in V$ is sent to all target nodes in a set $T \subseteq V$, which separates it from
- *Anycast*, where it is sufficient to deliver a message from a single source $s \in V$ to one arbitrary node $t \in T$ for a given set $T \subseteq V$.
- *Broadcast* is a special case of multicast with $T = V$, meaning that a message is distributed throughout the entire network. The inverse task of broadcast is also known as
- *Convergecast*, where messages from all nodes $v \in V$ are collected at a single target $t \in V$.

Every routing protocol that fulfills one of these tasks can also be used for all other tasks, if executed the correct number of times with the correct arguments. However, this can obviously generate a tremendous amount of overhead in comparison to specialized protocols and therefore research has been conducted on all of these problems.

The next sections present an overview of basic routing strategies for the broadcast, multicast and unicast problems.

Broadcast

The easiest way to distribute a message across a WSN is the so-called flooding protocol, which will be used to derive a specialized multicast algorithm in section 2.3.

The flooding protocol, initiated by a node $s \in V$, distributes a message M to all nodes of the WSN by forwarding M to all neighbors of s, which then also forward M to all of there neighbors and so forth. To ensure that this algorithm eventually terminates, it is necessary to introduce some kind of exit condition. For example, the flooding will obviously terminate if every node v forwards M at most once, which can be implemented either by keeping state at each node for every message or by managing a list of nodes in the message itself. Both approaches have the disadvantage of linearly increasing storage space requirement for an increasing number of messages or network nodes, respectively.

A simple solution to this problem is the introduction of a transmission counter i_t, carried in the message header, that keeps track of how many times this particular copy of the message has been forwarded. Utilizing this information the forwarding of a message can be stopped as soon as i_t exceeds the diameter of the network $\varnothing(G)$. Naturally, this requires the knowledge of $\varnothing(G)$ (or an upper bound for $\varnothing(G)$, such as $|V|$) at every node, but it reduces the required storage space to $log(\varnothing(G))$ (or $log(|V|)$, respectively). This technique can also be used to restrict the message dissemination to nodes with distance at most d to the node that initiated the execution by not forwarding any messages with $i_t \geq d$. This restriction is called *limited range flooding*.

However, there is another problem caused by this protocol, the so-called *Broadcast Storm Problem*, which describes the effects of massively redundant transmissions generated by flooding a WSN or *Mobile Ad-Hoc Network (MANET)* such as network contention and wireless medium access collisions [74]. This problem has been studied intensively and there are several proposals for more sophisticated flooding protocols. [4, 74, 84, 87]

Multicast

The general multicast problem in wireless ad hoc networks, i.e. the task of distributing message to a known set of nodes T, has received tremendous attention in the scientific community and several proposals for protocols have been made. Therefore the reader is referred to [5, 56] for an overview on this topic. The following chapter is dedicated to a special type of the multicast routing, where the main difference to this general multicast task is given by the fact that none of the nodes in the network is aware of the set of target nodes T, because T is defined by the neighborhood of a given node that does not participate in any of the implemented network protocols.

Unicast

The ability to transmit messages from a single source node to a single target is one of the basic problems that occur in WSNs and although this task can also be accomplished by the flooding algorithm above, it is not desirable to distribute every message across the entire network due to the tremendous transmission overhead. Therefore many researchers considered more efficient solutions for this routing problem and there have been countless proposals for multi-hop routing protocols in dynamic ad-hoc wireless networks and subsequent optimizations thereof, for example [7, 10, 14, 28, 36, 30, 32, 53, 54, 62, 65, 78, 81, 90], as well as competitive theoretical studies and experimental evaluations such as [2, 39, 57, 69, 70, 79, 93, 96].

The proposed routing protocols base their forwarding decisions on information about the network topology that is acquired either *proactively* during an initialization phase of the network or *reactively* based on route discovery mechanisms that are initiated once a payload is supposed to be delivered. Almost all of the protocols use one of the following techniques to store information about routing paths within the network:

1. The entire routing path is explicitly contained in the message header.
2. Each node in the network maintains a routing table, which is used to determine the next hop of a message.
3. Some sort of virtual address is assigned to each node v, spread across its neighborhood $N(v)$ and the decision about the next hop for a message is based on the addresses of available neighbors.

Many protocols use the third option such as *Compass Routing* [62], *Beacon Vector Routing (BVR)* [32], *GLIDER* [30], *ABVCap* [90] or *HBR* [36] based on different strategies to generate virtual addresses. However, several of these protocols use the lengths of shortest paths to a specified subset of vertices to compute these addresses, which leads to a problem commonly known as Metric Dimension. The goal is to determine a minimum set of nodes that yields a unique virtual addressing, if these addresses are built as tuples of distances to the selected nodes.

Definition 7. *An undirected graph $G = (V, E)$ has metric dimension at most k, if there is a vertex set $U \subseteq V$ such that $|U| \leq k$ and $\forall u, v \in V$, $u \neq v$, there is a vertex $w \in U$ such that $d_G(w, u) \neq d_G(w, v)$. The metric dimension of G is the smallest integer k such that G has metric dimension at most k.*

The metric dimension was independently introduced by Harary and Melter [38] and Slater [88].

METRIC DIMENSION	
Given:	An undirected graph $G = (V, E)$ and a positive integer $k \in \mathbb{N}$
Question:	Does G have metric dimension at most k?

Determining the metric dimension of a graph is a problem that also has an impact on other research fields such as chemistry [17], robotics [59] and combinatorial optimization [85]. METRIC DIMENSION is known to be NP-complete for general graphs [35], planar graphs [26], and even Gabriel unit disk graphs [46].

There are several algorithms for computing the metric dimension in polynomial time for special classes of graphs, as for example for trees [17, 59], wheels [41, 86], grid graphs [72], k-regular bipartite graphs [83], amalgamation of cycles [50], outerplanar graphs [26] and cactus block graphs [42]. The approximability of the metric dimension has been studied for bounded degree, dense, and general graphs in [40]. Upper and lower bounds on the metric dimension are considered in [16, 19] for further classes of graphs. Recently we have developed an algorithm based on dynamic programming to efficiently compute the metric dimension of special graph classes by decomposition techniques. [44]

2 Neighborhood Broadcast

A task that arises naturally in WSNs is the following special case of the multicast problem, which is called NEIGHBORHOOD BROADCAST:

> NEIGHBORHOOD BROADCAST
>
> *Given:* A WSN with undirected communication graph $G = (V, E)$ and two nodes $v \in V$ and $s \in N_G^1(v)$.
> *Task:* Distribute a message originating at node s to all nodes in $N_G^1(v)$ under the assumption that v does not obey any of the implemented protocols.

Although this problem implicitly occurs in several areas of ad-hoc networking, it has, to the best of my knowledge, never been explicitly considered in literature.

In [91] the authors focus on building data aggregation trees that span the neighborhood of a sensor node v. To discover and initially contact all neighbors of v they implicitly try to perform a NEIGHBORHOOD BROADCAST by briefly describing the usage of limited range flooding. However, sending a message with the technique they mention is only guaranteed to reach all neighbors of v, if the flooding range is high enough to flood the entire network - even in cases where it is not necessary to do so.

The NEIGHBORHOOD BROADCAST problem generally occurs during collaborative fault detection [91], when the neighbors of a sensor node v want to exchange their observations in order to decide whether v should be excluded from the network due to faults or misbehavior. It is also required during the exclusion of a misbehaving or faulty sensor node v from the network, because every neighbor of v has to be instructed not to communicate with v anymore and it can be applied, for example, to the detection algorithm presented in [23].

A practical NEIGHBORHOOD BROADCAST algorithm also has applications in reactively repairing unicast routing paths or multicast routing structures after node failures: Regardless of the routing protocol in use, the collected information about a specific routing path might be outdated due to node failures as soon as the actual payload reaches a particular interior node of the computed routing path. In this case the initially discovered routing path is no longer available. And while it is possible to reinitialize the network or perform a new route discovery in this case, these approaches are usually expensive in terms of energy consumption and introduce considerable time delay until the message reaches its destination. Therefore it would be advantageous to have a simple repair mechanism for these cases that allows messages to reach their destination without reinitialization of the network or additional route discovery queries. Assuming that there is sufficiently redundant routing information to identify the successor of a failed node on the original routing path a message was supposed to take, this route repair mechanism has to perform a NEIGHBORHOOD BROADCAST in order to guarantee that the message reaches its successor, who can then resume normal routing operation.

For the most commonly used techniques of routing protocols the required redundancy is either automatically fulfilled or at least trivially achievable:

- For routing protocols that store the computed routing path in the message header, for example *Dynamic Source Routing (DSR)* [54], the message itself contains enough information and there is no need for additional data replication.
- For all routing protocols that base their decisions on precomputed *routing tables*, for example *Ad-Hoc On-demand Distance Vector Routing (AODV)* [78] or *Hierarchical Bipartition Routing (HBR)* [36], it is sufficient that each node v replicates its routing table within the 1-hop neighborhood $N^1(v)$.
- Routing protocols that base their forwarding decisions on some sort of (virtual) addressing of neighboring nodes, such as *Compass Routing* [62], *Beacon Vector Routing (BVR)* [32], *GLIDER* [30] or *ABVCap* [90], the desired property can be achieved by collecting all addresses from the 2-hop neighborhood $N^2(v)$ at every node v.

Known protocols for the general multicast problem are theoretically suitable for the NEIGHBORHOOD BROADCAST problem. However, they would require the initiating node to know the neighborhood in which the message is supposed to be distributed and this requires constant maintenance, especially when the network topology is subjected to frequent changes. And of course the mentioned flooding protocols are also applicable for this task, but introduce a considerable transmission overhead.

Whether it is possible to perform a NEIGHBORHOOD BROADCAST obviously depends on the structure of the communication graph and especially on the neighborhood of node v: If v is a separation vertex of the communication graph, it is impossible to distribute a message across $N^1(v)$ without the help of v itself.

But even if v is not a separation vertex, it might be necessary to broadcast a message M across the entire network in order to distribute it to all neighbors

$N^1(v)$: Consider the cycle C_n with n vertices as a network topology:

$$C_n \quad := \quad (\{v_1, \ldots, v_n\}, \{\{v_i, v_{i+1} \mid 1 \leq i < n\}\} \cup \{v_n, v_1\})$$

If any one of the nodes $v_1, \ldots, v_n$ does not obey implemented network protocols, for example v_i, then the only remaining path between both neighbors in $N^1(v_i) = \{v_{i-1}, v_{i+1}\}$ traverses all other nodes and therefore every multicast algorithm has to distribute M to all nodes. However, theoretical worst cases like this are rather unlikely in a real sensor network and therefore a specialized protocol with a better performance than flooding the entire network is desirable for this scenario and will be developed in the remainder of this chapter.

Since it is assumed that the node v does not participate in any protocol, the subgraph induced by the neighborhood of v, without v itself, is relevant to the design of a NEIGHBORHOOD BROADCAST protocol. This relates to the graph theoretic concept of local connectivity, that was originally introduced by Chartrand and Pippert in [18] and will be extended in the next section.

2.1 Local Connectivity

Definition 8. *For an undirected graph $G = (V, E)$ and positive integers $k, d \in \mathbb{N}$ a vertex $v \in V$ is called* d-locally k-connected *in G, if $G|_{N^d[v]}$ is k-connected. A subset $U \subseteq V$ of the vertices is called a* d-locally k-connected vertex set*, if every vertex in $G|_U$ is d-locally k-connected. The graph G is d-locally k-connected, if V is a d-locally k-connected vertex set.*

Note that the parameters d and k may be omitted, if they are equal to 1. For example 1-locally 1-connected can also be noted as locally connected, 1-locally 2-connected as locally 2-connected and so on.

Definition 9. *In an undirected graph $H = (V, E)$ a* Local Connectivity Component (LCC) *is a pair (v, C) consisting of a vertex $v \in V$ and a subset of vertices $C \subseteq N_H(v)$ such that $H|_C$ is a connected component of $H|_{N_H(v)}$. Let $LCC(H)$ denote the set of all local connectivity components of H.*

This definition is now extended further such that the neighborhoods depend only on a subset of the given edges. Also see Figure 2 for an example.

Definition 10. *For an undirected graph $H = (V, E)$ and a subgraph $G = (V, E')$ of H a* Local Connectivity Component (LCC) of H with respect to G *is a pair (v, C) consisting of a vertex $v \in V$ and a subset of vertices $C \subseteq N_G(v)$ such that $H|_C$ is a connected component of $H|_{N_G(v)}$. Let $LCC_G(H)$ denote the set of all local connectivity components of H with respect to G.*

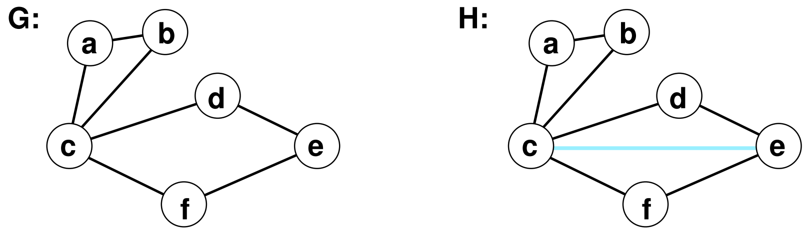

Fig. 2. The set $LCC_G(G)$ consists of the local connectivity components $(a, \{b, c\})$, $(b, \{a, c\})$, $(c, \{a, b\})$, $(c, \{d\})$, $(c, \{f\})$, $(d, \{e\})$, $(d, \{c\})$, $(f, \{e\})$, $(f, \{c\})$, $(e, \{d\})$ and $(e, \{f\})$, while the set $LCC_G(H)$ consists of $(a, \{b, c\})$, $(b, \{a, c\})$, $(c, \{a, b\})$, $(c, \{d\})$, $(c, \{f\})$, $(d, \{c, e\})$, $(f, \{c, e\})$, $(e, \{d\})$ and $(e, \{f\})$.

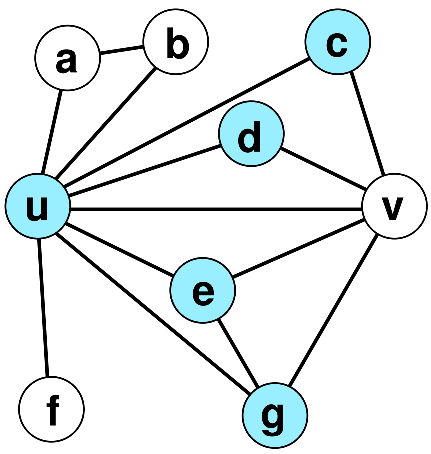

Fig. 3. Graph $G = (V, E)$: Vertex v is locally connected, because $N_G(v)$ (blue vertices) induces a connected subgraph of G due to the fact that every vertex in $N_G(v)$ is also adjacent to vertex u (Lemma 3). The vertices e and g are true twins (Definition 11).

The local connectivity of graphs is analyzed several times throughout the thesis and the following definition, lemma and corollary provide powerful tools for this purpose.

Definition 11. *Let $G = (V, E)$ be an undirected graph and $u, v \in V$, $u \neq v$. The vertices u and v are called* true twins, *if*

1. $\{u, v\} \in E$ *and*
2. $N^1(u) \setminus \{v\} = N^1(v) \setminus \{u\}$.

See Figure 3 for an example.

Lemma 3. *Let $G = (V, E)$ be an undirected graph and $u, v \in V$, $\{u, v\} \in E$. If $N^1_G(v) \setminus \{u\} \subset N^1_G(u)$, then vertex v is locally connected.*

Proof. Let $v_1, v_2 \in N^1_G(v)$ be two vertices adjacent to v. If either v_1 or v_2 is the vertex u, then there is a path between v_1 and v_2 in $G|_{N^1_G(v)}$, because $\{v_1, v_2\} \in E$ due to the fact that $\{v_1, v_2\} \setminus \{u\} \subset N^1_G(u)$. Otherwise v_1, u, v_2 is a path between v_1 and v_2 in $G|_{N^1_G(v)}$, because $\{v_1, v_2\} \subset N^1_G(u)$. Also see Figure 3 for an example. □

Corollary 2. *Let $G = (V, E)$ be an undirected graph and $u, v \in V$ true twins. Then both vertices u and v are locally connected in G.*

The possibility to perform a Neighborhood Broadcast depends on the smallest integer d for which the vertex v is d-locally connected.

Definition 12. *Let $G = (V, E)$ be an undirected graph and $v \in V$ a vertex. The smallest positive integer $d_{min} \in \mathbb{N}$, such that v is d_{min}-locally connected in G, is called the* local connectivity distance *of v in G. If none such integer exists, the local connectivity distance is said to be ∞.*

Observation 1 *The local connectivity distance of a vertex v in an undirected graph G is ∞ if and only if v is a separation vertex.*

2.2 Computing the Local Connectivity Distance

This section investigates how to algorithmically determine the local connectivity distance of a vertex v in a given undirected graph $G = (V, E)$. Formally, the following construction problem is considered:

> Local Connectivity Distance (LCD)
>
> *Given:* An undirected graph $G = (V, E)$ and a vertex $v \in V$.
> *Task:* Compute the local connectivity distance of v in G.

The LCD problem can be solved in polynomial time by utilizing standard methods from graph theory as follows: For a positive integer $i \in \mathbb{N}$ the neighborhood $N^i[v] = \bigcup_{j=1}^{i} N^j(v)$ can be determined in time $\mathcal{O}(|V| + |E|)$ by a breadth first search, started at vertex v, that keeps track of the distance $d(v, u)$ for every vertex $u \in V$. Afterwards the induced subgraph $G|_{N^i[v]}$ can be constructed in time $\mathcal{O}(|V| + |E|)$ and tested for connectivity using, for example, another breadth first search.

The local connectivity distance can be determined by performing this test consecutively for all $i \in \{1, \ldots, |V|\}$ in ascending order until the first connectivity test yields a positive result. The overall running time of this straightforward solution is obviously bounded by $\mathcal{O}(|V| \cdot (|V| + |E|)) = \mathcal{O}(|V|^2 + |V| \cdot |E|)$.

The remainder of this section is dedicated to a more efficient solution for the LCD problem that additionally provides the possibility for a distributed implementation in a WSN. The algorithmic idea is based on the following Theorem that is formulated using the terminology introduced in the next definition.

Definition 13. *Let $G = (V, E)$ be an undirected graph and $s \in V$ a vertex. A* shortest path tree for G at root s *is a tree $T = (V, E_T)$, $E_T \subseteq E$ such that $d_G(s, v) = d_T(s, v)$ for all vertices $v \in V$.*

A tree $T' = (V', E')$ of forest $T|_{V \setminus \{s\}}$ is called a branch *of T. The* root of branch *T' is the (uniquely determined) vertex $v' \in N^1_G(s)$ that belongs to T', i.e. $v' \in V'$. For all vertices $v \in V \setminus \{s\}$ let $r(v)$ denote the root of the branch of T that contains v.*

Furthermore, an edge $e = \{u, v\} \in E$ is called

- tree edge with respect to T, *if $e \in E_T$,*
- bridge with respect to T, *if u and v belong to different branches of T and*
- shortcut with respect to T, *if u and v belong to the same branch of T and $e \notin E_T$.*

Also see Figure 4 for an example of the notations introduced in this definition.

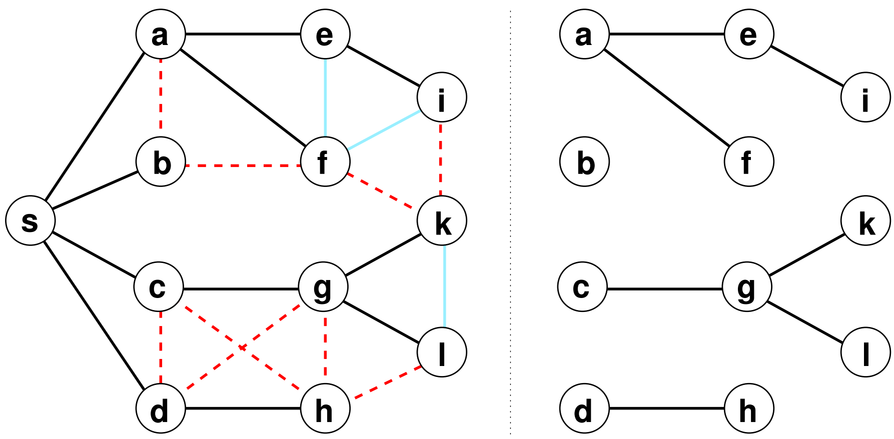

Fig. 4. Example for Definition 13: **Left**: Graph $G = (V, E)$ with a shortest path tree $T = (V, E_T)$ at root s (black edges). The dashed red edges are bridges with respect to T, the light blue edges are shortcuts with respect to T. **Right**: Forest $T|_{V \setminus \{s\}}$ that defines four branches of T, induced by the vertex sets $\{a, e, f, i\}, \{b\}, \{c, g, k, l\}$ and $\{d, h\}$. The root $r(e)$ of e is vertex a, the root $r(b)$ of b is vertex b, the root $r(k)$ of k is vertex c etc.

Theorem 4. *Let $G = (V, E)$ be an undirected graph, $s \in V$ a vertex and T a shortest path tree for G at root s. Furthermore let $d \in \mathbb{N}$ be a positive integer and $B \subset E$ the set of bridges with respect to T.*

Every edge $e \in B$ is given a positive integer weight representing the maximum distance from s to one of the vertices incident to e. Formally, define a mapping $f : B \to \mathbb{N}$ by

$$f(\{u, v\}) \quad := \quad max\{d_T(s, u), d_T(s, v)\}.$$

Additionally, define a set of edges E_1 between neighbors of s with positive integer weights $w : E_1 \to \mathbb{N}$ based on the bridges B and their weights f as follows:

$$\begin{aligned} E_1 \quad &:= \quad \{\{r(u), r(v)\} \mid \{u, v\} \in B\} \\ w(\{u, v\}) \quad &:= \quad min\{f(\{u', v'\}) \mid \{u', v'\} \in B \wedge r(u') = u \wedge r(v') = v\} \end{aligned}$$

Then vertex s is d-locally connected, if there is a subset $E' \subseteq E_1$ such that

1. *the graph $(N_G^1(s), E')$ is connected and*
2. *$\forall e \in E' : \quad w(e) \leq d$.*

Proof. Let $E' \subseteq E_1$ be a set of edges that satisfies the properties 1. and 2. above and $w, w' \in N_G^1(s)$, $w \neq w'$ two arbitrary neighbors of s. Then we can construct a path $w_1, \ldots, w_k$ in G for some positive integer $k \in \mathbb{N}$ such that $w_1 = w$, $w_k = w'$ and $\forall i \in \{1, \ldots, k\} : d_G(s, w_i) \leq d$ as follows, meaning that s is d-locally connected in G:

Since the graph $(N_G^1(s), E')$ is connected, there is a path between w and w' in $(N_G^1(s), E')$, i.e. there is a sequence $\{v_1, v_1'\}, \ldots, \{v_l, v_l'\}$ of edges of E' for some positive integer $l \in \mathbb{N}$ such that $v_1 = w$, $v_l' = w'$ and $\forall i \in \{1, \ldots, l-1\} : v_i' = v_{i+1}$. By definition of E_1, which is a superset of E', in conjunction with property 2. this means that there also is a sequence of bridges $\{u_1, u_1'\}, \ldots, \{u_l, u_l'\}$ satisfying the following properties:

a) $\forall i \in \{1, \ldots, l\} : \{u_i, u_i'\} \in B$
b) $\forall i \in \{1, \ldots, l-1\} : r(u_i') = r(u_{i+1})$
c) $r(u_1) = w$
d) $r(u_l) = w'$
e) $\forall i \in \{1, \ldots, l\} : w(\{u_i, u_i'\}) \leq d$

Therefore it is sufficient to prove that, for every pair v, v' of two distinct vertices with $r(v) = r(v')$, there is a path p between v and v' in G such that every vertex u in p satisfies $d_G(s, u) \leq max\{d_G(s, v), d_G(s, v')\}$. Let p be the (unique) path between v and v' in $T|_{V\setminus\{s\}}$. Such a path exists, because v and v' belong to the same branch of T due to $r(v) = r(v')$. Additionally, every vertex u in p satisfies $d_G(s,u) \leq max\{d_G(s,v), d_G(s,v')\}$, because T is a shortest path tree for G at root s and of course p also is a path in G, because T is a subgraph of G. □

Based on Theorem 2.2 the LCD problem can be solved by computing the smallest positive integer $d \in \mathbb{N}$ for which there is a set of edges $E' \subseteq E_1$ such that the conditions 1 and 2 hold. It is now shown that this computation can be achieved by solving the following Minimum Spanning Tree problem on the graph $(N^1_G(s), E_1)$ with edge weights w as defined in Theorem 2.2.

Definition 14. *Let $G = (V, E)$ be an undirected graph with positive edge weights $w : E \to \mathbb{N}$. A subgraph (V, E') of G is a* minimum spanning tree for G*, if (V, E') is connected and*

$$\forall E'' \subseteq E \qquad \left((V, E'') \textit{ is connected} \quad \Rightarrow \quad \sum_{e \in E''} w(e) \geq \sum_{e \in E'} w(e) \right).$$

Minimum Spanning Tree (MST)
Given: An undirected graph $G = (V, E)$ with edge weights $w : E \to \mathbb{N}$
Task: Compute a minimum spanning tree for G.

Although the MST problem asks for an edge set in which the sum of all weights is minimal while maintaining connectivity, it can also be used to compute the maximum weight necessary for achieving connectivity, because the maximum weight of an edge in a minimum spanning tree is independent of the tree itself as shown in the following Lemma.

Lemma 4. *Let $G = (V, E)$ be an undirected graph with positive edge weights $w : E \to \mathbb{N}$ and $T_1 = (V, E')$, $T_2 = (V, E'')$ two minimum spanning trees for G. Then $max\{w(e) \mid e \in E'\} = max\{w(e) \mid e \in E''\}$.*

Proof. Let $m_1 := max\{w(e) \mid e \in E'\}$, $m_2 := max\{w(e) \mid e \in E''\}$ and let $e_1 \in E'$ be an edge with $w(e_1) = m_1$. Assume that $m_1 \neq m_2$ and, without loss of generality, let $m_1 > m_2$. Then the subgraph $(V, E' \setminus \{e_1\})$ consists of exactly two connected components induced by vertex sets V_1 and V_2. Obviously $V_1 \cup V_2 = V$ and since (V, E'') is connected there is an edge $\{v_1, v_2\} \in E''$ with $v_1 \in V_1$ and $v_2 \in V_2$. Additionally $w(\{v_1, v_2\}) \leq m_2 < m_1 = w(e_1)$

and therefore T_1 is not a minimum spanning tree, because $(V, (E' \setminus \{e_1\}) \cup \{\{v_1, v_2\}\})$ is connected and

$$\sum_{e \in E'} w(e) \quad > \quad w(\{v_1, v_2\}) - w(e_1) + \sum_{e \in E'} w(e).$$

□

According to Theorem 2.2 and Lemma 4 the LCD problem can be solved by the following algorithm:

1. Compute a shortest path tree T for G at root s.
2. Determine edge set E_1 and the weights w as defined in Theorem 2.2.
3. Solve the MST problem on the graph $(N^1_G(s), E_1)$ with weights w.
4. Return the maximum edge weight that occurs in the computed minimum spanning tree.

In a centralized algorithm these steps can be implemented as follows:

1. The shortest path tree T at root s as defined in Definition 13 (where the length of a path equals the number of edges) can be computed in time $\mathcal{O}(|V| + |E|)$ by running a breadth first search on vertex s. A simple extension of this breadth first search allows the simultaneous computation of the distance $d_G(s, v)$ and the root $r(v)$ for every vertex $v \in V$, which is saved at vertex v.
2. Using the information collected in the previous step, one iteration over the edge set E is sufficient to decide whether an edge $e \in E$ is a bridge and to compute the weight $f(e)$ for all of these bridges. Let L be a list of all bridges and for a vertex $w \in N^1_G(s)$ let $p(s)$ be the position of w in the adjacency list of s. Now assign to every bridge $\{u, v\}$ in list L a key $(p(r(u)), p(r(v)))$ with $p(r(u)) < p(r(v))$, which is then used to sort L in linear time with bucket sort. Afterwards the edge set E_1 and their weights w can be easily constructed by iterating the sorted list once more. Therefore the overall running time for this step is $\mathcal{O}(|V| + |E|)$.
3. The MST problem as defined above with integer edge weights can theoretically be solved in linear time using the trans-dichotomous minimum spanning tree algorithm presented in [34].
4. This step can obviously be done in linear time.

The discussion above establishes the following Theorem.

Theorem 5. *The* LCD *problem can be solved in linear time.*

However, the linear time algorithm for computing integer weight minimum spanning trees in [34] is purely theoretical and not applicable in practice. If, for example, Prim's algorithm [80] is used for step 3, then the overall running of the algorithm above is in

$$\mathcal{O}(|V| + |E| + \Delta(G) \cdot log(\Delta(G))).$$

Additional to the theoretically achievable linear time implementation based on global topology knowledge, this approach is suitable for a distributed implementation in a WSN: If a node r wants to determine the minimum distance d such that the d-hop neighborhood $N_G^d[r]$ is connected, the network needs to cooperate in order to determine the edge set E_1 and the weights w. Once this is done and the collected data has been transmitted to node r, the steps 3 and 4 can be solved locally by r itself.

The shortest path tree T at root r can be built by a modified flooding algorithm, started at node r: Every transmitted message M contains a hop-counter i that is incremented after each transmission and keeps track of the distance to r. Every node v receiving such a message M updates a parent pointer to identify its parent p in the tree and notifies p that v is a child of p, if i is lower than the currently saved distance to r. Additionally M also contains the neighbor $u \in N^1(r)$ that originally transmitted the message, allowing every node v to identify the root $r(v)$ of its own branch. Based on this information it is possible to compute all bridges $e \in B$ and their weights $f(e)$ after T has been built by having every node v exchange the collected distance and branch information with all neighbors in $N^1(v)$.

Afterwards T can be used as a data aggregation tree to transmit the collected information to r while simultaneously computing the minimum weight $w(e)$ for every edge $e \in E_1$: Starting at the leafs of T the nodes send a list of edges in E_1 that result from their incident bridges to their respective parent in T. Every node that is not a leaf in T waits until it received these lists from its children in T, merges all lists including its own one by computing the minimum weight for every edge that is contained in one of the lists and sends the resulting list to its parent.

To avoid flooding the entire network, it is possible to use limited range flooding with increasing ranges to successively build larger subgraphs of T until r discovers connectivity within the considered neighborhood.

2.3 The *d*-Hop Bouncing Flood Protocol

In this section a practical algorithm for the Neighborhood Broadcast problem is presented and analyzed.

General Idea

The *k-Hop Bouncing Flood (k-HBF)* protocol uses the idea of carrying a transmission counter i_t in the message header that restricts the number of retransmissions for each message: The forwarding is stopped as soon as i_t reaches the given parameter $k \in \mathbb{N}$. This disseminates the message to all nodes of distance at most k to the node s, who initiated the protocol. This range limitation of the flooding process reduces the number of nodes transmitting the message, but the protocol is no longer guaranteed to deliver the message to all targets, as discussed above. Utilizing only information already present at the individual sensor nodes, the success probability can be increased considerably by resetting i_t to 0 at every node that is adjacent to v, which leads to M "bouncing" along the neighbors of v, see Figure 5 for an example. The idea is to distribute M not only to all nodes of distance at most k to s, but to all nodes with distance at most k to any neighbor of v or, equivalently, to every node with distance at most $k + 1$ to v. To achieve this it is not sufficient to reset i_t at every neighbor of v:

For example the node r in Figure 5: Before w resets the transmission counter to 0 and sends the message to its neighbors, node r already received M with counter value 2 from s, but r is required to relay the message from w again in order to deliver it to the yellow nodes.

Unlike conventional flooding it is therefore necessary that nodes can relay messages multiple times and simply marking a message M as "already seen" by a particular node to prevent infinite transmission loops by not relaying M again based on this mark is not sufficient anymore. In the k-HBF protocol, each node r that received M keeps track of the minimum $i_{min,r}$ of all transmission counter values contained in copies of M that reached r. Node r then relays M again at a later time, if and only if the new transmission counter is strictly lower than $i_{min,r}$, because in this case it might reach additional nodes. This modification guarantees that every node within the k-hop neighborhood of a node that performed a transmission counter reset will eventually receive the message, while ensuring that the transmissions terminate due to the observation that every node can transmit a message at most k times, before the minimum transmission counter for this node reaches 0.

Distributed Implementation

One of the biggest challenges remaining for real world deployment of sensor networks is testing and debugging of implemented algorithms in a realistic environment, which introduces additional problems that are usually not considered during simulation. In recent history there have been several real world experiments that demonstrated this issue [20, 25, 64, 75] and there has also

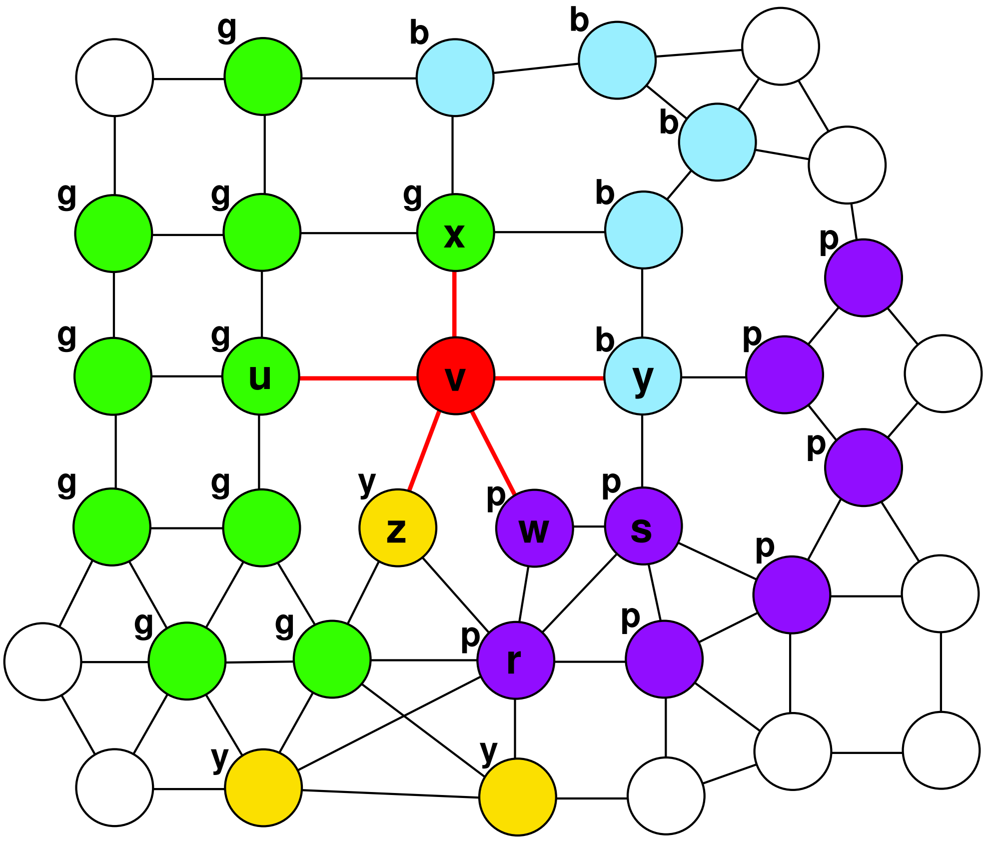

Fig. 5. 2-HBF example on $G = (V, E)$: Node u initiates the protocol for its neighbor v to distribute M to $N^1_G(v) = \{u, x, y, w, z\}$. The green nodes receive M due to their distance to u being at most 2 in $G|_{V \setminus \{v\}}$, the blue nodes receive M due to x resetting the transmission counter i_t, the magenta nodes due to the reset at y and the yellow nodes due to the reset at w.

been intensive work on testbed environments suitable to tackle this problem [51, 52, 68, 71, 95].

From this point of view, the d-HBF protocol offers the advantage of an easy and straightforward distributed implementation, which is demonstrated by the pseudo code given in Algorithm 1 for the main part of the protocol.

To guarantee termination this implementation uses the strategy of keeping the minimum transmission counter of a protocol invocation at each node, as discussed above.

Packets sent by the algorithm wrap around a given message M and add additional header fields required for the protocol itself in the format

Algorithm 1 k-HBF protocol: Distributed implementation at node u

```
m ← 1                                                    ▷ Counter for message ids
h ← HashMap                                              ▷ h : ID × NODE → ℕ
procedure SendPacket(m, s, v, k, i_t, M, T)
    Pass data to link layer:
    Send (m, s, v, k, i_t, M) to all nodes in T ⊆ N^1(u)
end procedure
procedure StartHBF(k, v, M)
    m ← m + 1
    SendPacket(m, u, v, k, 0, M, N^1(u))
end procedure
procedure ReceivePacket(m, s, v, k, i_t, M)
    i_{min,u} ← ∞
    if h contains (m, s) then
        i_{min,u} ← h((m, s))
    end if
    if i_t < i_{min,u} then
        if v ∈ N^1(u) then                               ▷ neighbor resets i_t
            SendPacket(m, s, v, k, 0, M, N^1(u))
            h((m, s)) ← −∞
            return
        end if
        if i_t < k then
            SendPacket(m, s, v, k, i_t + 1, M, N^1(u))
            h((m, s)) ← i_t
        end if
    end if
end procedure
```

(message id, initiator s, target v, k, i_t, M),
where the message id m is unique for each node s, meaning that (m, s) provides a unique message identifier throughout the network. The message header also contains the node v for which the protocol was initiated as well as the distance parameter k and the transmission counter i_t.

Several extensions of protocol arise from techniques commonly used in computer networks:

To ensure long-term stability of the network, the data structure used to organize the mapping between message identifiers and minimum transmission counters at each node should be implemented based on a soft-state approach in the sense that the entries are deleted after a reasonable amount of time to avoid memory leaks.

In the next section it is shown that k-HBF is guaranteed to deliver M to all neighbors of v, if $k \geq 2d-1$ and v is d-locally connected. Therefore the optimal choice of k depends on the local connectivity distance of v, which is usually unknown. If the application is able to decide whether a broadcast attempt was successful (which is possible via acknowledgements, for example, if $|N^1(v)|$ is known or the target is one particular neighbor of v as in the case of unicast route repair), the protocol can obviously be extended by successively increasing the distance parameter k as long as the previous attempt was unsuccessful. According to common practice one would double k after each failed attempt to achieve exponential growth of the search area, which compensates for a potentially fast changing network topology.

It is noteworthy that the k-HBF protocol can also be used to determine the local connectivity distance of a node v, if the network is able to decide whether an execution of the protocol was successful. While the distributed approach in section 2.2 is very efficient in terms of transmitted messages, it has to be performed proactively, because in order to determine the local connectivity distance of a node v the node itself has to participate in the computation. If this is not viable, it is also possible to determine the local connectivity distance of a node v without its participation by using the k-HBF protocol as follows:

An additional counter i_{max} in the transmitted messages can keep track of the maximum value that the counter i_t ever reached before it has been reset. If every neighbor $u \in N^1(v)$ computes the minimum value $k_{min}(u)$ of the i_{max} counters of all messages u received and transmits the result back to the initiating node s, then s is capable of determining the minimum value k_{min} (and therefore the local connectivity distance of v) such that the k_{min}-HBF protocol succeeds via

$$k_{min} \quad = \quad max\{k_{min}(u) \mid u \in N^1(v) \setminus \{s\}\}.$$

This extension can also be used to provide increased performance during future executions for the same node v by distributing the value k_{min} to all nodes in $N^1(v)$.

Theoretical Analysis

This section evaluates the k-HBF protocol in terms of message complexity and success guarantee. It is shown that the number of messages transmitted by an invocation of k-HBF in a WSN with communication graph G is at most $k \cdot \Delta(G)^{k+1}$. Furthermore a sufficient condition for the delivery guarantee is proven: k-HBF is guaranteed to deliver M to all neighbors of v, if $k \geq 2d-1$ and v is d-locally connected in G. Finally, it is also proven that the set of nodes participating in the k-HBF protocol is optimal with respect to all possible protocols for the Neighborhood Broadcast task that are restricted to the same topology information, i.e. every participating node also has to participate in every protocol that guarantees delivery.

Theorem 6. *Let $G = (V, E)$ be an undirected graph and $k \in \mathbb{N}$ a positive integer. The k-HBF protocol, initiated by a neighbor $s \in N^1(v)$ for a node $v \in V$ and a message M, transmits at most $k \cdot \Delta(G)^{k+1}$ messages.*

Proof. Every neighbor $u \in N^1(v)$ performs a limited range flooding of M with range k and therefore at most $\Delta(G)^k$ nodes receive the message due to the hop counter reset at node u. Furthermore v has at most $\Delta(G)$ neighbors, which means that at most $\Delta(G)^{k+1}$ nodes receive M during the execution of the k-HBF protocol. Finally, every node w can relay M at most k times, because afterwards the minimum transmission counter $i_{min,w}$ is either 0 or $-\infty$. In conjunction it follows that $k \cdot \Delta(G)^{k+1}$ is an upper bound for the number of transmitted messages.

The Theorem above demonstrates a theoretical worst case for number of messages transmitted the k-HBF protocol, assuming that every node in the considered part of the network actually has the maximum vertex degree $\Delta(G)$. Furthermore it assumes that the sets of nodes reached by each neighbor of v are pairwise disjoint, which is not possible, if v has more than one neighbor. Therefore the average number of transmitted messages in any real world application should be considerably lower than the presented upper bound. Of course the number of transmitted packets can be higher due to retransmissions, acknowledgements etc.

The following Lemma and the subsequent Theorem are used to establish the connection between the k-HBF protocol and the local connectivity distance of the node v it is executed for.

Lemma 5. *Let $G = (V, E)$ be an undirected graph and $d \in \mathbb{N}$ a positive integer. Let $v \in V$ be a vertex that is d-locally connected in G. Then, for every pair of vertices $s, t \in N^1(v)$, there is a path $p = w_1, \dots, w_k$ in $G|_{V\setminus\{v\}}$ between $w_1 = s$ and $w_k = t$ such that for all $i \in \{1, \dots, k - 2d + 1\}$ the path $w_i, w_{i+1}, \dots, w_{i+2d-1}$ of length $2d - 1$ contains at least one vertex of $N^1(v)$.*

Proof. Let $p' = u_0, \dots, u_m$ be a shortest path between $u_0 = s$ and $u_m = t$ in $H := G|_{N^d[v]}$. Since $v \in V$ is d-locally connected such a path exists and we also know that for every vertex u on p' the distance $d_G(u, v)$ is at most d.

Path p' will now be altered successively to fulfill the required condition as follows: Let l be the minimum index such that the path $u_l, \dots, u_{l+2d-1}$ does not contain a vertex adjacent to v. Now replace $u_l, \dots, u_{l+2d-1}$ with a path $u_l, \dots, s', \dots, u_{l+2d-1}$ for a vertex $s' \in N^1(v)$ such that both paths $u_l, \dots, s'$ and $s', \dots, u_{l+2d-1}$ have length at most $2d - 1$ as follows. This proves the Lemma, because during each iteration the length of a consecutive subsequence of the current path that violates the required condition is strictly shortened.

Let $d_l, \ldots, d_{l+2d-1}$ be the sequence D of distances to vertex v as defined by $d_i := d_G(v, u_i)$ for $i \in \{l, \ldots, l+2d-1\}$. Furthermore let $j \in \{l+1, \ldots, l+2d-1\}$ be the minimum index such that d_j is the second occurrence of this particular distance in D, meaning that there is an index $j' < j$ with $d_{j'} = d_j$. Since all distances d_i are between 2 and d by definition of p' it follows that $j \leq l + d - 1$.

Now let $v, s', q_2, \ldots, q_{d_j-1}, u_j$ be a shortest path between v and u_j in G. Note that p' does not contain vertex s', because p' contains a shortest path p'' between u_{l-1} and u_j such that p'' contains vertex $u_{j'}$. Therefore $d_H(s', u_j) = d_j - 1 < d_H(u_{l-1}, u_j)$. Then it holds that the length $L(q)$ of the path

$$q = u_l, \ldots, u_j, q_{d_j-1}, \ldots, q_2, s'$$

is at most $2d - 2$, because

$$\begin{aligned} L(q) &= L(u_l, \ldots, u_j) + L(u_j, q_{d_j-1}, \ldots, q_2, s') \\ &\leq L(u_l, \ldots, u_{l+d-1}) + d_j - 1 \\ &\leq d - 1 + d_j - 1 \leq 2d - 2. \end{aligned}$$

Furthermore it also holds that $L(q')$ for

$$q' = s', q_2, \ldots, q_{d_j-1}, u_j, u_{j+1}, \ldots u_{l+2d-1}$$

is at most $2d - 1$, because

$$\begin{aligned} L(q') = & \; L(s', q_2, \ldots, q_{d_j-1}, u_j) \\ & + L(u_j, u_{j+1}, \ldots u_{l+2d-1}) \\ < & \; L(u_{l-1}, u_l, \ldots, u_j) \\ & + L(u_j, u_{j+1}, \ldots, u_{l+2d-1}) \\ = & \; L(u_{l-1}, u_l, \ldots, u_{l+2d-1}) = 2d. \end{aligned}$$

□

Theorem 7. *Let $d \in \mathbb{N}$ be a positive integer and $v \in V$ a vertex with local connectivity distance d in an undirected graph $G = (V, E)$ that represents a WSN. Then the $(2d-1)$-HBF protocol, initiated by a neighbor $s \in N_G^1(v)$ for node v and a message M, distributes M to all nodes in $N_G^{2d}(v)$ even if v itself does not relay any messages.*

Proof. Let $t \in N_G^1(v)$ be an arbitrary neighbor of v. According to Lemma 5 there is a path p between s and t in $G|_{V \setminus \{v\}}$ such that every sub-path of length

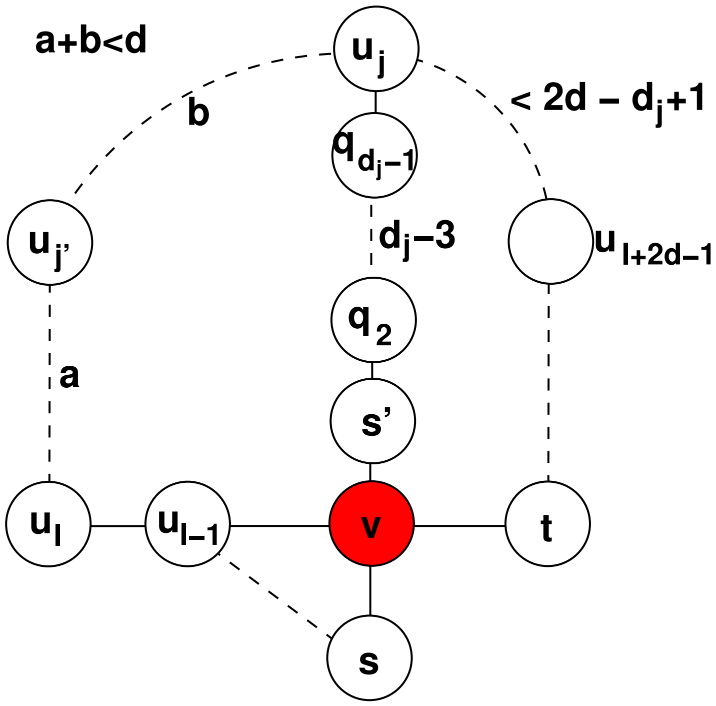

Fig. 6. Proof of Lemma 5: The dashed lines represent paths of the noted lengths

$2d-1$ contains at least one neighbor of v. Therefore the message sent by s is relayed along p, because the transmission counter i_t is reset after at most $2d-1$ transmissions and hence t receives M. And since every neighbor of v receives M and resets the transmission counter once, M is distributed to all nodes with hop-distance at most $2d-1$ to any neighbor of v, i.e. to all nodes with hop-distance at most $2d$ to v.

Although the d-hop neighborhood of v is connected, it is necessary to distribute the message across the $2d$-hop neighborhood of v, unless there is additional information about the network topology available. This is shown in the next Theorem.

Theorem 8. *Let $G=(V,E)$ be an undirected graph that represents a WSN, $d \in \mathbb{N}$ a positive integer and $v \in V$ a vertex with local connectivity distance d.*

Furthermore let $\mathcal{P}$ be a distributed algorithm with the following properties:

1. *$\mathcal{P}$ is initiated by a single node $s \in N_G^1(v)$ to distribute a message M.*
2. *$\mathcal{P}$ does not run on node v.*

3. *$\mathcal{P}$ is not randomized and the only information $\mathcal{P}$ utilizes about the network topology is the integer d and the knowledge about the 1-hop neighborhood $N_G^1(u)$ at each node u.*
4. *$\mathcal{P}$ terminates after a finite amount of steps and at that point every node in $N_G^1(v)$ received M.*

Then $\mathcal{P}$ transmits at least one message to every node in

$$N_G^{2d}[v] \quad = \quad \bigcup_{i=1}^{2d} N_G^i(v).$$

Proof. It is first shown that every node $u \in N_G^{2d}(v)$ has to receive at least one message.

Let $H := G|_{V \setminus \{v\}}$ be the induced subgraph of G that does not contain v and consider the graph

$$G' \quad := \quad (V, E \cup \{u, v\}).$$

Vertex v is d-locally connected in G': Since

$$N_G^{2d}(v) = \bigcup_{s' \in N_G^1(v)} N_G^{2d-1}(s') = \bigcup_{s' \in N_G^1(v)} N_H^{2d-1}(s'),$$

there is a path $w_1, \ldots, w_{2d}$ between $w_1 = s'$ for some vertex $s' \in N_G^1(v)$ and $w_{2d} = u$ in H. And for all $1 \leq i \leq 2d$ it holds that $d_{G'}(v, w_i) \leq d$, because $d_H(s', w_j) \leq d - 1$ for $j \in \{1, \ldots, d\}$ and $d_H(u, w_j) \leq d - 1$ for $j \in \{d + 1, \ldots, 2d\}$.

Vertex v is not $(d-1)$-locally connected in G': Every vertex $s' \in N_G^1(v)$ satisfies $d_H(s', u) \geq 2d - 1$, because $d_G(v, u) = 2d$. Therefore every path $w_1, \ldots, w_l$ between $w_1 = s'$ for some $s' \in N_G^1(v)$ and $w_l = u$ in H satisfies $l \geq 2d$ and thus it holds that $d_H(w_d, s') \geq d - 1$ and $d_H(w_d, u) \geq d - 1$. But then there is no path between neighbor $u \in N_{G'}^1(v)$ and any neighbor $s' \in N_G^1(v)$ in $G'|_{N_{G'}^{d-1}(v)}$, meaning that v is not $(d-1)$-locally connected in G'.

We will now compare the execution of $\mathcal{P}$ initiated at node s in G to the execution of $\mathcal{P}$ initiated at node s in G': Since v is d-locally connected and not $(d-1)$-locally connected both in G and G', the integer d is identical in both executions of $\mathcal{P}$ and by property 3 these two processes can only differ from each other due to a difference in the 1-hop neighborhood information at some node. However, the 1-hop neighborhood in G and G' is identical at all nodes except for v and u and $\mathcal{P}$ does not have access to the information at node v by property 2. Therefore both executions are identical until $\mathcal{P}$ sends a message to node u in G', which happens because of property 4, and thus u also receives a message in G.

Knowing that every node in $N_G^{2d}(v)$ has to receive at least one message, it follows that every node in $N_G^{2d}[v]$ has to receive at least one message, because it is possible to generate a network topology G'' that satisfies the preconditions of the Theorem while forcing every node in $N_{G''}^{2d-1}[v]$ to relay (and therefore receive) at least one message in order to reach all nodes in $N_{G''}^{2d}(v)$: G'' contains all vertices and edges from G and for every vertex $w \in N_G^{2d-1}[v]$ with distance $x := d_G(v, w)$ one additional vertex w' that is connected to w via a path of length $2d - x$. Then $w' \in N_{G''}^{2d}(v)$ and therefore w has to receive at least one message, which implies that w and all nodes on the path between w and w' have to relay that message. □

Experimental Analysis

Based on the theoretical analysis in the previous section, the experimental analysis of the k-HBF protocol is performed by solving the LCD problem rather than simulating the protocol itself. The computations are done on randomly generated graphs, using different graph models that are commonly used for simulations in the wireless network research community. The simulation software has been implemented in Java and executed in the Java Runtime Environment at version 8u74 on a system running the Kubuntu 12.04 LTS operating system, whose NativePRNG was used for random number generation.

For every set of parameters the results are computed for 100 graphs that are generated as follows:

1. 9000 vertices are placed randomly in a 3000×3000 square.
2. Edges are added according to one of the graph models described below.
3. The connected component containing the maximum number of vertices is determined and used for the simulation.

To generate edges the following models are used:

1. Unit Disk Graph (UDG) with radius $r \in \mathbb{R}$: An edge between two vertices is generated if and only if their euclidean is at most r. It is well known that this graph model is unrealistic for real life wireless networks and yet it is still used by many researchers for simulations. We use this model as a point of reference.
2. Waxman with parameters $\alpha \in [0, 1]$ and $r \in \mathbb{R}$: The random graph model introduced by Waxman in [92] captures important effects that occur in real life networks. Unlike the UDG model it does not guarantee the existence

of edges between vertices that are positioned close to each other and it also generates "long" edges. The Waxman model is often used by researchers, because it is implemented in the BRITE topology generator that can easily be used in conjunction with the *ns-3* network simulator. In this model an edge between two vertices u, v is added to the graph with probability

$$P(u,v) \quad = \quad \alpha \cdot exp\left(\frac{-d(u,v)}{0.5 \cdot \sqrt{2} \cdot r}\right)$$

where $d(u,v)$ denotes the euclidean distance between u and v. The scaling factor of $0.5 \cdot \sqrt{2}$ has been chosen such that, for $\alpha = 1$ and the values used for r, the average vertex degree of the generated graphs is similar to the UDG model with the same parameter r.

3. Locality with parameter $r \in \mathbb{R}$: The locality model that is mentioned in [94] and also implemented in the BRITE topology generator partitions the euclidean distances between two vertices into a finite amount of categories and assigns different, constant probabilities to each category. Based on the parameter r and the euclidean distance $d(u,v)$ between u and v, the edge $\{u,v\}$ is added with probability

$$P(u,v) \quad = \quad 0.8 - 0.1 \cdot (i-1),$$

if $r \cdot \frac{i-1}{4} < d(u,v) \leq r \cdot \frac{i}{4}$ for $i \in \{1,\ldots,8\}$. Edges between vertex pairs u, v with $d(u,v) > 2r$ are not added. Unlike the UDG model, edges between vertices that are positioned close to each other are not guaranteed while there still is an upper bound for the distance between adjacent vertices and in contrast to the Waxman model the probability does not decrease exponentially with the distance.

For every set of parameters, the *average vertex degree* δ and *the average graph diameter* $\varnothing$ are determined for comparability. Then, for every distance d_{min}, it is computed how many of the n *vertices* are d_{min}-locally connected, but not $(d_{min}-1)$-locally connected, i.e. have local connectivity distance d_{min}.

Two special cases are noted separately in the following tables: The *number of separation vertices* is given in row $d_{min} = \infty$, because these vertices are obviously never d-locally connected and it is not possible to perform a Neighborhood Broadcast for them. Also the *number* Δ_1 *of vertices with vertex degree* 1 is given separately due to the fact that they are trivially 1-locally connected and that their neighbor always is a separation vertex. The Δ_1 vertices are also contained in the number given for $d_{min} = 1$.

Since the existence of edges between vertices that are positioned close to each other are guaranteed and the fact that the maximum length of any edge

Table 1. Waxman $\alpha = 1$

r	50		
n	899170		
δ	7.63		
$\varnothing$	37.38		
Δ_1	4556		
d_{min}	#vertices	(%)	
1	34228	(3.80)	
2	797634	(88.7)	
3	61867	(6.88)	
4	777	(0.09)	
$5-6$	4	(0.00)	
∞	4660	(0.52)	

r	70	
n	899987	
δ	14.75	
$\varnothing$	22.07	
Δ_1	70	
d_{min}	#vertices	(%)
1	41529	(4.61)
2	854816	(94.98)
3	3569	(0.40)
4	2	(0.00)
∞	71	(0.01)

Table 2. Waxman $\alpha = 1$

r	90	
n	900000	
δ	24.09	
$\varnothing$	15.8	
Δ_1	1	
d_{min}	#vertices	(%)
1	60066	(6.67)
2	839379	(93.26)
3	554	(0.06)
∞	1	(0.00)

r	110	
n	900000	
δ	35.54	
$\varnothing$	12.14	
Δ_1	0	
d_{min}	#vertices	(%)
1	81543	(9.10)
2	818301	(90.92)
3	156	(0.02)
∞	0	(0.00)

Table 3. Waxman $\alpha = 0.75$

r	50	
n	895888	
δ	5.74	
$\varnothing$	42.2	
Δ_1	18821	
d_{min}	#vertices	(%)
1	20015	(2.23)
2	657711	(73.41)
3	189849	(21.19)
4	8373	(0.93)
5	331	(0.04)
6	28	(0.00)
∞	19581	(2.19)

r	70	
n	899928	
δ	11.06	
$\varnothing$	24.18	
Δ_1	578	
d_{min}	#vertices	(%)
1	12104	(1.34)
2	872618	(96.97)
3	14594	(1.62)
4	30	(0.00)
∞	582	(0.06)

Table 4. Waxman $\alpha = 0.75$

r	90	
n	899997	
δ	18.07	
$\varnothing$	16.98	
Δ_1	30	
d_{min}	#vertices	(%)
1	16325	(1.81)
2	881550	(97.95)
3	2092	(0.23)
∞	30	(0.00)

r	110	
n	900000	
δ	26.66	
$\varnothing$	13.06	
Δ_1	0	
d_{min}	#vertices	(%)
1	25332	(2.81)
2	874132	(97.13)
3	536	(0.06)
∞	0	(0.00)

Table 5. Waxman $\alpha = 0.5$

r	50	
n	874040	
δ	3.92	
$\varnothing$	51.83	
Δ_1	74781	
d_{min}	#vertices	(%)
1	16385	(1.87)
2	352649	(40.35)
3	328613	(37.60)
4	80075	(9.16)
5	12783	(1.46)
$6-9$	2326	(0.27)
∞	81209	(9.29)

r	70	
n	898911	
δ	7.38	
$\varnothing$	28.04	
Δ_1	5940	
d_{min}	#vertices	(%)
1	4847	(0.54)
2	770761	(85.74)
3	116337	(12.94)
4	906	(0.10)
5	23	(0.00)
∞	6037	(0.67)

Table 6. Waxman $\alpha = 0.5$

r	90	
n	899944	
δ	12.03	
$\varnothing$	19.1	
Δ_1	377	
d_{min}	#vertices	(%)
1	2012	(0.22)
2	883397	(98.16)
3	14144	(1.57)
4	8	(0.00)
∞	383	(0.04)

r	110	
n	899996	
δ	17.78	
$\varnothing$	14.56	
Δ_1	31	
d_{min}	#vertices	(%)
1	2103	(0.23)
2	894926	(99.44)
3	2934	(0.33)
∞	33	(0.00)

Table 7. Unit Disk Graph

r	40
n	784287
δ	5.13
$\varnothing$	222.63
Δ_1	20140
d_{min}	#vertices (%)
1	531542 (67.78)
2	71115 (9.07)
$3-6$	67179 (8.57)
$7-20$	36789 (4.69)
$21-50$	9104 (1.16)
$51-80$	1568 (0.20)
$81-121$	389 (0.05)
∞	66601 (8.49)

r	50
n	897819
δ	7.76
$\varnothing$	118.13
Δ_1	3307
d_{min}	#vertices (%)
1	750717 (83.62)
2	67635 (7.53)
3	32973 (3.67)
4	17447 (1.94)
5	9494 (1.06)
$6-10$	12118 (1.35)
$11-21$	840 (0.09)
∞	6595 (0.73)

Table 8. Unit Disk Graph

r	60
n	899868
δ	11.12
$\varnothing$	89.56
Δ_1	320
d_{min}	#vertices (%)
1	847605 (94.19)
2	36412 (4.05)
3	10895 (1.21)
4	3153 (0.35)
$5-11$	1254 (0.14)
∞	549 (0.06)

r	70
n	899983
δ	15.10
$\varnothing$	73.37
Δ_1	39
d_{min}	#vertices (%)
1	886854 (98.54)
2	11383 (1.26)
3	1470 (0.16)
4	163 (0.02)
$5-6$	51 (0.01)
∞	62 (0.01)

Table 9. Locality Model

r	30
n	790951
δ	3.75
$\varnothing$	190.64
Δ_1	69865
d_{min}	#vertices (%)
1	171022 (21.62)
2	365517 (46.21)
3	56484 (7.14)
$4-10$	65442 (8.27)
$11-40$	20853 (2.64)
$41-70$	1586 (0.2)
$71-102$	275 (0.03)
∞	109772 (13.88)

r	50
n	899844
δ	9.79
$\varnothing$	61.59
Δ_1	933
d_{min}	#vertices (%)
1	407176 (45.25)
2	487215 (54.14)
3	4190 (0.47)
4	268 (0.03)
5	12 (0.00)
6	7 (0.00)
∞	976 (0.11)

Table 10. Locality Model

r	70
n	899996
δ	19.00
$\varnothing$	39.14
Δ_1	10
d_{min}	#vertices (%)
1	722197 (80.24)
2	177787 (19.75)
3	2 (0.00)
∞	10 (0.00)

r	90
n	900000
δ	31.16
$\varnothing$	28.91
Δ_1	0
d_{min}	#vertices (%)
1	861893 (95.77)
2	38107 (4.23)
∞	0 (0.00)

is bounded in the purely theoretical unit disk graph model, the Tables 7 and 8 exhibit that the majority of vertices in this model is 1-locally connected, even in very sparse graphs. All graphs generated based on the Waxman model contain a significantly lower rate of 1-locally connected vertices and the number of vertices clearly spikes at local connectivity distance 2. In the locality model on the other hand, the spike gradually shifts from local connectivity distance 2 to local connectivity distance 1 with increasing vertex degree, presumably due to the still existing maximum edge length in this model.

Measuring the success rate of the k-HBF protocol as a percentage of non-separation vertices, the conducted simulations indicate a success rate of more than 80% across all considered graph models for the 5-HBF protocol, the minimum being defined by the very sparse graphs with an average vertex degree below 6. Restricted to graphs with an average vertex degree of at least 7, the success rate of the 5-HBF protocol is above 95% and the success rate of the 3-HBF protocol is still above 85%.

While one might intuitively presume that most of the vertices with higher local connectivity distance are close to the border of the geometric region that graph is placed across, the sample graphs taken during the simulation do not verify this conjecture: Typically these vertices offer some sort of “shortcut” through an otherwise sparse region of the graph, see Figure 7 for an example.

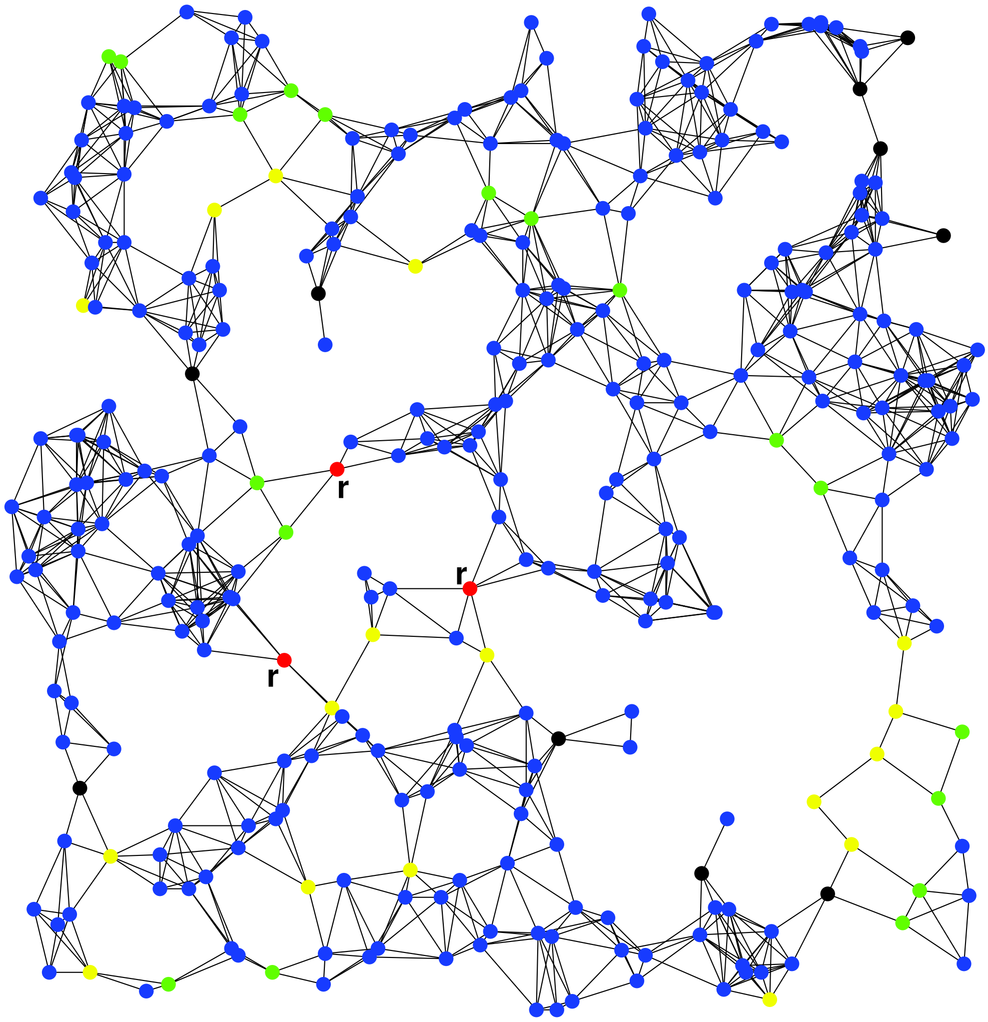

Fig. 7. An induced subgraph of a unit disk graph with radius $r = 50$: The red (r) vertices have local connectivity distance 5 due to their position between two “holes”. Other distances are blue ($d_{min} = 1$), green ($d_{min} = 2$), yellow ($d_{min} = 3$) and black ($d_{min} > 5$). Note that the local connectivity distance of some vertices at the border is not verifiable based on this image, because some incident edges have been removed.

3 Topology Control with Respect to Local Connectivity

Since the success rate of the d-HBF protocol and the overall capability to perform a Neighborhood Broadcast depends on the local connectivity of the WSN, it would be desirable to achieve local connectivity through the means of topology control, if the initial network is not locally connected. The following sections examine several approaches to this kind of topology control and investigate the computational complexity of the corresponding decision problems for graphs.

3.1 Maximum Locally *k*-Connected Vertex Set

The first problem asks for an induced subgraph with a maximum number of vertices that is locally k-connected as well as connected. In terms of sensor networks this problem corresponds to identifying a more reliable subnetwork and this information could be used, for example, as a criteria for energy saving mechanisms: Sensor nodes belonging to the identified subset might get prioritized when it comes to energy saving to keep a reliable backbone network operational as long as possible.

Determining maximum connected induced subgraphs $G|_U$ of a given graph G such that a specific graph property Π holds for $G|_U$ has been examined in great detail and it is known that these problems are NP-complete for many graph properties Π, which is shown in [35]. Note that this general result cannot be applied to the Maximum Locally k-Connected Vertex Set problem below, because the graph property "is locally connected" is not *hereditary*, i.e. it does not hold for all induced subgraphs of a graph G whenever it holds for G.

Maximum Locally k-Connected Vertex Set

Given: An undirected graph $G = (V, E)$ and a positive integer $m \in \mathbb{N}$
Question: Is there a vertex set $U \subseteq V$ of cardinality $|U| \geq m$, such that $G|_U$ is connected and locally k-connected?

Theorem 9. *The* Maximum Locally 1-Connected Vertex Set *problem is NP-complete.*

Maximum Locally k-Connected Vertex Set is in NP for all positive integers $k \in \mathbb{N}$, because for a given set of vertices $U \subseteq V$ a deterministic algorithm can decide in polynomial time whether every vertex $u \in U$ is locally k-connected in $G|_U$ by using network flow techniques to determine the connectivity of $G|_{N^1(u)}$ [29].

The NP-hardness of Maximum Locally 1-Connected Vertex Set is shown by the following polynomial time reduction from Satisfiability that constructs a graph G and an integer m from an arbitrary instance $(X, \mathcal{C})$ for Satisfiability such that $(X, \mathcal{C})$ is satisfiable if and only if there is a connected, locally connected vertex set of cardinality at least m in G.

Let $(X, \mathcal{C})$ be an instance for Satisfiability with variable set X and clause set $\mathcal{C}$. Assume w.l.o.g. that there is no clause $c \in \mathcal{C}$ such that c contains the positive literal x and the negative literal $\overline{x}$ for a variable $x \in X$ (if such clauses exist they can be removed from $\mathcal{C}$, because they are satisfied for every truth assignment of X). Also assume that for every variable $x \in X$ there is at least one clause $c \in C$ that contains either the literal x or the literal $\overline{x}$.

Define $m := 2 + 2 \cdot |X| + 4 \cdot |\mathcal{C}|$ and construct G by adding a copy of a graph for every variable in X and for every clause in $\mathcal{C}$ as follows.
For every variable $x_i \in X$ add a cycle $H_i := (V_i, E_i)$ with 4 vertices, that is

$$\begin{aligned} V_i &:= \{x_{i,1}, x_{i,2}, \overline{x_{i,1}}, \overline{x_{i,2}}\} \\ E_i &:= \{\{x_{i,1}, x_{i,2}\}, \{\overline{x_{i,1}}, \overline{x_{i,2}}\}, \{x_{i,1}, \overline{x_{i,1}}\}, \{x_{i,2}, \overline{x_{i,2}}\}\} \end{aligned}$$

For every clause $c_j \in \mathcal{C}$ add a path $P_j := (V'_j, E'_j)$ with 4 vertices, that is

$$\begin{aligned} V'_j &:= \{c_{j,1}, c_{j,2}, c_{j,3}, c_{j,4}\} \\ E'_j &:= \{\{c_{j,1}, c_{j,2}\}, \{c_{j,2}, c_{j,3}\}, \{c_{j,3}, c_{j,4}\}\} \end{aligned}$$

These *variable gadgets* $C_i = (V_i, E_i)$ and *clause gadgets* $P_j = (V'_j, E'_j)$ are connected to each other by adding the following edges. Let $x_i \in X$ be a variable and $c_j \in \mathcal{C}$ a clause. If c_j contains the positive literal x_i, the edges $\{x_{i,1}, c_{j,1}\}$, $\{x_{i,1}, c_{j,2}\}$, $\{x_{i,2}, c_{j,2}\}$, $\{x_{i,2}, c_{j,3}\}$ and $\{x_{i,2}, c_{j,4}\}$ are added. If c_j contains the negative literal $\overline{x_i}$, the edges $\{\overline{x_{i,1}}, c_{j,1}\}$, $\{\overline{x_{i,1}}, c_{j,2}\}$, $\{\overline{x_{i,2}}, c_{j,2}\}$, $\{\overline{x_{i,2}}, c_{j,3}\}$ and $\{\overline{x_{i,2}}, c_{j,4}\}$ are added. Also see Figure 8 for an example.

Additionally, the graph G contains two vertices w_1 and w_2, the edge $\{w_1, w_2\}$ as well as all edges $\{w, c_{j,3}\}, \{w, c_{j,4}\}$ for $w \in \{w_1, w_2\}$ and $c_j \in \mathcal{C}$. The vertices w_1 and w_2 serve as a connection between all clause gadgets and thereby guarantee that the induced subgraph $G|_U$ will be connected.

Lemma 6. *Let $U \subseteq V$ and $c_j \in \mathcal{C}$ be a clause such that the corresponding vertices $\{c_{j,1}, c_{j,2}, c_{j,3}, c_{j,4}\}$ are in U, i.e. $V'_j \subseteq U$. Then all vertices in V'_j are locally connected in $G|_U$ if and only if there is a variable $x_i \in X$ such that either $x_i \in c_j$ and $\{x_{i,1}, x_{i,2}\} \subset U$ or $\overline{x_i} \in c_j$ and $\{\overline{x_{i,1}}, \overline{x_{i,2}}\} \subset U$.*

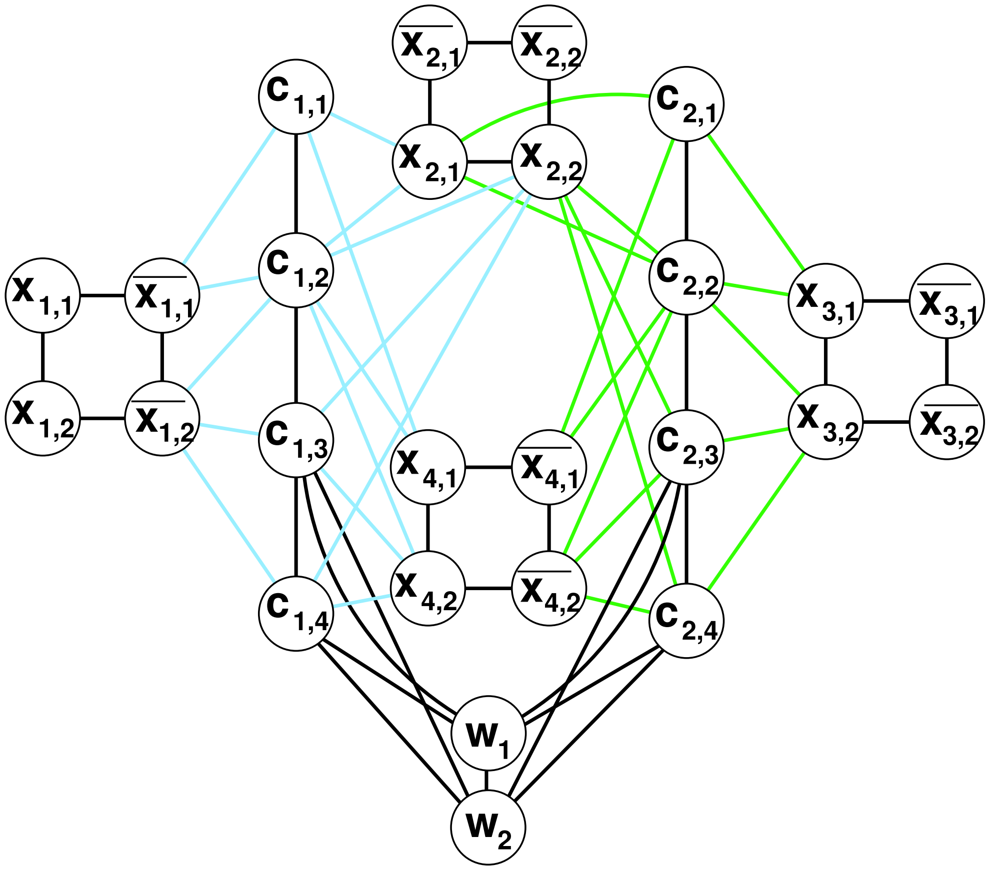

Fig. 8. Graph G constructed from instance $(\{x_1, x_2, x_3, x_4\}, \{c_1, c_2\})$, $c_1 = \{\overline{x_1}, x_2, x_4\}$, $c_2 = \{x_2, x_3, \overline{x_4}\}$. Edges between vertices from clause gadget H_1 and vertices from variables gadgets P_1, P_2, P_4 are colored blue and edges between vertices from clause gadget H_2 and vertices from variables gadgets P_2, P_3, P_4 are colored green.

Proof. $\Rightarrow$: Vertex $c_{j,2}$ is locally connected, so there is at least one simple path $p = c_{j,1}, v_1, \cdots, v_l, c_{j,3}$ in $G|_{N^1(c_{j,2})}$. Due to the construction of G, v_1 has to be either $x_{i_1,1}$ or $\overline{x_{i_1,1}}$ for some i_1 and v_l has to be either $x_{i_2,2}$ or $\overline{x_{i_2,2}}$ for some i_2. Additionally, every path between vertices from different variable gadgets contains at least one vertex from a clause gadget. But since the path $v_1, \cdots, v_l$ is simple, it contains neither $c_{j,1}$ nor $c_{j,3}$ and those are the only vertices from clause gadgets that are adjacent to $c_{j,2}$, meaning that $G|_{N^1(c_{j,2})}$ does not contain any other vertices from clause gadgets. Therefore $i_1 = i_2 =: i$ and because c_j does not contain both literals x_i and $\overline{x_i}$, it follows that either $x_i \in c_j$ and $\{x_{i,1}, x_{i,2}\} \subset U$ or $\overline{x_i} \in c_j$ and $\{\overline{x_{i,1}}, \overline{x_{i,2}}\} \subset U$.

$\Leftarrow$: The vertex $c_{j,1}$ is locally connected due to Lemma 3, because the edge $\{c_{j,1}, c_{j,2}\}$ is in $E|_U$ and $N^1_{G|_U}(c_{j,1}) \setminus \{c_{j,2}\} \subset N^1_{G|_U}(c_{j,2})$. Similarly, the vertex $c_{j,4}$ is locally connected, because $\{c_{j,3}, c_{j,4}\} \in E|_U$ and $N^1_{G|_U}(c_{j,4}) \setminus \{c_{j,3}\} \subset N^1_{G|_U}(c_{j,3})$. Without loss of generality, let there be a variable $x_i \in X$ such that $x_i \in c_j$ and $\{x_{i,1}, x_{i,2}\} \subset U$ (analogous argumentation for the case $\overline{x_i} \in c_j$). Then vertex $c_{j,3}$ is locally connected, because $c_{j,2}, x_{i,2}, c_{j,4}$ is a path in the subgraph induced by $N^1_{G|_U}(c_{j,3})$ and the fact that every vertex from a variable gadget that is adjacent to $c_{j,3}$, as well as the vertices w_1 and w_2, is also adjacent to $c_{j,4}$. Finally, the vertex $c_{j,2}$ is locally connected, because $c_{j,1}, x_{i,1}, x_{i,2}, c_{j,3}$ is a path in the subgraph induced by $N^1_{G|_U}(c_{j,2})$ and the fact that every vertex from a variable gadget that is adjacent to $c_{j,2}$ is also adjacent to either $c_{j,1}$ or $c_{j,3}$, see Figure 9. $\square$

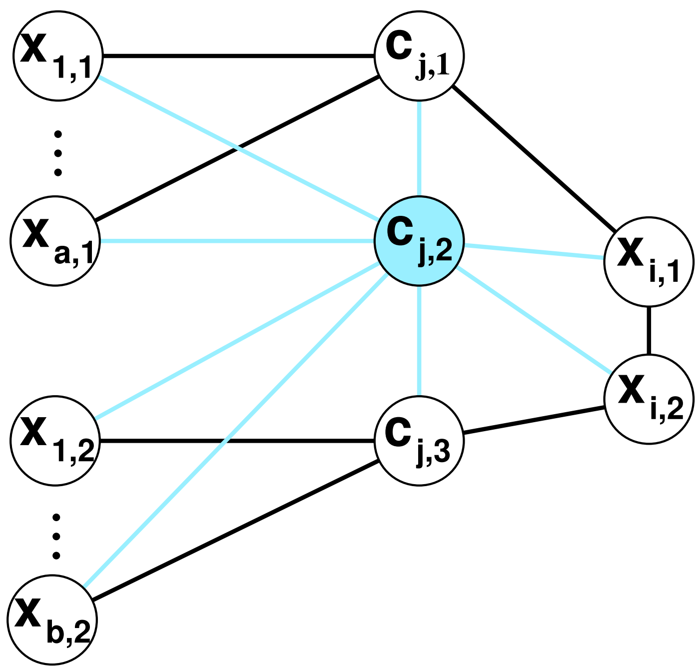

Fig. 9. A connected subgraph G' of the induced subgraph $G|_{N^1(c_{j,2})}$ for arbitrary numbers $a, b \in \mathbb{N}$ of neighbors inside adjacent variable gadgets. The blue vertex $c_{j,2}$ itself does not belong to G'.

Lemma 7. *Every locally connected vertex set $U \subset V$ contains at most two vertices from every variable gadget.*

Proof. Assume that U contains more than two vertices from a variable gadget (V_i, E_i). Without loss of generality, let $x_{i,1}$, $x_{i,2}$ and $\overline{x_{i,1}}$ be in U (analogous argumentation for all other cases due to symmetry). Then $x_{i,1}$ is not locally connected, which contradicts the assumption that U is a locally connected vertex set: Since there is no clause that contains both literals x_i and $\overline{x_i}$, there is no vertex from a clause gadget that is adjacent to both $x_{i,1}$ and $\overline{x_{i,1}}$. However, these two vertices are also not adjacent to w_1 or w_2 in G, which means that $N^1_{G|_U}(x_{i,1}) \cap N^1_{G|_U}(\overline{x_{i,1}}) = \emptyset$. Therefore there is no path between $\overline{x_{i,1}}$ and $x_{i,2}$ in the subgraph induced by $N^1_{G|_U}(x_{i,1})$. $\square$

The proof of Theorem 9 is now concluded by showing that $(X, \mathcal{C})$ is satisfiable if and only if there is a locally connected vertex set $U \subset V$ of cardinality $|U| \geq m = 2 + 2 \cdot |X| + 4 \cdot |\mathcal{C}|$ in $G = (V, E)$ such that $G|_U$ is connected.

$\Rightarrow$: Let $T : X \to \{0, 1\}$ be a satisfying truth assignment for $\mathcal{C}$. For every literal l over X define the set of clauses that contain l as $L(l) := \{c \in \mathcal{C} \mid l \in c\}$. Now set $U := \{w_1, w_2\} \cup V_\oplus \cup V_\ominus \cup V_c$ where

- $V_c := \bigcup_{c_j \in \mathcal{C}} \{c_{j,1}, c_{j,2}, c_{j,3}, c_{j,4}\}$,
- $V_\oplus := \{x_{i,1}, x_{i,2} \mid (T(x_i) = 1 \wedge L(x_i) \neq \emptyset) \vee (T(x_i) = 0 \wedge L(\overline{x_i}) = \emptyset)\}$,
- $V_\ominus := \{\overline{x_{i,1}}, \overline{x_{i,2}} \mid (T(x_i) = 0 \wedge L(\overline{x_i}) \neq \emptyset) \vee (T(x_i) = 1 \wedge L(x_i) = \emptyset)\}$.

Then U contains exactly $2+2\cdot|X|+4\cdot|\mathcal{C}|$ vertices and U is a locally connected vertex set: Because T satisfies $\mathcal{C}$, there is a variable x_i for every $c_j \in \mathcal{C}$ such that either $x_i \in c_j$ and $T(x_i) = 1$ or $\overline{x_i} \in c_j$ and $T(x_i) = 0$. In the former case it holds that $x_{i,1}, x_{i,2} \in U$ and in the latter case $\overline{x_{i,1}}, \overline{x_{i,2}} \in U$. Thus all vertices $c_{j,1}, c_{j,2}, c_{j,3}, c_{j,4}$ are locally connected in $G|_U$ according to Lemma 6. Furthermore the vertices $x_{i,1}$ and $x_{i,2}$ for all $x_i \in X$ are locally connected in $G|_U$: The only neighbors of $x_{i,1}$ in $G|_U$ are $x_{i,2}$ and $N_1 := \{c_{j',1}, c_{j',2} \mid x_i \in c_{j'}\}$. Since $x_{i,2}$ is also adjacent to all vertices $c_{j',2} \in N_1$, there is a path $c_{j',2}, x_{i,2}, c_{j'',2}$ in $G|_{U \cap N^1(x_{i,1})}$ for all $c_{j',2}, c_{j'',2} \in N_1$. Additionally $\{c_{j',1}, c_{j',2}\} \in E|_{U \cap N^1(x_{i,1})}$, so there is a path between all pairs of vertices in $G|_{U \cap N^1(x_{i,1})}$, also see Figure 10. A similar argument demonstrates that $x_{i,2}$ is locally connected: There is a path $c_{j',2}, x_{i,1}, c_{j'',2}$ in $G|_{U \cap N^1(x_{i,2})}$ for all $c_{j',2}, c_{j'',2} \in N_2 := \{c_{j',2}, c_{j',3}, c_{j',4} \mid x_i \in c_{j'}\}$ and the edges $\{c_{j',2}, c_{j',3}\}$ and $\{c_{j',3}, c_{j',4}\}$ are in $E|_{U \cap N^1(x_{i,2})}$. Analogous arguments apply for $\overline{x_{i,1}}$ and $\overline{x_{i,2}}$. Finally, the two vertices w_1 and w_2 are locally connected, because they are true twins (see Corollary 2). $G|_U$ is connected due to the vertices w_1 and w_2, because these vertices are adjacent to a vertex from every clause gadget and the fact that every vertex from a variable gadget is adjacent to a vertex from a clause gadget.

$\Leftarrow$: Let U be a locally connected vertex set of cardinality at least $2 + 2 \cdot |X| + 4 \cdot |\mathcal{C}|$. According to Lemma 7, U contains at most two vertices from

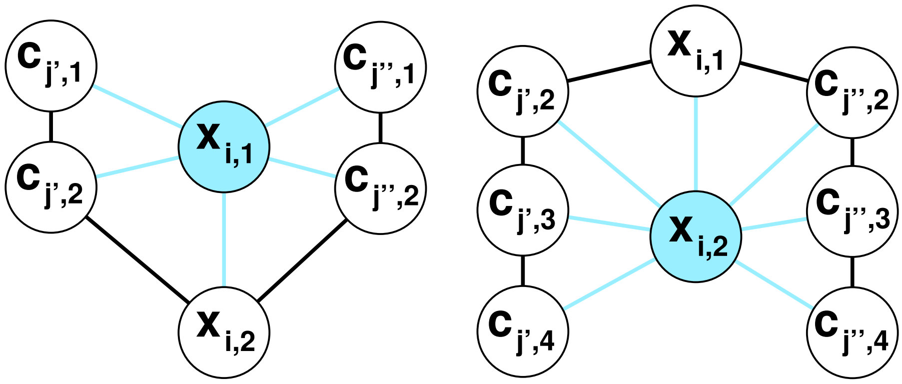

Fig. 10. The vertex sets $N^1(x_{i,1})$ (left, without the blue vertex) and $N^1(x_{i,2})$ (right, without the blue vertex) induce connected subgraphs for an arbitrary number of adjacent clause gadgets.

every variable gadget. Therefore U must contain all vertices $c_{j,1}, c_{j,2}, c_{j,3}, c_{j,4}$ of every clause gadget as well as the vertices w_1 and w_2.

Define a truth assignment $T : X \to \{0, 1\}$ as follows: If U contains both vertices $x_{i,1}$ and $x_{i,2}$, set $T(x_i) = 1$, otherwise set $T(x_i) = 0$. Then T satisfies $\mathcal{C}$: Let $c_j \in \mathcal{C}$ be a clause. Since $\{c_{j,1}, c_{j,2}, c_{j,3}, c_{j,4}\}$ is a locally connected vertex set, there is a variable $x_i \in X$ such that either $x_i \in c_j$ and $x_{i,1}, x_{i,2} \in U$ or $\overline{x_i} \in c_j$ and $\overline{x_{i,1}}, \overline{x_{i,2}} \in U$ as shown in Lemma 6. In the former case $T(x_i)$ has been set to 1 and in the latter case $T(x_i)$ has been set to 0 and therefore c_j is satisfied. □

3.2 Augmentation Problems

The term *augmentation problem* generally describes a problem regarding the question of how to achieve a desired graph property by adding additional edges to a given graph. The following subsections investigate which complementary edges $E' \subseteq \overline{E}$ have to be added to an undirected graph $G = (V, E)$ in order to attain connectivity in the subgraphs induced by certain neighborhoods. The nature of this question yields two different versions for each problem: The Weak version of this augmentation problem considers the neighborhoods defined by the original graph G, while the corresponding Strong version considers the altered neighborhoods defined by $(V, E \cup E')$. This section is concerned with proving the NP-completeness of both versions and presents a greedy algorithm for the Weak augmentation problem having a tight quality bound of $3/2$ for graphs without induced $K_{2,3}$ and a tight quality bound of $11/6$ for graphs without induced $K_{2,4}$. Recall that unit disk graphs never contain the $K_{2,3}$ as an induced subgraph (Corollary 1) and therefore the presented algorithm is a constant factor approximation for unit disk graphs. Furthermore Conjecture 1 implies that it also is a constant factor approximation for d-quasi unit disk graphs with $d \geq \sqrt{3}-1$. The presented greedy algorithm also provides an approximation factor of $1+ln(|V|)$ for the general case and it is conjectured that this bound is also tight.

In terms of sensor networks the task is to achieve locally connected graphs by adding additional communication links, which can be accomplished in several ways, depending on the prerequisites. For example:

– The sensor nodes might be capable of adjusting transmission power levels to establish additional radio links. This model will be defined in more detail in Chapter 4.
– The sensor nodes might also possess the limited ability to use arbitrary point to point transmissions via an additional internet uplink based on UMTS / LTE or similar techniques. Limitations to this approach are additional energy costs for these protocols and/or necessary financial investment for each data transmission. Local connectivity augmentation is still desirable in this scenario for repairing multicast routing structures.
– It is also possible to establish virtual communication links for this purpose based on suitable, predetermined routing paths.

Network design and graph augmentation problems related to connectivity have been studied extensively: The basic graph theoretic edge augmentation, i.e. adding a minimum number of additional edges to a k-connected graph in order to achieve $(k + 1)$-connectivity, is considered in [55], where an approximation algorithm is presented. In the context of communication networks, connectivity - or more precisely the number of either vertex or edge disjoint paths between given pairs of vertices - is commonly used as a metric for *survivability*, *reliability* and *fault tolerance*. Balakrishnan and Altinkemerin introduced hop constraints limiting the lengths of viable paths in [8], a technique that was subsequently considered in conjunction with edge or vertex disjoint paths for undirected as well as directed networks [11, 77]. We refer the reader to [58] for a more extensive overview on this topic.

The augmentation problems are formulated such that all complementary edges of the given graph are available for augmentation. It is noteworthy that the presented results also hold for the case that only a subset $\widetilde{E} \subseteq \overline{E}$ of the complementary edges is available for augmentation, because the NP-completeness is proven for the special case $\widetilde{E} = \overline{E}$. As for the greedy approach, the algorithm does not depend on the existence of specific edges and a restriction to this edge set also applies to any optimal solution.

LOCALLY **k**-CONNECTED WEAK AUGMENTATION

	LOCALLY k-CONNECTED WEAK AUGMENTATION
Given:	An undirected graph $G = (V, E)$ and a positive integer $m \in \mathbb{N}$.
Question:	Is there a set $E' \subseteq \overline{E}$ of cardinality $\lvert E'\rvert \leq m$ such that, for every vertex $v \in V$, the neighborhood $N^1_G(v)$ induces a k-connected subgraph of $(V, E \cup E')$?

Theorem 10. *The* LOCALLY 1-CONNECTED WEAK AUGMENTATION *problem is NP-complete.*

Proof. LOCALLY k-CONNECTED WEAK AUGMENTATION is in NP for all positive integers $k \in \mathbb{N}$, because for a given set E' of complementary edges a deterministic algorithm can decide in polynomial time whether the vertex sets $N^1_G(v)$ induce k-connected subgraphs in $(V, E \cup E')$ for all vertices $v \in V$.

The NP-hardness of LOCALLY 1-CONNECTED WEAK AUGMENTATION is shown by the following polynomial time reduction from SATISFIABILITY that constructs an undirected graph G and an integer m from an arbitrary instance $(X, \mathcal{C})$ of SATISFIABILITY with the following property: $(X, \mathcal{C})$ is satisfiable if and only if there is a set of complementary edges E' of cardinality $\lvert E'\rvert \leq m$ such that, for all vertices $v \in V$, $N^1_G(v)$ induces a connected subgraph of $(V, E \cup E')$. The integer m is set to $\lvert X\rvert$ and G is constructed as follows, also see Figure 11 for an example:

For every variable $x_i \in X$ the graph G contains a copy of the graph (V_i, E_i), with vertex set $V_i := \{x_i, u_i, u'_i, f_i, t_i\}$ and edge set
$E_i := \{\{u_i, u'_i\}, \{f_i, t_i\}, \{x_i, u_i\}, \{x_i, u'_i\}, \{x_i, f_i\}, \{x_i, t_i\}\}$.

For every clause $c_j \in \mathcal{C}$ the graph G contains two vertices c_j and c'_j and the edge $\{c_j, c'_j\}$.

Furthermore G also contains two vertices b_1 and b_2 as well as the following edges:

1. $\{b_1, b_2\}$
2. $\{\{b_1, f_i\}, \{b_2, f_i\}, \{b_1, t_i\}, \{b_2, t_i\} \mid x_i \in X\}$
3. $\{\{b_1, c_j\}, \{b_2, c_j\} \mid c_j \in \mathcal{C}\}$
4. $\{\{c_j, u_i\}, \{c_j, u'_i\}, \{c'_j, u_i\}, \{c'_j, u'_i\} \mid c_j \in \mathcal{C} \wedge (x_i \in c_j \vee \overline{x_i} \in c_j)\}$
5. $\{\{c_j, t_i\} \mid c_j \in \mathcal{C} \wedge x_i \in c_j\}$
6. $\{\{c_j, f_i\} \mid c_j \in \mathcal{C} \wedge \overline{x_i} \in c_j\}$

By construction, the neighborhood of all vertices except for the vertices x_i and c_j induce a connected subgraph of G and therefore also induce a connected subgraph of $(V, E \cup E')$ for all sets E':

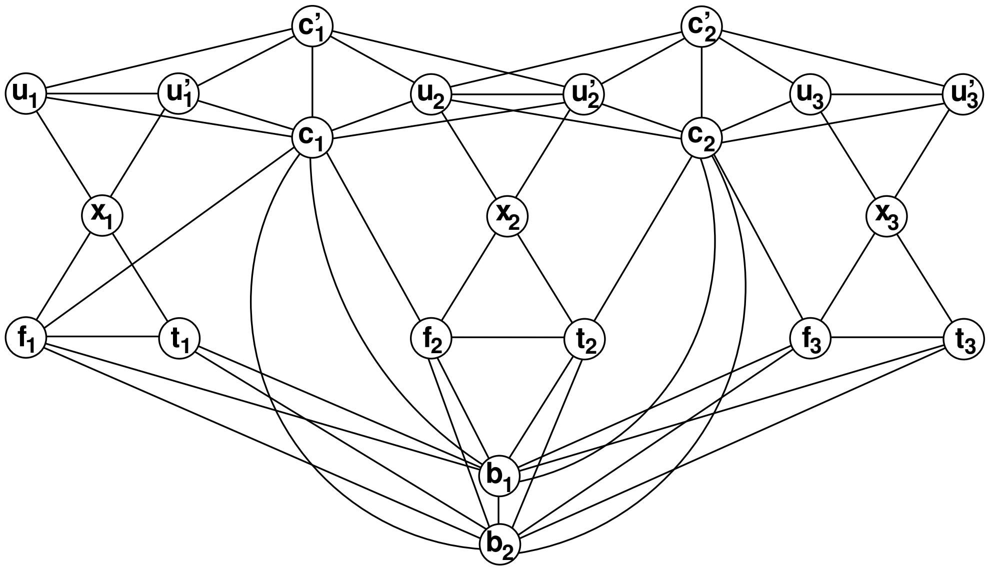

Fig. 11. Graph G constructed from instance $(\{x_1, x_2, x_3\}, \{c_1, c_2\})$ with $c_1 = \{\overline{x_1}, \overline{x_2}\}$ and $c_2 = \{x_2, \overline{x_3}\}$

- The vertices b_1 and b_2 are true twins and therefore locally connected in G, see Corollary 2.
- For all i, the vertices u_i and u'_i are true twins and therefore locally connected in G, see Corollary 2.
- For all j, the vertex c'_j is locally connected in G, because $\{c_j, c'_j\} \in E$ and $N^1_G(c'_j) \setminus \{c_j\} \subset N^1_G(c_j)$, see Lemma 3.
- For all i, the vertices t_i and f_i are true twins in the subgraph of G that does not contain the edges from 5 and 6 and every vertex that is adjacent to t_i or f_i due to those edges is also adjacent to their neighbor b_1.

We will now show that $(X, \mathcal{C})$ is satisfiable if and only if there is a set of complementary edges $E' \subseteq \overline{E}$ of cardinality at most $m = |X|$ such that, for all i, j, the vertex sets $N^1_G(x_i)$ and $N^1_G(c_j)$ induce connected subgraphs of $(V, E \cup E')$.

$\Rightarrow$: Let $T : X \to \{0, 1\}$ be a satisfying truth assignment for $(X, \mathcal{C})$.

$$E' := \{\{u'_i, t_i\} \mid x_i \in X \wedge T(x_i) = 1\} \cup \{\{u_i, f_i\} \mid x_i \in X \wedge T(x_i) = 0\}$$

Then $|E'| = m = |X|$ and the neighborhood $N^1_G(v)$ of all vertices $v \in V$ induces a connected subgraph of $(V, E \cup E')$, which only remains to show for the vertices x_i and c_j for the corresponding variables $x_i \in X$ and clauses $c_j \in \mathcal{C}$:

For every variable $x_i \in X$ the neighborhood $N^1_G(x_i) = \{u_i, u'_i, t_i, f_i\}$ of vertex x_i induces a connected subgraph of $(V, E \cup E')$, because E' contains either $\{u'_i, t_i\}$ or $\{u_i, f_i\}$, both of which merge the two connected components $G|_{\{u_i,u'_i\}}$ and $G|_{\{t_i,f_i\}}$ of $G|_{N^1_G(x_i)}$.

Let $x_{\pi(j)}$ be a variable that satisfies clause c_j under T. There is at least one such variable for each clause, because T satisfies $(X, \mathcal{C})$. Then the two connected components of $G|_{N^1_G(c_j)}$ are merged in $(V, E \cup E')$, because either $\{u_{\pi(j)}, f_{\pi(j)}\} \in E'$ and $\{c_j, f_{\pi(j)}\} \in E$ or $\{u'_{\pi(j)}, t_{\pi(j)}\} \in E'$ and $\{c_j, t_{\pi(j)}\} \in E$.

$\Leftarrow$: Let E' be a set of complementary edges with $|E'| \leq m$ such that the neighborhood of every vertex $v \in V$ induces a connected subgraph of $(V, E \cup E')$. $N^1_G(x_i)$ inducing a connected subgraph of $(V, E \cup E')$ for all vertices x_i implies that $|E'| \geq m$, because for all i, E' contains at least one of the following edges: $\{u_i, f_i\}$, $\{u_i, t_i\}$, $\{u'_i, f_i\}$, $\{u'_i, t_i\}$. Therefore we know that $|E'| = m$, meaning that E' contains exactly one of these edges for every index i. Define a truth assignment T by setting $T(x_i) = 1$ if E' contains an edge incident to t_i and $T(x_i) = 0$ if E' contains an edge incident to f_i. Then T satisfies $(X, \mathcal{C})$: Since $N^1_G(c_j)$ induces a connected subgraph of $(V, E \cup E')$ for all clauses c_j and E' contains only edges of the above form, it follows that there is at least one variable $x_{\pi(j)}$ for each c_j such that

- $x_{\pi(j)} \in c_j$ and $\{\{u_{\pi(j)}, t_{\pi(j)}\}, \{u'_{\pi(j)}, t_{\pi(j)}\}\} \cap E' \neq \emptyset$ or
- $\overline{x_{\pi(j)}} \in c_j$ and $\{\{u_{\pi(j)}, f_{\pi(j)}\}, \{u'_{\pi(j)}, f_{\pi(j)}\}\} \cap E' \neq \emptyset$.

Therefore, for every clause c_j, there is at least one variable $x_{\pi(j)}$ that satisfies c_j under T, meaning that T is a satisfying truth assignment. □

Greedy Algorithm for LCWA

This section presents and evaluates a greedy approach to the construction version of the Locally k-Connected Weak Augmentation problem, restricted to $k = 1$, that is defined as follows.

> Locally Connected Weak Augmentation (LCWA)
>
> *Given:* An undirected graph $G = (V, E)$ without isolated vertices.
> *Task:* Compute a set $E' \subseteq \overline{E}$ of minimum cardinality such that for all $v \in V$ the neighborhood $N^1_G(v)$ induces a connected subgraph of $(V, E \cup E')$.

The general idea behind Algorithm 2 is to greedily choose an edge e that reduces the number of local connectivity components as much as possible in

Algorithm 2 Greedy algorithm for LCWA

```
function AugmentationValue(V, E, E', {x, y})
    m ← 0
    L ← LCC_(V,E)((V, E ∪ E'))
    for all v ∈ N_(V,E)(x) ∩ N_(V,E)(y) do
        if ¬∃(v, C) ∈ L :   x ∈ C ∧ y ∈ C then
            m ← m + 1
        end if
    end for
    return m
end function
function GreedyWeakAugmentation(V, E)
    E' ← ∅
    r ← |LCC_G(G)| − |V|
    while r > 0 do
        m_max ← 0
        for all e ∈ Ē do                          ▷ search e_max with max. value
            m ← AugmentationValue(V, E, E', e)
            if m > m_max then
                m_max ← m
                e_max ← e
            end if
        end for
        E' ← E' ∪ {e_max}                         ▷ greedily add e_max to the solution
        r ← r − m_max                             ▷ update the number of LCCs
    end while
    return E'
end function
```

every step. For this purpose we call e m-valuable, if the addition of e to the current graph reduces the number of LCCs by m, formally noted in the next definition. Although the given implementation obviously has a low polynomial running time, it is optimized for clarity rather than efficiency. Utilizing suitable data structures to continuously update the values of all complementary edges after an initialization phase allows the reduction of the running time to an even lower polynomial function.

Definition 15. *For a positive integer $m \in \mathbb{N}$, a complementary edge $e \in \overline{E}$ of an undirected graph $H = (V, E)$ is* m-valuable with respect to a subgraph *$G = (V, E')$ of H (equivalently: has* value m with respect to G*), if*

$$m \quad = \quad LCC_G(H) - LCC_G((V, E \cup \{e\})).$$

Definition 16. *For a positive integer $m \in \mathbb{N}$, a complementary edge $e \in \overline{E}$ of an undirected graph $H = (V, E)$ is* m-valuable with respect to a subgraph

$G = (V, E')$ *of* H *(equivalently: has* value m with respect to G*), if*

$$m \quad = \quad |LCC_G(H)| - |LCC_G((V, E \cup \{e\}))|.$$

Also see Figure 12 for an example.

Theorem 11. *Let* $m \in \mathbb{N}$ *be a positive integer and* $G = (V, E)$ *an undirected graph without isolated vertices such that every complementary edge* $e \in \overline{E}$ *is at most* m*-valuable with respect to* G*. Furthermore let* $E_A \subseteq \overline{E}$ *be the solution computed by Algorithm 2 for instance* G *and* $E_{OPT} \subseteq \overline{E}$ *be an optimal solution for* LCWA*. Then*

$$|E_A| \quad \leq \quad \sum_{j=1}^{m} \frac{1}{j} \cdot |E_{OPT}|.$$

Proof. Let $e_1, \ldots, e_{|E_A|}$ be the sequence of edges chosen by Algorithm 2 and $G_i := (V, E \cup \{e_1, \ldots e_{i-1}\})$ the partially augmented graph after the insertion of the first $i-1$ edges. The current value of $e \in \overline{E}$ is denoted by $f_{G_i}(e)$, i.e.

$$f_{G_i}(e) = m' \Leftrightarrow e \text{ is } m'\text{-valuable in } G_i \text{ with respect to } G$$

Algorithm 2 computed a solution for LCWA and terminates as soon as the current graph G_i has exactly $|V|$ local connectivity components with respect to G. The number of LCCs that remain to be merged is denoted as d_i and given by:

$$\begin{aligned} d_0 \quad &:= \quad |LCC_G(G)| \quad - \quad |V| \\ d_i \quad &:= \quad d_{i-1} \quad - \quad f_{G_i}(e_i) \end{aligned} \tag{1}$$

Furthermore let $k_i \in \mathbb{Z}$ be the largest integer that satisfies

$$d_i \quad > \quad k_i \cdot |E_{OPT}| \tag{2}$$

Obviously $k_0 < m$, because there is a solution with $|E_{OPT}|$ edges, each of which has value at most m. For all intermediate steps it holds that there is at least one edge $e \in E_{OPT}$ of value $f_{G_i}(e) > k_i$, because otherwise $|E_{OPT}|$ would not be enough to merge all remaining LCCs. Due to the greedy selection process of Algorithm 2, it follows that the next edge chosen has at least the value $k_i + 1$, i.e.

$$f_{G_i}(e_i) \quad \geq \quad f_{G_i}(e) \quad \geq \quad k_i + 1 \tag{3}$$

For $l_i := |E_{OPT}|/(k_i + 1)$ it holds that $k_{i+l_i} < k_i$ which is proven by contradiction as follows. Assume that

$$k_{i+l_i} \quad \geq \quad k_i, \tag{4}$$

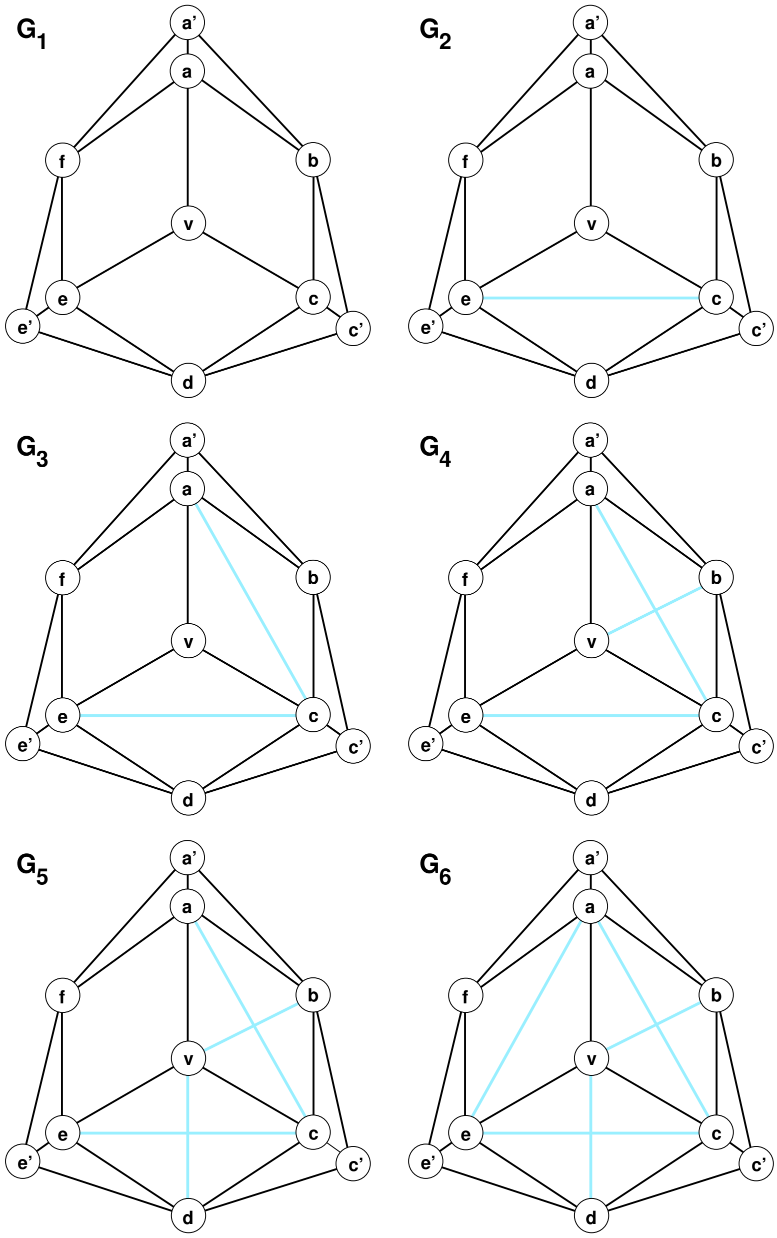

Fig. 12. Edge value deterioration example (all values with respect to G_1): One optimal solution for the LCWA instance G_1 is $E_{OPT} = \{\{a, e\}, \{a, c\}, \{c, e\}, \{b, v\}, \{d, v\}\}$. While all of these edges have value 2 in G_1, edge $\{a, e\}$ has value 1 in G_3, G_4 and G_5. Furthermore edge $\{f, v\}$ has value 2 in G_1, G_2 and G_3, value 1 in G_4 and value 0 in G_5 and G_6.

Table 11. Upper bounds for k_i

i	k_i
0	$\leq m-1$
$\frac{\lvert E_{OPT}\rvert}{m}$	$\leq m-2$
$\frac{\lvert E_{OPT}\rvert}{m} + \frac{\lvert E_{OPT}\rvert}{m-1}$	$\leq m-3$
$\frac{\lvert E_{OPT}\rvert}{m} + \frac{\lvert E_{OPT}\rvert}{m-1} + \frac{\lvert E_{OPT}\rvert}{m-2}$	$\leq m-4$
$\ldots$	$\ldots$
$\sum_{j=2}^{m} \frac{\lvert E_{OPT}\rvert}{j}$	≤ 0
$\sum_{j=1}^{m} \frac{\lvert E_{OPT}\rvert}{j}$	≤ -1

then

$$\begin{aligned}
k_i \cdot |E_{OPT}| &\overset{(4)}{\leq} k_{i+l_i} \cdot |E_{OPT}| \overset{(2)}{<} d_{i+l_i} \\
&\overset{(1)}{=} d_i - \sum_{j=i+1}^{i+l_i} f_{G_j}(e_j) \overset{(3),(4)}{\leq} d_i - l_i(k_i+1) \\
&= d_i - |E_{OPT}| \\
&\Rightarrow d_i > (k_i+1) \cdot |E_{OPT}|
\end{aligned}$$

However, this contradicts the maximality of k_i and therefore it follows that $k_{i+l_i} < k_i$. Starting from $k_0 < m$ the repeated application of this inequality yields upper bounds for the values k_i as given in Table 11. Since $k_s < 0$ for $s := |E_{OPT}| \cdot \sum_{j=1}^{m} \frac{1}{j}$, the number d_s of LCCs that remain to be merged after the addition of at most s edges equals 0 and Algorithm 2 terminates. □

Lemma 8. *Let $m \geq 2$ be a positive integer and $G = (V, E)$ an undirected graph without induced $K_{2,m}$, i.e. there is no subset $U \subseteq V$ such that $G|_U$ is isomorphic to the complete bipartite graph $K_{2,m}$. Then every complementary edge $e \in \overline{E}$ is at most $(m-1)$-valuable.*

Proof. Assume that there is a m-valuable complementary edge $e := \{x_1, x_2\} \in \overline{E}$. Let

$$(w_1, C_1), \ldots, (w_m, C_m) \in LCC_G(G) \setminus LCC_G((V, E \cup \{e\}))$$

be m LCCs in G that vanish by adding the edge $\{x_1, x_2\}$ such that all vertices w_i, $1 \le i \le m$ are pairwise distinct. These local connectivity components exist, because within the neighborhood of one vertex every edge can merge at most two connected components together. Then $G|_U$ for $U := \{x_1, x_2, w_1, \ldots w_m\}$ is isomorphic to $K_{2,m}$:

1. $x_1, x_2, w_1, \ldots w_m$ are pairwise distinct,
2. $\{x_1, x_2\} \notin E$,
3. $\forall 1 \le j \le m \quad \forall 1 \le i \le 2 : \qquad \{x_i, w_j\} \in E$ and
4. $\forall 1 \le j \le m \quad \forall 1 \le i \le m : \qquad \{w_i, w_j\} \notin E$

While the first three properties are obvious due to the definition of the x_i and w_j vertices, the forth property emerges from the following observation:

Assume that $\{w_i, w_j\} \in E$ for some indices $i \neq j$. Then there is a local connectivity component $(w_j, C) \in LCC_G(G)$ such that $x_1, x_2, w_i \in C$ and thus $(w_j, C) \in LCC_G((V, E \cup \{\{x_1, x_2\}\}))$. On the other hand $x_1, x_2 \in C_j$ by definition of (w_j, C_j), which implies $C = C_j$ and therefore contradicts the existence of (w_j, C_j). □

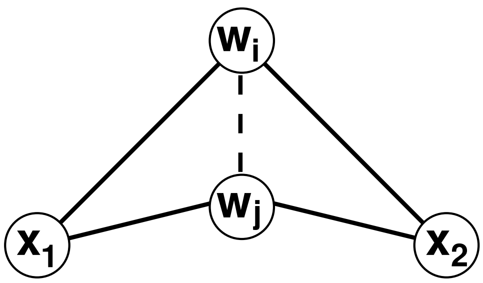

Fig. 13. Lemma 8: None of the edges $\{w_i, w_j\}$ is in E, because otherwise edge $\{x_1, x_2\}$ would not reduce the number of connected components within the neighborhoods of w_i and w_j.

Corollary 3. *Theorem 11 and Lemma 8 imply that Algorithm 2 is a constant factor approximation algorithm for the* LCWA *problem with instances restricted to undirected graphs that do not contain the $K_{2,m+1}$ as an induced subgraph for any constant positive integer $m \in \mathbb{N}$, which, for example, is the case for graphs with a constant maximum vertex degree. For $m = 2$, which holds for UDGs according to Lemma 1, the approximation factor is* 3/2 *and for $m = 3$ the approximation factor is* 11/6.

It is well known that the topology of real life wireless networks does not strictly obey the theoretical unit disk graph model [61] and therefore real networks might in fact contain the complete bipartite graph $K_{2,3}$ as an induced subgraph. However, the UDG model at least provides some indication that networks whose topology emerges from range assignments to points in the euclidean space are unlikely to contain the $K_{2,m}$ for arbitrarily large positive integers $m \in \mathbb{N}$.

Theorem 12. *Algorithm 2 is a* $1 + ln(\Delta(G))$ *factor approximation algorithm for the general* LCWA *problem.*

Proof. The maximum value of any complementary edge is bounded by the maximum vertex degree $\Delta(G)$ due to Lemma 8 and based on the upper bound shown in Theorem 11 the claim is proven by the following inequality.

$$\sum_{i=1}^{\Delta(G)} \frac{1}{i} \quad < \quad 1 + \int_1^{\Delta(G)} \frac{1}{x}\,\mathrm{d}x \quad = \quad 1 + ln(\Delta(G))$$

□

Theorem 13. *The upper bound of* $3/2$ *given in Theorem 11 for* $m = 2$ *is tight.*

Proof. Consider the graph $G = (V, E)$ given in Figure 14. For this example we have $|V| = 16$ and $|LCC(G)| = 20$, because the neighborhood of the vertices u, v, w, x induces two connected components each, while the neighborhood of all other vertices induces a connected subgraph. Any optimal solution for instance G of LCWA contains two edges, for example the 2-valuable edges $\{a, d\}$ and $\{c, f\}$. On the other hand, the edge $\{b, e\}$ is also 2-valuable in G and might therefore be chosen first by Algorithm 2. In this case the values of $\{a, d\}$ and $\{c, f\}$ decrease by one and there are no 2-valuable edges left, because there are no vertices adjacent to both u and x. Therefore the algorithm has to chose two more edges, which leads to $|E_A| = 3$, while $|E_{OPT}| = 2$. □

Analogously to the former case for $m = 2$, the upper bound for $m = 3$ is also tight as shown in the following Theorem. The presented worst case example is based on a very specific induced subgraph consisting of 62 vertices, making it highly unlikely to occur in real life networks.

Theorem 14. *The upper bound of* $3/2$ *given in Theorem 11 for* $m = 3$ *is tight.*

Proof. Consider the graph $G = (V, E)$ described in Figure 15. Analogously to the former example, the vertices a'_i and b'_j provide local connectivity for all

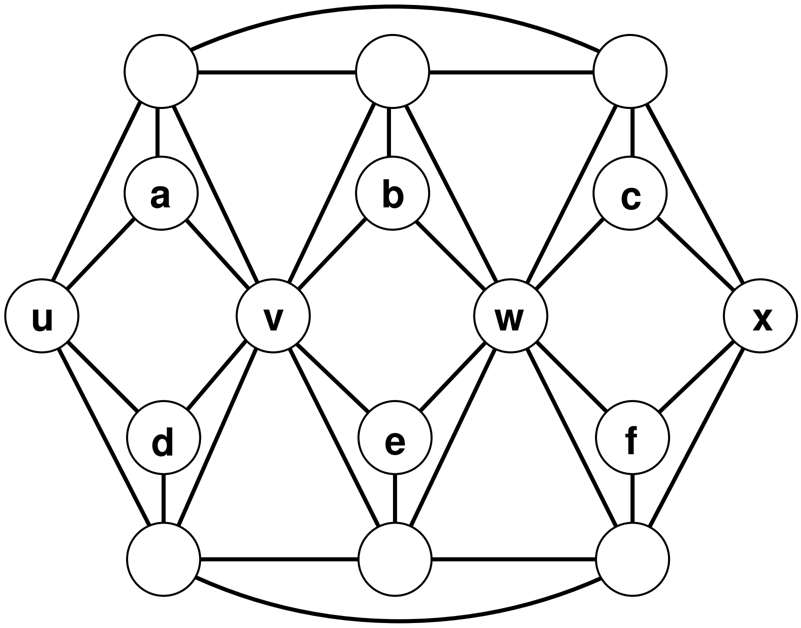

Fig. 14. Graph G, example for Theorem 13

vertices in G except the c_k vertices. Furthermore, for all $k \in \{1, \ldots 18\}$, the neighborhood $N_G(c_k)$ induces exactly two connected components of G.

One optimal solution for this instance of the LCWA problem is the following subset $E_{OPT} \subset \overline{E}$ of 3-valuable edges:

$$E_{OPT} := \{\{a_2, b_2\}, \{a_4, b_4\}, \{a_5, b_5\}, \{a_7, b_7\}, \{a_8, b_8\}, \{a_{10}, b_{10}\}\}$$

On the other hand, Algorithm 2 might choose edges as follows:

1. $\{a_3, b_3\}$, decreasing the values of $\{a_2, b_2\}$,$\{a_4, b_4\}$ and $\{a_7, b_7\}$ to 2,
2. $\{a_9, b_9\}$, decreasing the values of $\{a_5, b_5\}$,$\{a_8, b_8\}$ and $\{a_{10}, b_{10}\}$ to 2,
3. $\{a_1, b_1\}$, decreasing the values of $\{a_2, b_2\}$ and $\{a_4, b_4\}$ to 1,
4. $\{a_6, b_6\}$, decreasing the values of $\{a_5, b_5\}$ and $\{a_7, b_7\}$ to 1,
5. $\{a_{11}, b_{11}\}$, decreasing the values of $\{a_8, b_8\}$ and $\{a_{10}, b_{10}\}$ to 1.

Afterwards there are no more 2-valuable edges left and Algorithm 2 has to choose 6 more edges, for example the entire set E_{OPT}, which leads to a solution of cardinality 11, while the optimal solution given above has cardinality 6. □

The idea of the proofs for the Theorems 13 and 14 is based on the observation that the worst case examples for the following combinatorial problem provide a lower bound for the number of edges chosen by Algorithm 2:

Given a sequence $a_{0,1}, \ldots, a_{0,k}$ of $k \in \mathbb{N}$ non-negative integers, what is the largest integer $l \in \mathbb{N}$ such that there is an *elimination sequence of length* l, i.e. a sequence of l sequences satisfying the following properties?

1. $\forall 1 \leq j \leq k \quad \forall 1 \leq i \leq l : \qquad 0 \leq a_{i,j} \leq a_{i-1,j}$
2. $\forall 1 \leq i \leq l :$
   $\sum_{j=1}^{k} a_{i,j} \leq \sum_{j=1}^{k} a_{i-1,j} - max\{a_{i-1,j} \mid 1 \leq j \leq k\}$
3. $\exists 1 \leq j \leq k \quad a_{l-1,j} > 0$
4. $\forall 1 \leq j \leq k \quad a_{l,j} = 0$

With respect to the algorithm, the initial sequence $a_{0,1}, \ldots, a_{0,k}$ is given by arbitrarily ordering the edges of an optimal solution $e_1, \ldots, e_k$ and defining $a_{0,i}$ to be the value of e_i in the graph $(V, E \cup \{e_1, \ldots, e_{i-1}\})$ with respect to G. It directly follows that the maximum of this sequence is at most the maximum value of any edge in the given graph and also that $\sum_{i=1}^{k} a_{0,i} = |LCC_G(G)| - |V|$. Each time Algorithm 2 chooses an edge e'_j the current value of e'_j is subtracted from $\sum_{i=1}^{k} a_{j-1,i}$, because the addition of e'_j to the current graph reduces the number of local connectivity components by this value, which also reduces the overall values of $e_1, \ldots, e_k$, thereby defining the next sequence $a_{j,1}, \ldots, a_{j,k}$. Furthermore, the value of the next edge chosen by the algorithm is at least the maximum of the sequence $a_{j-1,1}, \ldots, a_{j-1,k}$ due to the greedy property. Finally, the algorithm computed a solution and terminates after having chosen l edges as soon as $a_{l,j} = 0$ for all $1 \leq j \leq k$.

There are instances for this combinatorial problem having

$$l \quad = \quad \left(\sum_{i=1}^{m} \frac{1}{i} \right) \cdot k$$

with m being the maximum of the sequence $a_{0,1}, \ldots, a_{0,k}$, see Table 12 for an example with $m = 4$ and $k = 12$. With respect to graphs, the worst case example given in Theorem 13 for $m = 2$ and the example of Theorem 14 for $m = 3$ correspond exactly to the analogous elimination sequences for $k = 2$ and $k = 6$, respectively.

Extrapolation of this concept in conjunction with Lemma 8 and the upper bound proven in Theorem 11 suggests the following conjecture.

Conjecture 2. Algorithm 2 has a tight quality bound of

$$\sum_{i=1}^{m} \frac{1}{i}$$

for undirected graphs G in which $m \in \mathbb{N}$ is the largest positive integer such that G contains the complete bipartite graph $K_{2,m}$ as an induced subgraph.

Table 12. Worst case example with maximum 4

$a_{0,1},\ldots,a_{0,12}$	$4,4,4,4,4,4,4,4,4,4,4,4$
$a_{1,1},\ldots,a_{1,12}$	$3,3,3,3,4,4,4,4,4,4,4,4$
$a_{2,1},\ldots,a_{2,12}$	$3,3,3,3,3,3,3,3,4,4,4,4$
$a_{3,1},\ldots,a_{3,12}$	$3,3,3,3,3,3,3,3,3,3,3,3$
$a_{4,1},\ldots,a_{4,12}$	$2,2,2,3,3,3,3,3,3,3,3,3$
$a_{5,1},\ldots,a_{5,12}$	$2,2,2,2,2,2,3,3,3,3,3,3$
$a_{6,1},\ldots,a_{6,12}$	$2,2,2,2,2,2,2,2,2,3,3,3$
$a_{7,1},\ldots,a_{7,12}$	$2,2,2,2,2,2,2,2,2,2,2,2$
$a_{8,1},\ldots,a_{8,12}$	$1,1,2,2,2,2,2,2,2,2,2,2$
$a_{9,1},\ldots,a_{9,12}$	$1,1,1,1,2,2,2,2,2,2,2,2$
$a_{10,1},\ldots,a_{10,12}$	$1,1,1,1,1,1,2,2,2,2,2,2$
$a_{11,1},\ldots,a_{11,12}$	$1,1,1,1,1,1,1,1,2,2,2,2$
$a_{12,1},\ldots,a_{12,12}$	$1,1,1,1,1,1,1,1,1,1,2,2$
$a_{13,1},\ldots,a_{13,12}$	$1,1,1,1,1,1,1,1,1,1,1,1$
$a_{14,1},\ldots,a_{14,12}$	$0,1,1,1,1,1,1,1,1,1,1,1$
$\ldots$	$\ldots$
$a_{25,1},\ldots,a_{25,12}$	$0,0,0,0,0,0,0,0,0,0,0,0$

Conjecture 2 is not formally proven at this point because of the rapidly increasing size and complexity of the necessary graphs for larger values of m. This results from the fact that the worst case example graph that has to be constructed for a specific value of m inevitably has to contain several copies of the worst case example for $m-1$. Based on this observation it should theoretically be possible to proof the conjecture using an inductive argument.

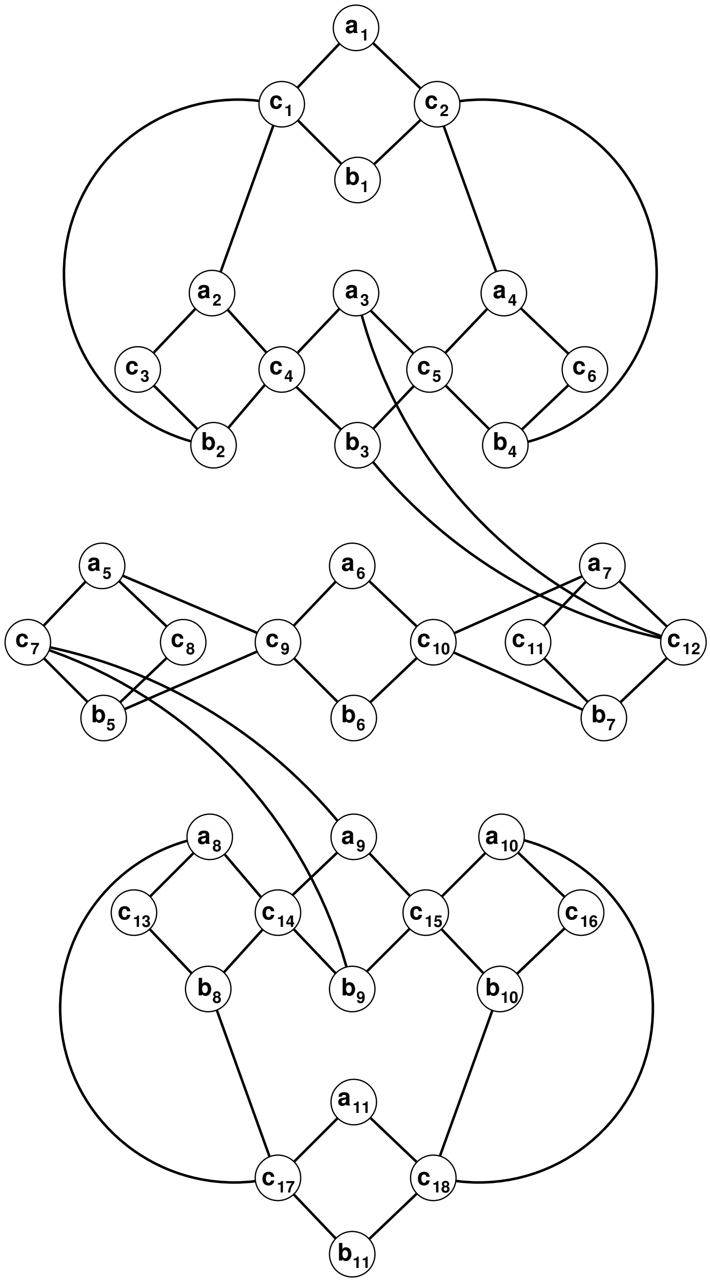

Fig. 15. Induced subgraph H used in the proof of Theorem 14: The entire graph G additionally contains, for every vertex a_i, one vertex a'_i that is adjacent to a_i as well as all vertices in $N_H(a_i)$ and for every vertex b_i, one vertex b'_i that is adjacent to b_i as well as all vertices in $N_H(b_i)$. Finally the vertex sets $\{a'_1, \ldots, a'_{11}\}$ and $\{b'_1, \ldots, b'_{11}\}$ induce complete subgraphs of G.

LOCALLY ***k*-CONNECTED STRONG AUGMENTATION**

In terms of sensor networks it is usually desirable to only utilize the additional communication links in case of a failure, because their usage is generally expensive in terms of power or others resources. This scenario is best represented by the LCWA problem above as it allows to utilize the additional edges for connectivity while providing the option to exclude them from other algorithms such as beaconing or watchdog protocols. On the other hand, if one insists on treating the additional edges as normal communication links for all algorithms in question, the following LCSA version is the more natural description of the problem.

LOCALLY k-CONNECTED STRONG AUGMENTATION

Given:	An undirected graph $G = (V, E)$ and a positive integer $m \in \mathbb{N}$
Question:	Is there a set of complementary edges $E' \subseteq \overline{E}$ with $\lvert E' \rvert \leq m$ such that $(V, E \cup E')$ is locally k-connected?

Theorem 15. *The* LOCALLY 1-CONNECTED STRONG AUGMENTATION *problem is NP-complete.*

Proof. LOCALLY k-CONNECTED STRONG AUGMENTATION is in NP for all positive integern $k \in \mathbb{N}$, because for a given set of complementary edges $E' \subseteq \overline{E}$ a deterministic algorithm can decide in polynomial time whether $H := (V, E \cup E')$ is locally k-connected by determining the connectivity of $G|_{N_H^1(v)}$ for each vertex $v \in V$.

The NP-hardness of LOCALLY 1-CONNECTED STRONG AUGMENTATION is shown by the following polynomial time reduction that constructs a graph G and an integer m from an arbitrary instance $(X, \mathcal{C})$ of SATISFIABILITY such that $(X, \mathcal{C})$ is satisfiable if and only if there is a set of complementary edges $E' \subseteq \overline{E}$ of cardinality $\lvert E' \rvert \leq m$ for which $(V, E \cup E')$ is locally connected.

Let $(X, \mathcal{C})$ be an instance for SATISFIABILITY with the set of variables X and the set of clauses $\mathcal{C}$. Define $m := \lvert X \rvert$ and construct G by adding a copy of a graph for every variable in X and every clause in $\mathcal{C}$:

For every variable $x_i \in X$ insert a circle with 4 vertices (V_i, E_i) with vertex set $V_i := \{x_{i,1}, x_{i,2}, \overline{x_{i,1}}, \overline{x_{i,2}}\}$ and edge set

$$E_i := \{\{x_{i,1}, \overline{x_{i,1}}\}, \{x_{i,1}, \overline{x_{i,2}}\}, \{x_{i,2}, \overline{x_{i,1}}\}, \{x_{i,2}, \overline{x_{i,2}}\}\}.$$

For every clause $c_j \in \mathcal{C}$ insert a path with three vertices (V'_j, E'_j) having $V'_j := \{c_{j,1}, c_{j,2}, c_{j,3}\}$ and $E'_i := \{\{c_{i,1}, c_{i,2}\}, \{c_{i,2}, c_{i,3}\}\}$.

These *variable gadgets* and *clause gadgets* are connected to each other by adding edges as follows: If clause $c_j \in \mathcal{C}$ contains literal x_i add $\{c_{j,2}, x_{i,1}\}$ and $\{c_{j,2}, x_{i,2}\}$, if c_j contains literal $\overline{x_i}$ add $\{c_{j,2}, \overline{x_{i,1}}\}$ and $\{c_{j,2}, \overline{x_{i,2}}\}$. In both cases add all edges $\{c_{j,1}, x_{i,1}\}$, $\{c_{j,1}, \overline{x_{i,1}}\}$, $\{c_{j,3}, x_{i,2}\}$, $\{c_{j,3}, \overline{x_{i,2}}\}$, see Figure 16 for an example.

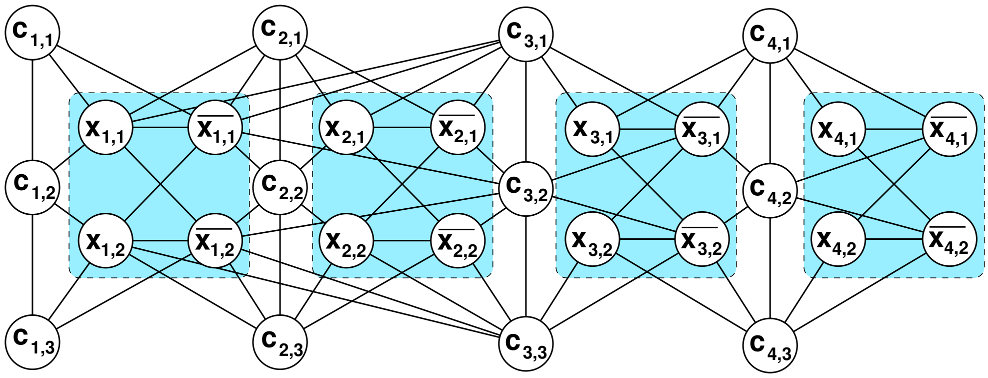

Fig. 16. Graph G constructed from instance $(X, \mathcal{C})$ with $X = \{x_1, x_2, x_3, x_4\}$ and $\mathcal{C} = \{\{x_1\}, \{\overline{x_1}, x_2\}, \{\overline{x_1}, \overline{x_2}, \overline{x_3}\}, \{\overline{x_3}, \overline{x_4}\}\}$

Lemma 9. *Let $H = (V_H, E_H)$ be an undirected graph and $E'_H \subseteq \overline{E_H}$ a set of complementary edges such that $(V_H, E_H \cup E'_H)$ is locally connected. Furthermore let $c \in V_H$ be a vertex that is not locally connected in H and $(c, \{v\})$ a local connectivity component. Then there is at least one edge $e \in E'_H$ such that $e \cap \{c, v\} \neq \emptyset$.*

Proof. Since c is locally connected in $(V_H, E_H \cup E'_H)$ and not locally connected in H, there is a second LCC (c, C), $C \neq \{v\}$ in H, but only one LCC (c, C') in $(V_H, E_H \cup E'_H)$ having $C \cup \{v\} \subseteq C'$. Therefore E'_H contains either one edge $e \in E'_H$ incident to v, connecting v to any vertex in $N^1_{(V_H, E_H \cup E'_H)}(c) \setminus \{v\}$ or one edge $e \in E'_H$ incident to c, extending the neighborhood of vertex c by a vertex v' for which $\{v, v'\} \in E_H$. □

The application of Lemma 9 to the graph G constructed for Theorem 15 provides the following Lemma which is used to conclude the proof:

Lemma 10. *Every complementary edge set E' for G of cardinality at most $|X|$ such that $(V, E \cup E')$ is locally connected contains either $\{x_{i,1}, x_{i,2}\}$ or $\{\overline{x_{i,1}}, \overline{x_{i,2}}\}$ for every variable gadget generated for variable $x_i \in X$.*

Proof. By the construction of G it follows, for every variable $x_i \in X$, that $(x_{i,1}, \{\overline{x_{i,2}}\})$ as well as $(\overline{x_{i,1}}, \{x_{i,2}\})$ are local connectivity components. Now Lemma 9 implies that E' contains at least one complementary edge incident to either $x_{i,1}$ or $\overline{x_{i,2}}$ and at least one complementary edge incident to either $\overline{x_{i,1}}$ or $x_{i,2}$. In conjunction with the bounded size $|E'| \leq |X|$ it follows that E' contains exactly one complementary edge incident to either $x_{i,1}$ or $\overline{x_{i,2}}$ and exactly one complementary edge incident to either $\overline{x_{i,1}}$ or $x_{i,2}$. Therefore we also know that E' does not contain any edges incident to vertices other than $x_{i,1}$, $\overline{x_{i,2}}$, $\overline{x_{i,1}}$ and $x_{i,2}$. It remains to show that E' does not contain any complementary edge e between vertices from different variable gadgets, which is shown by contradiction as follows: Due to symmetry we can assume without loss of generality that $e = \{x_{i,1}, x_{j,1}\}$ for two variables $x_i \neq x_j$. Since neither $x_{i,1}$ nor $\overline{x_{i,2}}$ is contained in another edge $e' \in E'$, $e' \neq e$ it follows that $(x_{i,1}, \{\overline{x_{i,2}}\})$ is a local connectivity component in $(V, E \cup E')$, because $x_{i,1}$ obviously has more than one neighbor in $(V, E \cup E')$. This however contradicts the assumption that $(V, E \cup E')$ is locally connected. $\square$

The proof of Theorem 15 is now concluded by showing that $(X, \mathcal{C})$ is satisfiable if and only if there is a set of complementary edges $E' \subseteq \overline{E}$ with $|E'| \leq m := |X|$ such that $(V, E \cup E')$ is locally connected.

$\Rightarrow$: Let $T : X \to \{0, 1\}$ be a satisfying truth assignment for $\mathcal{C}$. Define E' by adding one edge for every variable $x_i \in X$. If $T(x_i) = 1$, add $\{x_{i,1}, x_{i,2}\}$ and if $T(x_i) = 0$, add $\{\overline{x_{i,1}}, \overline{x_{i,2}}\}$. Then $(V, E \cup E')$ is locally connected: For every variable gadget (V_i, E_i) there is one edge $e \in E'$ between two vertices in V_i that are not adjacent in G. It is easy to verify that all vertices in V_i are locally connected due to e. Furthermore all vertices $c_{j,1}$ and $c_{j,3}$ are already locally connected in G and all vertices $c_{j,2}$ are locally connected, because T satisfies all clauses $c_j \in \mathcal{C}$: For every vertex $c_{j,2}$ the neighborhood $N^1(c_{j,2})$ induces a graph consisting of two connected components C and C'. C contains vertex $c_{j,1}$ as well as the vertices $x_{i,1}$ ($\overline{x_{i,1}}$) for all $x_i \in c_j$ ($\overline{x_i} \in c_j$) and C' contains vertex $c_{j,3}$ as well as the vertices $x_{i,2}$ ($\overline{x_{i,2}}$) for all $x_i \in c_j$ ($\overline{x_i} \in c_j$). Since T satisfies c_j, there is at least one i such that $x_i \in c_j$ and $\{x_{i,1}, x_{i,2}\} \in E'$ or $\overline{x_i} \in c_j$ and $\{\overline{x_{i,1}}, \overline{x_{i,2}}\} \in E'$ and therefore there is an edge $e \in E'$ such that $e \cap C \neq \emptyset$ and $e \cap C' \neq \emptyset$, meaning that $(c_{j,2}, C \cup C')$ is a local connectivity component in $(V, E \cup E')$, which implies that $c_{j,2}$ is locally connected in $(V, E \cup E')$.

$\Leftarrow$: Let E' be a set of complementary edges set of cardinality at most $|X|$ such that $(V, E \cup E')$ is locally connected. According to Lemma 10, E' consists of either $\{x_{i,1}, x_{i,2}\}$ or $\{\overline{x_{i,1}}, \overline{x_{i,2}}\}$ for every variable $x_i \in X$ and no other edges.

Define a truth assignment $T : X \to \{0, 1\}$ by setting $T(x_i) = 1$, if $\{x_{i,1}, x_{i,2}\} \in E'$ and $T(x_i) = 0$, if $\{\overline{x_{i,1}}, \overline{x_{i,2}}\} \in E'$. Then T satisfies $\mathcal{C}$: Because $(V, E \cup E')$ is locally connected, the vertex $c_{j,2}$ is locally connected for all clauses $c_j \in \mathcal{C}$, which implies that, for every vertex $c_{j,2}$, there is an index i such that either $\{x_{i,1}, x_{i,2}\} \in E'$ and $x_{i,1}, x_{i,2} \in N^1(c_{j,2})$ or $\{\overline{x_{i,1}}, \overline{x_{i,2}}\} \in E'$ and $\overline{x_{i,1}}, \overline{x_{i,2}} \in N^1(c_{j,2})$. In the former case $x_i \in c_j$ and $T(x_i)$ has been set to 1 and in the latter case $\overline{x_i} \in c_j$ and $T(x_i)$ has been set to 0, thus T satisfies c_j. □

Locally *k*-Connected Weak Vertex Augmentation

In contrast to the former augmentation problems, the next two problems aim for connectivity of the subgraph induced by the neighborhood of a single vertex $v \in V$ instead of connectivity of the subgraphs induced by the neighborhoods of all vertices. Analogously to the former problems, there is a Weak and a Strong version of this Vertex Augmentation problem.

Locally k-Connected Weak Vertex Augmentation

Given:	An undirected graph $G = (V, E)$, a vertex $v \in V$ and a positive integer $m \in \mathbb{N}$
Question:	Is there a set $E' \subseteq \overline{E}$ of cardinality $\lvert E' \rvert \leq m$ such that $N^1_G(v)$ induces a connected subgraph of $(V, E \cup E')$?

Theorem 16. *The* Locally 1-Connected Weak Vertex Augmentation *problem is solvable in linear time.*

Proof. If $G|_{N^1_G(v)}$ consists of more than one connected component, the only possibility to achieve connectivity in the Weak model is the addition of complementary edges between vertices from different connected components.

Let n be the number of connected components in $G|_{N^1_G(v)}$, which can be determined in time $\mathcal{O}(|V|+|E|)$ by a depth first search. Let H be the complete graph (e.g. the graph that contains all possible edges) that consists of one vertex for every connected component of $G|_{N^1_G(v)}$ and T be a spanning tree for H. The number of edges in T in the minimum number of complementary edges that have to be added to the graph such that the neighborhood $N^1_G(v)$ induces a connected subgraph. And because a tree with n vertices contains exactly $n-1$ edges, the instance can be accepted if $m \geq n-1$ and rejected otherwise. □

LOCALLY k-CONNECTED STRONG VERTEX AUGMENTATION

LOCALLY k-CONNECTED STRONG VERTEX AUGMENTATION	
Given:	An undirected graph $G = (V, E)$, a vertex $v \in V$ and a positive integer $m \in \mathbb{N}$
Question:	Is there a set $E' \subseteq \overline{E}$ of cardinality $\lvert E' \rvert \leq m$ such that v is locally k-connected in $(V, E \cup E')$?

Theorem 17. *The* LOCALLY 1-CONNECTED STRONG VERTEX AUGMENTATION *problem is NP-complete.*

Proof. LOCALLY k-CONNECTED STRONG VERTEX AUGMENTATION is in NP for all $k \in \mathbb{N}$, because for a given set of complementary edges $E' \subseteq \overline{E}$ a deterministic algorithm can decide in polynomial time whether v is locally k-connected in $(V, E \cup E')$.

The NP-hardness is shown by the following polynomial time reduction that transforms an instance $(X, \mathcal{C})$ of SATISFIABILITY into an undirected graph $G = (V, E)$ with a designated vertex $v \in V$ and a positive integer $m := \lvert X \rvert$ such that $(X, \mathcal{C})$ is satisfiable if and only if there is a subset $E' \subseteq \overline{E}$ of cardinality $\lvert E' \rvert \leq m = \lvert X \rvert$ for which v is locally connected in $(V, E \cup E')$.

G contains two vertices s and v and the edge $\{s, v\}$ as well as one copy of the graph

$$G_x \quad := \quad (\{x, \hat{x}, \overline{x}\}, \{\{x, \hat{x}\}, \{\hat{x}, \overline{x}\}\})$$

and the edges $\{s, x\}$, $\{s, \overline{x}\}$, $\{v, \hat{x}\}$ for every variable $x \in X$. G also contains a vertex c for every clause $c \in \mathcal{C}$ and the edges $\{\{c, x\} \mid x \in c\}$, $\{\{c, \overline{x}\} \mid \overline{x} \in c\}$. See Figure 17 for an example. Formally, G is defined by the following sets.

$$\begin{aligned}
V := \;& \{s, v\} \quad \cup \quad \{x, \hat{x}, \overline{x} \mid x \in X\} \quad \cup \quad \mathcal{C} \\
E := \;& \{\{s, v\}\} \quad \cup \quad \{\{s, x\}, \{s, \overline{x}\} \mid x \in X\} \\
& \cup \quad \{\{\hat{x}, x\}, \{\hat{x}, \overline{x}\} \mid x \in X\} \quad \cup \quad \{\{v, \hat{x}\} \mid x \in X\} \\
& \cup \quad \{\{c, x\} \mid x \in X \wedge x \in c\} \quad \cup \quad \{\{c, \overline{x}\} \mid x \in X \wedge \overline{x} \in c\}
\end{aligned}$$

Then $(X, \mathcal{C})$ is satisfiable if and only if there is a set $E' \subseteq \overline{E}$ containing at most $m = \lvert X \rvert$ complementary edges such that v is locally connected in $H := (V, E \cup E')$:

$\Rightarrow$: Let $T : X \to \{0, 1\}$ be a satisfying truth assignment for $\mathcal{C}$. Define

$$E' \quad := \quad \{\{v, x\} \mid T(x) = 1\} \quad \cup \quad \{\{v, \overline{x}\} \mid T(x) = 0\}.$$

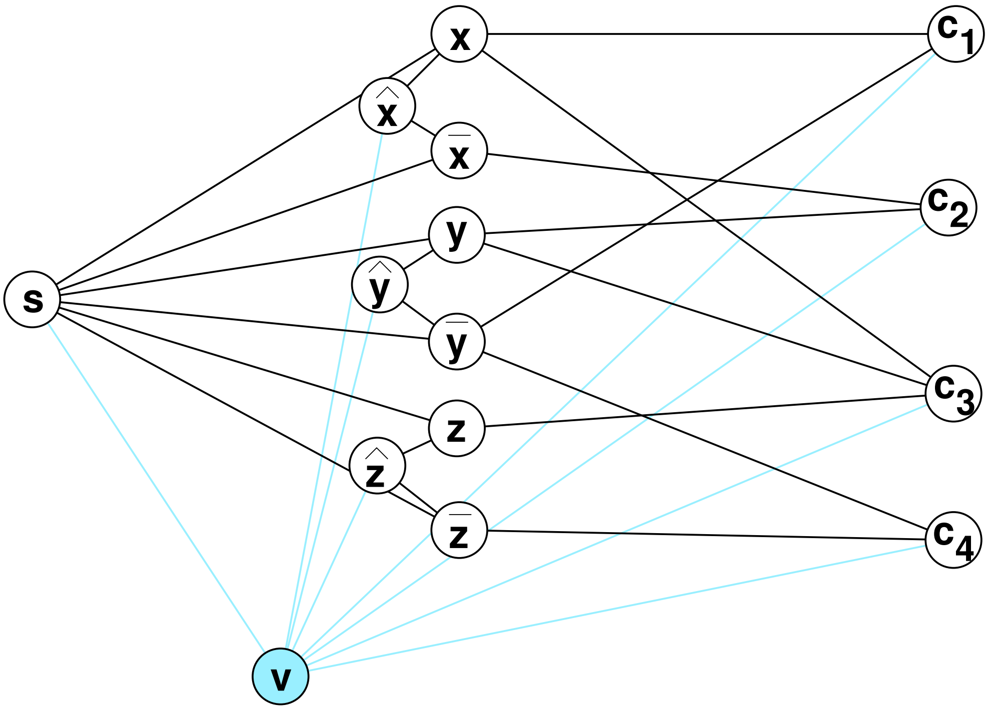

Fig. 17. Graph G constructed from instance $(X, \mathcal{C})$ with $X = \{x, y, z\}$ and $\mathcal{C} = \{c_1, c_2, c_3, c_4\}$, $c_1 = \{x, \overline{y}\}$, $c_2 = \{\overline{x}, y\}$, $c_3 = \{x, y, z\}$, $c_4 = \{\overline{y}, \overline{z}\}$.

Then v is locally connected in H, because for every vertex $w \in N_H^1(v)$ there is a path p_w between w and s that only contains vertices from $N_H^1(v)$: If $w = \hat{x}$ for some variable $x \in X$, then either $x \in N_H^1(v)$ or $\overline{x} \in N_H^1(v)$ and thus either $p_w = w, x, s$ or $p_w = w, \overline{x}, s$. If $w = c$ for some clause $c \in \mathcal{C}$, then there is a literal $l \in c$ such that $\{v, l\} \in E'$, because T satisfies c and thus $p_w = w, l, s$. Finally, for $w = x$ or $w = \overline{x}$, $p_w = w, s$, because $\{w, s\} \in E$.

$\Leftarrow$: Let $E' \subseteq \overline{E}$ be a set of complementary edges of cardinality $|E'| \leq m = |X|$ such that v is locally connected in $(V, E \cup E')$.

It can be assumed without loss of generality that E' does not contain any edge $\{\hat{x}, \hat{y}\}$ for two variables $x, y \in X$: If $\{\hat{x}, \hat{y}\} \in E'$ for two distinct variables $x, y \in X$ and v is not locally connected in $J := (V, E \cup E' \setminus \{\{\hat{x}, \hat{y}\}\})$, then $J|_{N_J^1(v)}$ consists of exactly two connected components. Let C_x and C_y be the two connected components of $J|_{N_J^1(v)}$ that contain vertex $\hat{x}$ and $\hat{y}$, respectively. Without loss of generality let C_x be the connected component that contains vertex s. Then there is no path between $\hat{y}$ and s in $J|_{N_J^1(v)}$, meaning that E' contains neither $\{v, y\}$ nor $\{v, \overline{y}\}$. Then $E'' := (E' \setminus \{\{\hat{x}, \hat{y}\}\}) \cup \{\{v, y\}\}$ is a set of complementary edges such that $|E''| = |E'|$ and v is locally connected in $(V, E \cup E'')$.

$H|_{N_H^1(v)}$ is connected, therefore there is a path between $\hat{x}$ and s in $H|_{N_H^1(v)}$ for all variables $x \in X$. Since $\{\hat{x}, \hat{y}\} \notin E'$ for all $x, y \in X$, it follows that for every variable $x \in X$ at least one of the following three conditions holds true.

1. $\{v, x\} \in E' \quad \Leftrightarrow \quad x \in N_H^1(v)$
2. $\{v, \overline{x}\} \in E' \quad \Leftrightarrow \quad \overline{x} \in N_H^1(v)$
3. $\{\hat{x}, u\} \in E'$ for some vertex $u \in V : \forall y \in X : u \neq \hat{y}$

Utilizing an argument similar to the one before we can now assume without loss of generality that there is no variable $x \in X$ for which the third condition holds: If $\{\hat{x}, u\} \in E'$ for some variable $x \in X$, then $E'' := (E' \setminus \{\{\hat{x}, u\}\}) \cup \{\{v, x\}\}$ is a set of complementary edges such that $|E''| = |E'|$ and v is locally connected in $(V, E \cup E'')$.

In conjunction with the fact that $|E'| \leq |X|$ this implies that for every variable $x \in X$ either the first or the second condition holds true, which means that $|E'| = |X|$. This allows the definition of a satisfying truth assignment T as follows:

Define $T : X \to \{0, 1\}$ by setting $T(x) = 1$ if $\{v, x\} \in E'$ and $T(x) = 0$ otherwise. Then T satisfies all clauses $c \in \mathcal{C}$: Because $H|_{N_H^1(v)}$ is connected, there is a path $c, l, \ldots, s$ between vertex c and vertex s. Since E' does not contain any edges incident to c, vertex l has to be a vertex corresponding to a literal that occurs in c and thus $\{v, l\} \in E'$. If $l = x$ is the positive literal of a variable $x \in X$, then $T(x)$ has been set to 1, otherwise $T(x)$ has been set to 0 and therefore T satisfies c. $\square$

3.3 Edge Removal Problems

After having considered topology control problems that aim to achieve local connectivity, this section is concerned with a type of problems that try to preserve local connectivity while removing as many edges as possible from a given graph to reduce the number of communication links that the individual sensor nodes have to maintain. Removing edges from the communication graph of a wireless network while preserving a specific graph property has been considered for several graph properties such as planarity or a specified minimum vertex degree, see for example [66, 82] for an overview of this topic. Due to the nature of local connectivity, there are two versions of the problem just as in the case of graph augmentation: While the Weak version of this graph thinning approach aims for connectivity in the subgraphs induced by the original neighborhoods of each vertex, the Strong version asks for a set of edges such that the remaining graph is locally connected. Note that in the strong version of the problem the remaining graph is additionally required to be connected, because otherwise it would theoretically be possible to remove all edges based on the observation that isolated vertices are locally connected. It will be shown that both versions of this Graph Thinning problem are NP-complete. Since removing edges from a graph is a somewhat opposite task of adding edges to a graph, these problems are closely related to the augmentation problems considered above and therefore the NP-hardness can be proven by making suitable modifications to the polynomial time reduction from Theorem 10.

Locally ***k***-Connected Weak Graph Thinning

	Locally k-Connected Weak Graph Thinning
Given:	A locally k-connected graph $G = (V, E)$ and a positive integer $m \in \mathbb{N}$.
Question:	Is there a set $E' \subseteq E$ of cardinality $\lvert E' \rvert \geq m$ such that, for every vertex $v \in V$, the neighborhood $N^1_G(v)$ induces a k-connected subgraph of $(V, E \setminus E')$?

Theorem 18. *The* Locally 1-Connected Weak Graph Thinning *problem is NP-complete.*

Proof. Locally k-Connected Weak Graph Thinning is in NP for all $k \in \mathbb{N}$, because for a given set of edges $E' \subseteq E$ a deterministic algorithm can decide in polynomial time whether the neighborhood $N^1_G(v)$ induces a k-connected subgraph of $(V, E \setminus E')$ for all vertices $v \in V$.

The NP-hardness of Locally 1-Connected Weak Graph Thinning is shown by a polynomial time reduction from Satisfiability that transforms an instance $(X, \mathcal{C})$ into a locally connected graph $G = (V, E)$ and a positive integer $m \in \mathbb{N}$ such that $(X, \mathcal{C})$ is satisfiable if and only if there is a set $E' \subseteq E$ of cardinality $\lvert E' \rvert \geq m$ such that, for every vertex $v \in V$, the neighborhood $N^1_G(v)$ induces a connected subgraph of $(V, E \setminus E')$.

For this purpose the reduction from Theorem 10 is extended as follows: Let $H = (V_H, E_H)$ be the graph that is constructed in the proof of Theorem 10 for instance $(X, \mathcal{C})$. Define $G = (V, E)$ by $V := V_H \cup \{v_e \mid e \in E_H\}$ and $E := E_H \cup \{\{v_e, w_1\}, \{v_e, w_2\} \mid e = \{w_1, w_2\} \in E_H\} \cup \{\{u_i, f_i\}, \{u'_i, t_i\} \mid x_i \in X\}$. In other words, an additional vertex v_e is generated for every edge $e \in E_H$ and connected to both end vertices of e. Afterwards all edges of type $\{u_i, f_i\}$

and $\{u_i', t_i\}$ are added to G. Integer m is set to the number of variables $|X|$ just as in Theorem 10.

It directly follows from the proof of Theorem 10 and the following two observations that G is locally connected:

1. All newly added vertices are obviously locally connected and they are only adjacent to two vertices, which are also adjacent to each other. Therefore the vertices v_e do not generate additional connected components within the neighborhood of any vertex.
2. The inclusion of the $\{u_i, f_i\}$ and $\{u_i', t_i\}$ edges alters the neighborhood of their incident vertices, but does not introduce any new connected components, because all of these vertices are also adjacent to the respective x_i vertex.

The proof is now concluded by showing that $(X, \mathcal{C})$ is satisfiable if and only if there is a set $E' \subseteq E$ of cardinality $|E'| \geq m$ such that, for every vertex $v \in V$, the neighborhood $N_G^1(v)$ induces a connected subgraph of $(V, E \setminus E')$:

$\Rightarrow$: Let $T : X \to \{0,1\}$ be a satisfying truth assignment for $(X, \mathcal{C})$. Set $E' := \{\{u_i, f_i\} \mid x_i \in X \wedge T(x_i) = 1\} \cup \{\{u_i', t_i\} \mid x_i \in X \wedge T(x_i) = 0\}$. Obviously, $|E'| = |X| = m$ and, for every vertex $v \in V$, the neighborhood $N_G^1(v)$ induces a connected subgraph of $(V, E \setminus E')$ due to the same arguments as in the proof of Theorem 10.

$\Leftarrow$: Let E' be a set of edges with $|E'| \geq m$, such that, for every vertex $v \in V$, the neighborhood $N_G^1(v)$ induces a connected subgraph of $(V, E \setminus E')$. First, we can observe that E' does not contain any edge $e \in E_H$, because otherwise $(V, E \setminus E')|_{N_G^1(v_e)}$ would not be connected. Furthermore we can also observe that E' does not contain any of the edges $\{v_e, w_1\}$ or $\{v_e, w_2\}$ that are generated for an edge $e = \{w_1, w_2\} \in E_H$: If $\{v_e, w_1\} \in E'$ for any edge $e \in E_H$, then the graph $(V, E \setminus E')|_{N_G^1(w_2)}$ is not connected (analogous argument for $\{v_e, w_2\}$). Therefore E' only contains edges of type $\{u_i, f_i\}$ or $\{u_i', t_i\}$ and because $(V, E \setminus E')|_{N_G^1(x_i)}$ is connected for all vertices x_i, it follows that $|E'| = |X| = m$ and that E' contains exactly one of the edges $\{u_i, f_i\}$ or $\{u_i', t_i\}$ for all $x_i \in X$. Due to the same arguments as in the proof of Theorem 10, this implies the satisfiability of $(X, \mathcal{C})$, because for all clauses $c_j \in \mathcal{C}$ the neighborhood of vertex c_j induces a connected subgraph of $(V, E \setminus E')$. $\square$

LOCALLY ***k***-CONNECTED STRONG GRAPH THINNING

LOCALLY k-CONNECTED STRONG GRAPH THINNING	
Given:	A connected, locally k-connected graph $G = (V, E)$ and a positive integer $m \in \mathbb{N}$.
Question:	Is there a set $E' \subseteq E$ of cardinality $\|E'\| \geq m$ such that $(V, E \setminus E')$ is connected and locally k-connected?

Theorem 19. *The* LOCALLY 1-CONNECTED STRONG GRAPH THINNING *problem is NP-complete.*

Proof. LOCALLY k-CONNECTED STRONG GRAPH THINNING obviously belongs to NP for all positive integers $k \in \mathbb{N}$ due to the same arguments as in the former theorems. As for the NP-hardness, the reduction of Theorem 18 can be used.

Let $G = (V, E)$ be the graph generated for instance $(X, \mathcal{C})$. Obviously G is connected, because the vertices b_1 and b_2 are adjacent to at least one vertex from each of the subgraphs created for the variables and clauses in $(X, \mathcal{C})$. Then $(X, \mathcal{C})$ is satisfiable if and only if there is a set $E' \subseteq E$ of cardinality $|E'| \geq m := |X|$ such that $(V, E \setminus E')$ is connected and locally connected:

$\Rightarrow$: Analogous to the proof of Theorem 18 due to the following observation: The alteration of the neighborhoods due to the removal of an edge $\{u_i, f_i\}$ or $\{u_i', t_i\}$ generated for a variable $x_i \in X$ is irrelevant for the local connectivity, because of the corresponding x_i vertex that is adjacent to both end vertices of the edge.

$\Leftarrow$: Let $E' \subseteq E$ be a set of cardinality $|E'| \geq m = |X|$ such that $(V, E \setminus E')$ is connected and locally connected. Utilizing the observation above we already know from the proof of Theorem 18 that $(X, \mathcal{C})$ is satisfiable, if there are $m = |X|$ edges of the type $\{u_i, f_i\}$ or $\{u_i', t_i\}$ whose removal will result in a locally connected graph and it is also easy to verfiy that this graph is connected. Therefore it only remains to show that E' cannot contain any edges other than $\{u_i, f_i\}$ or $\{u_i', t_i\}$ for a variable $x_i \in X$ to proof the theorem.

This will be proven by contradiction. Let $e \in E'$ be an edge such that $\forall i \in \{1, \ldots, |X|\} : e \notin \{\{u_i, f_i\}, \{u_i', t_i\}\}$. Then one of the following two cases applies.

Case 1. A vertex of degree 2 that is adjacent to both vertices in e has been generated, meaning that $\exists v_e \in V : N_G^1(v_e) = e$. Let u and v be the two vertices incident to e, i.e. $e = \{u, v\}$. Now it follows that either $\{v_e, u\} \in E'$ or $\{v_e, v\} \in$

E', because otherwise v_e would not be locally connected in $(V, E\setminus E')$. Without loss of generality let $\{v_e, v\} \in E'$. If it also holds that $\{v_e, u\} \in E'$, then v_e is isolated in $(V, E \setminus E')$, which contradicts the connectivity of $(V, E \setminus E')$. Therefore $\{v_e, u\} \notin E'$ which leads to $(u, \{v_e\})$ being a local connectivity component of $(V, E\setminus E')$. In conjunction with $(V, E\setminus E')$ being locally connected we know that v_e is the only connected component induced by the neighborhood $N^1_{(V,E\setminus E')}(u)$ of u, implying that E' contains all edges incident to u except for $\{u, v_e\}$. Again, this contradicts the assumption that $(V, E \setminus E')$ is connected.

Case 2. Edge e is incident to a vertex $v_{e'}$ of degree 2 that has been generated for an edge e' and $N^1_G(v_{e'}) = e'$. Let u and v be the two vertices incident to e' and without loss of generality let $e = \{v_{e'}, v\}$. Then $\{v_{e'}, u\} \notin E'$, because $v_{e'}$ is not isolated in $(V, E \setminus E')$. But then $e' \in E'$, because u is locally connected in $(V, E \setminus E')$. Therefore *Case 1* is applicable to edge e'.

□

3.4 Complexity Results for *k*-Connectivity

In this section it is briefly discussed how all of the NP-hardness results from the sections 3.1 and 3.2 for $k = 1$ can be extended to arbitrary k using the same basic idea:

Adding an additional vertex v to the graph that is adjacent to all other vertices yields a polynomial time reduction

$$k - \Pi \qquad \leq_p \qquad (k+1) - \Pi,$$

assuming that the following condition holds:

($\star$) Every solution for the construction version of $k - \Pi$ results in a $(k+1)$-connected graph.

This $\star$-condition is mandatory to ensure that the required local connectivity condition automatically holds for the newly added vertex v. The correctness of these reductions is then given by the following observation.

Definition 17. *For an undirected graph $G = (V, E)$ let $\gamma(G)$ be the* connectivity *of G, meaning the largest positive integer $k \in \mathbb{N}$ such that G is k-connected.*

Observation 2 *Let $G = (V, E)$ be an undirected graph and*

$$G^+ \quad := \quad (V \cup \{v\}, E \cup \{\{v, u\} \mid u \in V\})$$

the graph with one additional vertex v that is adjacent to all other vertices. Then, for every subset of vertices $U \subseteq V$,

$$\gamma(G^+|_{U\cup\{v\}}) \quad = \quad \gamma(G|_U) + 1.$$

Using the NP-hardness results for $k = 1$ above as starting points for inductive arguments, Observation 2 also provides validity of the $\star$-condition for arbitrary k, if it holds for $k = 1$. Therefore the only thing that remains to show is that the $\star$-condition holds for $k = 1$. For the MAXIMUM LOCALLY k-CONNECTED VERTEX SET problem this is given by the following lemma.

Lemma 11. *Let $G = (V, E)$ be an undirected, connected graph that is locally connected. Then G is 2-connected.*

Proof. Assume that G is not 2-connected. Then there is a separation vertex $w \in V$, meaning that $G|_{V\setminus\{w\}}$ is not connected. Let $u, v \in V$ be a pair of vertices for which there is no path in $G|_{V\setminus\{w\}}$. Since G is connected there is path $u, \ldots, u', w, v', \ldots v$ in G. Because w is locally connected there is a path p between u' and v' in $G|_{N^1(w)}$ (which is a subgraph of $G|_{V\setminus\{w\}}$) and thus $u, \ldots, p, \ldots, v$ is a path between u and v in $G|_{V\setminus\{w\}}$. □

For the augmentation problems in section 3.2 it is sufficient to show that the problem remains NP-hard when the instances are restricted to 2-connected graphs, because augmenting a graph G by adding additional edges does not reduce the connectivity of G. For all presented reductions it is easy to verify that the constructed graph is 2-connected, if the reduction is started from CONNECTED SAT.

4 Connectivity of Networks with Two Power Levels

Wireless networks with adjustable transmission power levels have received considerable attention in literature as it is related energy management and therefore has direct impact on the lifetime of the network, because the replacement of batteries in a large scale network is practically difficult or even impossible.

There are several studies on the so-called RANGE ASSIGNMENT (RA) problem with different preliminaries, for example [21, 37]The RA problem is typically defined as computing a range assignment $f : P \to \mathbb{R}^+$ for a set of points $P \subset \mathbb{R}^n$ $(1 \leq n \leq 3)$, representing the nodes in the network, such that the total energy $\sum_{p\in P} c(f(p))$ is minimal (c being a cost function according to a radio wave propagation model) under the constraint that the graph (P, E), $E := \{(p, q) \in P^2 \mid \|p - q\|_2 \leq f(p)\}$ is strongly connected, where $\|p - q\|_2$ denotes the Euclidean distance between p and q. It is well known that the corresponding decision problem is NP-hard for both the 2- and 3-dimensional euclidean space [60, 22]. Variations of the problem include requiring connectivity of the resulting undirected graph (where an edge $\{u, v\}$ is present if and only if both directed edges (u, v) and (v, u) are present) instead of strong connectivity [9, 12] and restricting the possible power level values from real number to a discrete space [27] or limiting the number of power levels to two [13]. Most recently we introduced a range assignment problem that demands a minimum node degree for all nodes in the network [43].

The connectivity problem considered in this chapter is motivated by wireless sensor networks in which the nodes have two transmission power levels, a *min-power* and a *max-power level*. Such networks are usually represented as directed graphs in that the directed edges represent connections from source nodes to target nodes. Again, since almost all communication protocols are based on symmetric connections, we are mainly interested in the underlying symmetric networks that are represented by undirected graphs.

Our goal is to find a minimum number k of nodes such that if these k nodes use max-power and the remaining nodes use min-power then the resulting underlying symmetric network is connected.

4.1 2-Level Symmetric Range Assignment

The problem is now defined more formally for both the symmetric and the asymmetric case. For a node $u \in V$, let $d_{\min}(u) \subseteq V$ and $d_{\max}(u) \subseteq V$ be the set of nodes reachable from u with min-power or max-power, respectively. That is, $d_{\min}$ and $d_{\max}$ can be considered as mappings from V to the power set $\mathcal{P}(V)$ of V, i.e., to the set of all subsets of V.

Definition 18. *For a finite set V, a subset $U \subseteq V$ and two mappings $d_{\min} : V \to \mathcal{P}(V)$, $d_{\max} : V \to \mathcal{P}(V)$ with $d_{\min}(v) \subseteq d_{\max}(v) \quad \forall v \in V$ the set of* symmetric min-power edges *is defined by*

$$E_{\min}(U) := \{\{u, v\} \mid u, v \in U \wedge u \neq v \wedge v \in d_{\min}(u) \wedge u \in d_{\min}(v)\},$$

and the set of symmetric max-power edges *by*

$$E_{\max}(U) := \{\{u, v\} \mid u, v \in U \wedge u \neq v \wedge v \in d_{\max}(u) \wedge u \in d_{\max}(v)\}.$$

Furthermore define the min-max-power graph

$$G(V, d_{\min}, U, d_{\max}) \quad := \quad (V, E_{\min}(V) \cup E_{\max}(U)).$$

For the asymmetric case define the set of directed min-power edges *as*

$$E^{+}_{\min}(U) := \{(u, v) \mid u \in U \wedge v \in d_{\min}(u) \wedge u \neq v\},$$

the set of directed max-power edges *as*

$$E^+_{\max}(U) := \{(u,v) \ \mid\ u \in U \wedge v \in d_{\max}(u) \wedge u \neq v\}$$

and the directed min-max-power graph

$$G^+(V, d_{\min}, U, d_{\max}) \quad := \quad (V, E^+_{\min}(V) \cup E^+_{\max}(U)).$$

The min-max-power graph $G(V, d_{\min}, U, d_{\max})$ is also denoted by $G(U)$ if V, $d_{\min}$, and $d_{\max}$ are known from the context. Graph $G(\emptyset)$ is also called the *min-power graph* whereas graph $G(V)$ is also called the *max-power graph*. Analogously the directed min-max-power graph $G^+(V, d_{\min}, U, d_{\max})$ is also denoted by $G^+(U)$.

If the min-power mapping $d_{\min}$ is symmetric, meaning that $\forall u, v \in V$: $v \in d_{\min}(u) \Leftrightarrow u \in d_{\min}(v)$, the min-max-power graph $G(U)$ represents the underlying symmetric network for the case that the nodes of U use max-power and the remaining nodes of $V \setminus U$ use min-power. This kind of symmetry is automatically given, for example, if the mapping $d_{\min}$ corresponds to the vertex adjacency given by the UDG model for a given range as described in the following definition.

Definition 19. *For a finite set $P \subset \mathbb{R}$ and a real number $r \in \mathbb{R}$ define*

$$d_r(v) \quad := \quad \{u \in P \setminus \{v\} \quad \mid \quad \|v - u\|_2 \leq r\}.$$

The definition of the min-max-power graph is used to define the 2-Level Symmetric Range Assignment problem as follows.

2-Level Symmetric Range Assignment (2LSRA)

Given: A node set V and two mappings $d_{\min} : V \to \mathcal{P}(V)$ and $d_{\max} : V \to \mathcal{P}(V)$ such that the max-power graph $G(V)$ is connected.
Task: Compute a set $U \subseteq V$ of minimum cardinality such that the min-max-power graph $G(U)$ is connected.

While formulated in an entirely different way, the 2LSRA problem is basically equivalent to the problem called Max Power Users given in [67], which was renamed to $\{0,1\}$-MPST by the authors of [76]. And although this work has been conducted independently of [67], the concept used for an approximation algorithm is remarkably similar: In this chapter a family of approximation algorithms, called Approx2LSRA$_k$ and based on a positive integer $k \in \mathbb{N}$, is developed and analyzed. The authors of [67] introduce two approximation algorithms which are in fact equivalent to our special cases Approx2LSRA$_3$ and Approx2LSRA$_4$. However, the best known approximation ratio possibly for this problem is given in [76] where the authors prove that the algorithm presented in [3] for the more general problem of arbitrary power levels achieves an approximation ratio of 3/2 for the special case of two power levels. It is noteworthy that the algorithm given in [3] is based on a rather complex approximation scheme for the classical Steiner Tree problem, while the idea used in this chapter and in [67] is a simple, fast and easy to implement greedy approach.

We will start by proving the NP-completeness in the following section, before a family of approximation algorithm called Approx2LSRA$_k$ based on a positive integer $k \in \mathbb{N}$ is introduced and analyzed.

4.2 NP-Completeness

The decision problem that corresponds to 2LSRA, i.e. the problem of finding a vertex set U of size at most k, for an additionally given integer k, such that $G(U)$ is connected, is NP-complete. The problem obviously is in NP, because deciding whether the min-max-power graph $G(U)$ is connected for a given set U can be accomplished by a depth first search on the graph $G(U)$ in linear time. The NP-hardness will be shown in this section by demonstrating that the following restriction to the problem already is NP-hard. In this euclidean version of the two level range assignment problem the mappings $d_{\min}$ and $d_{\max}$ are not arbitrary, but implicitly given by the pairwise distances between vertex positions in the 2-dimensional euclidean space as given in Definition 19. One can think of this model as an extension to the UDG model in which a second

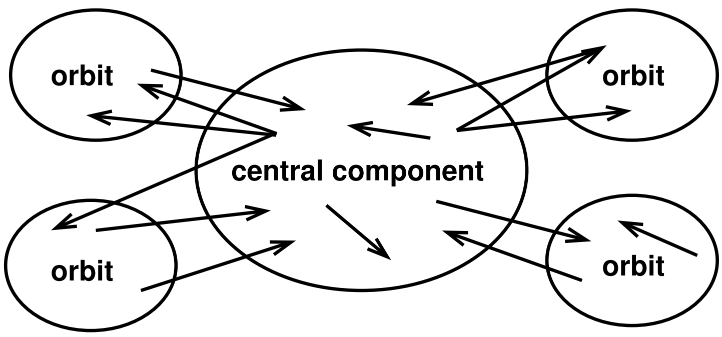

Fig. 18. 2LERAS: Several orbit components around one central component yield a star topology. The arrows indicate possible directed max-power edges.

max-power radius is given. The 2LSERA problem then asks for a minimum number of points from a given set for which the larger radius has to be used in order to obtain a connected graph.

2-Level Symmetric Euclidean Range Assignment (2LSERA)	
Given:	A finite set $P \subset \mathbb{R}^2$, a positive integer $k \in \mathbb{N}$ and two ranges $r_1, r_2 \in \mathbb{R}$, $r_1 < r_2$ such that the max-power graph $G(P)$ is connected for the mappings $d_{\text{min}} = d_{r_1}$ and $d_{\text{max}} = d_{r_2}$.
Question:	Is there a set $U \subseteq P$ of cardinality $\lvert U \rvert \leq k$ such that the min-max-power graph $G(U)$ is connected?

The main part on this proof is given in the proof of Theorem 6.1 of [13], where the authors show that the following 2LERAS problem is NP-complete. 2LERAS basically is the asymmetric version of the 2LSERA problem above with an additional restriction to star topologies, meaning that the strongly connected components of the directed min-power graph $(P, E^+_{\text{min}}(P))$ consist of one central component induced by a vertex set C and $m \in \mathbb{N}$ so-called orbit components induced by vertex sets $V_1, \ldots, V_m$ satisfying the following property:

For all directed max-power edges (s, t) (that are not directed min-power edges) with source vertex s in one of the m orbit components, the target vertex t belongs either to the same orbit component or the central component, see Figure 18 for an illustration. Due to this property strong connectivity can only be achieved by attaching single orbit components to the central component, which results in a star-like topology.

2-Level Euclidean Range Assignment for Stars (2LERAS)	
Given:	A finite set $P \subset \mathbb{R}^2$, a positive integer $k \in \mathbb{N}$ and two ranges $r_1, r_2 \in \mathbb{R}$, $r_1 < r_2$ such that the graph $(P, E^+_{\text{min}}(P) \cup E^+_{\text{max}}(P))$ is strongly connected for the mappings $d_{\text{min}} = d_{r_1}$ and $d_{\text{max}} = d_{r_2}$ and that the following condition holds: Let $C, V_1, \ldots, V_m$ be the vertex sets of the strongly connected components of $(P, E^+_{\text{min}}(P))$. Then, for all $(u, v) \in E^+_{\text{max}} \setminus E^+_{\text{min}}$ and for all $1 \leq i \leq m$ it holds that $(u \in V_i \quad \Rightarrow \quad v \in V_i \quad \vee \quad v \in C)$.
Question:	Is there a set $U \subseteq P$ of cardinality $\lvert U \rvert \leq k$ such that the graph $(P, E^+_{\text{min}}(P) \cup E^+_{\text{max}}(U))$ is strongly connected?

Theorem 20. 2LSERA *is NP-complete.*

Proof. Obviously 2LSERA $\in$ NP, because a deterministic algorithm can decide in polynomial time whether the min-max-power graph $G(U)$ is connected for a given subset of the vertices U.

Since every instance (P, k, r_1, r_2) for 2LERAS can also be considered as an instance for 2LSERA, it only remains to show that the following holds true:

There is a subset $Q \subseteq P$ of cardinality $|Q| \leq k$ such that the graph $G^+(Q) := (P, E^+_{\text{min}}(P) \cup E^+_{\text{max}}(Q))$ is strongly connected if and only if there is a set $U \subseteq P$ of cardinality $|U| \leq k$ such that the min-max-power graph $G(U)$ is connected.

First, we can observe that the vertex sets $C, V_1, \ldots, V_m$ of the strongly connected components of $G^+(\emptyset)$ are also the vertex sets of the connected components of $G(\emptyset)$, because the mapping d_{min} emerges from euclidean distances and therefore is symmetric.

$\Rightarrow$: Let Q be a subset of the vertices with $|Q| \leq k$ such that $G^+(Q)$ is strongly connected. We can assume without loss of generality that Q contains exactly one vertex from each of the sets $V_1, \ldots, V_m$, because having two vertices from the same orbit component is completely redundant with respect to strong connectivity and can therefore be avoided: Due to the restriction that all directed max-power edges for which the source vertex belongs to one of the orbit components have a target vertex either in the same orbit or in the central component, it is sufficient to include one source vertex with an incident edge towards the central component in any solution. Let $Q_C := Q \cap C$ be the subset of vertices that belong to the central component. Then the cardinality of Q_C is given by $|Q_C| = |Q| - m$. Graph $G^+(Q)$ being strongly connected implies that the set

$$T_Q \quad := \quad \{t \in V_1 \cup \cdots \cup V_m \;\mid\; \exists s \in Q_C : t \in d_{r_2}(s)\}$$

contains at least one vertex from each of the sets $V_1, \ldots, V_m$. Let $T \subseteq T_Q$ be an arbitrary subset that contains exactly one vertex from each V_i, $1 \leq i \leq m$. Then the set $U := Q_C \cup T$ has cardinality $|U| = |Q| - m + |T| = |Q| \leq k$ and the min-max-power graph $G(U)$ is connected, because for every pair (C, V_i) of vertex sets that induce connected components of $G(\emptyset)$ the set U contains two vertices $u_i \in C$, $v_i \in V_i$ such that $v_i \in d_{\text{max}}(u_i)$ and $u_i \in d_{\text{max}}(v_i)$, which implies that $\{u_i, v_i\} \in E_{max}(U)$.

$\Leftarrow$: Let U be a subset of the vertices with $|U| \leq k$ such that $G(U)$ is connected. Due to the symmetry of the mapping d_{max} it is obvious that the set U itself satisfies the required property. $\square$

Corollary 4. 2LSRA *is NP-complete.*

4.3 A family of Approximation Algorithms

If the min-max-power graph $G(U)$ is connected for some set $U \subseteq V$ then U contains at least one vertex from the vertex set of each connected component of the min-power graph $G(\emptyset)$. That is, the number $|CC(G(\emptyset))|$ of connected components of $G(\emptyset)$ is a lower bound for the cardinality of U. On the other hand, it is easy to find a set $U \subseteq V$ with at most $2(|CC(G(\emptyset))| - 1)$ vertices such that $G(U)$ is connected. Such a set can be determined by a simple spanning tree algorithm as follows: Let H be the graph that contains a vertex for every connected component of $G(\emptyset)$ and an edge between two vertices $C, C' \in CC(G(\emptyset))$ if there are vertices $u \in C, u' \in C'$ such that $u' \in d_{\max}(u)$ and $u \in d_{\max}(u')$. Let T be a spanning tree for H. Then for every edge $\{C, C'\}$ of T we can select two vertices $u \in C, v \in C'$ from different connected components of $G(\emptyset)$. Let U_T be the set of all these vertices selected for T. Then the min-max-power graph $G(U)$ is obviously connected and U_T contains at most $2(|CC(G(\emptyset))| - 1)$ vertices.

Observation 3 *Any minimal solution for an instance of* 2LSRA *contains between* $|CC(G(\emptyset))|$ *and* $2(|CC(G(\emptyset))| - 1)$ *vertices.*

The algorithm starts with the min-power graph $(V, E_0) := G(\emptyset)$ and an empty vertex set $U_0 := \emptyset$ that is successively extended to vertex sets $U_i := U_{i-1} \cup M_i$ by adding vertex sets M_i such that (V, E_i) has less connected components than (V, E_{i-1}), where $E_i := E_{i-1} \cup E_{\max}(M_i)$. This is done until (V, E_i) is connected, which implies that $G(U_i)$ is connected, because (V, E_i) is a subgraph of $G(U_i)$. To achieve a good result with this approach it seems natural to choose these vertex sets M_i in a greedy fashion by maximizing the ratio

$$\frac{|CC((V, E_{i-1}))| - |CC((V, E_i))|}{|M_i|}.$$

For example, if each M_i consists of two vertices and the number of connected components is reduced by one at each extension, then the algorithm computes a solution of size at most $2(|CC(G(\emptyset))| - 1)$. This is equivalent to a spanning tree solution. If each M_i consists of three vertices and the number of connected components is reduced by two at each extension, then the algorithm computes a solution of size at most $\frac{3}{2}(|CC(G(\emptyset))| - 1)$.

Definition 20. *A set of vertices $M \subseteq V$ of cardinality $|M| = k$ is called a* k-merging *for a graph $G = (V, E)$ and a mapping $d_{\max} : V \to \mathcal{P}(V)$, if*

1. *the graph* $(M, E_{\max}(M))$ *is connected, and*
2. *the* k *vertices of* M *are in* k *different connected components of* G*.*

If a k-merging M_i is added to vertex set U_{i-1}, then

$$\frac{|CC((V, E_{i-1}))| - |CC((V, E_i))|}{|M_i|} \quad = \quad \frac{k-1}{k}.$$

The approximation algorithm Approx2LSRA_k shown in Figure 3 has a fixed parameter $k \geq 2$. Starting with $k' = k$, it successively gathers k'-mergings as long as possible. Afterwards k' is decremented and the algorithm proceeds in the same way until (V, E_i) is connected, which will occur at the latest during the iteration for $k' = 2$, where all remaining 2-mergings are considered.

Observation 4 *Algorithm* Approx2LSRA_k *always finds a solution for an instance* $(V, d_{\min}, d_{\max})$ *of* 2LSRA.

For every positive integer $k \in \mathbb{N}$ the algorithm Approx2LSRA_k listed in Figure 3 can be implemented such that its running time is polynomial in $|V|$ and $|E|$, because the running time is dominated by the computation of all subsets $M \subseteq V$ with $|M| = k$ in line 7. Therefore we get a polynomial time algorithm for every constant integer k.

Algorithm 3 Algorithm Approx2LSRA_k for a fixed integer $k \geq 2$

1: **function** $\text{Approx2LSRA}_k(V, d_{\min}, d_{\max})$
2:     $k' \leftarrow k$
3:     $i \leftarrow 0$
4:     $U_0 \leftarrow \emptyset$
5:     $E_0 \leftarrow E_{\min}(V)$
6:     **while** (V, E_i) is not connected **do**
7:         **while** there is a k'-merging $M_i \subseteq V$ for (V, E_i) **do**
8:             $U_{i+1} \leftarrow U_i \cup M_i$
9:             $E_{i+1} \leftarrow E_i \cup E_{\max}(M_i)$
10:             $i \leftarrow i + 1$
11:         **end while**
12:         $k' \leftarrow k' - 1$
13:     **end while**
14:     **return** U_i
15: **end function**

Upper Bounds on the Quality of Approx2LSRA_k

Let $U_{OPT}(I)$ be an optimal solution for an instance $I = (V, d_{\min}, d_{\max})$ of 2LSRA. We show that Approx2LSRA_k for a positive integer $k \geq 2$ computes a solution $U_k(I)$ such that

$$\frac{|U_k(I)|}{|U_{OPT}(I)|} \quad \leq \quad \frac{1}{k-1} + \sum_{i=1}^{k-1} \frac{1}{i^2}.$$

Lemma 12. *Let $F_0 = (V, E_0)$ be a forest with $n = |V|$ vertices and $m = |E_0|$ edges and let $p \in \mathbb{N}$, $1 \leq p \leq n-1$ and $l \in \mathbb{N}$ be positive integers.*
Furthermore let $F_i = (V, E_i)$, $1 \leq i \leq l$, be a sequence of forests such that F_l contains only trees with less than p edges and $E_i \subseteq E_{i-1}$, $|E_i| = |E_{i-1}| - p$.
If $m > \frac{n \cdot (p-1)}{p}$ then

1. *F_0 contains a tree with at least p edges,*
2. *each forest F_i,*
$$i \quad < \quad \frac{1}{p} \cdot \left(m - \frac{n \cdot (p-1)}{p} \right),$$
contains a tree with at least p edges and
3. $$l \quad \geq \quad \left\lceil \frac{1}{p} \cdot \left(m - \frac{n \cdot (p-1)}{p} \right) \right\rceil.$$

Proof.

1. Forest F_0 consists of $n - m$ trees. If $m > (n-m) \cdot (p-1)$, then at least one of these $n - m$ trees has more than $p - 1$ edges and thus at least p edges.
$$m > (n-m) \cdot (p-1) \quad \Leftrightarrow \quad m > \frac{n \cdot (p-1)}{p}$$
2. Forest F_i has $m - i \cdot p$ edges. If
$$i < \frac{1}{p} \cdot \left(m - \frac{n \cdot (p-1)}{p} \right)$$
then F_i has more than $m - \frac{1}{p} \cdot \left(m - \frac{n \cdot (p-1)}{p} \right) \cdot p = \frac{n \cdot (p-1)}{p}$ edges and by Lemma 12.1. at least one tree with p edges.
3. Follows from 2. and the fact that we still can remove at least p more edges if there is a tree with p edges.

□

Lemma 13. *Let $k \in \mathbb{N}, k \geq 3$ be a positive integer and U_{OPT} an optimal solution for an instance of* 2LSRA. *Then* Approx2LSRA$_k$ *always finds at least*

$$\left\lceil \frac{1}{k-1} \cdot \left((|CC(G(\emptyset))| - 1) - \frac{|U_{OPT}| \cdot (k-2)}{k-1} \right) \right\rceil$$

k-mergings.

Proof. Let $M_0, \ldots, M_{l-1}$ be the k-mergings chosen by APPROX2LSRA$_k$ in Line 7 for a given instance $V, d_{\min}, d_{\max}$ of 2LSRA and $E_1, \ldots, E_l$ the edge sets computed in line 9 starting with edge set E_0 of the min-power graph $G(\emptyset)$.

Let $H = (CC(G(\emptyset)), E_H)$ be the undirected graph that has a vertex for every connected component of $G(\emptyset)$ and an edge $\{C_1, C_2\}$ if and only if there is an edge $\{u, v\}$ in the min-max-power graph $G(U_{\text{OPT}})$ with $u \in C_1$ and $v \in C_2$.

For $i = 0, \ldots, l$ we define a tree T_i and a forest $F_i = (U_{\text{OPT}}, E'_i)$ that satisfies the following invariant:

(I1) The vertex set R of every tree of forest F_i, $0 \leq i \leq l$, is a $|R|$-merging for graph (V, E_i).

Let $T_0 = (CC(G(\emptyset)), E_T)$ be a spanning tree of H and F_0 be a forest that contains for every edge $\{C_1, C_2\} \in E_T$ exactly one edge $\{u, v\}$ of $G(U_{\text{OPT}})$ with $u \in C_1$ and $v \in C_2$. Invariant (I1) above obviously holds true for F_0.

For every k-merging $M_i = \{u_1, \ldots, u_k\}$, $0 \leq i < l$, we successively define trees $T_{i,1}, \ldots, T_{i,k}$ and forests $F_{i,1}, \ldots, F_{i,k}$ starting with $T_{i,1} := T_i$ and $F_{i,1} := F_i$ such that $T_{i+1} := T_{i,k}$ and $F_{i+1} := F_{i,k}$.

For $j = 2, \ldots, k$, tree $T_{i,j}$ and forest $F_{i,j}$ are defined by merging two vertices of tree $T_{i,j-1}$ and removing one edge from forest $F_{i,j-1}$, respectively.

Let C^* be the vertex of $T_{i,j-1}$ that contains the vertices $u_1, \ldots, u_{j-1}$ and let C be the vertex of $T_{i,j-1}$ that contains u_j. Choose any edge $\{C', C''\}$ from the path between C^* and C in $T_{i,j-1}$, replace the two vertices C^*, C by one new vertex $C^* \cup C$ in $T_{i,j-1}$, and remove edge $\{u, v\}$ with $u \in C'$ and $v \in C''$ from $F_{i,j-1}$.

If Invariant (I1) holds true for F_i, then it holds true for F_{i+1}, because the construction above guarantees that for every simple path $v_1, \ldots, v_m$ in F_{i+1} the vertices v_i, $1 \leq i \leq m$, are in m different connected components of (V, E_{i+1}).

Since forest F_0 has $|U_{\text{OPT}}|$ vertices and $|CC(G(\emptyset))| - 1$ edges, by Lemma 12.3, algorithm APPROX2LSRA$_k$ finds at least

$$\left\lceil \frac{1}{k-1} \cdot \left((|CC(G(\emptyset))| - 1) - \frac{|U_{\text{OPT}}| \cdot (k-2)}{k-1} \right) \right\rceil$$

k mergings. □

Theorem 21. *Let $I = (V, d_{\min}, d_{\max})$ be an instance of* 2LSRA *and $U_{OPT}(I)$ an optimal solution. Algorithm* APPROX2LSRA$_k$*, for a fixed integer $k \geq 2$, computes a solution $U_k(I)$ such that*

$$\frac{|U_k(I)|}{|U_{OPT}(I)|} \quad \leq \quad \frac{1}{k-1} + \sum_{i=1}^{k-1} \frac{1}{i^2}.$$

Proof. Let l_i for $3 \leq i \leq k$ denote the number of i-mergings chosen by Approx2LSRA$_k$. Furthermore, let $c_k := |CC(G(\emptyset))|$ be the number of connected components of the min-power graph and let c_i for $2 \leq i \leq k-1$ be the number of connected components before Approx2LSRA$_k$ searches for i-mergings for the first time, that is $c_i := |CC((V, E_{s(i)}))|$ with $s(i) := \sum_{j=i+1}^{k} l_j = l_k + \ldots + l_{i+1}$. Then we know for $3 \leq i \leq k$ that

$$c_{i-1} = c_i - (i-1)l_i \quad \Leftrightarrow \quad l_i = \frac{c_i - c_{i-1}}{i-1}. \tag{5}$$

Let $d_{\min,s(i)} : V \to \mathcal{P}(V)$ be defined by $u \in d_{\min,s(i)}(v)$ if and only if $\{u, v\} \in E_{s(i)}$ for all $u, v \in V$. Then $(V, E_{s(i)})$ is the min-power graph of instance $I_{s(i)} := (V, d_{\min,s(i)}, d_{\max})$. Now we can apply Lemma 13 for $k = i$ on instance $I_{s(i)}$ and get

$$\begin{aligned}
& \frac{c_i - c_{i-1}}{i-1} \geq \frac{1}{i-1}\left(c_i - 1 - \frac{i-2}{i-1} \cdot |U_{OPT}(I_{s(i)})|\right) \\
\Leftrightarrow \quad & c_{i-1} - 1 \leq \frac{i-2}{i-1} \cdot |U_{OPT}(I_{s(i)})| \leq \frac{i-2}{i-1} \cdot |U_{OPT}(I)|,
\end{aligned} \tag{6}$$

because $|U_{OPT}(I_{s(i)})| \leq |U_{OPT}(I)|$. We can derive an upper bound for $|U_k(I)|$ as follows.

$$\begin{aligned}
& |U_k(I)| \\
\leq\ & \sum_{i=3}^{k} i \cdot l_i + 2(c_2 - 1) \overset{(5)}{=} \sum_{i=3}^{k} i \cdot l_i + 2\left(c_k - 1 - \sum_{i=3}^{k}(i-1)l_i\right) \\
=\ & 2(c_k - 1) - \sum_{i=3}^{k}(i-2)l_i \overset{(5)}{=} 2(c_k - 1) - \sum_{i=3}^{k} \frac{i-2}{i-1}(c_i - c_{i-1}) \\
=\ & 2(c_k - 1) + \sum_{i=3}^{k} \frac{i-2}{i-1}(c_{i-1} - 1) - \sum_{i=3}^{k} \frac{i-2}{i-1}(c_i - 1) \\
=\ & 2(c_k - 1) + \frac{c_2 - 1}{2} - \frac{(k-2)(c_k - 1)}{k-1} + \sum_{i=3}^{k-1}\left(\frac{i-1}{i} - \frac{i-2}{i-1}\right)(c_i - 1) \\
=\ & \left(2 - \frac{k-2}{k-1}\right) \cdot (c_k - 1) + \frac{c_2 - 1}{2} + \sum_{i=3}^{k-1} \frac{c_i - 1}{i(i-1)} \\
\overset{(6)}{\leq}\ & \left(2 - \frac{k-2}{k-1}\right) \cdot (c_k - 1) + \frac{|U_{OPT}(I)|}{4} + \sum_{i=3}^{k-1} \frac{|U_{OPT}(I)|}{i^2}
\end{aligned}$$

Since $c_k - 1 < |U_{OPT}(I)|$, see Observation 3, we get

$$\begin{aligned} |U_k(I)| &< \left(2 - \frac{k-2}{k-1} + \frac{1}{4} + \sum_{i=3}^{k-1} \frac{1}{i^2}\right) \cdot |U_{OPT}(I)| \\ &= \left(\frac{1}{k-1} + \sum_{i=1}^{k-1} \frac{1}{i^2}\right) \cdot |U_{OPT}(I)|. \end{aligned}$$

□

Corollary 5. Approx2LSRA$_3$ *is a* $7/4$ *factor approximation for* 2LSRA.

Observation 5 *The upper bound in Theorem 21 tends to* $\frac{\pi^2}{6}$ *for* $k \to \infty$. *[6]*

Lower Bounds on the Quality of Approx2LSRA$_k$

This section presents worst cases for the algorithms Approx2LSRA$_k$ and derives lower bounds on their quality. For $k = 3$ we demonstrate that the upper bound of 7/4 obtained in section 4.3 is tight.

Theorem 22. *Let* $k \in \mathbb{N}$, $k \geq 3$ *be a positive integer. For an instance* I *of* 2LSRA *let* $U_k(I)$ *be the solution computed by* Approx2LSRA$_k$ *and* $U_{OPT}(I)$ *an optimal solution. For all* $\epsilon > 0$ *there is an instance* I *such that*

$$\frac{|U_k(I)|}{|U_{OPT}(I)|} > \frac{3k-2}{2k-2} - \epsilon.$$

Proof. For a positive integer $t \in \mathbb{N}$ let I_t be the instance for 2LSRA defined as follows, also see Figure 19.

$$\begin{aligned} V :=& \{(0,0,0)\} \quad \cup \quad \{(d,3,0) \mid 1 \leq d \leq t\} \\ & \cup \quad \{(d,r,c) \mid 1 \leq d \leq t, 1 \leq r \leq 3, 1 \leq c < k\} \\ E_{\min}(V) :=& \{\{(0,0,0),(d,3,0)\} \mid 1 \leq d \leq t\} \\ & \cup \{\{(d,2,c),(d,3,c)\} \mid 1 \leq d \leq t, 1 \leq c < k\} \\ E_{\max}(V) :=& E_{\min}(V) \quad \cup \quad \{\{(0,0,0),(d,2,1)\} \mid 1 \leq d \leq t\} \\ & \cup \{\{(d,r,c),(d,r,c+1)\} \mid 1 \leq d \leq t, r \in \{2,3\}, 1 \leq c < k-1\} \\ & \cup \{\{(d,3,0),(d,3,1)\} \mid 1 \leq d \leq t\} \\ & \cup \{\{(d,1,c),(d,2,c)\} \mid 1 \leq d \leq t, 1 \leq c < k\} \end{aligned}$$

The (unique) optimal solution for I_t is

$$U_{OPT}(I_t) = \{(0,0,0)\} \cup \{(d,r,c) \mid 1 \leq d \leq t, r \in \{1,2\}, 1 \leq c < k\},$$

meaning that $|U_{OPT}(I_t)| = 1 + 2(k-1)t$. Algorithm APPROX2LSRA$_k$, in the worst case, successively gathers all k-mergings $\{(d,3,c) \mid 0 \le c < k\}$ for $1 \le d \le t$ first, followed by the remaining 2-mergings $\{(d,1,c),(d,2,c)\}$ for $1 \le d \le t$ and $1 \le c < k$. Thus we get

$$q(k,t) := \frac{|U_k(I_t)|}{|U_{OPT}(I_t)|} = \frac{kt + 2(k-1)t}{1 + 2(k-1)t} = \frac{k + 2(k-1)}{\frac{1}{t} + 2(k-1)} = \frac{3k-2}{\frac{1}{t} + 2k - 2}$$

and $\lim_{t\to\infty}(q(k,t)) = \frac{3k-2}{2k-2}$, which implies the existence of a positive integer $t_\epsilon \in \mathbb{N}$ such that $q(k,t_\epsilon) > \frac{3k-2}{2k-2} - \epsilon$. □

Corollary 6. *The upper bound of* $7/4$ *on the quality of* APPROX2LSRA$_3$ *is tight.*

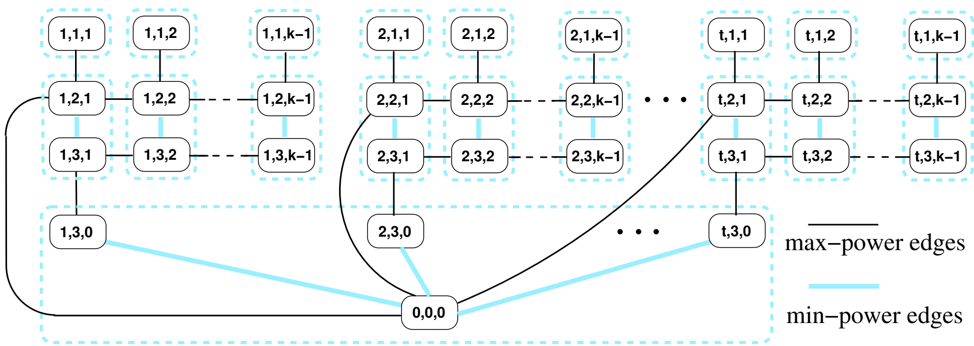

Fig. 19. Instance I_t of Theorem 22: The min-power edges $E_{\min}(V)$ are drawn as thick blue lines, the max-power edges $E_{\max}(V)$ are drawn as thin black lines and the connected components of the min-power graph $G(\emptyset)$ are enclosed by dashed boxes.

Efficient Implementation of APPROX2LSRA$_3$

For $k = 3$ the algorithm can be implemented in almost linear time using a union-find data structure D to organize the vertex sets of the connected components of (V, E_i).

Assume that the mappings $d_{\min}$ and $d_{\max}$ are explicitly given as relations $d_{\min} \subset V \times \mathcal{P}(V)$ and $d_{\max} \subset V \times \mathcal{P}(V)$ of size $s_{\min} := |V| + \sum_{v\in V} |d_{\min}(v)|$ and $s_{\max} := |V| + \sum_{v\in V} |d_{\max}(v)|$, respectively. The implementation is based on the following three steps.

1. *Initialization*: Generate the set of min-power edges $E_{\min}(V)$ and the set of max-power edges $E_{\max}(V)$. This can be done in time $\mathcal{O}(s_{\min})$ and $\mathcal{O}(s_{\max})$,

respectively. Afterwards D can be initialized with the connected components of $G(\emptyset)$ during one iteration over the min-power edges by performing two find operations $u' = find(u)$, $v' = find(v)$ as well as one union operation $union(u', v')$ for each edge $\{u, v\} \in E_{\min}(V)$.

The total number of $find$ and $union$ operations for this step is linear in $|V|$ and $|E_{\min}(V)|$ and therefore also linear in $s_{\min}$.

2. *Finding* 3*-mergings (also see Algorithm 4):* For every vertex $v \in V$ examine all incident edges $\{v, u\} \in E_{\max}(V)$ after identifying the connected component of v via $C_v = find(v)$. If $C_u := find(u) \neq C_v$, vertex u and its component C_u are temporarily saved until a second vertex u' adjacent to v is found, such that $C_v \neq C_{u'}$ and $C_u \neq C_{u'}$. Then $\{v, u, u'\}$ is a 3-merging that is added to the solution and the three connected components C_v, C_u and $C_{u'}$ are merged via two union operations.

   Since every edge in $E_{\max}(V)$ has to be considered only twice, the total number of $find$ and $union$ operations for this step is linear in $|V|$ and $|E_{\max}(V)|$ and therefore also linear in $s_{\max}$.
3. *Finding* 2*-mergings:* For every edge $\{u, v\} \in E_{\max}(V)$ add $\{u, v\}$ to the solution and call $union(u, v)$, if $find(u) \neq find(v)$. Again, the total number of $find$ and $union$ operations for this step is linear in $s_{\max}$.

Theorem 23. Approx2LSRA$_3$ *can be implemented such that the running time is in* $\mathcal{O}(f(s_{\min}, s_{\max}) \cdot \alpha(f(s_{\min}, s_{\max}), |V|))$ *where* α *is the inverse Ackermann function and* $f \in \mathcal{O}(s_{\min} + s_{\max})$.

Proof. The total number of $union$ and $find$ operations $f(s_{\min}, s_{\max})$ is linear in $s_{\min}$ and $s_{\max}$ as discussed above. Tarjan and Leeuwen show in [89] that any sequence of $m \in \mathbb{N}$ $union$ and $find$ operations on a union-find data structure saving $n \in \mathbb{N}$ elements can be performed in time $\mathcal{O}(m \cdot \alpha(m, n))$. Fredman and Saks show in [33] that this bound in tight. □

4.4 Maximum Merging

Observation 6 *For a* 2LSRA *instance there is not necessarily an optimal solution that contains a merging of maximum size, see Figure 20. Furthermore, computing a merging of maximum size is not feasible for the design of an approximation algorithm due to the following Theorem 24.*

Algorithm 4 Determining a maximal number 3-mergings utilizing a union-find data structure for the vertex sets of the connected components. In this implementation V is an array of all vertices and $V[i].N$ denotes a vertex set containing all neighbors that are adjacent to $V[i]$ via the max-power edges $E_{\max}(V) \setminus E_{\min}(V)$

```
procedure Find-3-Mergings
    for i = 0 ; i < |V| ; i++ do
        y ← z ← ∅
        x ← find(V[i])
        for u ∈ V[i].N do
            if y == ∅ then
                w ← u
                y ← find(u)
                if x == y then
                    y ← ∅
                end if
            else
                z ← find(u)
                if (x ≠ z) ∧ (y ≠ z) then              ▷ {V[i], w, u} is a 3-merging
                    union(x, y)
                    union(x, z)
                    V[i].N ← V[i].N \ {u, w}       ▷ Do not consider u, w multiple times
                    i ← i − 1                       ▷ V[i] might be part of another 3-merging
                    break                           ▷ Restart outer loop for V[i]
                end if
            end if
        end for
    end for
end procedure
```

Maximum Merging	
Given:	An undirected min-power graph $G = (V, E)$, a mapping $d_{\max} : V \to \mathcal{P}(V)$ and a positive integer $k \in \mathbb{N}$.
Question:	Is there a merging U of cardinality $|U| \geq k$ in G?

Theorem 24. Maximum Merging *in NP-complete.*

Proof. Maximum Merging is obviously in NP, because it can be verified in polynomial time whether a given subset $U \subseteq V$ of the vertices is a merging by computing the connected components of G and $(U, E_{\max}(U))$. The NP-hardness is now shown by giving a polynomial time reduction from Directed Hamiltonian Path (DHP) that transforms a directed graph $H = (V_H, E_H)$

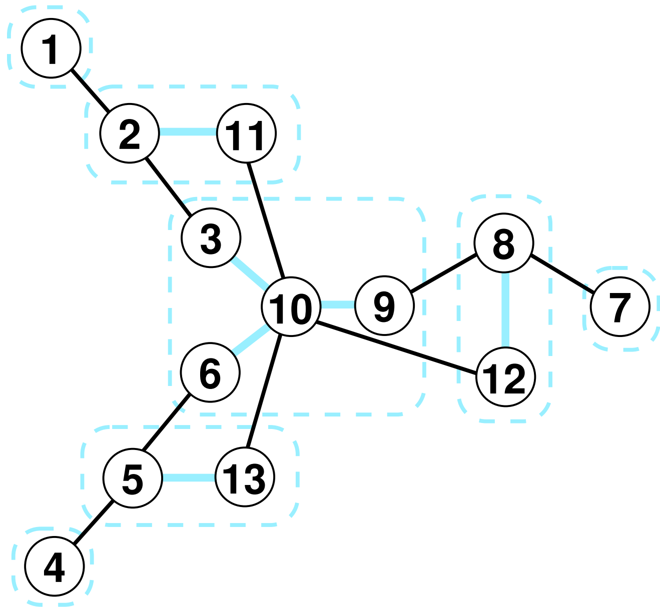

Fig. 20. An instance of 2LSRA: The min-power edges are drawn as thick blue lines, the max-power edges are drawn as black lines and the connected components of the min-power graph are enclosed by dashed boxes. The (unique) optimal solution is $\{1, 2, 3, 4, 5, 6, 7, 8, 9\}$ while the merging of maximum size is $\{10, 11, 12, 13\}$.

into an undirected graph $G = (V, E)$, a mapping $d_{\max} : V \to \mathcal{P}(V)$ and a positive integer $k := 2|V_H|$ such that there is a simple path across all vertices in H if and only if there is merging U of cardinality $|U| \geq k$ in G.

The reduction generates exactly two connected components for every vertex $v \in V_H$, one of which contains a vertex t_v as well as a vertex $t_{(u,v)}$ for every edge $(u, v) \in E_H$ while the other one contains a vertex s_v as well as a vertex $s_{(v,u)}$ for every edge $(v, u) \in E_H$. The set $E_{\max}(V)$ consists of all edges between these type t and type s vertices inserted for a vertex $v \in V_H$ as well as one edge $\{s_e, t_e\}$ for every edge $e \in E_H$ (see Figure 21). Formally,

$$\begin{aligned} V &:= \bigcup_{e \in E_H} \{s_e, t_e\} \cup \bigcup_{v \in V_H} \{s_v, t_v\} \\ E &:= \big\{\{s_v, s_{(v,u)}\} \mid v \in V_H \wedge (v, u) \in E_H\big\} \\ &\quad \cup \big\{\{t_v, t_{(u,v)}\} \mid v \in V_H \wedge (u, v) \in E_H\big\} \end{aligned}$$

and $d_{\max} : V \to \mathcal{P}(V)$ is a mapping that results in the following set of max-power edges:

$$\begin{aligned} E_{\max}(V) \quad = \quad & \{\{s_e, t_e\} \mid e \in E_H\} \\ \cup \quad & \left\{\{t_{(u,v)}, s_{(v,w)}\} \mid (u,v) \in E_H \wedge (v,w) \in E_H\right\} \\ \cup \quad & \left\{\{t_v, s_{(v,w)}\} \mid v \in V_H \wedge (v,w) \in E_H\right\} \\ \cup \quad & \left\{\{t_{(u,v)}, s_v\} \mid v \in V_H \wedge (u,v) \in E_H\right\}. \end{aligned}$$

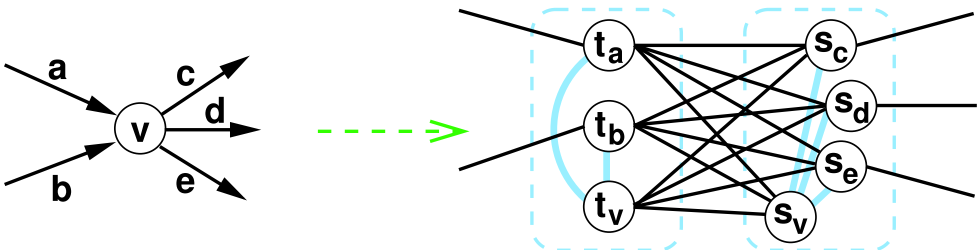

Fig. 21. Reduction of Theorem 24: Every vertex $v \in V_H$ of the instance $H = (V_H, E_H)$ for DHP is transformed into two connected components. The min-power edges are drawn as thick blue lines, the max-power edges are drawn as black lines and the connected components of the min-power graph are enclosed by dashed boxes.

The proof is now completed by showing that there is a Hamiltonian path in H if and only if there is merging U of cardinality $|U| \geq k$ in G:

$\Rightarrow$: Let $p := (v_1, v_2, \ldots, v_n)$ be a Hamiltonian path in H. Then

$$U \quad := \quad \{t_{v_1}, s_{v_n}\} \quad \cup \quad \{s_e, t_e \mid e = (v_i, v_{i+1}), 1 \leq i < n\}$$

is a merging of cardinality $k = 2|V_H|$: U contains exactly one vertex from each of the $2|V_H|$ connected component in G, because p is a simple path. Also, the graph $(U, E_{\max}(U))$ is connected, because $\{s_e, t_e\} \in E_{\max}(V)$ for all $e \in E_H$ and $\{s_e, t_e\} \subseteq U$ for all $e \in \{(v_i, v_{i+1}) \mid 1 \leq i < n\}$.

$\Leftarrow$: Let U be a merging of cardinality $|U| \geq 2|V_H|$ in G. G contains exactly $2|V_H|$ connected components, therefore $|U| = 2|V_H|$. It also follows that $|E_{\max}(U)| \geq |U| - 1$, because $(U, E_{\max}(U))$ is connected. By construction of G, every vertex of U has degree at most 2 in $(U, E_{\max}(U))$, meaning that

$$|E_{\max}(U)| \quad = \quad \frac{1}{2} \sum_{u \in U} deg(u) \quad \leq \quad |U|,$$

which implies that the connected graph $(U, E_{\max}(U))$ is either a path or a cycle. Additionally, the sequence of vertices representing this path (or cycle) alternates between type s and type t vertices due to the construction of G and every other edge is of the form $\{s_e, t_e\}$ for some edge $e \in E_H$ while the remaining edges are between the two connected components generated for one vertex $v \in V_H$. Therefore H contains a Hamiltonian path. $\square$

5 Conclusion

Wireless ad-hoc sensor networks consist of devices that commonly operate without a central control authority and therefore each node has to make decisions autonomously based on information that is locally available. Complex tasks require the nodes to exchange information with each other and reach a consensus about how a goal can be achieved. Hence successful message delivery by routing protocols is one of the most basic building blocks for any network. We have seen that a special case of multicast routing, the NEIGHBORHOOD BROADCAST, naturally occurs when trying to provide reliability of routing protocols in the present of node failures and that it also is an important part of security and intrusion detection when sensor nodes want to detect and exclude misbehaving nodes from the network. The presented k-HBF network protocol, parameterized with a positive integer $k \in \mathbb{N}$, successfully solves this problem in an optimal manner, if the parameter k is chosen according to the local connectivity distance of the node in question. The problem of how to determine the local connectivity distance of a node in a distributed environment was also addressed and it has been discussed how the k-HBF protocol can be used in cases where it is not viable to compute the local connectivity distance in advance. Finally, simulations conducted on state of the art random graph models for wireless network topology generation suggest a very high success rate of the protocol for very small values of k in real world environment. This empirical analysis assumes a static network and it would be a suitable topic for future research to analyze the protocol in networks that are subjected to frequent changes, for example due to mobile nodes.

While network topologies in which the nodes have a low local connectivity distance would be advantageous for performing neighborhood broadcasts, aiming for local connectivity distance 1 via topology control has been proven to be computationally difficult: Adding a minimum number of edges to a given graph in order to achieve local connectivity distance 1 for all nodes - and in the strong model even for a single node - is NP-complete. The same holds true for computing the maximum connected induced subgraph of a given communication graph and also for removing as many edges as possible from a locally connected network while preserving connectivity and local connectivity. We have also seen that most of the NP-completeness results presented in this thesis are easily transferable to the case where k-connectivity of the neighborhoods in question is required instead of 1-connectivity. However, it remains open whether these topology control problems also remain NP-complete, if they are extended to consider d-hop neighborhoods for an arbitrary constant distance $d > 1$. While this intuitively seems likely, it is not a direct implication of the presented results for the special case $d = 1$, if the distance parameter d is not given in the instance of the problem.

Fortunately, one of the most practical topology control problems for supporting neighborhood broadcasts, the LCWA problem, can be approximated within a factor of $1+ln(\Delta(G))$ by a greedy algorithm. It is noteworthy that this greedy algorithm does not require global topology knowledge and therefore can theoretically be implemented in a distributed environment. An efficient and practical network protocol that realizes this algorithm in a sensor network is a suitable topic for future research.

Regarding the symmetric connectivity of wireless networks with two power levels, we have seen that the previously known greedy approximation algorithm for computing a set of nodes that have to use max-power can be extended to provide better results at the cost of higher polynomial computation time. However, the presented lower bounds for this family of algorithms also demonstrate that this technique is not capable of outperforming the best known approximation algorithm for this problem that is based on approximating the classical Steiner Tree problem.

References

1. *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies, 13-17 March 2005, Miami, FL, USA*. IEEE, 2005.
2. M. Ahyar, M. Yani, Elfisa, and R.F. Sari. Comparison of energy efficiency and routing packet overhead in single and multi path routing protocols over S-MAC for wireless sensor network. In *Sixth UKSim/AMSS European Symposium on Computer Modeling and Simulation, EMS 2012, Malta, November 14-16, 2012*, pages 406–411. IEEE, 2012.
3. E. Althaus, G. Calinescu, I.I. Mandoiu, S. Prasad, N. Tchervenski, and A. Zelikovsky. Power efficient range assignment for symmetric connectivity in static ad hoc wireless networks. *Wirel. Netw.*, 12(3):287–299, May 2006.
4. A. Qayyum L. and Viennot and A. Laouiti. Multipoint relaying for flooding broadcast messages in mobile wireless networks. In *System Sciences, 2002. HICSS. Proceedings of the 35th Annual Hawaii International Conference on*, pages 3866–3875, 2002.
5. E.M. Royer andC.K. Toh. A review of current routing protocols for ad hoc mobile wireless networks. *Personal Communications, IEEE*, 6(2):46–55, Apr 1999.
6. T. Apostol. A proof that Euler missed: evaluating zeta(2) the easy way. *Math. Intelligencer*, 5(3):59–60, 1983.
7. A. Awad, S. Lei, R. German, and F. Dressler. Advantages of virtual addressing for efficient and failure tolerant routing in sensor networks. In *Wireless On-Demand Network Systems and Services, 2009. WONS 2009. Sixth International Conference on*, pages 111–118, 2009.
8. A. Balakrishnan and K. Altinkemer. Using a hop-constrained model to generate alternative communication network design. *ORSA Journal on Computing*, 4(2):192–205, 1992.
9. D. Blough, M. Leoncini, G. Resta, and P. Santi. On the symmetric range assignment problem in wireless ad hoc networks. In *Foundations of Information Technology in the Era of Network and Mobile Computing*, pages 71–82. Springer, 2002.
10. P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. *Wireless Networks*, 7(6):609–616, 2001.
11. Q. Botton, B. Fortz, L. Gouveia, and M. Poss. Benders decomposition for the hop-constrained survivable network design problem. *INFORMS journal on computing*, 25(1):13–26, 2013.
12. G. Călinescu, I. Măndoiu, and A. Zelikovsky. Symmetric connectivity with minimum power consumption in radio networks. In *Foundations of Information Technology in the Era of Network and Mobile Computing*, pages 119–130. Springer, 2002.
13. P. Carmi and M.J. Katz. Power assignment in radio networks with two power levels. *Algorithmica*, 47(2):183–201, 2007.
14. A. Caruso, S. Chessa, S. De, and A. Urpi. Gps free coordinate assignment and routing in wireless sensor networks. In *INFOCOM* [1], pages 150–160.
15. A. Cerpa, J. Wong, L. Kuang, M. Potkonjak, and D. Estrin. Statistical model of lossy links in wireless sensor networks. In *IPSN*, pages 81–88. IEEE, 2005.
16. G.G. Chappell, J.G. Gimbel, and C. Hartman. Bounds on the metric and partition dimensions of a graph. *Ars Comb.*, 88, 2008.
17. G. Chartrand, L. Eroh, M.A. Johnson, and O. Oellermann. Resolvability in graphs and the metric dimension of a graph. *Discrete Applied Mathematics*, 105(1-3):99–113, 2000.
18. G. Chartrand and R.E. Pippert. Locally connected graphs. *Casopis Pest. Mat*, 99:158–163, 1974.
19. G. Chartrand, C. Poisson, and P. Zhang. Resolvability and the upper dimension of graphs. *Computers and Mathematics with Applications*, 39(12):19–28, 2000.

20. K. Chin, J. Judge, A. Williams, and R. Kermode. Implementation experience with MANET routing protocols. *Computer Communication Review*, 32(5):49–59, 2002.
21. A. Clementi, A. Ferreira, P. Penna, S. Perennes, and R. Silvestri. The minimum range assignment problem on linear radio networks. In *Algorithms-ESA 2000*, pages 143–154. Springer, 2000.
22. A. Clementi, P. Penna, and R. Silvestri. Hardness results for the power range assignment problem in packet radio networks. In *Randomization, Approximation, and Combinatorial Optimization. Algorithms and Techniques*, pages 197–208. Springer, 1999.
23. J. Clulow and T. Moore. Suicide for the common good: A new strategy for credential revocation in self-organizing systems. *SIGOPS Oper. Syst. Rev.*, 40(3):18–21, July 2006.
24. S.A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, STOC '71, pages 151–158, New York, NY, USA, 1971. ACM.
25. S. Desilva and S. Das. Experimental evaluation of a wireless ad hoc network. In Ton Engbersen and E. K. Park, editors, *Proceedings Ninth International Conference on Computer Communications and Networks, ICCCN 2000, 16-18 October 2000, Las Vegas, Nevada, USA*, pages 528–534. IEEE, 2000.
26. J. Diaz, O.M. Pottonen, M. Serna, and E.J. van Leeuwen. On the complexity of metric dimension. http://arxiv.org/pdf/1107.2256, 2012.
27. H. Du, J. Xiaohua, and X. Hu. Energy efficient multicast routing in ad hoc wireless networks with discrete levels of transmission power. In *Communications, Circuits and Systems, 2005. Proceedings. 2005 International Conference on*, volume 1, pages 319–323 Vol. 1, May 2005.
28. E. H. Elhafsi, N. Mitton, and D. Simplot-Ryl. Cost over progress based energy efficient routing over virtual coordinates in wireless sensor networks. In *WOWMOM*, pages 1–6. IEEE, 2007.
29. S. Even and R. Tarjan. Network flow and testing graph connectivity. *SIAM J. Comput.*, 4(4):507–518, 1975.
30. Q. Fang, J. Gao, L.J. Guibas, V. de Silva, and L. Zhang. GLIDER: gradient landmark-based distributed routing for sensor networks. In *INFOCOM* [1], pages 339–350.
31. G. Fonseca, V. Pereira de Sa, R. Machado, and C. Figueiredo. On the recognition of unit disk graphs and the distance geometry problem with ranges. *Discrete Applied Mathematics*, (0):–, 2014.
32. R. Fonseca, S. Ratnasamy, J. Zhao, C.T. Ee, D.E. Culler, S. Shenker, and I. Stoica. Beacon vector routing: Scalable point-to-point routing in wireless sensornets. In *NSDI*. USENIX, 2005.
33. M. Fredman and M. Saks. The cell probe complexity of dynamic data structures. In *Proceedings of the Twenty-first Annual ACM Symposium on Theory of Computing*, STOC '89, pages 345–354, New York, NY, USA, 1989. ACM.
34. M. Fredman and D. Willard. Trans-dichotomous algorithms for minimum spanning trees and shortest paths. *Journal of Computer and System Sciences*, 48(3):533 – 551, 1994.
35. M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman, 1979.
36. D. Gaußmann, S. Hoffmann, and E. Wanke. Hierarchical bipartition routing for delivery guarantee in sparse wireless ad hoc sensor networks with obstacles. In *ICWN*, pages 3–9. CSREA Press, 2012.
37. J. Gomez and A. Campbell. A case for variable-range transmission power control in wireless multihop networks. In *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, volume 2, pages 1425–1436 vol.2, March 2004.

38. F. Harary and R.A. Melter. On the metric dimension of a graph. *Ars Combinatoria*, 2:191–195, 1976.
39. D. Hasenfratz, A. Meier, C. Moser, J.J. Chen, and L. Thiele. Analysis, comparison, and optimization of routing protocols for energy harvesting wireless sensor networks. In *IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing, SUTC 2010 and IEEE International Workshop on Ubiquitous and Mobile Computing, UMC 2010, 7-9 June 2010, Newport Beach, California, USA*, pages 19–26. IEEE Computer Society, 2010.
40. M. Hauptmann, R. Schmied, and C. Viehmann. On approximation complexity of metric dimension problem. In *Proceedings of the 21st international conference on Combinatorial algorithms*, IWOCA'10, pages 136–139, Berlin, Heidelberg, 2011. Springer-Verlag.
41. M.C. Hernando, M. Mora, I.M. Pelayo, C. Seara, J. Cáceres, and M.L. Puertas. On the metric dimension of some families of graphs. *Electronic Notes in Discrete Mathematics*, 22:129–133, 2005.
42. S. Hoffmann, A. Elterman, and E. Wanke. A linear time algorithm for metric dimension of cactus block graphs. *Theoretical Computer Science*, 630:43 – 62, 2016.
43. S. Hoffmann, T. Kampermann, and E. Wanke. Minimizing the number of max-power users in ad-hoc wireless networks with minimum node degree requirements. In preparation for submission.
44. S. Hoffmann, D. Vietz, and E. Wanke. Efficient computation of the metric dimension by decomposing graphs. In preparation for submission.
45. S. Hoffmann and E. Wanke. Minimum power range assignment for symmetric connectivity in sensor networks with two power levels. http://arxiv.org/abs/1605.01752.
46. S. Hoffmann and E. Wanke. *Algorithms for Sensor Systems: 8th International Symposium on Algorithms for Sensor Systems, Wireless Ad Hoc Networks and Autonomous Mobile Entities, ALGOSENSORS 2012, Ljubljana, Slovenia, September 13-14, 2012. Revised Selected Papers*, chapter Metric Dimension for Gabriel Unit Disk Graphs Is NP-Complete, pages 90–92. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.
47. S. Hoffmann and E. Wanke. Generic route repair: Augmenting wireless ad hoc sensor networks for local connectivity. In *Proceedings of the 15th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2016.
48. S. Hoffmann and E. Wanke. The neighborhood broadcast problem in wireless ad hoc sensor networks. In preparation for submission, 2016.
49. IEEE. *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications (ANSI/IEEE Std 802.11, 1999 Edition (R2003)).* Institute of Electrical and Electronics Engineers, Inc., June 2003.
50. H. Iswadi, E.T. Baskoro, A.N.M. Salman, and R. Simanjuntak. The metric dimension of amalgamation of cycles. *Far East Journal of Mathematical Sciences (FJMS)*, 41(1):19–31, 2010.
51. S. Jadhav, T. Brown, S. Doshi, D. Henkel, and R. George. Lessons learned constructing a wireless ad hoc network test bed. In *1st International Workshop on Wireless Network Measurements*, 2005.
52. Z. Jin, B. Liang, Y. Shu, and O. Yang. Designing and implementing a mobile ad hoc network testbed. In *Electrical and Computer Engineering, 2002. IEEE CCECE 2002. Canadian Conference on*, volume 3, pages 1559–1564 vol.3, 2002.
53. D.B. Johnson. Routing in ad hoc networks of mobile hosts. In *First Workshop on Mobile Computing Systems and Applications, WMCSA 1994, Santa Cruz, CA, USA, December 8-9, 1994*, pages 158–163. IEEE Computer Society, 1994.
54. D.B. Johnson and D.A. Maltz. Dynamic source routing in ad hoc wireless networks. In *Mobile Computing*, pages 153–181. Kluwer Academic Publishers, 1996.

55. T. Jordán. On the optimal vertex-connectivity augmentation. *Journal of Combinatorial Theory, Series B*, 63(1):8–20, 1995.
56. L. Junhai, Y. Danxia, L. Xue, and F. Mingyu. A survey of multicast routing protocols for mobile ad-hoc networks. *Communications Surveys Tutorials, IEEE*, 11(1):78–91, First 2009.
57. A. Kaya, A. Zengin, and H. Ekiz. Comparison of the beacon-less geographic routing protocols in wireless sensor networks. In *Proceedings of the 7th International Wireless Communications and Mobile Computing Conference, IWCMC 2011, Istanbul, Turkey, 4-8 July, 2011*, pages 1093–1098. IEEE, 2011.
58. H. Kerivin and A. Mahjoub. Design of survivable networks: A survey. *Networks*, 46(1):1–21, 2005.
59. S. Khuller, B. Raghavachari, and A. Rosenfeld. Landmarks in graphs. *Discrete Applied Mathematics*, 70(3):217–229, 1996.
60. L. Kirousis, E. Kranakis, D. Krizanc, and A. Pelc. Power consumption in packet radio networks. In *STACS 97*, pages 363–374. Springer, 1997.
61. D. Kotz, C. Newport, and C. Elliott. The mistaken axioms of wireless-network research. Technical report, Dept. of Computer Science, Dartmouth College, July 2003.
62. E. Kranakis, H. Singh, and J. Urrutia. Compass routing on geometric networks. In *Proceedings of the 11th Canadian Conference on Computational Geometry, UBC, Vancouver, British Columbia, Canada, August 15-18, 1999*, 1999.
63. F. Kuhn, R. Wattenhofer, and A. Zollinger. Ad-hoc networks beyond unit disk graphs. In *Proceedings of the 2003 Joint Workshop on Foundations of Mobile Computing*, DIALM-POMC '03, pages 69–78, New York, NY, USA, 2003. ACM.
64. K. Langendoen, A. Baggio, and O. Visser. Murphy loves potatoes: experiences from a pilot sensor network deployment in precision agriculture. In *20th International Parallel and Distributed Processing Symposium (IPDPS 2006), Proceedings, 25-29 April 2006, Rhodes Island, Greece*. IEEE, 2006.
65. J. Li, L. Gewali, H. Selvaraj, and M. Venkatesan. Hybrid greedy/face routing for ad-hoc sensor network. In *DSD*, pages 574–578. IEEE, 2004.
66. M. Li, Z. Li, and A. Vasilakos. A survey on topology control in wireless sensor networks: Taxonomy, comparative study, and open issues. *Proceedings of the IEEE*, 101(12):2538–2557, Dec 2013.
67. E.L. Lloyd, R. Liu, and S.S. Ravi. Approximating the minimum number of maximum power users in ad hoc networks. In Ioanis Nikolaidis, Michel Barbeau, and Evangelos Kranakis, editors, *Ad-Hoc, Mobile, and Wireless Networks*, volume 3158 of *Lecture Notes in Computer Science*, pages 1–13. Springer Berlin Heidelberg, 2004.
68. H. Lundgren, D. Lundberg, J. Nielsen, E. Nordström, and C. Tschudin. A large-scale testbed for reproducible ad hoc protocol evaluations. In *2002 IEEE Wireless Communications and Networking Conference Record, WCNC 2002, Orlando, Florida, USA, MArch 17-21, 2002*, pages 412–418. IEEE, 2002.
69. P.G. Lye and J.C. McEachen. A comparison of optimized link state routing with traditional routing protocols in marine wireless ad-hoc and sensor networks. In *40th Hawaii International Conference on Systems Science (HICSS-40 2007), CD-ROM / Abstracts Proceedings, 3-6 January 2007, Waikoloa, Big Island, HI, USA*, page 293. IEEE Computer Society, 2007.
70. Muhammet Macit, Vehbi Cagri Gungor, and Gurkan Tuna. Comparison of qos-aware single-path vs. multi-path routing protocols for image transmission in wireless multimedia sensor networks. *Ad Hoc Networks*, 19:132–141, 2014.
71. D. Maltz, J. Broch, and D. Johnson. Experiences designing and building a multi-hop wireless ad hoc network testbed. Technical Report CMU-CS-99-116, CMU School of Computer Science, March 1999.

72. R.A. Melter and I. Tomescu. Metric bases in digital geometry. *Computer Vision, Graphics, and Image Processing*, 25(1):113–121, 1984.
73. K. Menger. Zur allgemeinen kurventheorie. *Fundamenta Mathematicae*, 10(1):96–115, 1927.
74. S.Y. Ni, Y.C. Tseng, Y.S. Chen, and J.P. Sheu. The broadcast storm problem in a mobile ad hoc network. In Harel Kodesh, Victor Bahl, Tomasz Imielinski, and Martha Steenstrup, editors, *MOBICOM '99, The Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking, Seattle, Washington, USA, August 15-19, 1999.*, pages 151–162. ACM, 1999.
75. E. Nordström, P. Gunningberg, and H. Lundgren. A testbed and methodology for experimental evaluation of wireless mobile ad hoc networks. In Roberto Battiti, Mario Gerla, and Piero Spinnato, editors, *1st International Conference on Testbeds & Research Infrastructures for the DEvelopment of NeTworks & COMmunities (TRIDENTCOM 2005), 23-25 February 2005, Trento, Italy*, pages 100–109. IEEE Computer Society, 2005.
76. Z. Nutov and A. Yaroshevitch. Wireless network design via 3-decompositions. *Information Processing Letters*, 109(19):1136 – 1140, 2009.
77. G. Pastukhov, A. Veremyev V. and Boginski, and E. Pasiliao. Optimal design and augmentation of strongly attack-tolerant two-hop clusters in directed networks. *Journal of Combinatorial Optimization*, 27(3):462–486, 2014.
78. C.E. Perkins and E.M. Belding-Royer. Ad-hoc on-demand distance vector routing. In *2nd Workshop on Mobile Computing Systems and Applications (WMCSA '99), February 25-26, 1999, New Orleans, LA, USA*, pages 90–100. IEEE Computer Society, 1999.
79. N.N. Pham, J. Youn, and C. Won. A comparison of wireless sensor network routing protocols on an experimental testbed. In *IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC 2006), 5-7 June 2006, Taichung, Taiwan*, pages 276–281. IEEE Computer Society, 2006.
80. R. Prim. Shortest connection networks and some generalizations. *Bell System Technical Journal, The*, 36(6):1389–1401, Nov 1957.
81. S.S. Rizvi, M.A. Jafri, K.M. Elleithy, and A. Riasat. A novel optimization of the distance source routing (DSR) protocol for the mobile ad hoc networks (MANET). In Tarek M. Sobh, Khaled M. Elleithy, and Ausif Mahmood, editors, *Novel Algorithms and Techniques in Telecommunications and Networking, proceedings of the 2008 International Conference on Telecommunications and Networking (TeNe), part of the International Joint Conferences on Computer, Information, and Systems Sciences, and Engineering, CISSE 2008, Bridgeport, Connecticut, USA*, pages 269–274. Springer, 2008.
82. P. Santi. Topology control in wireless ad hoc and sensor networks. *ACM Comput. Surv.*, 37(2):164–194, June 2005.
83. S.W. Saputro, E.T. Baskoro, A.N.M. Salman, D. Suprijanto, and A.M. Baca. The metric dimension of regular bipartite graphs. *arXiv/1101.3624*, 2011.
84. Y. Sasson, D. Cavin, and A. Schiper. Probabilistic broadcast for flooding in wireless mobile ad hoc networks. In *2003 IEEE Wireless Communications and Networking, WCNC 2003, New Orleans, LA, USA, 16-20 March, 2003*, pages 1124–1130. IEEE, 2003.
85. A. Sebö and E. Tannier. On metric generators of graphs. *Mathematics of Operations Research*, 29(2):383–393, 2004.
86. B. Shanmukha, B. Sooryanarayana, and K.S. Harinath. Metric dimension of wheels. *Far East Journal of Applied Mathematics*, 8(3):217–229, 2002.
87. M. Sheng, J. Li, and Y. Shi. Relative degree adaptive flooding broadcast algorithm for ad hoc networks. *TBC*, 51(2):216–222, 2005.

88. P. Slater. Leaves of trees. *Congressus Numerantium*, 14(549-559):37, 1975.
89. R. Tarjan and J. van Leeuwen. Worst-case analysis of set union algorithms. *J. ACM*, 31(2):245–281, March 1984.
90. M.-J. Tsai, H.-Y. Yang, B.-H. Liu, and W.-Q. Huang. Virtual-coordinate-based delivery-guaranteed routing protocol in wireless sensor networks. *IEEE Transactions on Networking*, 17(4):1228–1241, 2009.
91. G. Wang, W. Zhang, Guohong Coa, and T. La Porta. On supporting distributed collaboration in sensor networks. In *Military Communications Conference, 2003. MILCOM '03. 2003 IEEE*, volume 2, pages 752–757 Vol.2, Oct 2003.
92. B.M. Waxman. Routing of multipoint connections. *Selected Areas in Communications, IEEE Journal on*, 6(9):1617–1622, Dec 1988.
93. S. Yessad, L. Bouallouche-Medjkoune, and D. Aïssani. Comparison of routing protocols in wireless sensor networks. In Hassane Alla and Louiza Bouallouche-Medjkoune, editors, *Proceedings of the 8th International Workshop on Verification and Evaluation of Computer and Communication Systems, VECoS 2014, Bejaïa, Algeria, September 29-30, 2014.*, volume 1256 of *CEUR Workshop Proceedings*, pages 135–142. CEUR-WS.org, 2014.
94. E.W. Zegura, K.L. Calvert, and S. Bhattacharjee. How to model an internetwork. In *INFOCOM '96. Fifteenth Annual Joint Conference of the IEEE Computer Societies. Networking the Next Generation. Proceedings IEEE*, volume 2, pages 594–602 vol.2, Mar 1996.
95. Y. Zhang and W. Li. An integrated environment for testing mobile ad-hoc networks. In *Proceedings of the 3rd ACM Interational Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc 2002, June 9-11, 2002, Lausanne, Switzerland*, pages 104–111. ACM, 2002.
96. A.M. Zungeru, L.M. Ang, and K.P. Seng. Classical and swarm intelligence based routing protocols for wireless sensor networks: A survey and comparison. *J. Network and Computer Applications*, 35(5):1508–1536, 2012.

List of Figures

List of Tables

Contributions to Papers with Multiple Authors

I wrote, or contributed to, the following papers that are related to the content of this dissertation.

[47]: The majority of this research has been conducted by myself and I wrote this paper. My co-author Egon Wanke contributed proofreading as well as suggestions regarding motivation, proofs and formalization.

[48]: The majority of this research has been conducted by myself and I wrote this paper. My co-author Egon Wanke contributed proofreading as well as helpful ideas regarding some theorems, simulation setup and evaluation.

[42]: Based on ideas from the master thesis of Alina Elterman, this paper has been written by Egon Wanke and myself with equal contributions.

[45]: This report has been written by Egon Wanke and myself with equal contributions.

[43]: During discussions and supervision of the bachelor thesis of Thomas Kampermann, I contributed to the formulation of lemmata and theorems and suggested the current formalization of the main proofs.

[44]: During discussions I contributed the reduction for a NP-completeness result and suggestions regarding a proof.

[46]: This publication is a short version of a paper that has been developed in equal parts by Egon Wanke and myself.

[36]: I contributed proofreading, ideas regarding motivation and data for simulations to this paper.