Computer Science 1
Computer Science Department

Nils Olberg

Approximation Algorithms for Group Communication Networks

Master Thesis
March 28, 2016

Thesis advisor: Priv.-Doz. Dr. Walter Unger
Second advisor: Prof. Rajmohan Rajaraman

Erklärung

Hiermit versichere ich, dass ich die Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt, sowie Zitate kenntlich gemacht habe.

Aachen, den 28.03.2016

_____

(Nils Olberg)

Contents

1 Introduction **1**
1.1 Preliminaries 2
1.2 Contributions of This Thesis 9

2 Communication Network Problems **11**
2.1 Broadcasting 11
2.2 Gossiping 14
2.3 Multicasting 16
2.4 Concurrent Multicasting 18
2.5 Summary 23

3 Radio Concurrent Multicasting **25**
3.1 Simple Upper Bounds 25
3.2 Sparse Partition Covers 28

4 Minimum Degree Diameter-Bounded Networks **35**
4.1 Randomized Rounding Approaches 38
4.2 Concurrent Multicast Algorithm Modification 41
4.3 Minimum Total-Degree Diameter-Bounded Group Networks . . . 46

5 Minimum Degree Graph Spanners **55**
5.1 Bicriteria Approximation Algorithms 56

6 Conclusion and Further Work **59**

Chapter 1

Introduction

Consider a communication network in which parties are supposed to distribute information within the network. The problem of planning communication networks arises in many real-world applications, e.g., the design of server clusters or sensor networks. A natural way to model such a network is to represent the parties by nodes of a graph and the communication links between parties by edges. Several communication models have been introduced for problems regarding message dissemination in networks. We focus our attention on two models to which we refer to as the telephone model [Rav94] and the radio model [IRRS15]. In both of these communication models we assume that the exchange of messages occurs at discrete time steps. Further, these models omit capacity restrictions for the communication links.

Problems that concern the distribution of messages in networks are often affiliated with the computation of sparse subgraphs of the underlying network graph. In order to guarantee low latency, two parties that are supposed to exchange information should not be separated by many hops in the network. Moreover, it is desirable that a party is only required to exchange information with few other parties. In the graph induced by the communication, a small number of hops and low connectivity correspond to a small distance between two nodes and a small maximum degree.

All optimization problems that we consider in this thesis are NP-hard. Hence, they cannot be solved optimally in polynomial time unless P=NP. Therefore, we focus on the development of approximation algorithms for these problems.

We are particularly interested in developing approximation algorithms for the concurrent multicast problem. This problem is a generalization of message dissemination problems such as broadcasting and gossiping, which have been studied intensively in the past decades. It is an open question whether the concurrent multicast problem in the telephone model allows for a polylogarithmic

approximation algorithm. Further, to the best of our knowledge there are no known results for radio concurrent multicasting until now.

1.1 Preliminaries

Graphs and Networks

For an undirected graph $G = (V(G), E(G))$ we denote the number of nodes by $n = |V(G)|$ and the number of edges by $m = |E(G)|$. Let $u, v \in V(G)$ be two nodes of G. A path from u to v in G is a sequence of nodes $u, ..., v$, such that for two consecutive nodes there is a corresponding edge in $E(G)$ and no node is repeated. The length of a path is the number of edges used. Let $\text{dist}(G, u, v)$ denote the distance between two nodes u and v in G, which is the length of a shortest path from u to v. If G is not connected and there is no such path, then $\text{dist}(G, u, v) = \infty$. We call $\text{rad}(G, v) = \max\{\text{dist}(G, v, v') \mid v' \in V(G)\}$ the *radius* of v in G and the *diameter* of graph G is defined as $\text{diam}(G) = \max\{\text{rad}(G, u', v') \mid u', v' \in V(G)\}$. For every graph G it holds that $\text{rad}(G, v) \leq \text{diam}(G) \leq 2 \cdot \text{rad}(G, v)$ for all $v \in V(G)$.
Given a pair (G, P), where G is a graph and $P = \{(s_1, t_1), ..., (s_k, t_k)\} \subseteq V(G) \times V(G)$, we refer to P as *sender-receiver pairs*. Further, we call $S_P = \{s_1, ..., s_k\}$ the senders, $R_P = \{t_1, ..., t_k\}$ the receivers and $V_P = S_P \cup R_P$ the terminals of instance (G, P). The diameter of (G, P) is defined as $\text{diam}(G, P) = \max\{\text{dist}(G, s_i, t_i) \mid i \in \{1, ..., k\}\}$.
The degree of v in G, denoted by $\deg(G, v)$, is the number of edges incident to v. We call $\Delta(G) = \max\{\deg(G, v') \mid v' \in V(G)\}$ the *maximum degree* of G. For a node $v \in V(G)$ let $N_\ell(G, v)$ denote the ℓ-neighborhood of v, which is the set of nodes that can be reached via a path of length at most ℓ from v. Moreover, $N(G, v)$, or simply $N(v)$ whenever it is clear from the context, is the set of *neighbors* of v, i.e., $N(G, v) = \{u \in V(G) \mid \{u, v\} \in E(G)\}$. Analogously, for a given set $S \subseteq V(G)$ we define the ℓ-neighborhood of S, namely $N_\ell(G, S)$, to be the nodes that can be reached from an arbitrary node $u \in S$ via a path of length at most ℓ and that are not contained in S. Let $\deg(G, S) = |\{\{u, v\} \in E(G) \mid u \in S, v \notin S\}|$ denote the degree of S in G. We further call $\text{bnd}(S) = \{v \in S \mid \exists u \in N_1(G, v) : u \notin S\}$ the *boundary* of S.
The graph *induced* by a subset of nodes S is the subgraph $G[S] = (S, E(S))$, where $E(S) \subseteq E(G)$ contains all edges with both endpoints in S. For a graph G and $k \in \mathbb{N}$, we call G^k the k-th power of G. G^k is the graph where each node of G is also connected to every node in its k-neighborhood, or formally $G^k = (V(G), E^k)$ with $E^k = \{\{u, v\} \mid u \in N_k(G, v), v \in V(G)\}$.

A set of edges $M \subseteq E(G)$ is a *matching* in G, if the graph $G' = (V(G), M)$ has maximum degree $\Delta(G') \leq 1$. An *independent set* is a subset of nodes $S \subseteq V(G)$, such that for all $u, v \in S$: $u \notin N(G, v)$. A subgraph T of G is called a *Steiner tree* for graph G with *terminals* $S \subseteq V(G)$, if T ensures that all nodes of S are connected. An inner-arborescence T is a directed tree, such that all edges are pointing towards its root r. Analogously, an outer-arborescence T is a directed tree with root r where all edges point away from r.

Telephone and Radio Model

Network communication problems such as broadcasting, gossiping or multicasting have been studied in the literature using different communication models. We consider the two of the most established models: the *radio* and the *telephone* model. In both models, the transmission of messages occurs at discrete time steps, which we call *rounds*. Often, our goal is to minimize the number of rounds to route messages from senders to receivers in a given network.
We use graphs to model communication networks as follows: the nodes in a graph G represent the parties of a network. In the radio model, a node is either *transmitting* or *idle* in each round. If a node v is transmitting, all the messages it has received so far are sent to all of its neighbors. A node $u \in N(G, v)$ can receive those messages if and only if u is idle and v is the only neighbor transmitting in the current round. We call the event where more than one node in the neighborhood of u transmits in a round a *collision* at u. A *schedule* in the radio model is a sequence $\mathcal{S} = S_1, ..., S_t$, where $S_i \subseteq V(G)$ is the set of nodes that are transmitting in round i. The length of schedule $\mathcal{S} = S_1, ..., S_t$ is the number of rounds in $\mathcal{S}$ and is further denoted by $|\mathcal{S}|$.
It is often a desirable property for a schedule to be collision-free in order to be sure that messages are received by $N(G, v)$ when v is transmitting. The next observation shows why the graph G^2 can be useful when working with radio schedules.

Observation 1.1. *A radio schedule $\mathcal{S} = S_1, ..., S_k$ has no collisions if and only if S_i is an independent set in G^2 for all $i \in \{1, ..., k\}$.*

The telephone model allows nodes to talk to at most one of their neighbors in one round. If u and v are neighbors and talk to each other in round i, they exchange all messages they have received so far, and thus share the same information after this round. Formally, a schedule in the telephone model is a sequence of matchings $\mathcal{S} = M_1, ..., M_t$, where an edge $\{u, v\} \in E(M_i)$ indicates that u and v exchange messages in round i. Analogously to the radio model, we define the length $|\mathcal{S}|$ of schedule $\mathcal{S}$ to be the length of the sequence.

Observation 1.2. *Let $\mathcal{S} = M_1, ..., M_t$ be a telephone schedule that ensures that node u sends its message to v. Then schedule $\mathcal{S}_R$, which is schedule $\mathcal{S}$ reversed, i.e., $\mathcal{S}_R = M_t, ..., M_1$, sends the message of v to u.*

Note that this observation does not hold for the radio model in general. In both models, $\mathcal{S}^\ell$ denotes the schedule where $\mathcal{S}$ is repeated ℓ times and hence $|\mathcal{S}^\ell| = \ell \cdot |\mathcal{S}|$. Further, $\mathcal{S} = \mathcal{S}_1\mathcal{S}_2$ is a schedule that first executes $\mathcal{S}_1$ and afterwards $\mathcal{S}_2$. It follows that $|\mathcal{S}| = |\mathcal{S}_1| + |\mathcal{S}_2|$.

Approximation Algorithms

Since all problems discussed in the following chapters are NP-hard, we can not compute optimal solutions for these problems in polynomial time, unless P = NP. However, there are approximation algorithms that run in polynomial time and have, unlike heuristics, quality guarantees for their output [WS11]. For all optimization problems that are studied in this thesis the objective value should be minimized. Therefore, we only give a definition of approximation algorithms for *minimization* problems. The definition for maximization problems is analogous.

Definition 1.3. *Let $\mathcal{P}$ be a minimization problem. An algorithm* ALG *is called an α-approximation for $\mathcal{P}$ if for every instance G of $\mathcal{P}$ with optimal solution value* OPT *the following holds:*

$$\text{val}(\text{ALG}(G)) \leq \alpha \cdot \text{OPT}$$

where ALG(G) *is the solution provided by algorithm* ALG *for instance G.*

We do not restrict α to be a constant, but also functions depending on a property of the instance, such as the number of nodes or the maximum degree. If $\alpha = c$ for some constant $c > 1$, we call the algorithm a *constant* approximation. Similarly, if $\alpha = O(\log n)$ or $\alpha = O(\log^c n)$ for a constant $c > 0$, where n denotes the size of the instance, ALG is referred to as a *logarithmic* or *polylogarithmic* approximation for $\mathcal{P}$ respectively.

Definition 1.4. *Let $\mathcal{P}$ be a minimization problem. An algorithm* ALG *is called an α-approximation with additive term β for $\mathcal{P}$ if for every instance G of $\mathcal{P}$ with optimal solution value* OPT *the following holds:*

$$\text{val}(\text{ALG}(G)) \leq \alpha \cdot \text{OPT} + \beta$$

where ALG(G) *is the solution provided by algorithm* ALG *for instance G.*

Hence, an additive approximation tolerates an additional violation of β. It follows that every α-approximation for a problem $\mathcal{P}$ can be interpreted as an α-approximation with additive term 0 for $\mathcal{P}$.
Whenever we talk about approximation algorithms in this thesis, we implicitly assume that their running time is polynomial in the size of the input.

Randomized Rounding

Randomized rounding is a common technique for designing approximation algorithms and was first described by Raghavan and Thompson [RT87]. The main idea is to convert a fractional solution of a linear program relaxation into an integral solution for the original integer linear program. We explain the concept of randomized rounding using the example of the *minimum set cover problem*, as done in [Sou]. The minimum set cover problem is an intensively studied NP-hard problem and can be formulated as follows.

Definition 1.5. *Given a set of elements $\mathcal{U}$ (the universe), a collection of sets $\mathcal{S} = \{S_1, ..., S_k\}$, such that $S_i \subseteq \mathcal{U}$ and $\bigcup_{i=1}^{k} S_i = \mathcal{U}$, and a cost function $c : \mathcal{S} \to \mathbb{R}^+$. The minimum set cover problem is to find a subcollection $\mathcal{S}' \subseteq \mathcal{S}$ with minimal costs $c(\mathcal{S}')$, where $c(\mathcal{S}') = \sum_{S \in \mathcal{S}'} c(S)$, such that $\cup_{S \in \mathcal{S}'} S = \mathcal{U}$.*

Here is an example instance for the minimum set cover problem: let $\mathcal{U} = \{1, 2, 3\}$ and $\mathcal{S} = \{S_1, S_2, S_3\}$, where $S_1 = \{1, 2\}$, $S_2 = \{1, 3\}$, $S_3 = \{1, 2, 3\}$, $c(S_1) = 2$, $c(S_2) = 3$ and $c(S_3) = 8$. First of all, note that $(\mathcal{U}, \mathcal{S})$ is a valid instance since $S_1 \cup S_2 \cup S_3 = \{1, 2, 3\} = \mathcal{U}$. The collection $\mathcal{S}' = \{S_1, S_2\}$ is an optimal solution in this case because $\mathcal{S}'$ *covers* the universe, i.e., $S_1 \cup S_2 = \mathcal{U}$, and there is no other solution $\mathcal{S}''$ that covers the universe with $c(\mathcal{S}'') < c(\mathcal{S}') = 5$.
The minimum set cover program can be expressed as an integer linear program as follows.

$$\text{minimize} \sum_{S \in \mathcal{S}} c(S) x_S$$

$$\text{subject to:} \sum_{S : u \in S} x_S \geq 1 \quad u \in S \tag{1.1}$$

$$x_S \in \{0, 1\} \quad S \in \mathcal{S} \tag{1.2}$$

For every set $S \in \mathcal{S}$ we introduce a variable x_S, which indicates whether S is contained in solution $\mathcal{S}'$ or not. The objective function ensures that we choose a subset of $\mathcal{S}$ of minimal cost. Constraints 1.1 further guarantee that every element of the universe is covered by at least one set in the solution.

Solving integer linear programs in general is known to be NP-hard. However, for our purposes we consider the relaxation of the linear program.

$$\begin{aligned} \text{minimize} \quad & \sum_{S \in \mathcal{S}} c(S) x_S \\ \text{subject to:} \quad & \sum_{S: u \in S} x_S \geq 1 \quad u \in S \qquad (1.3) \\ & 0 \leq x_S \leq 1 \quad S \in \mathcal{S} \qquad (1.4) \end{aligned}$$

Let x^* be an optimal solution to the relaxation with objective value $z = \text{val}(x^*)$. Note that for every x_S^* we have that $0 \leq x_S^* \leq 1$, and thus we can interpret the value of x_S^* as a probability that set S is chosen for an integral solution. Algorithm 1, which is a randomized algorithm, determines an optimal fractional solution and then turns it into an integral solution. In other words, the algorithm *rounds* a fractional to an integral solution.

Algorithm 1 MINIMUM SET COVER

Input: $(\mathcal{U}, \mathcal{S})$

1. Solve the linear program relaxation for the instance $(\mathcal{U}, \mathcal{S})$ and let x' be the provided optimal solution.
2. Let $x \in \{0,1\}^{|\mathcal{S}|}$ with $x_S = 0$ for all $S \in \mathcal{S}$.
3. **for** $1, ..., 2\lceil \ln n \rceil$ **do**
4. **for** $S \in \mathcal{S}$ **do**
5. Set $x_S = 1$ with probability x'_S.
6. **end for**
7. **end for**

Output: x

Theorem 1.6. *Algorithm 1 is a $2\lceil \ln n \rceil$-approximation for the minimum set cover problem in expectation.*

Proof. In each of the $2\lceil \ln n \rceil$ iterations, x_S is set to 1 with probability x'_S for each $S \in \mathcal{S}$. It follows that the increase of $\text{val}(x)$ is at most

$$\sum_{S \in \mathcal{S}} \mathbb{E}[c(S) x_S] = \sum_{S \in \mathcal{S}} c(S) \cdot \Pr[x_S = 1] = \sum_{S \in \mathcal{S}} c(S) x'_S = \text{val}(x')$$

in one iteration. Note that $\text{val}(x')$ is a lower bound for the optimal integral solution. It follows that $\mathbb{E}[\text{val}(x)] \leq 2\lceil \ln n \rceil \cdot \text{val}(x') \leq 2\lceil \ln n \rceil \cdot \text{OPT}$.
For the rest of the proof we show that x is feasible with high probability. Therefore, we estimate the probability that element $u \in \mathcal{U}$ is covered in one iteration by the solution induced by x. Without loss of generality, suppose u is contained in t sets $S_1, ..., S_t$. Since x' is feasible for the linear program

relaxation, we know by Constraints 1.3 that $\sum_{i=1}^{t} x'_{S_i} \geq 1$. The probability of the event 'u is not covered in one iteration' is maximized when $x'_{S_i} = \frac{1}{t}$ for all $i \in \{1, ..., t\}$, i.e., every of the t sets is equally likely to be added to the solution. We can bound this probability as follows.

$$\Pr[u \text{ is not covered in one iteration}] \leq \Pi_{i=1}^{t}(1 - x'_{S_i}) \leq \left(1 - \frac{1}{t}\right)^t \leq \frac{1}{e}$$

There are $2\lceil \ln n \rceil$ iterations, and thus the probability that an element u is not covered at the end of the algorithm is

$$\Pr[u \text{ is not covered}] \leq \left(\frac{1}{e}\right)^{2\lceil \ln n \rceil} \leq \frac{1}{n^2}.$$

Since there are n elements, it follows that the expected number of elements that have not been covered is bounded by $n \cdot \frac{1}{n^2} = \frac{1}{n}$ and thus the solution is feasible with high probability. □

Using *derandomization*, which will not be covered in this thesis, this algorithm can be converted into a deterministic algorithm with the same approximation guarantee.

Multi-Commodity Flows With Length Bound

Consider a graph G together with a set of pairs of vertices $\{(s_1, t_1), ..., (s_k, t_k)\} = P \subseteq V(G) \times V(G)$ and a length bound L. Let G' be the directed version of G, i.e., $G' = (V(G), \{(u, v) \mid \{u, v\} \in E(G)\})$. Each pair $(s_i, t_i) \in P$ represents a commodity i.

Definition 1.7. *The length-bounded minimum node-congestion problem is to find a flow-assignment $f : E(G') \times \{1, ..., k\} \to [0, 1]$, such that*

1. *s_i sends a unit flow to t_i, $i \in \{1, ..., k\}$,*
2. *the length of each i-flow path is bounded by L,*
3. *and the maximum node-congestion $\text{val}(f) = \max_{v \in V(G)} c(v)$ is minimized.*

The congestion of node v is defined as $c(v) = \sum_{i=1}^{k} \sum_{u \in N(v)} f_i(v)$, where $f_i(v)$ is the flow that belongs to commodity i and that goes trough v. The following linear program captures the length-bounded minimum node-congestion problem.

minimize C

subject to

$$\sum_{(s_i,u)\in E(G')} x_{i,1,s_i,u} = 1 \qquad i = 1,...,k \qquad (1.5)$$

$$\sum_{\ell=2}^{L} \sum_{(u,t_i)\in E(G')} x_{i,\ell-1,u,t_i} - x_{i,\ell,t_i,u} = 1 \qquad i = 1,...,k \qquad (1.6)$$

$$\sum_{\ell=2}^{L} \sum_{(u,v)\in E(G')} x_{i,\ell-1,u,v} - x_{i,\ell,v,u} = 0 \qquad i = 1,...,k \quad v \in V(G') \setminus \{t_i\} \qquad (1.7)$$

$$\sum_{\ell=1}^{L} \sum_{(u,v)\in E(G')} x_{i,\ell,u,v} \leq C \qquad u \in V(G') \qquad (1.8)$$

$$\sum_{\ell=1}^{L} \sum_{(u,v)\in E(G')} x_{i,\ell,u,v} \leq C \qquad v \in V(G') \qquad (1.9)$$

$$x_{i,\ell,u,v} \in \{0,1\} \qquad (u,v) \in E(G') \quad i = 1,...,k \quad \ell = 1,...,L-1 \qquad (1.10)$$

The objective function guarantees to minimize the maximum node-congestion. Variable $x_{i,\ell,u,v}$ indicates how much flow of commodity i is sent from u to v at distance ℓ from source s_i. Equations 1.5 ensure that a unit flow leaves the source of every commodity. Similarly, Constraints 1.6 ensure that each sink receives the corresponding unit flow. Constraints 1.7 represent the flow conservation for each commodity. Finally, Constraints 1.8 and 1.9 ensure that the maximum node-congestion is updated correctly.

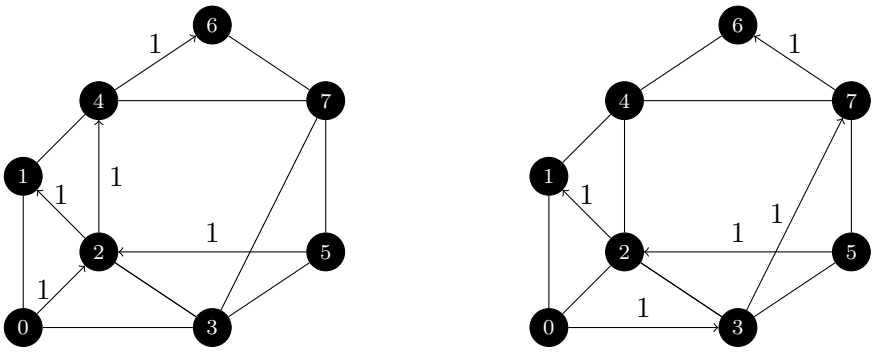

Figure 1.1: Two flow-assignments for pairs $P = \{(0,6),(5,1)\}$. In the left assignment node 2 has congestion $c(2) = 2$. In the right assignment $c(2) = 1$.

Albeit finding an integral flow-assignment of minimal node-congestion is well known to be NP-hard [Rav94], using randomized rounding we can derive a good approximation for this problem. The following result is one of the foundations for Algorithm 11.

Theorem 1.8 ([Rav94])**.** *Let (G, P) be an instance of the length-bounded minimum node-congestion problem with optimal solution f^*. In polynomial time, one can compute a feasible flow f for (G, P) using randomized rounding, such that* $\mathrm{val}(f) = O(\mathrm{val}(f^*) + \log n)$.

Further Background

Theorem 1.9 plays a crucial role for the development of two algorithms in this thesis.

Theorem 1.9 ([EK06])**.** *Suppose there exists a Steiner tree for graph G with terminals $S \subseteq V(G)$, diameter L and maximum degree D. In polynomial time, we can compute a Steiner tree T for G with terminals S, such that* $\mathrm{diam}(T) = O\left(\frac{\log |S|}{\log \log |S|} \cdot L\right)$ *and* $\Delta(T) = O\left(\frac{\log |S|}{\log \log |S|} \cdot D\right)$.

For the development of Algorithm 11, we further require the next two results, which are mentioned in [Rav94].

Theorem 1.10 ([Rav94])**.** *Let T be a tree with an even number of marked nodes. Then there is a pairing $(u_1, v_1), ..., (u_k, v_k)$ of of the marked nodes, such that the (u_i, v_i)-paths in T are edge-disjoint.*

Lemma 1.11 ([Rav94])**.** *Let G be a directed graph in which every node has exactly one outgoing edge. In polynomial time, one can find an edge-induced subgraph H of G, such that*

1. *H is a forest of directed trees,*
2. *each of these trees is an inward arborescence of at least two nodes,*
3. *every node of G is contained in exactly one of the trees.*

1.2 Contributions of This Thesis

To the best of our knowledge, we provide the first non-trivial upper bounds for concurrent multicasting in the radio model. Further, we introduce two network design problems, namely the minimum degree diameter-bounded network problem and the minimum total-degree diameter-bounded group network problem, which are both related to concurrent multicasting in the telephone model. It is shown that the algorithm for telephone concurrent multicasting of Nikzad and Ravi [NR14] can be modified to provide a $(2^{O(\log \log k \cdot \sqrt{\log k})}, 2^{O(\log \log k \cdot \sqrt{\log k})})$-bicriteria approximation for the minimum degree diameter-bounded network problem, where k corresponds to the number of terminals. For the minimum

total-degree diameter-bounded group network problem we are able to derive a polylogarithmic bicriteria approximation algorithm. We argue that the bicriteria approximation algorithm for minimum degree diameter-bounded networks implies a $(2^{O(\log\log n\cdot\sqrt{\log n})}, 2^{O(\log\log n\cdot\sqrt{\log n})})$-bicriteria approximation for minimum degree graph spanners. Moreover, we show that the minimum degree k-spanner problem admits an $\left(O\left(\frac{\log^2 n}{\log\log n}\cdot k\right), O\left(\frac{\log^2 n}{\log\log n}\right)\right)$-bicriteria approximation.

Chapter 2

Communication Network Problems

This chapter addresses problems regarding message dissemination in the radio and the telephone model, namely broadcasting, gossiping, multicasting and concurrent multicasting. With the exception of the concurrent multicast problem, which is a generalization of broadcasting, gossiping and multicasting, each of these problems has been studied extensively in both communication models. We present known results and point out some interesting properties of these problems. In the last section, we give a short summary of all upper and lower bounds that are mentioned in this thesis.

2.1 Broadcasting

Consider a network represented by an undirected graph G, where one of the nodes of G, namely $s \in V(G)$, holds a message M_s. A broadcast schedule for instance (G, s) is a schedule $\mathcal{S}$, which disseminates M_s in the whole network, i.e., every node of G receives M_s at some point.

Definition 2.1. *The broadcast problem is to find a broadcast schedule of minimum length.*

A trivial lower bound for the minimum broadcast time in both communication models is $\mathrm{rad}(G, s)$ since a node v can only forward the message over a link to nodes in $N(G, v)$.

Telephone Model

In the telephone model, a broadcast schedule can be illustrated as a pair (T, f), where T is an outer-arborescence and $f : E(T) \to \mathbb{N}$ maps each edge of the arborescence to a natural number. The value $f(u, v)$ represents the round in which v first received the message M_s and u was the node sending M_s to v.

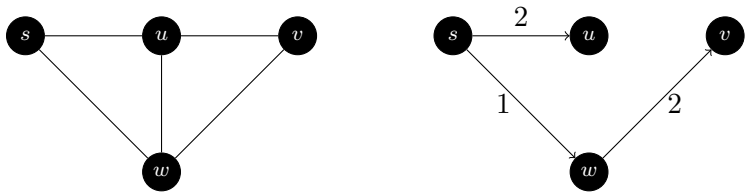

Figure 2.1: Network G and a broadcast schedule (T, f) for (G, s).

For a tree T we define the poise of T to be the sum of its diameter and its maximum degree, i.e., $\text{poise}(T) = \text{diam}(T) + \Delta(T)$.

Definition 2.2. *Given a graph G, the minimum poise spanning tree problem is to find a spanning tree T for G with minimal poise.*

A reduction from the Hamiltonian path problem shows that the minimum poise spanning tree problem is NP-hard [Rav94]. Let T be a spanning tree for an arbitrary graph G with n vertices. A lower bound for $\text{poise}(T)$ is $\Omega\left(\frac{\log n}{\log\log n}\right)$ because a spanning tree has least poise if it has branching k and k levels. It holds that $k = \Omega\left(\frac{\log n}{\log\log n}\right)$ if $n = k^k$.
In order to illustrate the relation between the poise of a spanning tree and a schedule, let (G, s) be an instance of the broadcast problem and let (T, f) be an optimal solution for (G, s). The undirected subgraph T' of G corresponding to T is a spanning tree for G and has small poise because of two reasons: if the distance between s and an arbitrary node v would be large, it would take many rounds for M_s to arrive at v because in one round a node can only share information with a direct neighbor. Hence, $\text{diam}(T')$ should be small. At the same time, the maximum degree of T' is small because every node can talk to at most one neighbor in each round. If v has degree d, then v needs at least d rounds to send the message to all of its neighbors.
The following theorem highlights the connection between the minimum poise spanning tree problem and broadcasting in the telephone model.

Theorem 2.3 ([Rav94])**.** *Let G be a graph, T a minimum poise spanning tree for G and $s \in V(G)$. Further,* br *denotes the length of an optimal broadcast schedule for (G, s) in the telephone model.*

$$\Omega(\text{poise}(T)) \leq \text{br}(G, s) \leq \Omega\left(\frac{\log n}{\log\log n} \cdot \text{poise}(T)\right)$$

Albeit this correlation, the maximum degree in contrast to the radius of s, which is always within a factor of 2 of the diameter, is not a lower bound for $\text{br}(G, s)$. Consider a graph G, which consists of a binary tree T of height h

with root s and an additional node t, as illustrated in Figure 2.2. Node t is further connected to all leaves of T. Since T has $\Theta(n)$ leaves, we have that $\Delta(G) = \Theta(n)$ as well. However, there is a schedule $\mathcal{S}$ that distributes M_v for an arbitrary node $v \in V(G)$ in $O(\log n)$ rounds. The message can be delivered to s in $\lceil \log n \rceil + 1$ rounds because $\text{diam}(G) \leq \lceil \log n \rceil + 1$. From there each parent in T sends M_v to both children. After $2\lceil \log n \rceil$ rounds, each node of T has received the message. The leaf of T that first receives M_v can forward the message to t.

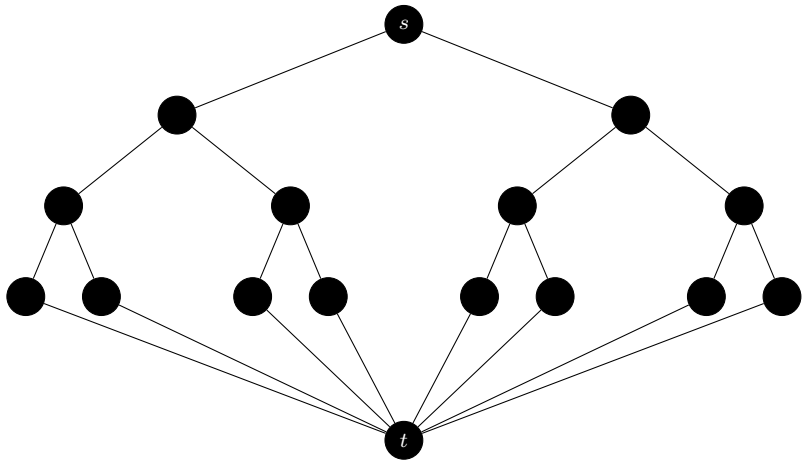

Figure 2.2: Example graph with $h = 4$. Although $\Delta(G) = \Omega(n)$, the gossiping task can be completed in $O(\log n)$ rounds.

Apart from the radius of s in graph G, $\lceil \log n \rceil$ is a lower bound for $\text{br}(G, s)$ in the telephone model since in every round the number of informed nodes can at most double. Computing a minimum length broadcast schedule in the telephone model is known to be NP-hard, even for 3-regular planar graphs [Rav94]. It is further NP-hard to approximate the broadcast problem within a factor of $3 - \epsilon$ for any $\epsilon > 0$ [EK05].
The first polylogarithmic approximation algorithm for the broadcast problem in the telephone model was presented by Ravi. This algorithm achieves an approximation factor of $O\left(\frac{\log^2 n}{\log \log n}\right)$. The currently best approximation algorithm is an $O\left(\frac{\log n}{\log \log n}\right)$-approximation by Elkin and Kortsarz [EK06].

Radio Model

The broadcast problem has also been studied intensively in the radio model. A first notable difference between telephone and radio broadcasting is the lack of the relationship between a minimum poise spanning tree and the minimum broadcast time. Given a star-graph G with center node s, it is easy to see that

$$\text{poise}(G) = \text{diam}(G) + \Delta(G) = 2 + n - 1 = n + 1.$$

Yet a schedule $\mathcal{S} = S_1$ with $S_1 = \{s\}$, i.e., a schedule in which only s transmits once, is a valid solution to the broadcast problem.
It has been shown that radio broadcasting is NP-hard, even for planar graphs [GPX07]. Several suggestions have been made for lower bounds of approximation algorithms different from the diameter. Alon et al. showed that there is a family of graphs where the minimal broadcast time is at least $\Omega(\text{diam}(G) + \log^2 n)$ [ABLP91]. This result tightly matches the currently best known approximation algorithm for the broadcast problem by Kowalski and Pelc.

Theorem 2.4 ([KP07])**.** *Given a graph G and a node $s \in V(G)$. In polynomial time we can compute a radio broadcast schedule $\mathcal{S}$ of length $O(\text{diam}(G)+\log^2 n)$ for (G, s).*

2.2 Gossiping

Another prominent communication network problem is *gossiping*. Suppose each node v of a graph G holds a message M_v. Each of these messages should be distributed to all nodes in the network. There are models where nodes can only send a certain number of messages per round. We allow nodes to pass on all the messages they have received so far once they are talking to a neighbor or transmitting depending on the communication model. A gossip schedule $\mathcal{S}$ ensures that M_v, $v \in V(G)$, is received by every node in G.

Definition 2.5. *Given a graph G, the gossip problem is to find a gossip schedule for G of minimum length.*

Telephone Model

In the telephone model, the length of an optimal gossip schedule can be related to the length of an optimal broadcast schedule as follows.

Theorem 2.6. *Let $\text{gs}(G)$ denote the length of an optimal gossip schedule for graph G. For any graph G and $s \in V(G)$ it holds that*

$$\text{br}(G, s) \leq \text{gs}(G) \leq 2\text{br}(G, s),$$

where $\text{br}(G, s)$ is the length of an optimal broadcast schedule for (G, s).

Proof. The first inequality is trivial since a gossip schedule is always also a valid broadcast schedule for an arbitrary sender $s \in V(G)$. The second inequality

is further easily proven by showing the existence of a gossip schedule of length $2\text{br}(G, s)$. Let $\mathcal{S} = M_1, ..., M_t$ be an optimal broadcast schedule for an arbitrary node s. Consider the reversed schedule $\mathcal{S}_R = M_t, ..., M_1$. If schedule $\mathcal{S}_R$ is executed all messages have been gathered in node s. Running $\mathcal{S}$ afterwards ensures that all vertices receive all messages since $\mathcal{S}$ is a broadcast schedule for s. Both schedules have length $\text{br}(G, s)$, which completes the proof. □

Hence, results for the broadcast problem immediately imply results for the gossip problem and vice versa. It follows that it is NP-hard to approximate the gossip problem in the telephone model within a factor of $3/2 - \epsilon$ for any $\epsilon > 0$. Moreover, we can conclude that there is a sublogarithmic $O\left(\frac{\log n}{\log \log n}\right)$-approximation for gossiping by modifying the approximation algorithm for broadcasting by Elkin and Kortsarz.

Radio Model

Another difference between radio and telephone model is the relationship between broadcasting and gossiping. In the radio version, it is generally not sufficient to run a broadcast schedule reversed and then in the original order to distribute all messages in the network. For a star-graph G with n nodes and a center node s it is not hard to see that $n \cdot \text{br}(G, s) \leq \text{gs}(G)$, where $\text{gs}(G)$ is the length of an optimal gossip schedule for G and $\text{br}(G, s)$ is the length of an optimal broadcast schedule for (G, s). As mentioned earlier, there is an optimal broadcast schedule $\mathcal{S} = \{s\}$ of length $|\mathcal{S}| = 1$. However, a gossip schedule requires at least n rounds because every node has to transmit at least once. If $u, v \in V(G) \setminus \{s\}$ were to transmit at the same time there would be a collision and their unique neighbor s would not receive any messages.
So far, the best upper bound for radio gossiping was published by Gasieniec et al.

Theorem 2.7 ([GPX07])**.** *For an arbitrary graph G a schedule $\mathcal{S}$ of length $O(\text{diam}(G) + \Delta(G) \log n)$ can be computed in polynomial time.*

The proof and the algorithm can be found in [GPX07]. It is further shown that every planar graph G admits an efficiently computable gossip schedule of length $3 \cdot \text{diam}(G)$.
It is worth emphasizing that the maximum degree is not a lower bound for the minimal gossip time. This can be illustrated by the same graph family we used to argue that this holds for broadcasting in the telephone model. An example graph is illustrated in Figure 2.2. The minimal gossip time for a graph of this family is $O(\log n)$, albeit $\Delta(G) = \Omega(n)$.

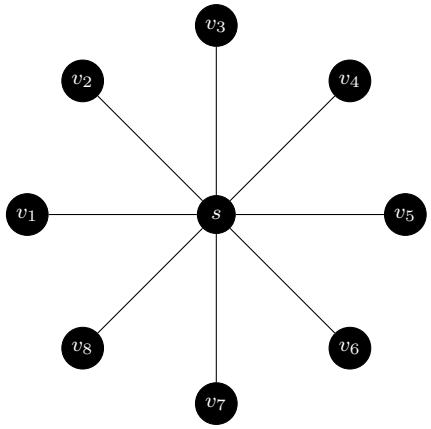

Figure 2.3: The optimal broadcast time for sender s is 1. In contrast, a gossip schedule requires at least n rounds.

The following theorem shows that it is much harder to approximate the radio version of the gossip problem than the telephone version.

Theorem 2.8 ([IRRS15])**.** *It is not possible to approximate the radio gossip problem in polynomial time to within a factor of $n^{1/2-\epsilon}$, $\epsilon > 0$, unless $P = NP$.*

2.3 Multicasting

This section addresses the *multicast problem*. Multicasting is a natural generalization of broadcasting, where instead of informing all nodes of a graph only a subset $R \subseteq V(G)$, which we call the *receivers*, is required to receive message M_s of a sender $s \in V(G)$. Nodes of $V(G) \setminus R$ can still be used to distribute the message in the network. A multicast schedule for an instance (G, s, R) is a schedule $\mathcal{S}$, which makes sure that all nodes of R receive message M_s.

Definition 2.9. *The multicast problem is to find a multicast schedule of minimum length.*

Note that in both the telephone and the radio model it is trivial to give a k-approximation for the multicast problem, where k is the number of receivers, just as it is trivial to give an n-approximation for the broadcast problem. For each receiver $v \in R$ one can compute a shortest path from s to v. Let L denote the length of the longest of these paths, which is always a lower bound for an optimal solution. In at most L rounds, one receiver can obtain M_s by sending the message along the shortest path from s to v. Since there are k receivers and this can be done for each receiver consecutively, which requires no further coordination, this gives a k-approximation.

Telephone Model

Analogous to broadcast schedules we can represent a multicast schedule not only as a sequence of matchings but as a pair (T, f), where T is an outer-arborescence rooted in s and a function $f(E(T)) \to \mathbb{N}$. The difference is that T does not have to span the whole graph G but only connect s to every receiver of R. Since the multicast problem is a more general version of the broadcast problem, it inherits all hardness properties from the latter. Hence, it is NP-hard to approximate the telephone version of the multicast problem within a factor of $3 - \epsilon$ for any $\epsilon > 0$.
There have been several publications suggesting polylogarithmic approximation algorithms for the multicast problem in the telephone model. The currently best upper bound for telephone multicasting is due to an $O\left(\frac{\log k}{\log \log k}\right)$-approximation, which is the same result we mentioned for broadcasting in the telephone model [EK06]. It remains an open question whether there exists a constant approximation for the multicast problem in the telephone model.
Theorem 2.10 shows that multicasting in the telephone model can be approximated within a constant factor for graph classes with constant maximum degree.

Theorem 2.10. *Consider a graph class $\mathcal{G}$ with constant maximum degree, i.e., there exists a constant $c > 0$ such that $\Delta(G) \leq c$ for all $G \in \mathcal{G}$. Then there is a constant approximation algorithm for the multicast problem in the telephone model restricted to this graph class.*

Proof. Given an instance (G, s, R), we show that we can efficiently compute a schedule of length $\Delta(G) \cdot L$, where $L = \max\{\text{dist}(G, s, v) \mid v \in R\}$. Since L is the maximum length of a shortest path from s to a receiver $v \in R$, the graph $G' = G[N_L(G, s)]$, which is the graph induced by the L-neighborhood of s, contains all receivers. Clearly, if we compute an outer-arborescence T rooted in s, which spans the whole graph G', using breadth-first-search, it holds that $\text{rad}(T, s) \leq L$. Further, we observe that $\Delta(T) \leq \Delta(G)$ because T is a subgraph of G. Let $\mathcal{S}$ be a greedy schedule for (G, s, R), i.e., each node sends to each of its children in T in an arbitrary order. Then a node at distance ℓ from s receives the message after at most $\Delta(G) \cdot \ell$ rounds. □

Radio Model

Although the multicast problem has been studied intensively in the telephone model, we are not aware of any results regarding the radio version. From the result for broadcasting we can derive the following statement.

Theorem 2.11. *For graph G with sender s and receivers R one can compute a multicast schedule of length $O(L + \log^2 n)$ in polynomial time, where $L = \max_{v \in R} \operatorname{dist}(G, s, v)$.*

Proof. Take the graph $G' = G[N_L(G, s)]$ induced by the L-neighborhood of s, which contains all receivers of R and where the radius of s is at most L. By Theorem 2.4 we can obtain a broadcast schedule $\mathcal{S}$ of length $O(L + \log^2 n)$ for (G', s, R). The schedule $\mathcal{S}$ informs every node in $V(G')$ and since $R \subseteq V(G')$, this completes the proof. □

2.4 Concurrent Multicasting

Concurrent multicasting is a generalization for both multicasting and gossiping. To the best of our knowledge this problem was introduced in [NR14], where it is referred to as *multi-commodity multicasting*. This name might be misleading since we are not routing commodities through a network but distributing information. Given a graph G and a set of k sender-receiver pairs $P = \{(s_1, t_1), ..., (s_k, t_k)\} \subseteq V(G) \times V(G)$, a concurrent multicast schedule makes sure that a message M_i is sent from s_i to t_i. We further assume that all messages are distinct.

Definition 2.12. *Let G be a graph and $P \subseteq V(G) \times V(G)$. The concurrent multicast problem is to find a concurrent multicast schedule of minimum length for (G, P).*

Two interesting properties of concurrent multicast schedules are captured by the next observations and hold for both communication models.

Observation 2.13. *Let G be a graph and P represent a set of sender-receiver pairs. For all $P' \subseteq P$ it holds that the length of an optimal schedule for (G, P') is at most the length of an optimal schedule for (G, P).*

This trivially holds because a concurrent multicast schedule for P is always a valid schedule for P' as well. Let $s \in S_P$ be an arbitrary sender. Consider the set $P_s = \{(s', t') \in P \mid s = s'\}$, which is the set P restricted to sender s. Then P_s together with G and s defines an instance of the multicast problem. It follows that the length of a multicast schedule with sender $s \in S_P$ and receivers R_{P_s} is always a lower bound for a concurrent multicast schedule for instance (G, P).

Observation 2.14. *Let $\mathcal{S}$ be a concurrent multicast schedule for an instance (G, P), and suppose P contains two pairs (v_1, v_2) and (v_2, v_3). Then schedule $\mathcal{S}^2$ makes sure that the message of v_1 also reaches v_3.*

The schedule $\mathcal{S}$ is a concurrent multicast schedule for (G, P). Thus, we know that the message of v_1 is at v_2 after one execution of $\mathcal{S}$. After the second execution, it is consequently delivered to v_3. This observation plays a crucial part in the algorithm of Nikzad and Ravi, and can be generalized as follows.

Observation 2.15. *Given a sequence of pairs of nodes in a network, such that* $(v_1, v_2), (v_2, v_3), ..., (v_{t-2}, v_{t-1}), (v_{t-1}, v_t) \in P$. *If a valid schedule* $\mathcal{S}$ *for* (G, P) *is repeated* t *times, then the message of* v_1 *is also delivered to* v_t.

Telephone Model

As mentioned earlier, the concurrent multicast problem was introduced by Nikzad and Ravi in [NR14]. They provide a $2^{O(\log\log k \cdot \sqrt{\log k})}$-approximation, which we describe in more detail in this section. Although $2^{O(\log\log n \cdot \sqrt{\log n})}$ is super polylogarithmic, it is worth mentioning that $\sqrt[c]{n} = 2^{\omega(\log\log n \cdot \sqrt{\log n})}$ for any constant $c > 0$.

The algorithm operates in three different phases. In phase one, it starts by guessing the length L of an optimal schedule. This can be done in linear time because there are at most n possible values for L. The set of pairs P induces a graph G_P, which we call the *demand graph* of instance (G, P). In G_P we have an undirected edge $\{s, t\}$ for every pair $(s, t) \in P$ as illustrated by an example in Figure 2.4. The algorithm potentially performs a reduction of the sender-receiver pairs depending on the number of terminals $|V_P|$. This can be done by applying the next theorem to the demand graph G_P.

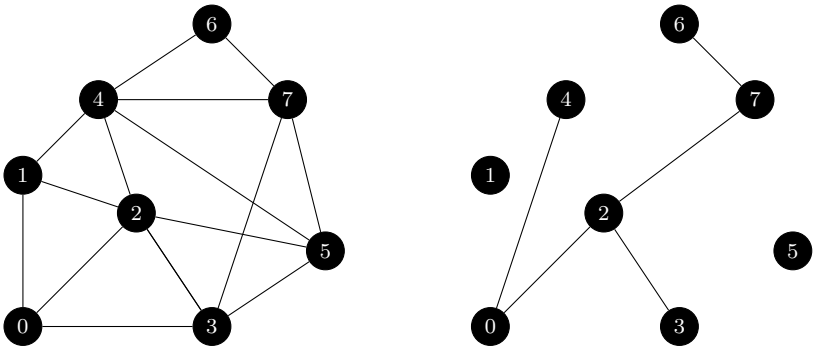

Figure 2.4: Graph G and demand graph G_P for pairs $P = \{(0, 4), (2, 0), (3, 2), (7, 2), (6, 7)\}$.

Theorem 2.16 ([NR14])**.** *In polynomial time, one can compute a subgraph* H *for an arbitrary undirected graph* G, *such that* $|E(H)| \leq 2n \log n$ *and for each* $\{u, v\} \in E(G)$ *it holds that* $\text{dist}(H, u, v) \leq 8 \log n$.

How to compute a subgraph with the properties of Theorem 2.16 is described in detail in [NR14]. Suppose we compute such a graph H_P for a demand graph G_P using Theorem 2.16. Note that in H_P the distance between two nodes increases at most by a factor of $8 \log n$ compared to G_P. By Observation 2.15 schedule $\mathcal{S}^{8 \log n}$, where $\mathcal{S}$ is a concurrent multicast schedule $\mathcal{S}$ for (G, P') and P' is the set of sender-receiver pairs corresponding to H_P, is ensured to be a valid schedule for the original instance (G, P). In the analysis of the algorithm's performance this reduction is used to bound the number of iterations.
The second phase consists of greedily finding node disjoint paths of length at most L from s to t for every pair $(s, t) \in P$. Initially, we set $\mathcal{P}$ to be the empty set. For every pair $(s, t) \in P$ a shortest path from s to t is computed in the graph $G[V(G) \setminus \mathcal{P}]$. If this path is not longer than L, all nodes of the path are added to $\mathcal{P}$. If the number of paths found in one iteration exceeds a certain threshold, the pairs that were satisfied, i.e., the algorithm added s and t to $\mathcal{P}$, are removed from P and phase two is executed again. Otherwise, the algorithm proceeds to phase three.
In the last phase, a new instance (G, P') is constructed and solved recursively, where P' only consists of senders that were satisfied during the last execution of phase two. This set of senders is further denoted as S. Since the threshold was not exceeded, this instance is guaranteed to be much smaller than the original instance. This phase can further be divided in three different steps:

1. The algorithm derives a function $f : V_P \to S$, which assigns each terminal of P to a sender in S. It further creates and runs a schedule $\mathcal{S}$, where each terminal $v \in V_P$ is ensured to send all the messages it has gathered so far to $f(v)$.

2. Let $P' = \{(f(s), f(t)) : (s, t) \in P\}$. Solve the instance (G, P') recursively and run the solution schedule.

3. Run schedule $\mathcal{S}$ reversed, such that for every $v \in V_P$ $f(v)$ is ensured to send its messages to v.

The auxiliary multicast instance to determine the assignment function f and schedule $\mathcal{S}$ is constructed as follows: create a binary tree T with root r of height $\lceil \log |S| \rceil$ and add it to G. Replace every leaf of T by exactly one node of S and compute an $O(\log k)$-approximation for instance (G, r, V_P) of the multicast problem. The obtained multicast schedule $\mathcal{S}$ induces a tree $T_{\mathcal{S}}$ in the graph, where every node $v \in V_P \setminus S$ has exactly one predecessor $v' \in S$ in $T_{\mathcal{S}}$ on the path to r. We define f to be $f(v) = v$, if $v \in S$, and $f(v) = v'$, if $v \in V_P \setminus S$. Algorithm 2 is a formal description of the algorithm. It is still an open question

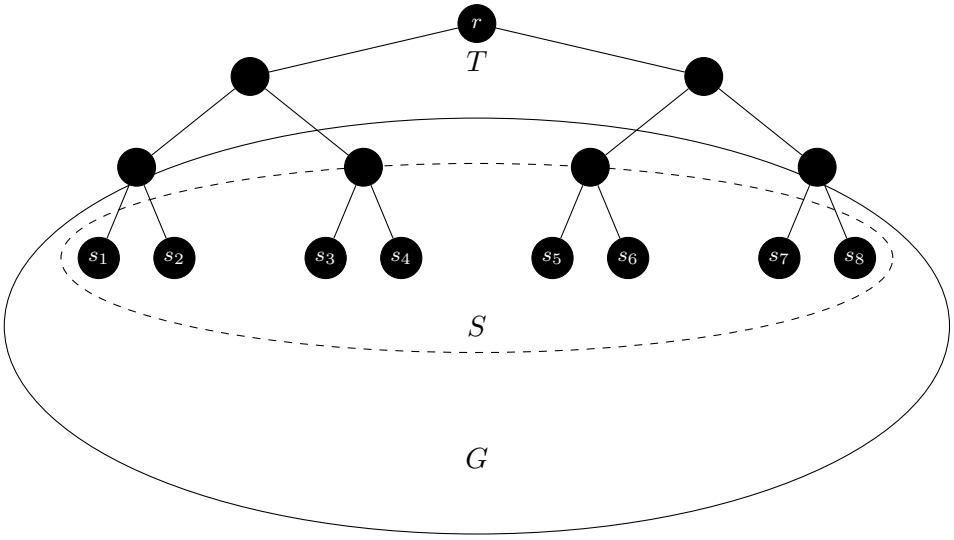

Figure 2.5: The graph G' of the auxiliary instance.

whether there is a polylogarithmic approximation algorithm for the concurrent multicast problem in the telephone model.

Radio Model

Until this point, there have not been any publications regarding concurrent multicasting in the radio model and to the best of our knowledge we provide the first non-trivial approximation algorithms for this problem. Gossiping is a special case of concurrent multicasting, and thus we can conclude from Theorem 2.8 that there is no polynomial time algorithm with approximation factor $O(n^{1/2-\epsilon})$, $\epsilon > 0$, unless P = NP. Since there are no results apart from the $\Omega(n^{1/2-\epsilon})$-hardness for the radio version of the concurrent multicast problem so far we only point out some properties of radio schedules that will be of further interest in the next chapter.

Theorem 2.17. *Let $\mathcal{S}$ be a collision-free schedule in which each node transmits at least once. If $\mathcal{S}$ is repeated ℓ times, then all messages at v have been delivered to every node in $N_\ell(G, v)$ for all $v \in V(G)$.*

Proof. The proof is by induction. Suppose $\ell = 1$, then obviously the statement is true. If $\ell > 1$, take an arbitrary node $v \in V(G)$ and let $u \in N_\ell(G, v)$ be a node in the ℓ-neighborhood of v. By the induction hypothesis, all nodes in $N_{\ell-1}(G, v)$ have received M_v after $\ell - 1$ repetitions. Node u has a neighbor u' in $N_{\ell-1}(G, v)$ because u is in the ℓ-neighborhood of v. Since u' will be transmitting at least once in $\mathcal{S}$ and $\mathcal{S}$ is collision-free u will receive M_v after ℓ executions of $\mathcal{S}$ at latest. □

Algorithm 2 MM [NR14]

Input: $G = (V(G), E(G))$, $P \subseteq V(G) \times V(G)$

{Phase 1}

1. Guess the optimal length L of a concurrent multicast schedule.
2. **if** $|P| > 2|V_P| \log |V_P|$ **then**
3. Apply Theorem 2.16 to G_P to reduce the number of pairs.
4. **end if**

{Phase 2}

5. $\mathcal{P} \leftarrow \varnothing$, $X \leftarrow \varnothing$, $S \leftarrow \varnothing$
6. **for** $i = 1, ..., k$ **do**
7. Find the shortest path P_i in $G[V(G) \setminus \mathcal{P}]$ from s_i to t_i.
8. **if** $|P_i| \leq L$ **then**
9. $\mathcal{P} \leftarrow \mathcal{P} \cup V(P_i)$, $X \leftarrow X \cup \{(s_i, t_i)\}$, $S \leftarrow S \cup \{s_i\}$
10. **end if**
11. **end for**

{Phase 3}

12. **if** $|X| \geq 2^{\log |P| - \sqrt{\log |P|}}$ **then**
13. For all $(s_i, t_i) \in X$ send the message of s_i along P_i to t_i simultaneously.
14. MM$(G, P \setminus X, L)$
15. **else**
16. Construct and solve the auxiliary multicast instance to obtain schedule $\mathcal{S}$ and the assignment function $f : V_P \to S$.
17. Run schedule $\mathcal{S}$ in the reverse order to send all messages from $v \in V_P$ to $f(v)$.
18. $P' \leftarrow \{(f(s_i), f(t_i)) \mid (s_i, t_i) \in P\}$
19. MM(G, P')
20. Run schedule $\mathcal{S}$ so that $f(v)$ sends all its messages to v for all $v \in V_P$.
21. **end if**

Theorem 2.17 together with Observation 1.1, which states that a schedule $\mathcal{S}$ is collision-free if and only if all sets S_i are independent sets in G^2, allows for the following conclusion: if we are given a coloring with q colors for G^2, then we can easily derive a q-approximation for an instance (G, P). A coloring is a function $c : V(G) \to \{1, ..., q\}$ that assigns each node in a graph G one of q colors, such that $c(u) \neq c(v)$ if u and v are neighbors. It immediately follows that the subset of nodes induced by color i, i.e., $C_i = \{v \in V(G) \mid c(v) = i\}$, is an independent set in G for all $i \in \{1, ..., q\}$.

Definition 2.18. *Let $c : V(G) \to \{1, ..., q\}$ be a coloring for G^2. The schedule induced by coloring c is $\mathcal{S}(c) = C_1, ..., C_q$, where $C_i = \{v \in V(G) \mid c(v) = i\}$.*

It is not hard to see that $\mathcal{S}(c)$ is always free of collisions and has length $|\mathcal{S}(c)| = q$. By Theorem 2.17 we observe that $\mathcal{S}(c)^L$, where $L = \text{diam}(G, P)$, is a valid concurrent multicast schedule of length $q \cdot L$. Since L is a lower bound for the

length of a valid concurrent multicast schedule we obtain a q-approximation. Unfortunately, apart from $\text{diam}(G, P)$ there are no good estimations for lower bounds.

The problem of coloring a graph with the least possible number of colors is NP-hard and further hard to approximate [Zuc07]. From the obvious fact that $\Delta(G^2) \leq \Delta(G)^2$ for an arbitrary graph G, and because every graph can be colored greedily with $\Delta(G) + 1$ colors in polynomial time [WP67], we get the following corollary.

Corollary 2.19. *The radio concurrent multicast problem can be approximated in polynomial time within a factor of* $\Delta(G)^2 + 1$.

Hence, for a graph class $\mathcal{G}$ with bounded degree $\Delta(\mathcal{G}) = O(\log^c n)$, $c > 0$, we can guarantee a polylogarithmic approximation factor. If the maximum degree of $\mathcal{G}$ can be bounded by some constant, $\mathcal{G}$ even allows for an $O(1)$-approximation of the radio concurrent multicast problem. Corollary 2.19 also suggests that the hard instances for radio concurrent multicasting are the ones with high maximum degree. The intuition is that a schedule is more likely to have collisions if the underlying graph has high maximum degree.

2.5 Summary

The following tables summarize the results for the communication network problems that we discussed in this chapter. Moreover, it reveals an upper bound for radio concurrent multicasting, which we describe in detail in the next chapter. In the broadcast and gossip case let L denote the diameter of the graph. For the multicast and the concurrent multicast problem, L is the maximum distance between a sender and a receiver. The number of terminals in the case of multicasting and concurrent multicasting is denoted by k.

Upper Bounds - Telephone Model		
Broadcasting	$O\left(\frac{\log n}{\log \log n}\right) \cdot \text{OPT}$	[EK06]
Gossiping	$O\left(\frac{\log n}{\log \log n}\right) \cdot \text{OPT}$	[EK06]
Multicasting	$O\left(\frac{\log k}{\log \log k}\right) \cdot \text{OPT}$	[EK06]
Concurrent Multicasting	$2^{O(\log \log k \cdot \sqrt{\log k})} \cdot \text{OPT}$	[NR14]

Upper Bounds - Radio Model		
Broadcasting	$O(L + \log^2 n)$	[KP07]
Gossiping	$O(L + \Delta(G) \cdot \log n)$	[GPX07]
Multicasting	$O(L + \log^2 n)$	[KP07]
Concurrent Multicasting	$O\left(\frac{\log^2 n}{\log \log n}(L + D)\right)$	This

Lower Bounds - Telephone Model		
Broadcasting	$(3 - \epsilon)$ hard	[EK05]
Gossiping	$(3/2 - \epsilon)$ hard	[EK05]
Multicasting	$(3 - \epsilon)$ hard	[EK05]
Concurrent Multicasting	$(3 - \epsilon)$ hard	[EK06]

Lower Bounds - Radio Model		
Broadcasting	$O(L + \log^2 n)$	[ABLP91]
Gossiping	$\Omega(n^{1/2-\epsilon})$ hard	[IRRS15]
Multicasting	$O(L + \log^2 n)$	[ABLP91]
Concurrent Multicasting	$\Omega(n^{1/2-\epsilon})$ hard	[IRRS15]

Chapter 3

Radio Concurrent Multicasting

We present an $O(\sqrt{n} + \Delta(G) \log n)$-approximation algorithm, which does not require any prior results, for the concurrent multicast problem in the radio model. This upper bound can easily be improved for planar graphs using a result regarding the chromatic number of powers of graphs. In the second part of this chapter, we introduce clusters, clusterings and partition covers, and we describe how sparse partition covers can be efficiently computed. This will further lead to a polylogarithmic approximation algorithm for radio concurrent multicasting with an additive term of $O\left(\frac{\log^2 n}{\log \log n} \cdot \Delta(G)\right)$.

3.1 Simple Upper Bounds

This section addresses simple approximation algorithms with easy-to-prove approximation guarantees compared to the more sophisticated algorithm in the second part of this chapter. The algorithms of this section allow us to derive first upper bounds for radio concurrent multicasting in general and planar graphs.

$O(\sqrt{n} + \Delta(G) \log n)$-Approximation

We first present an approximation algorithm for general graphs whose approximation factor depends on the number of nodes and the maximum degree of a graph G. For a given instance (G, P) of the radio concurrent multicast problem let $L = \text{diam}(G, P)$. The intuition behind Algorithm 3 is the following: the nodes are divided into three disjoint sets X, Y and Z such that $X \cup Y \cup Z = V(G)$. The set X is the union of neighborhoods of nodes with high degree in G^2. Remaining nodes with relatively small degree in G^2 are assigned to Y. The set Z contains all nodes that are in the neighborhood of X. Let $X_Z = X \cup Z$ and $Y_Z = Y \cup Z$. We use the fact that components of $G[X_Z]$ have

small diameter and vertices in $G[Y_Z]$ have a small 2-neighborhood to bound the approximation factor. For both graphs $G[X_Z]$ and $G[Y_Z]$ we compute schedules $\mathcal{S}_X$ and $\mathcal{S}_Y$, which are executed consecutively in order to avoid collisions.

Algorithm 3 RADIO CM I

Input: $G = (V(G), E(G))$, $P \subseteq V(G) \times V(G)$

1. $X \leftarrow \varnothing$, $H \leftarrow G^2$
2. **while** $\exists v \in V(H) : \deg(H, v) \geq \sqrt{n}$ **do**
3. Add $N(H, v)$ to X and remove $N(H, v)$ from H.
4. **end while**
5. $Z \leftarrow N(G, X)$
6. $Y \leftarrow V(G) \setminus (X \cup Z)$
7. Color H with $\sqrt{n}$ colors and let $c(v)$ denote the color of $v \in Y \cup Z$.
8. Let $\mathcal{S}_Y = \mathcal{S}(c)$ be a schedule for $G[Y \cup Z]$ where all nodes of color i transmit in round i.
9. Determine a gossip schedule $\mathcal{S}_X$ for $G[X \cup Z]$ of length $O(\sqrt{n} + \Delta(G) \log n)$ using Theorem 2.7.
10. Repeat schedule $\mathcal{S} = \mathcal{S}_X \mathcal{S}_Y$ for L times.

Lemma 3.1. *Schedule $\mathcal{S}^L$ computed by Algorithm 3 is a valid concurrent multicast schedule for (G, P).*

Proof. We show a more general statement, which together with the fact that $L = \max\{\text{dist}(s_i, t_i) \mid i \in \{1, ..., k\}\}$ implies the theorem: after one execution of $\mathcal{S}$ all nodes in G have successfully transmitted to their neighborhood. It is not hard to see that this holds for $v \in Y_Z$ since the schedule S_Y was derived from a valid coloring for $G^2[Y_Z]$. Hence, a node $u \in N_2(G, v)$ is never transmitting at the same time as v. It remains to show that the statement holds for $v \in X$. From the fact that Z is the neighborhood of X it follows that $N(G, v) \subseteq X_Z$ for all $v \in X$. Since schedule S_X is a gossip schedule for X_Z all neighbors of v receive the messages v has gathered so far during the execution. □

Theorem 3.2. *Algorithm 3 is an $O(\sqrt{n} + \Delta(G) \log n)$-approximation for the concurrent multicast problem in the radio model.*

Proof. First, we show that $\text{diam}(G[X_Z]) \leq 5\sqrt{n} + 2$, which together with Theorem 2.7 allows for the computation of a gossip schedule for $G[X_Z]$ with the desired upper bound on the length. Consider one iteration of the while-loop and let v be the node chosen by the algorithm with $\deg(H, v) \geq \sqrt{n}$. When $N(H, v)$ is added to X, the diameter of a component in $G[X]$ can increase by at most 5 because the diameter of $N(H, v)$ is at most 4 since for every

node $u \in N(H, v)$ we have that $\text{dist}(G, v, u) \leq 2$. Further, if two components C_1 and C_2 are merged, the diameter of the merged component is at most $\text{diam}(C_1) + \text{diam}(C_2) + 1$.

Note that the while-loop can be executed at most $\sqrt{n}$ times because in every iteration at least $\sqrt{n}$ nodes are removed from H. It follows that a component in $G[X]$ can have diameter at most $5\sqrt{n}$ at the end of the while-loop. Since $Z = N(G, X)$ the diameter of $G[X_Z]$ is at most $5\sqrt{n} + 2$, and thus the length of $\mathcal{S}_X$ can be bounded by $O(\sqrt{n} + \Delta(G) \log n)$.

It was mentioned earlier that every graph can be colored greedily with $\Delta(G)+1$ colors, and we claim that $G[Y_Z]$ is $\sqrt{n}$-colorable. For the sake of contradiction assume that there is a node $v \in Y_Z$ with $\deg(G[Y_Z], v) \geq \sqrt{n}$. Then v would have been removed by the algorithm together with $N(H, v)$. From a coloring $c : Y_Z \to \{1, ..., \sqrt{n}\}$ the algorithm can derive a collision-free schedule $\mathcal{S} = C_1, ..., C_{\sqrt{n}}$, where $C_i = \{v \in Y_Z \mid c(v) = i\}$.

We know that L is a trivial lower bound for the length of an optimal concurrent multicast schedule for (G, P). Since $|\mathcal{S}|$ is bounded by $O(\sqrt{n}+\Delta(G) \log n)$, the schedule $\mathcal{S}^L$ computed by Algorithm 3 is an $O(\sqrt{n}+\Delta(G) \log n)$-approximation. □

Planar Graphs

The upper bound provided by Algorithm 3 can easily be improved for planar graphs. Agnarsson and Halldórsson showed how the chromatic number of graph powers of planar graphs can be bounded in terms of the maximum degree. The chromatic number $\chi(G)$ of a graph G is the minimum number of colors required for a valid coloring of G.

Theorem 3.3 ([AH03])**.** *Let G be a planar graph. The chromatic number $\chi(G^2)$ is bounded by $O(\Delta(G))$.*

The proof for an even more general version of this theorem can be found in [AH03] and is constructive. Given such a coloring $c : V(G) \to \mathbb{N}$, we know that schedule $\mathcal{S}(c)$ has $O(\Delta(G))$ rounds, where in round i all nodes of color $c(v) = i$ are transmitting. As mentioned before, such a schedule is collision-free because $C_i = \{v \in V(G) \mid c(v) = i\}$ is an independent set in G^2. By Theorem 2.17 we get that schedule $\mathcal{S}(c)^L$, $L = \text{diam}(G, P)$, is a valid concurrent multicast schedule for (G, P), and thus an $O(\Delta(G))$-approximation.

3.2 Sparse Partition Covers

We first define the terms *cluster*, *clustering* and *partition cover*. After explaining what sparse partition covers are and how they can be computed, we derive an additive approximation for radio concurrent multicasting. The results for computing sparse partition covers are described in more detail in [AP90] and [Pel00]. We use a different notation that better fits our needs.

Clusters, Clusterings and Partition Covers

Throughout this section, let $G = (V(G), E(G))$ be an undirected graph.

Definition 3.4. *A cluster $C \subseteq V(G)$ is a subset of nodes of G, such that the induced subgraph $G[C]$ is connected.*

The radius $\text{rad}(v, C)$ of a node v in a cluster C is the radius of v in the induced subgraph $G[C]$. Analogously, we define the diameter $\text{diam}(C)$ of cluster C to be the diameter of $G[C]$. We call a collection of clusters $\mathcal{C} = \{C_1, ..., C_k\}$ a *clustering* in G and $\text{diam}(\mathcal{C}) = \max_{C \in \mathcal{C}} \text{diam}(C)$ the diameter of clustering $\mathcal{C}$.

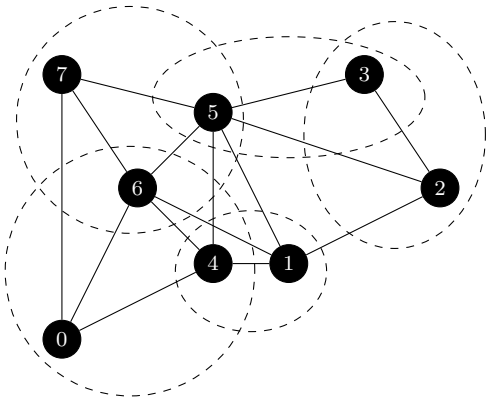

Figure 3.1: A clustering $\mathcal{C}$, which consists of five clusters, with $\text{diam}(\mathcal{C}) = 1$.

A partial partition is a special case of a clustering, where all clusters are disjoint.

Definition 3.5. *A partial partition $\mathcal{C}$ in G is a collection of clusters $\mathcal{C} = \{C_1, ..., C_k\}$, such that $C_i \cap C_j = \varnothing$ for all $i \neq j$, $i, j \in \{1, ..., k\}$.*

For the purpose of message dissemination in a network we use clusters to divide a graph in several regions and derive schedules for each of these regions separately. There is an important difference between clusterings in general and partial partitions.

Observation 3.6. *Let $\mathcal{C}$ be a partial partition and $\mathcal{S}_C$ a radio schedule in the graph $G[C \setminus \mathrm{bnd}(C)]$ for all $C \in \mathcal{C}$. Then all these schedules can be executed in parallel without causing any collisions between the clusters.*

This observation follows from the fact that every node belongs to at most one cluster, and the only nodes that could potentially cause a collision are the ones in the boundary of C because they might interfere with neighboring clusters. In a general clustering on the other hand collisions might occur at any node when all schedules run in parallel.

Definition 3.7. *$\mathcal{P}$ is a partition cover for a clustering $\mathcal{C}$, if $\mathcal{P}$ is a collection of partial partitions $\{\mathcal{C}_1, ..., \mathcal{C}_k\}$ with $\mathcal{C}_i = \{C_i^1, ..., C_i^{k_i}\}$, $k_i = |\mathcal{C}_i|$, $i \in \{1, ..., k\}$, such that for all $C \in \mathcal{C}$ there exists a partial partition $\mathcal{C}_i \in \mathcal{P}$ and a cluster $C_i^j \in \mathcal{C}_i$ with $C \subseteq C_i^j$.*

In other words, each cluster of a clustering $\mathcal{C}$ is fully contained in at least one of the clusters of a partition cover $\mathcal{P}$ for $\mathcal{C}$. We refer to the elements of a partition cover, which are all partial partitions, as *layers*. The diameter of $\mathcal{P}$ is denoted by $\mathrm{diam}(\mathcal{P}) = \max_{\mathcal{C}_i \in \mathcal{P}} \mathrm{diam}(\mathcal{C}_i)$. Further, the degree of partition cover $\mathcal{P}$ is the number of layers, i.e., $\Delta(\mathcal{P}) = |\mathcal{P}|$. We use $\Delta(\mathcal{P})$ to measure the sparseness of $\mathcal{P}$.

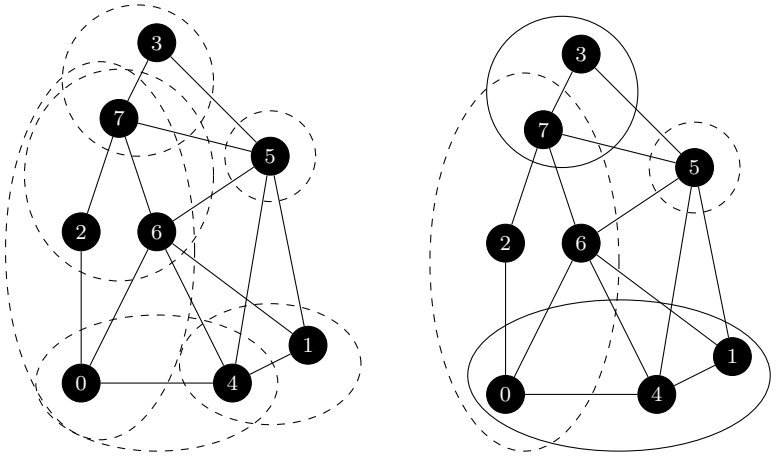

Figure 3.2: A clustering $\mathcal{C}$ and a partition cover $\mathcal{P}$ for $\mathcal{C}$, which consists of two layers. Each layer contains only disjoint clusters.

The reason partition covers are introduced is the following: suppose we are given a set of nodes $V \subseteq V(G)$ and the goal is to distribute the message M_v of every node $v \in V$ to the ℓ-neighborhood of v. We will see that it is possible to compute a partition cover $\mathcal{P}$ with relatively small degree $\Delta(\mathcal{P})$ in polynomial time, such that every ℓ-neighborhood is fully contained in $C \setminus \mathrm{bnd}(C)$ for one

of the clusters of $\mathcal{P}$. Furthermore, the maximum diameter of a cluster in $\mathcal{P}$ will still be within a reasonable factor of ℓ.
We can then compute a gossip schedule $\mathcal{S}_C$ for each cluster of $\mathcal{P}$. Since every ℓ-neighborhood is fully subsumed by a cluster of $\mathcal{P}$, it is ensured that the final schedule is feasible. Moreover, all schedules belonging to the same layer can be executed simultaneously by Observation 3.6 without causing further collisions. Because $\Delta(\mathcal{P})$ is relatively small, all layers can be processed consecutively without blowing up the total length of the final schedule.

Computing Sparse Partition Covers

We first describe a procedure, which will be used as a subroutine in Algorithm 5 for the computation of sparse partial covers, i.e., partition covers that consist of a small number of layers. Algorithm 4 takes a clustering $\mathcal{C}$ together with a parameter κ as input and outputs a subset of the clusters $\mathcal{X}_1 \subseteq \mathcal{C}$ together with a partial partition $\mathcal{X}_2$, such that each cluster of $\mathcal{X}_1$ is subsumed by a cluster of $\mathcal{X}_2$. Further, the diameter of a cluster in $\mathcal{X}_2$ is guaranteed to be at most $(2\kappa - 1) \cdot \operatorname{diam}(\mathcal{C})$.

Algorithm 4 PARTIAL PARTITION [Pel00]

Input: clustering $\mathcal{C}$, stretch bound κ
1. $\mathcal{U} \leftarrow \mathcal{C}$, $\mathcal{X}_1 \leftarrow \varnothing$, $\mathcal{X}_2 \leftarrow \varnothing$
2. **while** $\mathcal{U} \neq \varnothing$ **do**
3. Pick an arbitrary cluster $C \in \mathcal{U}$.
4. $\mathcal{Z} \leftarrow \{C\}$, $Y \leftarrow Z$
5. **repeat**
6. $\mathcal{Y} \leftarrow \mathcal{Z}$, $Y \leftarrow Z$
7. $\mathcal{Z} \leftarrow \{C \in \mathcal{U} \mid C \cap Y \neq \varnothing\}$
8. $Z \leftarrow \bigcup_{C \in \mathcal{Z}} C$
9. **until** $|\mathcal{Z}| \leq |\mathcal{C}|^{1/\kappa} \cdot |\mathcal{Y}|$
10. $\mathcal{U} \leftarrow \mathcal{U} \setminus \mathcal{Z}$
11. $\mathcal{X}_2 \leftarrow \mathcal{X}_2 \cup \{Y\}$
12. $\mathcal{X}_1 \leftarrow \mathcal{X}_1 \cup \mathcal{Y}$
13. **end while**

Output: $(\mathcal{X}_1, \mathcal{X}_2)$

The algorithm starts by assigning all clusters to the set of unprocessed clusters $\mathcal{U}$. After choosing an arbitrary cluster C at the beginning of an iteration, Algorithm 4 merges C with all clusters that share a node with C. This continues until the size of the new emerged cluster does not increase enough. Afterwards, a new unprocessed cluster C' is chosen and the same merging process is applied to C'. At some point, either all clusters are processed or it is not possible to

add a new cluster without violating the condition of the clusters to be disjoint.

Theorem 3.8 ([Pel00])**.** *The clusterings $\mathcal{X}_1$ and $\mathcal{X}_2$ computed by Algorithm 4 have the following properties:*

1. *For every $C_1 \in \mathcal{X}_1$, there is a cluster $C_2 \in \mathcal{X}_2$, such that $C_1 \subseteq C_2$,*
2. *$\mathcal{X}_2$ is a partial partition,*
3. *$|\mathcal{X}_1| \geq |\mathcal{C}|^{1-1/\kappa}$,*
4. *$\operatorname{diam}(\mathcal{C}') \leq (2\kappa - 1) \cdot \operatorname{diam}(\mathcal{C})$.*

The proof of this theorem can be found in [Pel00]. The third property described in Theorem 3.8 plays a crucial role for the computation of sparse partition covers. Although not all clusters might be covered by $\mathcal{X}_2$, it guarantees that the number of covers subsumed by the partial partition $\mathcal{X}_2$ is at least relatively large. This further leads to a small number of iterations in Algorithm 5. Since every iteration of Algorithm 5 adds precisely one layer to $\mathcal{P}$ this allows for the computation of sparse partition covers.

Algorithm 5 PARTITION COVER [Pel00]

Input: clustering $\mathcal{C}$, stretch bound κ

1. $\mathcal{U} \leftarrow \mathcal{C}$, $\mathcal{P} \leftarrow \varnothing$
2. **while** $\mathcal{U} \neq \varnothing$ **do**
3. $\quad(\mathcal{X}_1, \mathcal{X}_2) \leftarrow$ PARTIALPARTITION$(\mathcal{U}, \kappa)$
4. $\quad\mathcal{P} \leftarrow \mathcal{P} \cup \{\mathcal{X}_2\}$
5. $\quad\mathcal{U} \leftarrow \mathcal{U} \setminus \mathcal{X}_1$
6. **end while**

Output: $\mathcal{P}$

The set $\mathcal{U}$ represents the clusters that have not been covered by a cluster of $\mathcal{P}$ so far. The algorithm basically just produces partial partitions subsuming clusters of $\mathcal{U}$ until $\mathcal{U}$ is fully covered. It can be shown that the following properties hold for the partition cover $\mathcal{P}$ for $\mathcal{C}$. A proof for a slightly different version of Algorithm 5 is described in [Pel00].

Theorem 3.9 ([Pel00])**.** *The partition cover $\mathcal{P}$ computed by Algorithm 5 has the following properties:*

1. *$\mathcal{P}$ is a partition cover for $\mathcal{C}$,*
2. *$\operatorname{diam}(\mathcal{P}) \leq (2\kappa - 1) \cdot \operatorname{diam}(\mathcal{C})$,*
3. *$\Delta(\mathcal{P}) \leq 2\kappa \cdot |\mathcal{C}|^{1/\kappa}$.*

Additive Approximation Algorithm

Suppose we are given an instance (G, P) for radio concurrent multicasting. Throughout this section, let $S = S_P$ be the set of senders of instance (G, P), $k = |S|$ the number of senders and $L = \text{diam}(G, P)$. For our purposes Algorithm 5 is particularly useful if we choose parameter κ to be $\kappa = \log |\mathcal{C}|$.

Corollary 3.10. *For a clustering $\mathcal{C}$ one can compute a partition cover $\mathcal{P}$ in polynomial time with the following properties:*

1. *$\mathcal{P}$ is a partition cover for $\mathcal{C}$,*
2. *$\text{diam}(\mathcal{P}) \leq 2 \log |\mathcal{C}| \cdot \text{diam}(\mathcal{C})$,*
3. *$\Delta(\mathcal{P}) \leq 4 \log |\mathcal{C}|$.*

Using this corollary, we can now describe an approximation algorithm for the concurrent multicast problem, which strongly improves the upper bound derived in the previous section.

Algorithm 6 RADIO CM II

1. Let $\mathcal{C}$ be a collection of clusters $N_{L+1}(G, s_i)$ for all $i \in \{1, ..., k\}$.
2. Compute a partition cover $\mathcal{P}$ for $\mathcal{C}$ using Corollary 3.10, which consists of $O(\log k)$ partial partitions, such that the diameter of a cluster is bounded by $O(L \log k)$.
3. **for** $C \in \mathcal{C}_i$, $\mathcal{C}_i \in \mathcal{P}$ **do**
4. $\quad C \leftarrow C \setminus \text{bnd}(C)$
5. $\quad$ Determine a gossip schedule $\mathcal{S}_C$ of length $O(L \log k + \Delta(G) \log n))$ for cluster C.
6. **end for**
7. **for** $\mathcal{C}_i \in \mathcal{P}$ **do**
8. $\quad$ Run all gossip schedules belonging to layer $\mathcal{C}_i$ at the same time.
9. **end for**

Lemma 3.11. *Let C be a cluster and $s \in C$. If $N_{L+1}(G, s) \subseteq C$, then $N_L(G, s) \subseteq C \setminus \text{bnd}(C)$.*

Proof. We prove this lemma by showing that every node $u \in \text{bnd}(C)$ has distance at least $L + 1$ to s. For the sake of contradiction, suppose the distance is smaller than $L + 1$. By the definition of $\text{bnd}(C)$, there is a neighbor v of u outside of C. But this means that the distance between s and v is at most $L+1$ and thus $v \in N_{L+1}(s) \subseteq C$, a contradiction. □

Lemma 3.12. *Let $\mathcal{S}$ be a schedule computed by Algorithm 6 for instance (G, P). Then $\mathcal{S}$ is a valid radio concurrent multicast schedule for (G, P).*

Proof. Consider the partition cover $\mathcal{P}$ before the boundary of each cluster in $\mathcal{P}$ is removed, and let $s \in S$ be an arbitrary sender. There exists a cluster C in the partition cover, such that $N_{L+1}(G, s) \subseteq C$. By Lemma 3.11 the same holds for the L-neighborhood of s after removing the boundary of C. Schedule $\mathcal{S}_C$ is a gossip schedule and thus ensures that the message of s is delivered to all terminals t, $(s, t) \in P$. It remains to show that there are no collisions between clusters of the same layer. Since the boundary of each cluster was removed and by Observation 3.6, the lemma follows. □

By Theorem 2.7 we can compute a gossip schedule $\mathcal{S}_C$ for any cluster C of the partition cover with $|\mathcal{S}_C| = O(L \log k + \Delta(G) \log n)$. Since all layers of $\mathcal{P}$ are processed consecutively, we get the following performance guarantee of Algorithm 6.

Theorem 3.13. *Given an instance (G, P), Algorithm 6 computes a radio concurrent multicast schedule of length $O((L \log k + \Delta(G) \log n) \log k)$, where k corresponds to the number of senders.*

Note that L is a lower bound for any concurrent multicast schedule for instance (G, P), which implies the next corollary.

Corollary 3.14. *Algorithm 6 is an $O(\log^2 k)$-approximation plus additive term $O(\Delta(G) \log k \log n)$ for the concurrent multicast problem in the radio model.*

Chapter 4

Minimum Degree Diameter-Bounded Networks

In this chapter, we introduce the minimum degree diameter-bounded network problem and highlight its connection to concurrent multicasting in the telephone model. This relationship motivates the development of approximation algorithms for minimum degree diameter-bounded networks. We consider different linear program formulations of the latter and argue why standard randomized rounding approaches fail to lead to a polylogarithmic approximation algorithm. However, the computation of minimum degree diameter-bounded networks might be of interest in itself. We show that Algorithm 2 by Nikzad and Ravi for the telephone concurrent multicast problem can be modified to a bicriteria approximation for the minimum degree diameter-bounded network problem. Afterwards, we introduce another network design problem, namely the minimum total-degree diameter-bounded group network problem. This problem has an interesting connection to the minimum degree diameter-bounded network problem, although this connection is not that obvious. Finally, we present a polylogarithmic bicriteria approximation algorithm for the minimum total-degree diameter-bounded group network problem.
For a pair (G, P), where G is a graph and P is a set of sender-receiver pairs, we define the poise of (G, P) to be $\text{poise}(G, P) = \text{diam}(G, P) + \Delta(G)$. This notion was introduced by Nikzad and Ravi in [NR14] and generalizes the notion of the poise for trees.

Definition 4.1. *Given a graph G and $P \subseteq V(G) \times V(G)$, the minimum poise network problem is to find a subgraph H, which minimizes* $\text{diam}(H, P) + \Delta(H)$.

Since the minimum poise spanning tree problem is a special case of the minimum poise network problem we can conclude that the latter is NP-hard. An interesting result regarding minimum poise networks is highlighted in Theorem

4.2 and illustrates the connection between concurrent multicasting in the telephone model and the minimum poise network problem.

Theorem 4.2 ([NR14])**.** *Given a concurrent multicast schedule $\mathcal{S}$ for a graph G and a set of pairs of vertices P, there is a polynomial time algorithm, which computes a subgraph H with poise $O(|\mathcal{S}|)$. Similarly, given a subgraph H of G, there is a polynomial time algorithm, which computes a schedule $\mathcal{S}$ of length $O\left(\frac{\log^3 n}{\log\log n} \cdot \text{poise}(H, P)\right)$.*

The first statement of the theorem is easily shown: let $E \subseteq E(G)$ be the set of edges used by schedule $\mathcal{S}$. If we choose the subgraph H to be $H = (V(G), E)$, it holds that $\Delta(H) \leq |\mathcal{S}|$ and $\text{diam}(H, P) \leq |\mathcal{S}|$ because every node can only talk to at most one neighbor in one round. As a consequence we get that

$$\text{diam}(H, P) + \Delta(H) \leq 2|\mathcal{S}|$$

and thus $\Omega(\text{poise}(H^*, P))$ is a lower bound for the length of an optimal schedule, where H^* is an optimal solution to the minimum poise network problem. The second statement requires more work and a proof can be found in [NR14].
Theorem 4.2 allows for an approximation for concurrent multicasting as described in [NR14]: one can easily compute a subgraph H with $\text{poise}(H, P) \leq \text{diam}(G, P) + \Delta(G)$ by taking the union of shortest (s, t)-paths for all pairs $(s, t) \in P$. Together with the fact that $\text{diam}(G, P)$ is a trivial lower bound for the length of an optimal schedule $\mathcal{S}$, Theorem 4.2 can be used to derive an $O\left(\frac{\log^3 n}{\log\log n}\right)$-approximation plus an additive term of $O\left(\frac{\log^3 n}{\log\log n} \cdot \Delta(G)\right)$.
Further, Theorem 4.2 motivates the development of approximation algorithms for the minimum poise network problem since it implies that a polylogarithmic approximation for the minimum poise network problem immediately leads to a polylogarithmic approximation for the concurrent multicast problem and vice versa.
In this chapter, we analyze the minimum degree diameter-bounded network problem, which can be interpreted as a more general version of the minimum poise network problem.

Definition 4.3. *Given a graph G, $P \subseteq V(G) \times V(G)$ and a diameter bound L. The minimum degree diameter-bounded network problem is to find a subgraph H, such that each pair is connected in H,* $\text{diam}(H, P) \leq L$ *and $\Delta(H)$ is minimized.*

The main difference between these two problems is that in the case of minimum poise networks we are trying to find a subgraph H, where the sum of $\text{diam}(H, P)$

and $\Delta(H)$ is small. On the other hand, when we consider minimum degree diameter-bounded networks we try to bound $\text{diam}(H, P)$ and $\Delta(H)$ separately.

Definition 4.4. *An algorithm is called an (α, β)-bicriteria approximation algorithm for the minimum degree diameter-bounded network problem, if for any instance (G, P, L) the following holds: if an optimal solution has maximum degree D, then the algorithm outputs a subgraph H, such that* $\text{diam}(H, P) \leq \alpha \cdot L$ *and* $\Delta(H) \leq \beta \cdot D$.

Developing a polylogarithmic bicriteria approximation algorithm for the minimum degree diameter-bounded network problem, i.e., $\max\{\alpha, \beta\} = O(\log^c n)$ for some constant $c > 0$, would directly lead to a polylogarithmic approximation algorithm for the minimum poise network problem, as Lemma 4.5 shows.

Lemma 4.5. *If there is an (α, β)-bicriteria approximation algorithm for the minimum degree diameter-bounded network problem, then this algorithm can be used to derive a γ-approximation algorithm for the minimum poise network problem, where $\gamma = \max\{\alpha, \beta\}$.*

Proof. Suppose we have an (α, β)-approximation algorithm ALG for the minimum degree diameter-bounded network problem, and we are given an instance (G, P) for the minimum poise network problem. Let H^* be an optimal solution for (G, P) with $L = \text{diam}(H^*, P)$ and $D = \Delta(H^*)$. Note that $1 \leq L \leq n$ and $1 \leq D \leq n$ for any instance (G, P).

Algorithm 7 MDDBN TO MPN

Input: (G, P)
1. **for** $i = 1, ..., n$ **do**
2. Solve (G, P, i) using ALG and obtain H_i.
3. **end for**
4. Let $H = H_i$, such that $\text{poise}(H_i, P) \leq \text{poise}(H_j, P)$ for all $j \in \{1, ..., n\}$

Output: H

We show that Algorithm 7 is an γ-approximation for the minimum poise network problem, where $\gamma = \max\{\alpha, \beta\}$. Notice that there is a valid solution for instance (G, P, L), namely H^*. It follows that H_L computed by Algorithm 7 has $\text{diam}(H_L, P) \leq \alpha \cdot L$ and $\Delta(H_L) \leq \beta \cdot D$, and thus the poise of H_L is at most

$$\text{poise}(H_L, P) \leq \alpha \cdot L + \beta \cdot D \leq \gamma(L + D).$$

The lemma follows since the procedure outputs the graph of minimum poise. □

Moreover, Lemma 4.5 implies that there is no polynomial time algorithm for the minimum degree diameter-bounded network problem, unless P = NP. Note that an α-approximation for the minimum poise network problem on the other hand would not directly lead to an (α, α)-bicriteria approximation for the minimum degree diameter-bounded network problem. From Theorem 4.2 together with Lemma 4.5 we get the following corollary.

Corollary 4.6. *Given an (α, β)-bicriteria approximation for the minimum degree diameter-bounded network problem, one can approximate the concurrent multicast problem in the telephone model within a factor of $O\left(\frac{\log^3 n}{\log \log n} \cdot \gamma\right)$, where $\gamma = \max\{\alpha, \beta\}$.*

4.1 Randomized Rounding Approaches

This section addresses standard randomized rounding approaches for different linear program formulations of the minimum degree diameter-bounded network problem. One linear program formulation is based on the idea of sending a unit flow from s to t for every pair $(s, t) \in P$.

minimize D

subject to

$$\sum_{u \in N(s_i)} x_{i,1,s_i,u} = 1 \qquad i = 1, ..., k \quad (4.1)$$

$$\sum_{\ell=2}^{L} \sum_{u \in N(t_i)} x_{i,\ell-1,u,t_i} - x_{i,\ell,t_i,u} = 1 \qquad i = 1, ..., k \quad (4.2)$$

$$\sum_{\ell=2}^{L} \sum_{u \in N(v)} x_{i,\ell-1,u,v} - x_{i,\ell,v,u} = 0 \qquad i = 1, ..., k \quad v \in V(G) \setminus \{t_i\} \quad (4.3)$$

$$\sum_{\ell=1}^{L} x_{i,\ell,u,v} \leq y_{u,v} \qquad u, v \in V(G) \quad i = 1, ..., k \quad (4.4)$$

$$\sum_{\ell=1}^{L} x_{i,\ell,v,u} \leq y_{u,v} \qquad u, v \in V(G) \quad i = 1, ..., k \quad (4.5)$$

$$\sum_{u \in N(v)} y_{u,v} \leq D \qquad v \in V(G) \quad (4.6)$$

$$x_{i,\ell,u,v} \in \{0, 1\} \qquad (u, v) \in V(G) \times V(G) \quad i = 1, ..., k$$
$$\ell = 1, ..., L - 1 \quad (4.7)$$

$$y_{u,v} \in \{0, 1\} \qquad \{u, v\} \in E(G) \quad (4.8)$$

The objective function guarantees that the maximum degree of the solution is minimized. The first group of variables, namely the x variables, represents how much flow of commodity i is going from a node u to v. Variable $y_{u,v}$ indicates whether edge $\{u, v\}$ is part of the optimal solution H^* or not. Equations 4.1 ensure that a unit flow leaves the source of every commodity, while Constraints 4.2 ensure that each sink receives the corresponding unit flow. Constraints 4.3 represent the flow conservation. If a commodity sends flow from u to v along edge $\{u, v\}$, we have to make sure that $\{u, v\}$ is included in the solution. This is done by Constraints 4.4 and 4.5. Finally, Equations 4.6 update D to correspond to the maximum degree of the induced solution.

Given a feasible fractional solution (x, y) to this linear program, we are interested in rounding (x, y) to an integral solution (x', y'), such that the violation of the diameter constraint and the deviation from the objective value is within a polylogarithmic factor. If we had an algorithm that is capable of rounding (x, y) in such a way, then this would immediately lead to a polylogarithmic bicriteria approximation for the minimum degree diameter-bounded network problem.

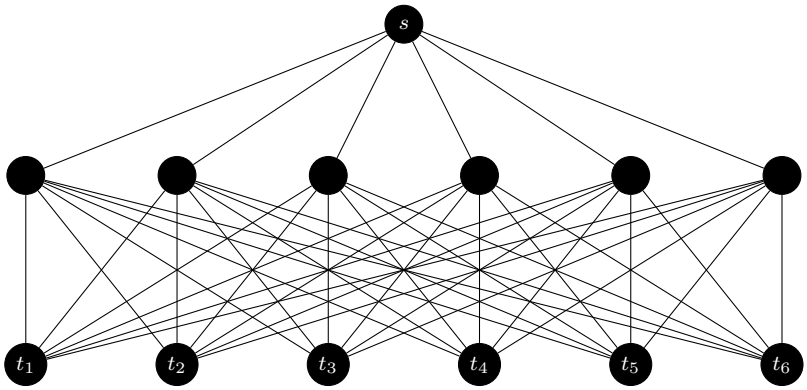

Figure 4.1: Example graph with $k = 6$. An optimal fractional solution might look very different from an optimal integral solution.

However, the instance in Figure 4.1 illustrates an issue with rounding fractional to integral solutions. Consider the general case of such an instance, (G_k, P_k), which consists of a complete bipartite graph with k nodes in both partite sets and an additional node s. Further, in the general case there are pairs (s, t_i) for all $i \in \{1, ..., k\}$.

Consider instance $(G_k, P_k, 2)$ of the minimum degree diameter-bounded network problem and the corresponding flow-based linear program. For all $k \in \mathbb{N}$ there is an optimal fractional solution (x^*, y^*) for the relaxation of this linear program, where s sends an i-flow of $\frac{1}{k}$ for each commodity i to all of its neighbors. Hence,

every non-terminal node receives a flow of $\frac{1}{k}$ for each commodity, and can thus forward it to the corresponding sink t_i. Such a flow ensures that every t_i receives the unit flow of commodity i. Moreover, we get that

$$\sum_{u \in N(v)} y_{u,v} \leq 2 \text{ for all } v \in V(G_k)$$

and every path induced by the flow has length at most 2. However, for any graph H that connects s to t_i, $i \in \{1, ..., k\}$, we get that $\text{diam}(H, P) = \Omega(\log k)$ if we restrict its maximum degree to be $\Delta(H) = O(1)$. Similarly, if $\text{diam}(H, P)$ is not supposed to exceed ℓ for some $\ell \in \mathbb{N}$, then we have that $\Delta(H) = \Omega\left(n^{1/\ell}\right)$. An optimal fractional solution might therefore look very different from an integral solution, and it is further not clear how one can round an optimal fractional solution to a good integral solution. Note that a similar issue arises for a flow-based linear program formulation of the minimum poise network problem.

minimize $L + D$

subject to

$$\sum_{u \in N(s_i)} x_{i,s_i,u} = 1 \qquad i = 1, ..., k \tag{4.9}$$

$$\sum_{u \in N(t_i)} x_{i,u,t_i} - x_{i,t_i,u} = 1 \qquad i = 1, ..., k \tag{4.10}$$

$$\sum_{u \in N(v)} x_{i,u,v} - x_{i,v,u} = 0 \qquad i = 1, ..., k \quad v \in V(G) \setminus \{t_i\} \tag{4.11}$$

$$\sum_{u,v \in V(G)} x_{i,u,v} \leq L \qquad i = 1, ..., k \tag{4.12}$$

$$x_{i,u,v} \leq y_{u,v} \qquad u, v \in V(G) \quad i = 1, ..., k \tag{4.13}$$

$$x_{i,v,u} \leq y_{u,v} \qquad u, v \in V(G) \quad i = 1, ..., k \tag{4.14}$$

$$\sum_{u \in N(v)} y_{u,v} \leq D \qquad v \in V(G) \tag{4.15}$$

$$x_{i,u,v} \in \{0, 1\} \qquad u, v \in V(G) \quad i = 1, ..., k \tag{4.16}$$

$$y_{u,v} \in \{0, 1\} \qquad u, v \in V(G) \tag{4.17}$$

We can also capture the minimum degree diameter-bounded network problem using a cut-based linear program formulation. Although this linear program has exponentially many constraints it is possible to compute an optimal fractional solution in polynomial time [KV07]. Unfortunately, with this approach we run

into similar problems as with the flow-based formulation.

minimize D

subject to

$$\sum_{\substack{\{u,v\}\in E(G):\\ u\in V,\ v\notin V}} x_{i,u,v} = 1 \qquad V \subseteq V(G):\ s_i \in V,\ t_i \notin V \quad i = 1,...,k \tag{4.18}$$

$$\sum_{u,v\in V(G)} x_{i,u,v} \leq L \qquad i = 1,...,k \tag{4.19}$$

$$x_{i,u,v} \leq y_{u,v} \qquad u,v \in V(G) \quad i = 1,...,k \tag{4.20}$$

$$x_{i,v,u} \leq y_{u,v} \qquad u,v \in V(G) \quad i = 1,...,k \tag{4.21}$$

$$\sum_{u\in N(v)} y_{u,v} \leq D \qquad v \in V(G) \tag{4.22}$$

$$x_{i,u,v} \in \{0,1\} \qquad u,v \in V(G) \quad i = 1,...,k \tag{4.23}$$

$$y_{u,v} \in \{0,1\} \qquad u,v \in V(G) \tag{4.24}$$

Again, the objective function ensures that the maximum degree is minimized. We have x variables, which indicate whether commodity i uses an edge to connect its s_i to t_i. Variable $y_{u,v}$ shows whether edge $\{u,v\}$ is contained in an optimal solution or not. Constraints 4.18 ensure that for each cut separating a nodes s and t with $(s,t) \in P$ there is an edge crossing the cut contained in the solution. Equations 4.19 guarantee the length bound is obeyed. Further, analogous to the flow LP formulation, Constraints 4.20, 4.21 and 4.22 of the linear program make sure that the degree bound is not violated.

4.2 Concurrent Multicast Algorithm Modification

Unfortunately, we were not able to develop a polylogarithmic bicriteria approximation for the minimum degree diameter-bounded network problem, which would have led to a polylogarithmic approximation for the telephone concurrent multicast problem by Corollary 4.6. However, the minimum degree diameter-bounded network problem might be of interest in itself. Therefore, we show that Algorithm 2 can be modified to a $(2^{O(\log\log p\cdot\sqrt{\log p})}, 2^{O(\log\log p\cdot\sqrt{\log p})})$-bicriteria approximation algorithm for the minimum degree diameter-bounded network problem, where p corresponds to the number of terminals of an instance. Although we are not interested in finding a schedule for different sender-receiver pairs, we continue to use the notation from previous chapters.
Algorithm 8 is a formal description of our bicriteria approximation algorithm. Let (G,P,L) be an instance of the minimum degree diameter-bounded network

problem and H^* be an optimal solution for (G, P, L) with $\Delta(H^*) = D$. The first difference to Algorithm 2 is that instead of computing a schedule Algorithm 8 determines a subgraph H, such that $\operatorname{diam}(H, P) = 2^{O(\log\log p \cdot \sqrt{\log p})} \cdot L$. Further, Algorithm 8 does not guess the length of an optimal schedule but instead requires L as an additional input to G and P.

Algorithm 8 MDDBN

Input: Graph G, pairs of vertices P, diameter bound L.
{Phase 1}
1. **if** $\max\{\operatorname{dist}(s_i, t_i) \mid i = 1, ..., k\} > L$ **then**
2. Stop. There is no solution with $\operatorname{diam}(H, P) \leq L$.
3. **end if**
4. **if** $|P| > 2|V_P| \log |V_P|$ **then**
5. Apply Theorem 2.16 to G_P to reduce the number of pairs.
6. **end if**
{Phase 2}
7. $\mathcal{P} \leftarrow \varnothing$, $X \leftarrow \varnothing$, $S \leftarrow \varnothing$
8. **for** $i = 1, ..., |P|$ **do**
9. Find a shortest path P_i in $G[V(G) \setminus \mathcal{P}]$ from s_i to t_i.
10. **if** $|P_i| \leq k$ **then**
11. $\mathcal{P} \leftarrow \mathcal{P} \cup V(P_i)$, $X \leftarrow X \cup \{(s_i, t_i)\}$, $S \leftarrow S \cup \{s_i\}$
12. **end if**
13. **end for**
{Phase 3}
14. Let H be the union of all paths that were added to $\mathcal{P}$.
15. **if** $|X| \geq 2^{\log |P| - \sqrt{\log |P|}}$ **then**
16. $H \leftarrow H \cup \text{MDDBN}(G, P \setminus X, L)$
17. **else**
18. Construct an auxiliary graph G' and use Theorem 1.9 to obtain a degree-diameter-bounded Steiner tree T' and the assignment function f.
19. $P' \leftarrow \{(f(s_i), f(t_i)) \mid (s_i, t_i) \in P\}$
20. $H \leftarrow H \cup T' \cup \text{MDDBN}(G, P', (2c_0 + 1) \cdot L \log^2 |S|)$
21. **end if**

Output: H

Construction of the Auxiliary Instance

Suppose the algorithm arrives at Step 18. The auxiliary graph G' is defined exactly the same way as for Algorithm 2, and thus is the union of G and a binary tree T of height $\lceil \log |S| \rceil$ with root r. Every node of the set S replaces exactly one leaf of the binary tree T. Hence, the root r is connected to every terminal of V_P in G' as illustrated in Figure 2.5. Instead of treating this instance as an input to the multicast problem, we use Theorem 1.9 to compute a degree-diameter-

bounded Steiner tree T' for G with terminals $S \cup \{r\}$, such that $\operatorname{diam}(T') = O(L \log^2 p)$ and $\Delta(T') = O(D \log p)$. The following Lemma shows that it is always possible to find a Steiner tree with these properties using Theorem 1.9.

Lemma 4.7. *There exists a Steiner tree T' for the auxiliary graph G' with terminals $S \cup \{r\}$, such that* $\operatorname{diam}(T') \leq 4L + 2\lceil \log p \rceil$ *and* $\Delta(T') \leq D + 2$.

Proof. Without loss of generality assume that $D \geq 1$. Let H' be the union of H^*, H and T, where H is the union of all paths that were added to $\mathcal{P}$ and T is the binary tree of the auxiliary graph G'. It is easy to see that the maximum degree of H' is bounded by $D + 2$. Each node $v \in V(T) \setminus S$ has degree at most 3. Moreover, H and T share only nodes of S and for every node $v \in S$ we have that $\deg(H, v) = \deg(T, v) = 1$. Further, we have that $\deg(H^*, v) \leq D$ for all $v \in V(H^*)$ because $\Delta(H^*) = D$.
For every pair $(s, t) \in P$ that has not been connected via a path in H there exists an (s, t)-path in H^* since H^* is a valid solution for (G, P, L). By the maximality of H this (s, t)-path intersects with at least one path of H. It follows that s and t are both connected to one of the leaves of T in H' via a path of length at most $2L$. Moreover, each leaf of T is connected to r via a path of length $\lceil \log |S| \rceil$.
It follows that $\Delta(H') \leq D + 2$ and $\operatorname{diam}(H') \leq 2L + \lceil \log p \rceil$. Hence, a breadth first search spanning tree T' for H' with root r is a tree with the desired properties. □

From Theorem 1.9 and Lemma 4.7 we can conclude the next corollary.

Corollary 4.8. *In polynomial time, we can compute a Steiner tree T' for the auxiliary graph G' with terminals $S \cup \{r\}$, such that* $\operatorname{diam}(T') \leq c_0 \cdot L \log^2 |S|$ *and* $\Delta(T') \leq c_0 \cdot D \log |S|$, *where $c_0 > 1$ is a universal constant.*

This Steiner tree induces an assignment function $f : V_P \to S$, where every terminal $v \in V_P$ is assigned to its unique predecessor on the path to r in T'. Note that $S \subseteq V_P$, and for a node $v \in S$ we define $f(v)$ to be v.

Approximation Guarantee

Lemma 4.9 shows that the new instance created in Step 20 of Algorithm 8 always allows for an optimal solution H', such that $\operatorname{diam}(H', P)$ and $\Delta(H')$ is within a polylogarithmic factor of $\operatorname{diam}(H^*, P)$ and $\Delta(H^*)$ respectively.

Lemma 4.9. *Let H^* be an optimal solution for (G, P, L) with $\Delta(H^*) = D$. There exists a solution H' for $(G, P', (2c_0 + 1) \cdot L \log^2 |S|)$, which is the new*

instance that is created in Step 20 *of the algorithm, with* $\Delta(H') \leq (c_0 + 1) \cdot D \log |S|$.

Proof. Take the union of H^* and T', namely H', where T' is the Steiner tree derived from the auxiliary graph. First note that there is a path from s to t for every pair $(s,t) \in P$ in H^* of length at most L. For a given pair $(s,t) \in P$ consider the corresponding pair $(f(s), f(t)) \in P'$ in the new instance. By Corollary 4.8 the tree T' connects s to $f(s)$ and t to $f(t)$ via a path of length at most $c_0 \cdot L \log^2 |S|$. Hence, there is a path from $f(s)$ to $f(t)$ in H' of length at most $2c_0 \cdot L \log^2 |S| + L$. Since H^* has maximum degree D and T' has maximum degree $c_0 \cdot D \log |S|$, the maximum degree of their union can be bounded by $c_0 \cdot D \log |S| + D$. □

Theorem 4.10. *Algorithm* 8 *is a* $(2^{O(\log\log p \cdot \sqrt{\log p})}, 2^{O(\log\log p \cdot \sqrt{\log p})})$*-bicriteria approximation algorithm for the minimum degree diameter-bounded network problem, where* $p = |V_P|$ *denotes the number of terminals.*

Proof. First of all, we can bound the deviation from D, which we denote by $\beta(p)$ and depends on the number of terminals p, using the following equation.

$$\beta(p) \cdot D \leq \frac{2p \log p}{2^{\log p - \sqrt{\log p}}} + c_0 \cdot D \log p + \beta(2^{\log p - \sqrt{\log p}}) \cdot (c_0 + 1) \cdot D \log p$$

We get the first additive term on the right side from the greedy selection of shortest paths in the second phase of the algorithm. Further, we can bound the maximum degree of the tree T' that we obtain by solving the auxiliary instance by $\Delta(T') \leq c_0 \cdot D \log p$ using Corollary 4.8. The term $\beta(2^{\log p - \sqrt{\log p}}) \cdot (c_0 + 1) \cdot D \log p$ further provides an upper bound for a solution of the new instance that is created in Step 20 of the algorithm. By rearranging and simplifying the previous equation we get the next slightly weaker bound.

$$\begin{aligned}
\beta(p) &\leq \frac{1}{D} \cdot \left(\frac{2p \log p}{2^{\log p - \sqrt{\log p}}} + c_0 \cdot D \log p + \beta(2^{\log p - \sqrt{\log p}}) \cdot (c_0 + 1) \cdot D \log p \right) \\
&\leq 2^{\sqrt{\log p}} \cdot 2 \log p + \beta(2^{\log p - \sqrt{\log p}}) \cdot (2c_0 + 1) \cdot \log p \\
&\leq (2c_0 + 1) \cdot \log p \cdot \beta(2^{\log p - \sqrt{\log p}}) + 2^{2\sqrt{\log p}}
\end{aligned}$$

We show by induction that there is a constant c_1, such that $\beta(p) \leq 2^{c_1 \log\log p \cdot \sqrt{\log p}}$. The statement trivially holds for small p, i.e., if p is a constant. Using the induction hypothesis we get that

$$\begin{aligned}
\beta(p) &\leq (2c_0 + 1) \cdot \log p \cdot 2^{c_1 \log(\log p - \sqrt{\log p}) \cdot \sqrt{\log p - \sqrt{\log p}}} + 2^{2\sqrt{\log p}} \\
&\leq 2^{d_1 + \log\log p} \cdot 2^{c_1 \log\log p \cdot \sqrt{\log p} - (c_1 \log\log p)/2} + 2^{2\sqrt{\log p}}
\end{aligned}$$

$$= 2^{c_1 \log\log p \cdot \sqrt{\log p}} \cdot 2^{d_1 - ((c_1/2)-1)\log\log p} + 2^{2\sqrt{\log p}},$$

where d_1 is a constant as well. This expression can further be bounded by

$$2^{c_1 \log\log p \cdot \sqrt{\log p}} \cdot \left(2^{d_1 - ((c_1/2)-1)\log\log p} + 2^{2 - c_1 \log\log p \cdot \sqrt{\log p}}\right).$$

Since $\log\log p$ is monotone increasing there exists a constant c_1, such that for all $p \geq c_1$:

$$2^{d_1 - ((c_1/2)-1)\log\log p} + 2^{2 - c_1 \log\log p \cdot \sqrt{\log p}} \leq 1,$$

and thus we have that $\beta(p) \leq 2^{c_1 \log\log p \cdot \sqrt{\log p}}$, which completes the first part of the proof. It remains to show that $\alpha(p) = 2^{O(\log\log p \cdot \sqrt{\log p})}$, where $\alpha(p)$ is the violation of the diameter constraint L. Similar to $\beta(p)$ we can bound $\alpha(p)$ as follows.

$$\alpha(p) \cdot L \leq 8\log p \cdot (c_0 \cdot L\log^2 p + \alpha(2^{\log p - \sqrt{\log p}}) \cdot (2c_0 + 1) \cdot L\log^2 p)$$

The factor of $8\log p$ is due to the potential reduction of sender-receiver pairs. We get the other terms from the construction of tree T' for the auxiliary graph and the newly created instance. By simplifying this equation we get another upper bound for $\alpha(p)$.

$$\begin{aligned}
\alpha(p) &\leq 8\log p \cdot \frac{1}{L} \cdot (c_0 \cdot L\log^2 p + \alpha(2^{\log p - \sqrt{\log p}}) \cdot (2c_0 + 1) \cdot L\log^2 p) \\
&\leq 8\log p \cdot \frac{1}{L} \cdot \alpha(2^{\log p - \sqrt{\log p}}) \cdot 4c_0 \cdot L\log^2 p \\
&= 32c_0 \cdot \log^3 p \cdot \alpha(2^{\log p - \sqrt{\log p}})
\end{aligned}$$

Analogous to the previous part, we show by induction that there is a constant c_2, such that $\alpha(p) \leq 2^{c_2 \log\log p \cdot \sqrt{\log p}}$. Again, the statement holds for $p = O(1)$. By plugging in the induction hypothesis we get that

$$\begin{aligned}
\alpha(p) &\leq 32c_0 \cdot \log^3 p \cdot 2^{c_2 \log(\log p - \sqrt{\log p}) \cdot \sqrt{\log p - \sqrt{\log p}}} \\
&\leq 2^{d_2 + 3\log\log p} \cdot 2^{c_2 \log\log p \cdot \sqrt{\log p} - (c_2 \log\log p)/2} \\
&= 2^{c_2 \log\log p \cdot \sqrt{\log p}} \cdot 2^{d_2 - ((c_2/2)-3)\log\log p},
\end{aligned}$$

where d_2 is a constant. Note that there exists another constant c_2, such that for all $p \geq c_2$: $2^{d_2 - ((c_2/2)-3)\log\log p} \leq 1$. Thus, we have that $\alpha(p) \leq 2^{c_2 \log\log p \cdot \sqrt{\log p}}$.

□

4.3 Minimum Total-Degree Diameter-Bounded Group Networks

In this section, we consider the minimum total-degree diameter-bounded group network problem and argue why developing approximation algorithms for this problem might be helpful to derive a polylogarithmic approximation algorithm for the minimum degree diameter-bounded network problem. We further show that the minimum total-degree diameter-bounded group network problem admits a polylogarithmic bicriteria approximation.

Definition 4.11. *Given a graph G, a diameter bound L and k groups $V_1, ..., V_k \subseteq V(G)$. The minimum total-degree diameter-bounded group network problem is to find a collection of subgraphs $\mathcal{T} = T_1, ..., T_k$ such that*

1. *T_i is a tree and connects all nodes in V_i,*
2. *$\text{diam}(\mathcal{T}) = \max\{\text{diam}(T_i) \mid 1 \leq i \leq k\} \leq L$,*
3. *and $\Delta(\mathcal{T}) = \max_{v \in V(G)} \sum_{i=1}^{k} \deg(T_i, v)$ is minimized.*

For a node $v \in V(G)$ we call $\deg(\mathcal{T}, v) = \sum_{i=1}^{k} \deg(T_i, v)$ the total-degree of v in $\mathcal{T}$. Throughout this thesis we assume that the number of groups is polynomial in n. It is not hard to see that the minimum total-degree diameter-bounded group network problem to some extent generalizes the minimum poise spanning tree problem.

Theorem 4.12. *Unless P=NP, there is no polynomial time algorithm for the minimum total-degree diameter-bounded group network problem.*

Proof. We show that an α-approximation algorithm for the minimum total-degree diameter-bounded group network problem implies an α-approximation for the minimum poise spanning tree problem. Suppose algorithm ALG is an α-approximation for the minimum total-degree diameter-bounded group network problem, and we are given an instance G for the minimum poise spanning tree problem. Let T^* be a spanning tree for G of minimal poise with $\text{diam}(T^*) = L$ and $\Delta(T^*) = D$. Since we assume that the input graph G is connected we know that such a solution always exists.

Algorithm 9 MTDDBGN TO MPST

Input: (G, P)

1. **for** $i = 1, ..., n$ **do**
2. Solve instance $(G, i, V(G))$ and let T_i denote the unique tree in the provided approximate solution.
3. **end for**
4. Let T be the tree T_i with $\text{poise}(T_i, P) \leq \text{poise}(T_j, P)$ for all $j \in \{1, ..., n\}$.

Output: T

We claim that Algorithm 9 is an α-approximation for the minimum poise spanning tree problem. Note that $\mathcal{T}^* = T^*$ is a valid solution for $(G, L, V(G))$. We know that $\Delta(T_L) \leq \alpha \cdot D$ because ALG provides an α-approximation for $(G, L, V(G))$, and thus the poise of T_L is at most

$$\text{poise}(T_L, P) \leq L + \alpha D \leq \alpha(L + D).$$

Since the procedure outputs the graph of minimum poise, the claim of the theorem follows. □

Definition 4.13. *An algorithm is called an (α, β)-bicriteria approximation algorithm for the minimum total-degree diameter-bounded group network problem, if for any instance $(G, L, V_1, ..., V_k)$ the following holds: let $\mathcal{T}^*$ be an optimal solution for $(G, L, V_1, ..., V_k)$. Then the algorithm outputs a solution $\mathcal{T}$, such that* $\text{diam}(\mathcal{T}) \leq \alpha \cdot L$ *and* $\Delta(\mathcal{T}) \leq \beta \cdot \Delta(\mathcal{T}^*)$.

Connection to Minimum Degree Diameter-Bounded Networks

We highlight the connection to the minimum degree diameter-bounded network problem, which further implies a connection to telephone concurrent multicasting and motivates developing approximation algorithms for the minimum total-degree diameter-bounded group network problem. Given a valid solution $\mathcal{T} = T_1, ..., T_k$ for an instance $(G, L, V_1, ..., V_k)$, we call $H(\mathcal{T})$, which is the union of all T_i, the graph induced by $\mathcal{T}$. It is important to notice that $\Delta(H(\mathcal{T})) \leq \Delta(\mathcal{T})$ always holds.

Observation 4.14. *Let (G, P, L) be a minimum degree diameter-bounded network problem instance and $(G, L', V_1, ..., V_k)$, $L' \leq L$, an instance of the minimum total-degree diameter-bounded group network problem. If for every $(s, t) \in P$ there exists a group V_i, such that $s, t \in V_i$, then the graph induced by solution $\mathcal{T}$ for $(G, L, V_1, ..., V_k)$ is a valid solution for (G, P, L).*

This observation holds because a feasible solution to $(G, L, V_1, ..., V_k)$ will connect all nodes belonging to a group via a path of length at most L, and thus

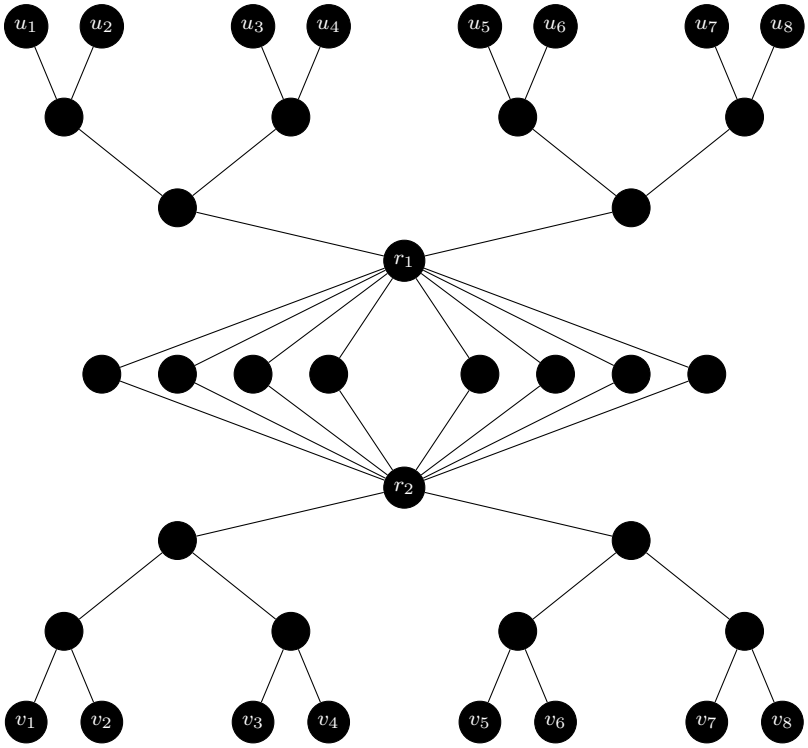

Figure 4.2: Example graph with $k = 8$.

especially s and t for every pair $(s, t) \in P$. However, there might be a huge gap between the maximum degree of $H(\mathcal{T})$ and the maximum degree of an optimal solution to the minimum degree diameter-bounded network problem as the following two examples show.

Consider a graph G, which consists of two binary trees with k leaves each, where the roots are connected via k different paths of length two, as pictured in Figure 4.2. Suppose there are precisely k pairs $(u_i, v_i) \in P$, $i \in \{1, ..., k\}$, where $k = 2^c$ for some $c \in \mathbb{N}$. An optimal solution H^* for (G, P, L) with $L = 2\lceil \log k \rceil + 2$ in the minimum degree diameter-bounded network problem would consist of both trees and one of the k paths between r_1 and r_2. Hence, this solution has maximum degree $\Delta(H^*) = 3$. Now take the instance $(G, L, V_1, ..., V_k)$ of the minimum total-degree diameter-bounded group network problem, where $V_i = \{u_i, v_i\}$ for all $i \in \{1, ..., k\}$. Note that this instance fulfills the requirements of Observation 4.14. An optimal solution $\mathcal{T}^*$ for $(G, L, V_1, ..., V_k)$ however might use all k different paths between r_1 and r_2. It follows that the graph $H(\mathcal{T}^*)$ induced by $\mathcal{T}^*$ might have maximum degree $\Delta(H(\mathcal{T}^*)) = \Omega(k)$. Yet there is a selection of groups $V_1, ..., V_{k'}$ for P, such that the maximum degree of the graph induced by an optimal solution for $(G, L, V_1, ..., V_{k'})$ matches $\Delta(H^*)$. Let (G, L, V) be a new instance for the minimum total-degree diameter-bounded

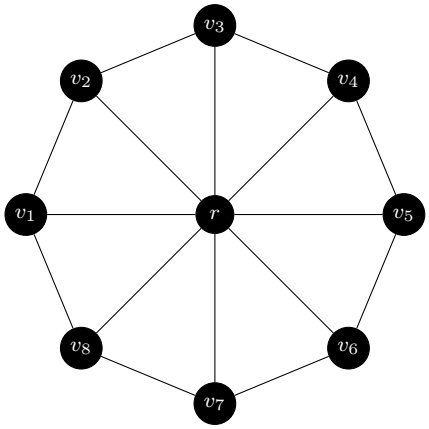

Figure 4.3: Wheel graph with $n = 9$ nodes.

group network problem, where V contains both u_i and v_i for all $i \in \{1, ..., k\}$. Then the induced graph $H(\mathcal{T}')$ of an optimal solution $\mathcal{T}'$ for (G, L, V) has $\text{diam}(H(\mathcal{T}'), P) = 2\lceil \log k \rceil + 2$ and $\Delta(H(\mathcal{T}')) = 3$ as well.
For the second example, consider a wheel graph G with center node r and outer nodes $v_1, ..., v_{n-1}$ as pictured in Figure 4.3. Together with this graph we are given pairs $P = \{(v_i, v_{i+1}) \mid 1 \leq i \leq n-2\} \cup \{(v_{n-1}, v_1)\}$. An optimal solution for instance $(G, P, 1)$ of the minimum degree diameter-bounded network problem would be the circle graph $H^* = G[V(G) \setminus \{r\}]$ with $\text{diam}(H^*, P) = 1$ and $\Delta(H^*) = 2$.
We can define a corresponding instance $(G, 1, V)$ of the minimum total-degree diameter-bounded group network problem by choosing $V = \{v_1, ..., v_{n-1}\}$. Note that for $(G, 1, V)$ no solution exists that obeys the diameter constraint. However, if we consider an instance $(G, 1, V_1, ..., V_{n-1})$, where $V_i = \{s_i, t_i\}$, $(s_i, t_i) \in P$, the graph induced by the unique optimal solution to this instance would match the subgraph H^*.
The next theorem shows that there is always a selection of groups, such that the induced graph of an optimal solution for the minimum total-degree diameter-bounded group network problem provides a good approximation for the minimum degree diameter-bounded network problem.

Theorem 4.15. *Let (G, P, L) be an instance for the minimum degree diameter-bounded network problem with optimal solution H and $k = |S_P|$. There exists an instance $(G, L', V_1, ..., V_{k'})$, $k' \leq k$, for the minimum total-degree diameter-bounded group network problem with optimal solution $\mathcal{T}^*$, such that*

- *the graph $H(\mathcal{T}^*)$ induced by $\mathcal{T}^*$ connects s and t for all $(s, t) \in P$,*
- *$\Delta(\mathcal{T}^*) \leq 4 \log k \cdot \Delta(H)$,*

- $L' \leq 4 \log k \cdot L$.

Proof. Consider the clustering $\mathcal{C}$, which consists of clusters $N_L(H, s)$ for all $s \in S_P$, i.e., for each sender we have the L-neighborhood of that sender in H as a cluster in $\mathcal{C}$. By Theorem 3.9, there exists a partition cover $\mathcal{P}$ for $\mathcal{C}$, such that $\text{diam}(\mathcal{P}) \leq 2L \cdot \log k$ and $\Delta(\mathcal{P}) \leq 4 \log k$. Let $\mathcal{C}' = \{C_1, ..., C_{k'}\}$ be the set that contains all clusters of the partition cover $\mathcal{P}$ and $L' = 2 \cdot \text{diam}(\mathcal{P})$. Notice that $k' \leq k$ and $L' \leq 4 \log k \cdot L$. We now show that there exists a solution for instance $(G, L', C_1, ..., C_{k'})$ with the desired properties, which implies the theorem.
Let $\mathcal{T} = T_1, ..., T_{k'}$, where T_i is a minimum diameter spanning tree for $H[C_i]$, $i \in \{1, ..., k'\}$. Since $\text{diam}(\mathcal{P}) \leq 2L \cdot \log k$ it immediately follows that $\text{diam}(\mathcal{T}) \leq 4 \log k \cdot L = L'$. Note that $H(\mathcal{T})$ is a valid solution for instance (G, P, L') because for each $(s, t) \in P$ there is a tree T_i, which connects s and t. Moreover, the degree of a node v is bounded by $\deg(T_j, v) \leq \deg(H[C_j], v) \leq \Delta(H)$ for all $j \in \{1, ..., k'\}$. Finally, we know that v appears in at most $4 \log k$ clusters because the number of layers of $\mathcal{P}$ is bounded by $4 \log k$. It follows that $\Delta(H(\mathcal{T})) \leq 4 \log k \cdot \Delta(H)$. □

Unfortunately, we do not know how to group the terminals V_P of an instance (G, P, L) of the minimum degree diameter-bounded network problem without an optimal solution H^* for (G, P, L). Suppose we have access to an oracle, which tells us how to group the terminals of (G, P, L). Together with a polylogarithmic approximation algorithm for the minimum total-degree diameter-bounded group network problem, which we describe in the remaining part of this chapter, this would lead to a polylogarithmic approximation algorithm for the minimum degree diameter-bounded network problem as Algorithm 10 demonstrates.

Algorithm 10 MTDDBN TO MDDBN

Input: (G, P, L)
1. Let the oracle group the terminals of P into groups $V_1, ..., V_{k'}$.
2. Solve instance $(G, L, V_1, ..., V_{k'})$ and let $\mathcal{T}$ be the provided approximate solution.

Output: $H(\mathcal{T})$

Polylogarithmic Bicriteria Approximation Algorithm

Algorithm 11 is a modification of an algorithm by Ravi, and our proof is very similar to the proof in [Rav94]. For the remaining part of this chapter, suppose we are given an instance $(G, L, V_1, ..., V_k)$ for the minimum total-degree

diameter-bounded group network problem and p denotes the maximum number of terminals in a group, i.e., $p = \max_{1 \leq i \leq k} |V_i|$. We show that Algorithm 11 is an $(O(\log p), O(\log p \log n))$-bicriteria approximation algorithm. Therefore, we first bound the number of iterations.

Lemma 4.16. *Algorithm* 11 *terminates after* $O(\log p)$ *iterations.*

Proof. First, notice that by definition $|\mathcal{C}_i| \leq p$ for all $i \in \{1, ..., k\}$. We show that in each iteration the number of clusters in each collection $\mathcal{C}_i$ decreases by a factor of at least one third. Let $\mathcal{C}_i$ be an arbitrary collection of clusters. In any iteration t, the number of edges in all the stars of $\mathcal{S}_i$, which we denote by $E(\mathcal{S}_i)$, represents the number of reductions of clusters because all clusters belonging to a star are merged to a new cluster. Note that for each $F \in \mathcal{F}_i$ the algorithm divides $E(F)$ in two sets and chooses the set with higher cardinality. Hence, the number of star edges due to tree F is at least $\lceil \frac{|F|-1}{2} \rceil$. By Lemma 1.11 each tree consists of at least two nodes and thus $\lceil \frac{|F|-1}{2} \rceil \geq \frac{|F|}{3}$. Moreover, Lemma 1.11 states that any node of G_i is contained in some tree F. Hence, we have that $\sum_{F \in \mathcal{F}_i} \frac{|F|}{3} \geq \frac{|G_i|}{3} = \frac{|\mathcal{C}_i|}{3}$. It follows that there can be at most $\lceil \log_{3/2} p \rceil$ iterations, and we can bound $\lceil \log_{3/2} p \rceil$ by $\lceil 2 \log p \rceil$. □

Lemma 4.17. *Let* C *be a cluster formed in iteration* t *of the algorithm with center* $\text{cen}(C) = v$*. Then the spanning tree* $T(C)$ *for cluster* C *has diameter at most* $2t \cdot L$*.*

Proof. We show that any node $u \in C$ has distance at most $t \cdot L$ to v in $T(C)$. The proof is by induction on t and the claim trivially holds for $t = 1$. For $t > 1$, suppose cluster C was formed by merging clusters $C_1, ..., C_{|S|}$ that were grouped by a star S, where v is the center of the star S. Further, let C_j be the cluster with $\text{cen}(C_j) = v$ before the merging process. For the construction of spanning tree $T(C)$ we take the union of all spanning trees $T(C_1), ..., T(C_{|S|})$ together with the flow paths. For this graph a breadth first search tree $T(C)$ is computed with root v. We now show that any node in $T(C)$ has distance at most $t \cdot L$ to v.

Let $u \in C$ be an arbitrary node in the cluster. Then u is either contained in one of the clusters $C_1, ..., C_{|S|}$ or part of a flow path of the integral solution. If u is on a flow path, the claim holds, since every flow path has length at most L. If u is in cluster C_j, then by the induction hypothesis there is a path of length $(t-1)L$ from u to v in $T(C_j)$. If u is in some other cluster $C_{j'}$, $j' \neq j$, we get by the induction hypothesis that there is a path from u to $\text{cen}(C_{j'})$ of length at most $(t-1)L$. Further there is a flow path from $\text{cen}(C_{j'})$ to v of length L, which adds up to a path of length $t \cdot L$. □

Since there are at most $O(\log p)$ iterations, the diameter of a tree $T_i(C)$ provided by the algorithm can be bounded by $O(L \log p)$. We can show a similar bound for the maximum total-degree of the solution.

Lemma 4.18. *Let $\mathcal{T}^* = T_1, ..., T_k$ be an optimal solution for $(G, L, V_1, ..., V_k)$ and $D = \Delta(\mathcal{T}^*)$. In each iteration of the algorithm, the total-degree of each node increases by at most $O(D + \log n)$.*

Proof. First, notice that the instance of the congestion flow problem of Algorithm 11 has at most $O(kp^2 + n)$ nodes because for every node of each group V_i we add at most $p + 1$ new nodes.
We show that there always exists an integral solution for the congestion flow problem with maximum node-congestion $2D$. Let V_i be an arbitrary group and suppose the number of center nodes in G_i is even. The tree T_i connects all nodes of V_i and has diameter at most L. Since the center nodes of G_i are all distinct, we can find a pairing of the centers such that the paths between pairs in T_i are edge-disjoint using Lemma 1.10. Consider the integral solution where for each of these pairs (v_1, v_2) we send the flow belonging to v_1 to v_2 and the flow of v_2 to v_1 along the pairing-edges. The length of each flow path is at most L since the diameter of T_i is bounded by L. Further, the congestion at any node is at most $2 \cdot \deg(T_i, v)$ because each edge in T_i is used at most twice and the paths are all edge-disjoint. Since $D = \Delta(\mathcal{T}^*) = \max_{v \in V(G)} \sum_{i=1}^{k} \deg(T_i, v)$, the constructed integral solution has a maximum node-congestion of at most $2D$. When there is an odd number of nodes in G_i, we can ignore one node and apply the above argument. Afterwards, the omitted node can be connected to a closest node in T_i without violating the node-congestion bound.
Because there exists a solution with maximum node-congestion $2D$ in each iteration we can obtain an integral solution of congestion at most $O(D + \log(kp^2 + n)) = O(D + \log n)$ by Theorem 1.8. The total-degree of each node due to the addition of flow paths is further bounded by its congestion, and thus $\max_{v \in V(G)} \sum_{i=1}^{n} \deg(H_i, v) = O(D + \log n)$. Since the tree $T_i(C')$ is a subgraph of H_i, $i \in \{1, ..., k\}$, the total-degree of each node increases by at most $O(D + \log n)$ in each iteration. □

Theorem 4.19. *Algorithm 11 is an $(O(\log p), O(\log n \log p))$-bicriteria approximation for the minimum total-degree diameter-bounded group network problem, where p is the maximum number of terminals in a group.*

Proof. This theorem is an immediate consequence of Lemma 4.16, Lemma 4.17 and Lemma 4.18 □

Algorithm 11 GROUP NETWORKS

Input: $(G, L, V_1, ..., V_k)$

1. For every group V_i: create a collection of clusters $\mathcal{C}_i$ consisting of a singleton for every node $v \in V_i$. Define the center of cluster $\{v\}$ to be $\text{cen}(\{v\}) = v$ and $T_i(\{v\}) = (\{v\}, \varnothing)$ to be the spanning tree of cluster $\{v\}$.
2. **while** $\exists \mathcal{C}_i : |\mathcal{C}_i| > 1$ **do**
3. For all $\mathcal{C}_i$ with $|\mathcal{C}_i| > 1$, let $U_i = \{v \in V(G) \mid \text{cen}(C) = v,\ C \in \mathcal{C}_i\}$. Denote the set of all those U_i by $\mathcal{U}$.
4. Set up a length-bounded minimum node-congestion problem as follows: for each node $v_j \in U_i$ define a commodity j and add a source node s_j to G together with an edge (s_j, v_j). Add a new node t_j to G representing the sink. Add a directed edge (v, t_j) for all $v \neq v_j, v \in U_i$. Set the diameter bound to $L + 2$.
5. Obtain an approximate optimal integral solution using Theorem 1.8. This provides a set of flow paths that connect the centers of clusters of each collection.
6. **for** $\mathcal{C}_i$: $|\mathcal{C}_i| > 1$ **do**
7. Let $G_i = (U_i, \varnothing)$. For every flow path belonging to group V_i add a directed edge (u, v) to G_i, which represents a flow path of length at most L from u to v.
8. Identify a collection $\mathcal{F}_i$ of disjoint directed trees that are subgraphs of G_i using Lemma 1.11.
9. **for** $F \in \mathcal{F}_i$ **do**
10. Partition the edges of F in odd and even levels. Let E be the one of the two edge sets with higher cardinality. The subgraph of F induced by E defines a collection of stars $\mathcal{S}_i$, where the edges of a star $S \in \mathcal{S}_i$ point to the center v_S of the star. Further, each star consists of at least 2 nodes.
11. **for** $S \in \mathcal{S}_i$ **do**
12. Let H_i be the union of spanning trees $T_i(C)$ that belong to $C \in \mathcal{C}_i$ with $\text{cen}(C) \in S$. Add the flow path of every node $v \in S \setminus \{v_S\}$ represented by (v, v_S) to H_i.
13. Merge all $C \in \mathcal{C}_i$ with $\text{cen}(C) \in S$ to a cluster C' with center v_S.
14. Compute a breadth first search spanning tree $T_i(C')$ for H_i starting in v_S.
15. **end for**
16. **end for**
17. **end for**
18. **end while**

Output: The unique spanning tree $T_i(C)$ for every group V_i.

Chapter 5

Minimum Degree Graph Spanners

Our results from the previous chapters have implications for the computation of graph spanners with small maximum degree. A k-spanner H for an undirected graph G is a subgraph of G, such that the distance between two nodes increases by a factor of at most k, i.e., $\text{dist}(H, u, v) \leq k \cdot \text{dist}(G, u, v)$ for all $u, v \in V(G)$. The parameter k is called the stretch factor of a k-spanner H.

Definition 5.1. *Given a graph G and a stretch factor $k \in \mathbb{N}$. The minimum degree k-spanner problem is to find a k-spanner H for G, such that for all k-spanner H' for G: $\Delta(H) \leq \Delta(H')$.*

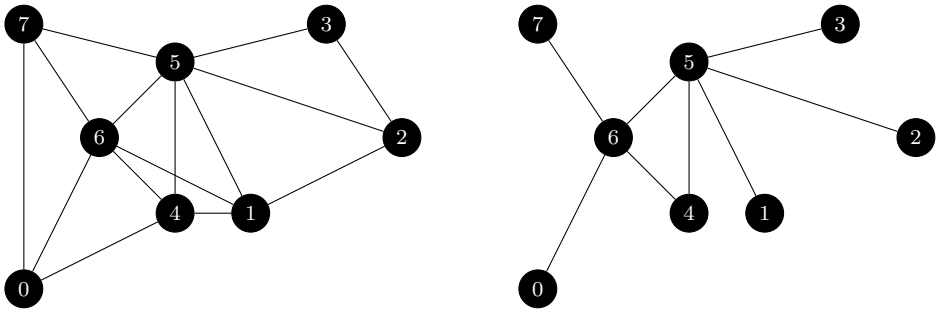

Figure 5.1: A graph G and a 2-spanner H for G. The distance between two nodes u and v in H is at most twice the distance between u and v in G.

Computing a minimum degree k-spanner for a graph G is known to be NP-hard [KP98]. We present two bicriteria approximation algorithms for the minimum degree k-spanner problem. The first is an immediate consequence of Algorithm 8 while the second algorithm uses sparse partition covers.

Definition 5.2. *An algorithm is called an (α, β)-bicriteria approximation algorithm for the minimum degree k-spanner problem, if for any instance (G, k) with optimal solution H^* the algorithm outputs a subgraph H, such that H is an $(\alpha \cdot k)$-spanner for G and $\Delta(H) \leq \beta \cdot \Delta(H^*)$.*

5.1 Bicriteria Approximation Algorithms

Connection to Minimum Degree Diameter-Bounded Networks

As we have seen earlier, Algorithm 2 of Nikzad and Ravi can be modified to a $(2^{O(\log\log p\cdot\sqrt{\log p})}, 2^{O(\log\log p\cdot\sqrt{\log p})})$-bicriteria approximation for the minimum degree diameter-bounded network problem. Here the parameter p corresponds to the number of terminals for a given instance. It is easy to see that this has implications for the minimum degree k-spanner problem since the minimum degree k-spanner problem is a special case of the minimum degree diameter-bounded network problem, where k corresponds to the diameter bound and $P = \{(u,v) \mid \{u,v\} \in E(G)\}$. Notice that $|P| \leq n(n-1)$ and thus the next theorem is an immediate consequence of Theorem 4.10.

Theorem 5.3. *Algorithm* 12 *is a* $(2^{O(\log\log n\cdot\sqrt{\log n})}, 2^{O(\log\log n\cdot\sqrt{\log n})})$*-bicriteria approximation for the minimum degree* k*-spanner problem.*

Algorithm 12 Minimum Degree Spanner I

Input: (G,k)

1. $P \leftarrow \{(u,v) \mid \{u,v\} \in E(G)\}$
2. Solve instance (G,P,k) using Algorithm 8 and let H be the provided $(2^{O(\log\log p\cdot\sqrt{\log p})}, 2^{O(\log\log p\cdot\sqrt{\log p})})$-approximation.

Output: H

Bicriteria Approximation Using Sparse Partition Covers

For small k, i.e., $k = O(\log^c n)$ for some constant $c > 0$, we can give an even better bicriteria approximation using sparse partition covers. Let D_k denote the maximum degree of an optimal solution to the minimum degree k-spanner for graph G. The number of possible values for D_k is bounded by n, and thus an algorithm can try each of these values in polynomial time. The following lemma together with Theorem 1.9 shows that a set of Steiner trees $\mathcal{T} = \{T_C \mid C \in \mathcal{C}\}$ with the properties described in Step 7 of Algorithm 13 can always be found.

Lemma 5.4. *Let* C *be a cluster in* $\mathcal{P}$*. There exists a tree* T*, which connects all vertices in* V_C*, such that* $\operatorname{diam}(T) \leq 4k^2 \log n$ *and* $\Delta(T) = D_k$*.*

Proof. Let H be a k-spanner for G with $\Delta(H) = D_k$. Since H is a k-spanner for G, each edge of G is replaced by a path of length at most k in H. It follows that for every edge $\{u,v\} \in \operatorname{core}(C)$ the corresponding path in H is fully contained in $G[C]$. Let T be a tree in $H[C]$, which connects all vertices of V_C and has

Algorithm 13 Minimum Degree Spanner II

Input: (G, k)

1. Guess the maximum degree D_k of a minimum degree k-spanner for G, e.g., by performing a binary search on $\{1, ..., n\}$.
2. $H \leftarrow \varnothing$
3. For every edge $\{u, v\} \in E(G)$ define cluster $C(u, v) = N_k(G, u) \cup N_k(G, v)$. Let $\mathcal{C}$ be the set that contains all of these clusters.
4. Compute a sparse partition cover $\mathcal{P}$ for $\mathcal{C}$, such that $\text{diam}(\mathcal{P}) = O(k \log n)$ and $\Delta(\mathcal{P}) = O(\log n)$, using Theorem 3.9.
5. **for** C in $\mathcal{P}$ **do**
6. Let $\text{core}(C) = \{\{u, v\} \in E(G) \mid C(u, v) \subseteq C\}$ and $V_C \subseteq C$ be the set of vertices that are incident to an edge in $\text{core}(C)$.
7. Compute a Steiner tree T_C for $G[C]$ with terminals V_C using Theorem 1.9, such that $\text{diam}(T_C) = O\left(\frac{\log^2 n}{\log \log n} \cdot k^2\right)$ and $\Delta(T_C) = O\left(\frac{\log n}{\log \log n} \cdot D_k\right)$.
8. **end for**
9. Let H be the union of all trees T_C, where C is a cluster of $\mathcal{P}$.

Output: H

minimal diameter. Obviously it holds that $\Delta(T) \leq \Delta(H)$. Since H replaces every edge in the core of the cluster by a path of length at most k in $G[C]$ and $\text{diam}(G[C]) \leq 2k \log n$, it follows that the diameter of $G[C]$ is at most $2k^2 \log n$. Further, T is a minimum diameter spanning tree for $G[C]$ and thus we get that $\text{diam}(T) \leq 4k^2 \log n$. □

It follows that each tree that is computed in Step 6 of Algorithm 13 has diameter at most $O\left(\frac{\log^2 n}{\log \log n} \cdot k^2\right)$ and maximum degree at most $O\left(\frac{\log n}{\log \log n} \cdot D_k\right)$. Note that for every edge $\{u, v\} \in E(G)$ there exists a cluster C in $\mathcal{P}$, which fully contains $C(u, v)$, and thus u and v are both contained in V_C. Since Algorithm 13 computes a tree that connects all nodes of V_C for every cluster C with diameter $O\left(\frac{\log^2 n}{\log \log n} \cdot k^2\right)$, it follows that H is an $O\left(\frac{\log^2 n}{\log \log n} \cdot k^2\right)$-spanner. Together with the fact that each node is contained in at most $O(\log n)$ clusters we get the next theorem.

Theorem 5.5. *Algorithm 13 is an* $\left(O\left(\frac{\log^2 n}{\log \log n} \cdot k\right), O\left(\frac{\log^2 n}{\log \log n}\right)\right)$*-bicriteria approximation for the minimum degree k-spanner problem.*

Chapter 6

Conclusion and Further Work

We described two of the most prominent models for message dissemination in communication networks, namely the telephone and the radio model. For each of these models we considered the broadcast, gossip, multicast and concurrent multicast problem, and presented known upper and lower bounds.

We first provided simple upper bounds for radio concurrent multicasting and further showed that for every instance (G, P) we can compute a concurrent multicast schedule $\mathcal{S}$ of length $O((\text{diam}(G, P) \log k + \Delta(G) \log n) \log k)$, where k corresponds to the number of senders of the instance.

We highlighted the connection between concurrent multicasting in the telephone model and networks with low poise. The minimum degree diameter-bounded network and the minimum total-degree diameter-bounded group network problem were introduced in the hope of leading to a polylogarithmic approximation algorithm for telephone concurrent multicasting. We showed that the Algorithm by Nikzad and Ravi can be modified to a $(2^{O(\log \log k \cdot \sqrt{\log k})}, 2^{O(\log \log k \cdot \sqrt{\log k})})$-bicriteria approximation for the minimum degree diameter-bounded network problem, where k denotes the number of terminals. Although the $(O(\log n), O(\log^2 n))$-bicriteria approximation algorithm for the minimum total-degree diameter-bounded group network problem did not directly imply a polylogarithmic approximation algorithm for telephone concurrent multicasting, this result might be of interest for other applications in the field of network design. Our observations and results had implications for the computation of minimum degree graph spanners, and we described two bicriteria approximation algorithms for the minimum degree k-spanner problem.

There are several directions for future research. The search for a polylogarithmic approximation algorithm for the concurrent multicast problem in the telephone model remains the main unsolved problem. Moreover, it would be of interest to know whether there is an even better bound for radio concurrent mul-

ticast schedules than $O((\text{diam}(G, P) \log |S_P| + \Delta(G) \log n) \log |S_P|)$. Another open question is whether there is a polylogarithmic bicriteria approximation algorithm for the minimum degree k-spanner problem.

Bibliography

[ABLP91] Noga Alon, Amotz Bar-Noy, Nathan Linial, and David Peleg. A lower bound for radio broadcast. *J. Comput. Syst. Sci.*, 43(2):290–298, 1991.

[AH03] Geir Agnarsson and Magnús M. Halldórsson. Coloring powers of planar graphs. *SIAM J. Discrete Math.*, 16(4):651–662, 2003.

[AP90] Baruch Awerbuch and David Peleg. Sparse partitions (extended abstract). In *31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume II*, pages 503–513, 1990.

[EK05] Michael Elkin and Guy Kortsarz. A combinatorial logarithmic approximation algorithm for the directed telephone broadcast problem. *SIAM J. Comput.*, 35(3):672–689, 2005.

[EK06] Michael Elkin and Guy Kortsarz. Sublogarithmic approximation for telephone multicast. *J. Comput. Syst. Sci.*, 72(4):648–659, 2006.

[GPX07] Leszek Gasieniec, David Peleg, and Qin Xin. Faster communication in known topology radio networks. *Distributed Computing*, 19(4):289–300, 2007.

[IRRS15] Jennifer Iglesias, Rajmohan Rajaraman, R. Ravi, and Ravi Sundaram. Rumors across radio, wireless, telephone. In *35th IARCS Annual Conference on Foundation of Software Technology and Theoretical Computer Science, FSTTCS 2015, December 16-18, 2015, Bangalore, India*, pages 517–528, 2015.

[KP98] Guy Kortsarz and David Peleg. Generating low-degree 2-spanners. *SIAM J. Comput.*, 27(5):1438–1456, 1998.

[KP07] Dariusz R. Kowalski and Andrzej Pelc. Optimal deterministic broadcasting in known topology radio networks. *Distributed Computing*, 19(3):185–195, 2007.

[KV07] Bernhard Korte and Jens Vygen. *Combinatorial Optimization: Theory and Algorithms*. Springer Publishing Company, Incorporated, 4th edition, 2007.

[NR14] Afshin Nikzad and R. Ravi. Sending secrets swiftly: Approximation algorithms for generalized multicast problems. In *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part II*, pages 568–607, 2014.

[Pel00] David Peleg. *Distributed Computing: A Locality-sensitive Approach*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.

[Rav94] R. Ravi. Rapid rumor ramification: Approximating the minimum broadcast time (extended abstract). In *35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994*, pages 202–213, 1994.

[RT87] Prabhakar Raghavan and Clark D. Thompson. Randomized rounding: a technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, 1987.

[Sou] Alexander Souza. Randomized algorithms & probabilistic methods. Lecture Notes, Summer Term 2011, Humboldt University Berlin.

[WP67] D. J. A. Welsh and M. B. Powell. An upper bound for the chromatic number of a graph and its application to timetabling problems. *The Computer Journal*, 10(1):85–86, January 1967.

[WS11] David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, 2011.

[Zuc07] David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory of Computing*, 3(6):103–128, 2007.