Communication Network Problems

Habilitationsschrift

von

Christian Schindelhauer

Fachbereich Mathematik/Informatik und Heinz Nixdorf Institut

Universität Paderborn

Paderborn, März 2002

Acknowledgments

This thesis and its preceding research were prepared at the following institutions:

- Technische Universität Darmstadt, Institut für Theoretische Informatik, research group of Prof. Dr. math. Rüdiger Reischuk
- Universität Lübeck, Institut für Theoretische Informatik, research group of Prof. Dr. math. Rüdiger Reischuk
- International Computer Science Institute, Berkeley, Theory Group, Prof. Richard Karp Ph.D.
- Heinz Nixdorf Institut, Universität Paderborn, Algorithms, Complexity Theory, Parallel Computing Group, Prof. Dr. math. Friedhelm Meyer auf der Heide

Furthermore I have to thank the Deutsche Forschungsgesellschaft (DFG) and the Deutscher Akademischer Austauschdienst (DAAD) for providing me Post Doc stipends, research grants, and traveling subsidies.

This work was also partially supported by the DFG-Sonderforschungsbereich 376 and by the Future and Emerging Technologies programme of the EU under contract number IST-1999-14186 (ALCOM-FT). Some results of this thesis have already been published in journals and conference proceedings, see [JRS94, JRS98, KSSV00, PS01, Sch00b, SW01]. I want to thank all co-authors for their contributions and for allowing the publication of our results within this habilitation thesis.

Furthermore, there is a (possibly incomplete) list of people helping in this work by discussions, suggestions, solutions, problems, remarks and hints to the literature: Nicolò Cesa-Bianchi, Fedor Fomin, Karin Genther, Matthias Grünewald, Andreas Jakoby, Richard Karp, Guy Kortsartz, Maciej Liskiewicz, Phil Long, Tamás Lukovszki, Friedhelm Meyer auf der Heide, Antonio Piccolboni, Rüdiger Reischuk, Scott Shenker, Marc Seroussi, Christian Scheideler, Frank Stephan, Klaus Volbert, Berthold Vöcking, Rolf Wanka, Manfred Warmuth, Roger Wattenhofer, Birgitta Weber, Marcelo Weinberg, Stephan Weis, and Thomas Zeugmann,

Finally, I thank my wife Barbara Schindelhauer (also for proof-reading) and my children Tina and Tom for some extra time in the final phase of the preparation of this thesis.

Paderborn, March 2002 Christian Schindelhauer

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The modern world relies on the availability of huge communication networks connecting computers and other electronic devices worldwide. Clearly, there is an enormous number of problems that must be solved. These problems can be manifold and very complex. We will make a theoretical approach of some specific issues in the context of network communication problems.

1.1 Broadcasting Information

Communication networks are often described as undirected graphs. The participants are nodes and information transmission is modeled by communication links between two nodes, abstracted by undirected edges. One of the basic information tasks is to spread an information from one node to all members of the communication network.

Assume that some people $A, B, C, \dots, J$ try to organize a party. Everybody knows only some other people, e.g. A knows B, C and D, while B knows A and E, C knows A, F and G etc. Such relations can be described by a graph where a player is depicted by a node and A knows B is presented by an edge. Such a relation graph is shown on the top of Figure 1.1.

Assume that A is the one who knows the time and location of the party. In the beginning A informs B. In a second round A and B can inform two participants in parallel, namely C and E. Then, the informed participants A, B, C, and E continue to inform all other players. In Figure 1.1 it is shown how the information spreads and how this information process terminates after six round.

In our simple mathematical model every communication step lasts one time unit and one node can only inform one neighbored node. Informed nodes can inform other nodes in parallel. This process begins with one or many informed nodes at time point 0. The broadcasting time is the number of time needed to inform all nodes. Note that given the graph and the informed nodes it is not clear how long it takes until all nodes are informed. Observe in Figure 1.2 that for the same graph an improved strategy achieves a broadcasting time of four.

In this context a variety of questions arise and we will discuss some of them.

- Time: How long does it take until all players have been informed in general?

  Figure 1.1 and 1.2 show that the time needed for broadcasting information depends on the strategy. Hence, to solve this questions we have to solve the following question first:

- Schedule: Which is the shortest schedule to organize the broadcast?

  It turns out that there is no known efficient algorithm that computes such schedules [MJ90]. However, this does not mean that for a specific given graph the problem cannot be solved at all.

- For which graph families can we solve this problem?

- Approximability: If the optimal broadcasting schedule cannot be computed, can we determine a good strategy which is only some additional rounds slower than the best schedule?

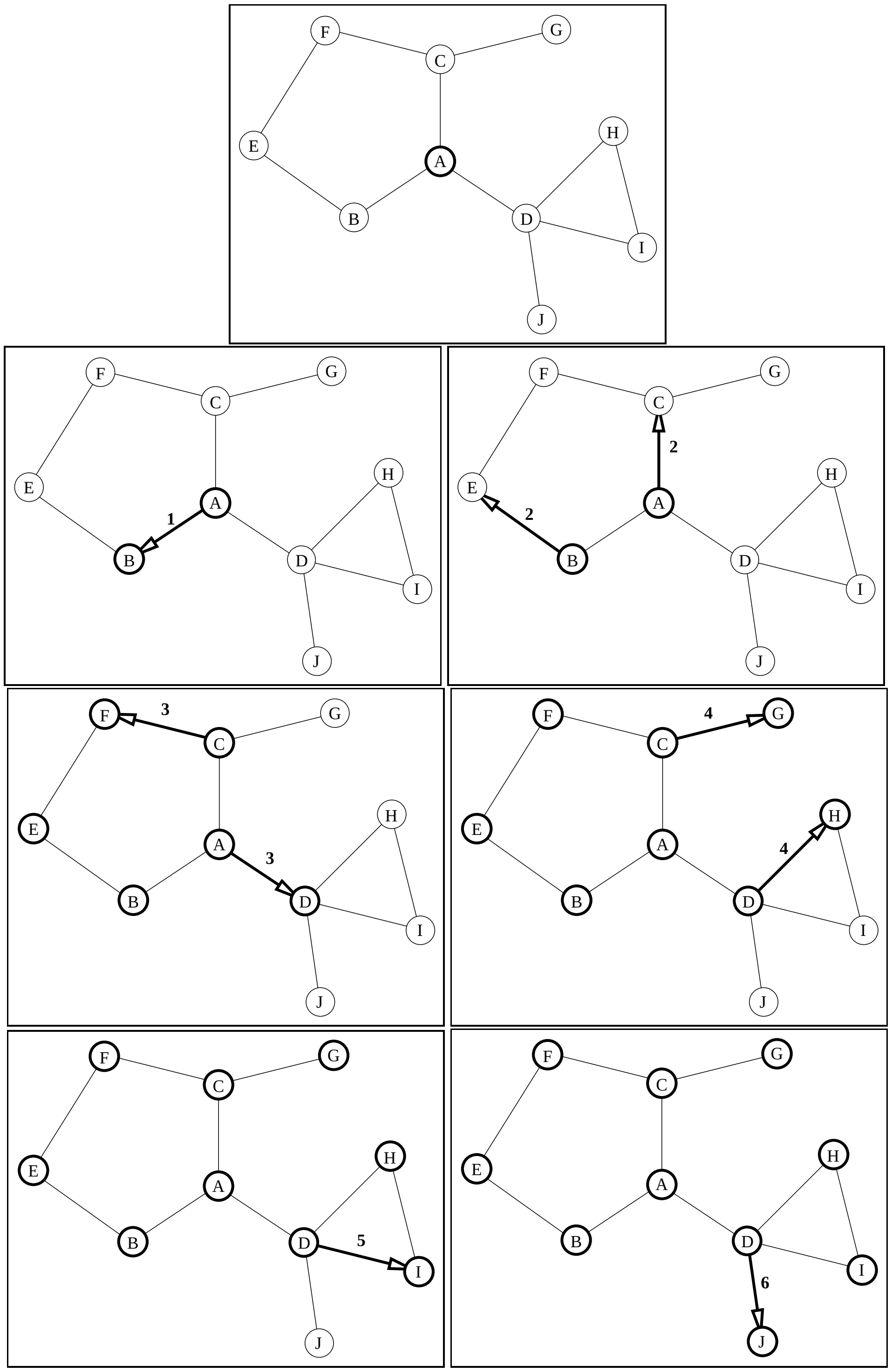

Figure 1.1: Player A starts the broadcasting process. Using this schedule six rounds suffice to inform all players.

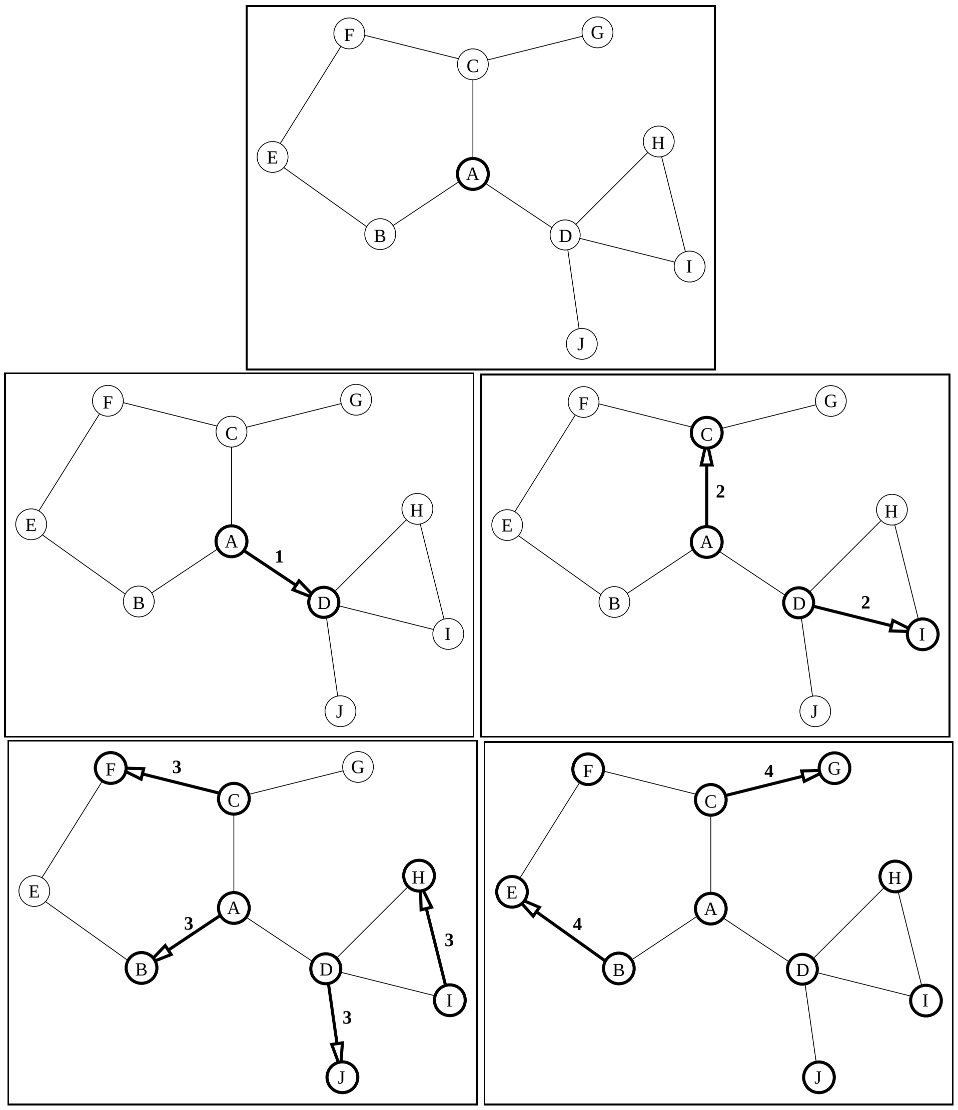

Figure 1.2: An improved broadcasting schedule with four rounds

1.2 Spreading Rumor and Viruses

The mathematical modeling of the spreading of infectious diseases goes back to Bernoulli in the 18th century. In the first half of the 20th century probabilistic methods have been used to predict the spreading behavior of diseases. For an overview see [Bai75]. In 1987 Pittel [Pit87] analyzed the expected contamination time for a very simple model.

> The virus spreads in rounds. In the beginning one individual is infected. In each round each individual contacts a random other of the n subjects. If the first individual is infected then the second will be contaminated this way and starts spreading the virus in the next round.

If we choose the contacted partner uniformly over all participants, then with probability $1 - 1/n$ two individuals are infected in the second round (because with probability $1/n$ the individual contacts itself). These two will infect two more in the second round with probability $1 - \frac{5}{n} + \frac{6}{n^2}$, and so on. It is intuitively clear that this way the number of infected beings grows exponentially unless a constant fraction of subjects is infected. Then, the number of uncontaminated participants will decrease exponentially. For this model Pittel showed that the expected time until all participants will be infected is $\log_2 n + \ln n + O(1)$.

Unlike in the broadcasting model discussed above, here the information strategy is not the problem. The virus, or the rumor, is spread as often as possible. Furthermore, the information process is robust. If in one round some participants fail to transmit the rumor, this does not change the general behavior. These may have been the arguments endorsing the application of this spreading mechanism to replicated databases under the name epidemic algorithms. In the landmark paper [DGH$^+$87] Demers et al. introduced this concept to the public.

They proposed to extend the information process as follows. In the above described standard model the caller informs the callee. In the case of contagious diseases it is not clear why the contacted subject should not also infect the contacting subject. This form of transmitting a virus is called a pull. It is not surprising that an additional transmission process speeds up the rumor spreading. However, in the second phase when at least half of the subjects are informed the number of informed players decreases double-exponentially, i.e. if $u(n)$ is the number of uninformed players in round t then $u(t) \leq \frac{cn}{2^{2^t}}$ for some constant c.

If we adopt the push and pull transmission scheme to broadcast information, we envisage a distributed system where new rumors pop up locally all the time and where in a time unit, e.g. one hour, every participating server contacts a random partner to exchange new rumors. So, the following questions arise:

- When is a rumor too old to be distributed?
- How many transmissions are necessary to inform all participants?
- Is there a robust termination mechanism ensuring fast broadcasting and small number of transmissions?

1.3 Bandwidth Allocation: Utilization versus Fairness

The Transmission Control Protocol (TCP) used in the Internet is an end-to-end congestion control mechanism. One of its main features is the so-called Additively Increasing and Multiplicatively Decreasing algorithm. In a simplified setting it works as follows:

> A router successively sends packets in a transmission rate. For every successful sent packet an acknowledgment appears after some time, this time is called the Round-Trip-Time (RTT). If the acknowledgment does not appear, then a sending failure appeared possibly due to exceeded bandwidth.
>
> The AIMD-mechanism works as follows. If packets are submitted successfully, then the acquired bandwidth is increased by an additive constant amount, if a failure occurs then the acquired bandwidth is reduced by a constant factor.

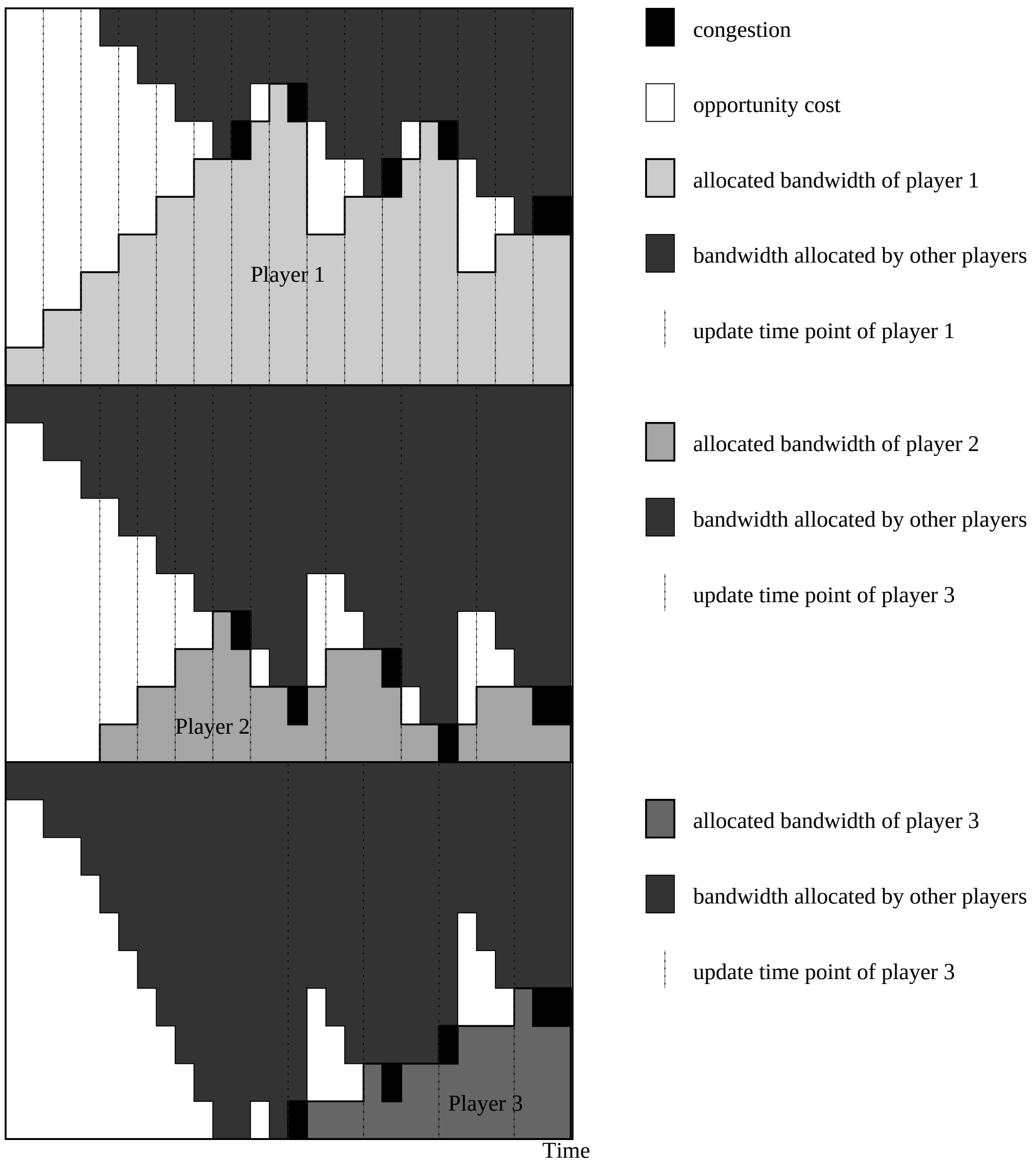
congestion
opportunity cost
allocated bandwidth of player 1
bandwidth allocated by other players
update time point of player 1
Player 1
allocated bandwidth of player 2
bandwidth allocated by other players
update time point of player 3
Player 2
allocated bandwidth of player 3
bandwidth allocated by other players
update time point of player 3
Player 3
Time

Figure 1.3: Schematic allocation behavior of TCP

Figure 1.3 shows a typical behavior of this protocol. We have three players sharing a link of a specific bandwidth. In the beginning player 1 starts and after some time player 2 and then player 3 joins in. They both linearly (additively) increase their share of the bandwidth until the sum of their allocations exceeds the bandwidth. In figure 1.3 this occurs four times. At the first occurrence player 2 is the one who recognizes it and he reduces his allocation by a constant factor (in our example he halves his packet rate).

Note that even in a best case scenario this AIMD-mechanism utilizes only a constant portion of the available bandwidth. One might argue that this is caused by the sparse feedback used by the allocation algorithm, who never learns the available bandwidth left-over by the other players. The player's only feedback is whether in the last round the allocated bandwidth was at most as high as the available bandwidth.

In the Internet packet failures are caused by the allocation behavior of other protocols. In Figure 1.3 player 1 receives an unfair high share of the bandwidth compared to player 2 and 3 although all players use the AIMD-scheme. One reason for this situation might be that player 1 updates his bandwidth more often than the other players, which might be caused by different round trip times (RTT). It seems that the timing behavior of updates is one of the obstacles to achieve fairness and full utilization in the Internet.

We will investigate the following questions:

- Is it possible to converge against full utilization for such restricted feedback?
- Is it possible to enable fairness under adversarial timing?
- Can we achieve fairness and full utilization?

1.4 Cost-distance Trees

Assume that the locations of the terminal sites of a network are given and the question is, how the interconnecting network has to be chosen. Then, the following parameters are important:

- The **size/cost** of the network.
- The **distance** between two terminal sites, if one uses only the network edges.
- The **topology**. Is the network planar, or even a tree.

In Figure 1.4 and 1.5 two different (subway) network structures can be seen. In Figure 1.4 we see a dense network, allowing short routes between all stations. In the other subway network in Figure 1.5 the network is sparser; long detours occur between some stations. However, the tree-like network topology of the second network may simplify many coordination problems.

In general, one expects that trees will increase the size of the network or the length of necessary detours. Clearly, long detours cannot be prevented at all. Therefore, the weighted distance will be considered which describes the average length of a route.

We consider a mixed measure that sums over two types of costs: the fixed costs proportional to the size of the network and the dynamic costs proportional to the average length of a route in the network. This measure is called the *weighted cost-distance* (WCD). Given a non-negative weighting $w : V \times V \to \mathbb{R}_0^+$ and a node set $V \subseteq \mathbb{R}^k$ we define it by the following equation:

$$\mathrm{WCD}_w(N) := \sum_{e \in E(N)} ||e||_2 + \sum_{u,v \in V} w(u,v) L_N(u,v),$$

where $L_N(u,v)$ denotes the length of the shortest path from u to v in N.

In this context spanner-graphs have been investigated. These are graphs G where for all nodes u, v it holds

$$L_N(u,v) \;\le\; c\, ||u,v||_2 \;,$$

where the constant c is called the stretch factor. Such spanner-graphs can be constructed efficiently. Furthermore, additional constraints such as small size can be fulfilled. Such a spanner is called a *light spanner* if for the size of the graph it holds for a constant k:

$$\sum_{e \in E(N)} ||e||_2 \le k \sum_{e \in E(\mathrm{MST}(V))} ||e||_2 \;,$$

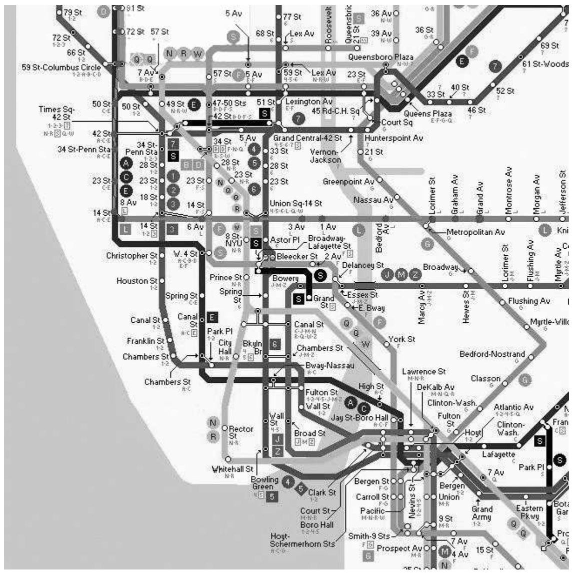
Figure 1.4: The subway network of New York downtown

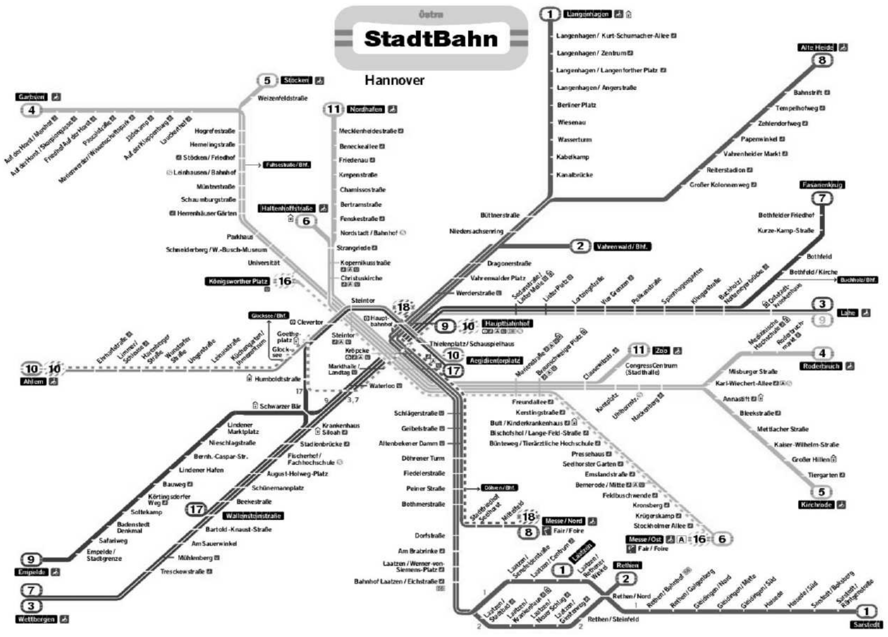

Figure 1.5: The subway network of Hanover

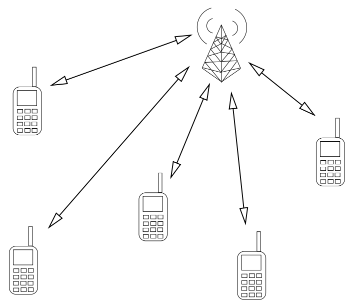

Figure 1.6: Traditional mobile phone networks. No interconnections between mobile phones

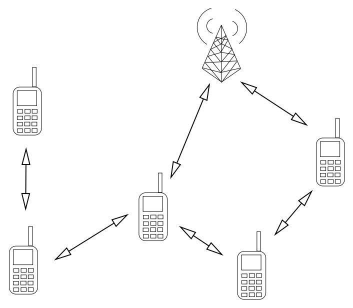

Figure 1.7: A mobile ad hoc network. Mobile phones can connect to closely located partners

where MST(V) denotes the minimum spanning tree of V.

Clearly, such light spanners approximate the optimal weighted cost-distance graph by a factor of $\max c, k$. However the question remains:

- How well can trees optimize the weighted cost-distance?

1.5 Mobile ad hoc Communication

The standard approach for wireless communication networks is to install a central radio station that relays communication between partners and connect the mobile communication partners to the other wired communication partners, see Figure 1.6.

Obviously, this network topology has a number of disadvantages. First of all, the relay station is a natural bottleneck. All information must pass this station and if the number of mobile phones increases the capacity will be exhausted at some point. Another problem is that the position and the transmission power of the central antenna determines areas without reception.

A different approach is to use *mobile ad hoc networks* which use neighbored radio stations as relay stations, see Figure 1.7. Communication between mobile phones can be done without a designated radio station using multiple hops between neighbored mobile devices. This decreases the transmission power needed by each mobile device. Furthermore an increase in the number of participants will also increase the available bandwidth.

However it is not clear what kind of basic network is the best choice to optimize parameters like dilation, energy and data throughput.

- Which mobile ad hoc networks optimize dilation, energy and data throughput?
- Can all of these measures be optimized at the same time?

Chapter 2

Main Results

In the previous chapter of this habilitation thesis we presented some introductory examples of communication network problems. This chapter is dedicated to a comprehensive summary of the main results and models presented in chapters 4–10. In the next chapter 3 we clarify some mathematical and computer scientific notations.

2.1 Overview

In this habilitation thesis we discuss communication network problems of the following areas.

1. **Broadcast information**

   Broadcasting in processor networks means disseminating a single piece of information, which is originally known only at some nodes, to all members of the network. This is done in a sequence of rounds by pairwise message exchange over the communication lines of the network.

   In this thesis we consider two different models:

   - In chapter 4 and 5 we consider broadcasting in the *telephone model* for a given undirected graph describing the bidirectional connection between processors. Here, in one round each processor can send a message to at most one of its neighbors. The goal is to inform all processors using as few rounds as possible. This number is called the *minimum broadcasting time* of the network.

     Given the network structure and the informed node we are interested in the optimal strategy, when which neighbored node has to be informed to minimize the number of rounds. We characterize the algorithmic complexity and want to find graph families where the optimal solution can be found.
   - For *randomized rumor spreading* presented in chapter 6 the connections between nodes change in every round and are given by a random addressing function. In one round a processor has to decide whether to send its information. Thus, in addition to minimizing the number of rounds for broadcasting information we have to take care of the quantity of messages. This must be done by a distributed algorithm because of the randomized communication structure.

2. **Distributed allocation of network bandwidth**

   The transport control protocol (TCP) of the Internet uses a distributed algorithm to determine the packet rate of connections between computers A and B. Here A adjusts the packet rate by an algorithm that uses only information whether the last transmission was successful, i.e. packets are not dropped.

   We will investigate two types of questions. In chapter 7 we generalize this model by a game-theoretic approach dealing with a player, representing a host, and an adversary, representing all other hosts on the network. In our model the adversary chooses the available bandwidth to maximize the cost of the player. The player can choose a packet rate, i.e. allocate bandwidth, and suffers two types of costs:

- Allocation of more bandwidth than available: Then the player's cost account for *re-transmission delay and overhead*.
- Allocation of less bandwidth than available: Then the player suffers *opportunity costs*.

The algorithm will never learn the available bandwidth in a round and thus cannot compute the total cost it is actually suffering. However, we present a probabilistic algorithm which uses the little feedback information available, i.e. whether the allocated bandwidth was smaller than the available bandwidth.

It turns out that such an algorithm solves a general class of *online prediction games* of all discrete forecasting problems that provide sufficient feedback information. This algorithm can bound the relative loss, called *regret* compared to the best constant choice of packet rate. We can show that the average regret of a round tends to 0 when the number of rounds increases.

The allocation behavior of such a bandwidth allocation algorithm tries to reach *full utilization*, i.e. the packet rate equals the available bandwidth, while it does not guarantee that every participating processor receives a *fair share* of the available bandwidth. On the other hand, TCP tries to distribute fair shares while it does not converge against full utilization even in a best case situation. In chapter 8 we concentrate on the relationship of full utilization and fairness. It has been observed that the allocation behavior of TCP depends highly on the time points when a host reevaluates and adjusts its packet rate. To describe robust allocation algorithms we consider an *adversarial timing* for the updates of the participating players. It turns out that in this model fairness and full utilization cannot be both satisfied and we present fair distributed allocation schemes if the algorithms receive the residual bandwidth as feedback information.

3. **Designing efficient communication networks**

Given a node set in Euclidean space we are interested in finding the network that optimizes the costs of the network. In this thesis we consider the following two optimization constraints.

- The *weighted cost-distance model* is a measure that applies also to other networks, e.g. street, or railway networks. In chapter 9 we describe the overall cost of a network by a static component, called cost, and a dynamic component, called weighted distance. The cost simply depends on the size of the network, i.e. the sum of the edge lengths. While the weighted distance combines a non-negative weighting between network nodes describing the demand or occurring traffic, and the shortest distance between nodes using the network.

  We will see that the optimization problem can be approximated by so-called spanners. Then, we concentrate how the weighted cost-distance increases if we restrict the network topology to trees. It turns out that this increase can be upper-bounded and lower-bounded by a logarithmic factor.

- In chapter 10 we concentrate on the question how *mobile ad-hoc networks* can be optimized. Such networks consist of radio stations which establish a node-to-node communication network. For this, every node, i.e. radio station, can adjust the transmission power such that the transmission radius is given by the distance to the addressed node. Because only one frequency is available, radio signals can interfere and thus additional constraints appear for the routing problem.

  We define the path system of a mobile ad hoc network as the union of all paths used for routing information. Because of the radio interferences the choices of these paths cannot be done independently. Therefore we want to find restricted basic networks, whose edges give a natural upper bound on the number of interference of path systems that use only edges of this basic network.

  We present reasonable definitions for *energy, dilation* and *congestion* of path systems and give algorithms that output optimal or nearly-optimal networks for these measures. We show that two of these measures cannot be optimized by the same path system.

2.2 Broadcasting in Planar and Decomposable Graphs (Chapter 4)

Broadcasting in processor networks means disseminating a single piece of information, which is originally known only at some nodes, to all members of the network. This is done in a sequence of rounds by pairwise message exchange over the communication lines of the network. In one round each processor can send a message to at most one of its neighbors. The goal is to inform everybody using as few rounds as possible. This number is called the *minimum broadcasting time* of the network. This short description constitutes the *telephone model* for broadcasting in undirected graphs.

Given a graph and a subset of nodes, the sources, the problem is to determine its specific broadcast time, or more generally to find a broadcast schedule of minimal length. This is known as a an $\mathcal{NP}$-hard problem. We are interested to find out more about the computational complexity of this problem. In particular, we ask which graph topology allows the efficient computation of a broadcast schedule and for which graph families the problems remains $\mathcal{NP}$-hard.

For the lower bounds we consider the decision problem of broadcasting: The MULTIPLE SOURCE BROADCASTING DECISION PROBLEM (**MB**) is: Given a set of sources, an undirected graph and a deadline, determine whether there exists a broadcasting strategy informing all nodes within the deadline. Furthermore we consider the SINGLE SOURCE BROADCASTING DECISION PROBLEM (**SB**), which is MB reduced to the case of a single source. We show the following results:

- MB restricted to planar graphs with degree 4 and deadline 4 is $\mathcal{NP}$-complete.
- SB restricted to graphs with degree 3 and is $\mathcal{NP}$-complete even when the deadline is logarithmic.
- SB restricted to planar graphs of degree 3 is $\mathcal{NP}$-complete.

On the other hand, we investigate for which classes of graphs this problem can be solved efficiently and show that broadcasting and even a more general version of this problem becomes easy for graphs with good decomposition properties. The solution strategy can efficiently be parallelized, too.

For this purpose we have to extend the notion of graph decomposition to measure its properties more exactly. A careful inspection of the possibilities how information can flow within a component and between different components of a graph will be required. For the internal flow components that are connected behave most favorably, but in general connectivity cannot always be achieved by a tree decomposition into small components.

In particular, we consider the following two types of tree decompositions:

- **Edge decomposition:**

  For a graph $G = (V, E)$ an **edge decomposition graph** $H = (V_H, E_H)$ provides the following properties:

  - The nodes G_i of V_H represent induced sub-graphs $G_i = (V_i, E_i)$ of G such that the V_i are pairwise disjoint and $V = \bigcup_{G_i \in V_H} V_i$.
  - $\{G_i, G_j\} \in E_H$ iff there is an edge between a node of G_i and a node of G_j.

  A graph G is **(κ, μ, c)–edge decomposable** if there exists an edge decomposition graph $H = (V_H, E_H)$ such that for all $G_i \in V_H$: cut$(G_i) \leq \kappa$, $|V_i| \leq \mu$ and $\mathtt{cc}(G_i) \leq c$ where $\mathtt{cc}(G_i)$ denotes the number of connected components of G_i. The cut of a node G_i is the union of all edges of G that connect G_i to other components.

  Such an edge decomposition graph H is called an **(κ, μ, c)–edge decomposition tree** of G if H is a tree.

- **Node decomposition:**

  For a graph $G = (V, E)$ a graph $H = (V_H, E_H)$ is a **node decomposition graph** if

  - The nodes G_i of V_H represent sub-graphs $G_i = (V_i, E_i)$ of G such that $V = \bigcup_{G_i \in V_H} V_i$ and $E = \bigcup_{G_i \in V_H} E_i$;

– For each node v holds: if $v \in V_i \cap V_j$ then H contains a path π from V_i to V_j such that v belongs to every node V_l in π.

A graph $G = (V, E)$ is called **(κ, μ, c)–node decomposable** if there exists a node decomposition graph $H = (V_H, E_H)$ such that for all $G_i \in V_H$ holds cut$(G_i) \leq \kappa$,$|V_i| \leq \mu$, and $\mathtt{cc}(G_i) \leq c$. Here, we define the **cut** of a node G_i as the union of all cuts cut(G_i, G_j) to neighbored sub-graph G_j, where cut$(G_i, G_j) := V_i \cap V_j$.

Such a decomposition H is called a **(κ, μ, c)–node decomposition tree** of G if H is a tree.

We present algorithms that construct the optimal broadcasting schedule in polynomial graphs, if one of the following decompositions is known for the given graph:

- The graph has a $\left(\mathrm{O}\left(\frac{\log n}{\log\log n}\right), \mathrm{O}\left(\frac{\log n}{\log\log n}\right), \mathrm{O}(1)\right)$–edge decomposition tree.
- The graph has bounded degree d and a $\left(\mathrm{O}\left(\frac{\log n}{\log\log n}\right), \mathrm{O}(\log n), \mathrm{O}(1)\right)$–edge decomposition tree.
- The graph has a $\left(\mathrm{O}(1), \mathrm{O}\left(\frac{\log n}{\log\log n}\right), O(1)\right)$–node decomposition tree.
- The graph has maximal degree $d = \mathrm{O}\left(\frac{\log\log n}{\log\log\log n}\right)$ and a $\left(\mathrm{O}\left(\frac{\log n}{\log\log n}\right), \mathrm{O}\left(\frac{\log n}{\log\log\log n}\right), \mathrm{O}(1)\right)$–node decomposition tree.
- The graph has constant degree and a $\left(\mathrm{O}\left(\frac{\log n}{\log\log n}\right), \mathrm{O}(\log n), \mathrm{O}(1)\right)$–node decomposition tree.

The algorithm even works for a more general version of the broadcasting problem. Furthermore, it can be parallelized efficiently to yield $\mathcal{NC}$-algorithms.

A preliminary version of this chapter has been published in [JRS94] and a journal version appeared in the Journal of Discrete Applied Mathematics [JRS98].

2.3 On the Inapproximability of Broadcasting (Chapter 5)

Then, we investigate the computational complexity of approximating the broadcasting time of a given graph. We show the following results.

- There is no efficient $\frac{3}{2} - \epsilon$ approximation algorithm for the broadcast time of a network with a single source unless $\mathcal{P} = \mathcal{NP}$. It is $\mathcal{NP}$-hard to distinguish between graphs having broadcast time smaller than $b \in \Theta(n^c)$ and graphs with broadcast time larger than $(\frac{3}{2} - \epsilon)b$ for any $\epsilon > 0$ and some $c > 0$, where n denotes the number of nodes.

- For the additive approximation of the broadcast time we show a tight lower bound of $\sqrt{n}$, i.e., we show that it is $\mathcal{NP}$-hard to distinguish between graphs with broadcast time smaller than b and larger than $b + \Omega(\sqrt{n})$.

- For graphs with degree 3 we show that it is $\mathcal{NP}$-hard to decide whether the broadcast time is $b \in \Theta(\log n)$ or $b + \Theta(\sqrt{b})$ in the case of multiple sources. For graphs with single sources and degree 3, it is $\mathcal{NP}$-hard to distinguish between graphs with broadcast time smaller than $b \in \Theta(\sqrt{n})$ and larger than $b + c\sqrt{\log b}$.

We prove these statements by polynomial time reductions from set-cover and E3-SAT. A preliminary version of this chapter was presented at the 3rd International Workshop on Approximation Algorithms for Combinatorial Optimization Problems [Sch00b].

2.4 Randomized Rumor Spreading (Chapter 6)

In this chapter we discuss the problem of broadcasting information in a processor network in a different communication model. We investigate the class of so-called epidemic algorithms that are commonly used for the lazy transmission of updates to distributed copies of a database. These algorithms use a simple randomized communication mechanism to ensure robustness. Suppose n players communicate in parallel rounds in each of which every player calls a randomly selected communication partner. In every round, players can generate rumors (updates) that are to be distributed among all players. Whenever communication is established between two players, each one must decide which of the rumors to transmit.

The communication graph $G_t = (V, E_t \subseteq V \times V)$ of round $t \geq 1$ is obtained by a distributed, randomized process. In each round, each player u chooses a communication partner v from V at random and u *calls* v. In round t, the rumor and other information can be exchanged in both directions along the edges of G_t. Whenever a connection is established between two players, each one of them (if holding the rumor) has to decide whether to transmit the rumor to the other player, typically without knowing whether this player has received the rumor already. Regarding the flow of information, we distinguish between push and pull transmissions. Assume player u calls player v.

- The rumor is *pushed* if u tells v the rumor.
- The rumor is *pulled* if v tells u the rumor.

The major problem (arising due to the randomization) is that players might not know which rumors their partners have already received. For example, a standard algorithm based on push-communication, i.e. forwarding each rumor from the calling to the called players, for $\Theta(\log n)$ rounds needs to transmit the rumor $\Theta(n \log n)$ times in order to ensure that every player finally receives the rumor with high probability.

We investigate whether such a large communication overhead is inherent to epidemic algorithms. On the positive side, we show that the communication overhead can be reduced significantly if we use push and pull–communications:

- We start with a simple **push&pull algorithm** that terminates transmission when the rumor is $\log_3 n + O(\log\log n)$ rounds old. It turns out that this algorithm needs only $O(n \log\log n)$ transmissions and broadcasts the rumor with high probability, i.e. with probability of at least $1 - \frac{1}{n^c}$ for any fixed $c > 1$.
- Shenker proposed a distributed termination mechanism using a counter indicating indirectly the spread of the rumor. We show that this **min-counter algorithm** performs as well as the push&-pull algorithm.
- In order to improve the robustness, we devise a distributed termination scheme, called the **median-counter algorithm**, that is provably robust against adversarial node failures as well as stochastic inaccuracies in establishing the random connections.

  In particular, we show that the efficiency of the algorithm does not rely on the fact that players choose their communication partners uniformly from the set of all players. We show that the median-counter algorithm takes $O(\log n)$ rounds and needs only $O(n\ \log\log n)$ transmissions regardless of the probability distribution used for establishing the random connections as long as all players act independently and each player uses the same distribution $\mathcal{D} : V \to [0, 1]$ to select its communication partner.
- On the negative side, we show that any *address-oblivious* algorithm (i.e., an algorithm that does not use the addresses of communication partners) needs to send $\Omega(n \log\log n)$ messages for each rumor regardless of the number of rounds.
- Furthermore, we give a general lower bound showing that time- and communication-optimality cannot be achieved simultaneously using random phone calls, that is, every algorithm that distributes a rumor in $O(\log n)$ rounds needs $\omega(n)$ transmissions.

These results were presented at the Symposium on Foundations of Computer Science [KSSV00].

2.5 Online Prediction with Partial Feedback (Chapter 7)

Bandwidth allocation in the Internet is managed by the *Transport Control Protocol (TCP)*. When host A transmits information to B it is A who regulates the *packet rate*, called *allocated bandwidth*. However A never learns the actual available bandwidth but receives a feedback from which A learns whether it over- or underestimated this value. Papadimitriou, Koutsoupias and Karp [KKPS00] investigate protocols optimizing the cost needed to find the optimal packet rate when the available bandwidth is constant. We extend this approach and investigate the dynamic case where in every round the available bandwidth is chosen by an adversary.

The cost $\ell(y,g)$ of allocating g and available bandwidth is given and reflects two major components: *opportunity costs* due to sending less than the available bandwidth when $g < y$, and *re-transmission delay and overhead* due to dropped packets when $g > y$. The goal of the host A is to minimize the total cost incurred over all periods. In [KKPS00] the following cost models are introduced: the *gentle cost function*, $C_\alpha(y,g) = y - g$ when $g \leq y$ and $C_\alpha(y,g) = \alpha(g-y)$ when $g > y$; and the the *severe cost function*, $S(y,g) = y - g$ when $g \leq y$ and $S(y,g) = y$ when $g > y$. We model the feedback by the *threshold feedback function* $f(y,g) = 0$, if the allocated bandwidth was higher than the available bandwidth, i.e. $y < g$ and $f(y,g) = 1$ elsewhere.

Given the loss function ℓ and the feedback function f in this model it is impossible to minimize the absolute loss. Therefore we only compete with the best constant choice of bandwidth allocation. We generalize this bandwidth allocation problem to an online-prediction problem, by allowing any discrete feedback and the loss function:

We investigate the problem of predicting a sequence when the information about the previous elements (feedback) is only partial and possibly dependent on the predicted values. This setting can be seen as a generalization of the classical multi-armed bandit problem and accommodates as a special case a natural bandwidth allocation problem. According to the approach adopted by many authors, we give up any statistical assumption on the sequence to be predicted. We evaluate the performance against the best constant predictor (regret), as it is common in iterated game analysis.

We describe the problem as a game between a player choosing an action g_t and an adversary choosing the action y_t at time t. There are K possible actions available to the player, without loss of generality from the set $[K] = \{1,\dots,K\}$, and R actions in the set $[R]$ from which the adversary can pick from. At every time step the player suffers a loss equal to $\ell(y_t,g_t) \in [0,1]$.

The game is played in a sequence of trials $t = 1,2,\dots,T$. The adversary has full information about the history of the game, whereas the player only gets a feedback according to the function $f(y,g)$. Hence the $R \times K$-matrices L and F, with $L_{ij} = \ell(i,j)$ and $F_{ij} = f(i,j)$ completely describe an instance of the problem. At each round t the following events take place.

1. The adversary selects an integer $y_t \in [R]$.
2. Without knowledge of the adversary's choice, the player chooses an action by picking $g_t \in [K]$ and suffers a loss $x_{g_t}(t) = \ell(y_t,g_t)$.
3. The player observes $f_t = f(y_t,g_t)$.

Note that due to the introduction of the feedback function this is a generalization of the partial information game of [ACBFS95].

Let $W(T) := \sum_{t=1}^T x_{g_t}(t) = \sum_{t=1}^T \ell(y_t,g_t)$ be the total loss of player A choosing $g_1,\dots,g_T$. We measure the performance of the player by the expected *regret* R_A, which is the difference between the total loss of A and the total loss of the best constant choice g^*, that is $\sum_{t=1}^T x_{g_*}(t)$.

$$R_A := \max_{y_1,\dots,y_T} \mathbf{E}\left[\sum_{t=1}^T \ell(y_t,g_t) - \min_j \sum_{t=1}^T \ell(y_t,j)\right]$$

where each y_i is a function of $g_1,\dots,g_{t-1}$. In some works the corresponding min-max problem is investigated, transforming the loss into a reward. The two settings are equivalent, as it is easy to check.

- We present an algorithm $\mathbf{FeedExp3}$ which solves this online prediction problem with a sub-linear expected regret. More specific, if there exists a matrix G such that $F\,G = L$ for feedback matrix F and loss matrix L then the expected regret $\mathbf{E}[R_{\text{FeedExp3}}(T)]$ of algorithm FeedExp3 after T steps is bounded by
$$\mathbf{E}[R_{\text{FeedExp3}}(T)] \;=\; O(T^{3/4}(\ln T)^{1/2}K^{1/2})\;,$$
where T is the number of rounds and K the number of available choices.

- We apply this theorem to the original bandwidth allocation problem and show that the expected regret with respect to
  - the *severe cost function* is bounded by $O(T^{\frac{3}{4}}(\log T)^{\frac{1}{2}}K^2)$;
  - the *gentle cost function* is bounded by $O(\alpha T^{\frac{3}{4}}(\log T)^{\frac{1}{2}}K^2)$;
  - the severe cost function *in the continuous case* is bounded by $O(T^{\frac{7}{8}}(\log T)^{\frac{1}{2}})$;
  - the gentle cost function *in the continuous case* is bounded by $O(\alpha T^{\frac{7}{8}}(\log T)^{\frac{1}{2}})$.

  The continuous case reflects the case where the algorithm can choose any real number in the interval $[0,1]$ and it also applies if the number of discrete choices is larger than the number of rounds.

- We show that for any discrete loss function ℓ and feedback function f only one of two situations can occur: Either there is a prediction strategy that achieves small regret as the FeedExp3 algorithm, or there is a sequence which cannot be predicted by any algorithm without incurring a regret of $O(T)$. For this, we show how to construct a sequence that no algorithm can predict without incurring a linear regret with probability at least $1/2$.

A preliminary version of this chapter was presented at the 14th Conference on Computational Learning Theory and the 5th European Conference on Computational Learning Theory [PS01].

2.6 Bandwidth Allocation under Adversarial Timing (Chapter 8)

Congestion control algorithms like, e.g., TCP have to meet the demands of high utilization and fairness simultaneously. We study the trade-off of these two objectives in a plain model consisting of players, shared resources with bounded bandwidth capacities and rate update events, i.e., points of time at which players can adjust their shares of occupied bandwidth. The times at which players can perform their rate update operations is determined by an adversary. As feedback we allow players to receive the size of the unused bandwidth.

We investigate infinite games, where players can enter and leave at any time but focus our analysis on those periods in which the system is closed, i.e., the set of players that perform update operations is fixed. The major novelty of our model is the adversarial timing of the rate update events.

- We start our investigation with bandwidth allocation on a **single bus**. The adversary specifies a sequence of events $\sigma = \sigma_1\sigma_2\sigma_3\ldots$, where each event σ_t is a tuple (i,x) with $i \in K$ and $x \in \{\text{enter}, \text{leave}, \text{update}\}$. With each player $i \in K$, we associate a positive rate variable r_i whose value is zero if the player is inactive, that is, the initial value of r_i is zero and r_i is reset to zero whenever the adversary calls (i, leave). The adversary calls update operations only for the active player. In particular, if the adversary calls (i, update) then player i can set r_i to any positive value. At any given time, we define the share of bandwidth b_i that player i receives by $b_i = r_i$ if $\sum_{i\in K} r_i \leq B$, and 0 otherwise. Thus, the share of bandwidth of all players is zero when the system is overloaded. (For analogous models see, e.g., [KKPS00].)

  A fair and efficient allocation protocol aims to set the rates in such a way that all players in the system get almost the same share of bandwidth and the unused bandwidth is as small as possible.

  It turns out that a very simple protocol, called **Virtual Player Protocol (VPP)**, suffices to achieve fairness: Suppose player j performs an update operation. Let $\bar r = \max\{B - \sum_{i\in K} r_i, 0\}$ denote the unused bandwidth immediately before the update operation. Then player j sets $r_j := \frac{\alpha}{\alpha+1}(r_j + \bar r)$.

For $\alpha = 1$ we show that in a *closed system period*, where the same k players participate, after some $O(k^2 \log(k/\epsilon)$ phases, all allocated bandwidth are almost equal, i.e. $\frac{b_i}{b_j} \leq 1 + \epsilon$, while the residual bandwidth is at most $O(B/(k+1)$ (Actually we state a more general theorem for all $\alpha > 0$). Note that a phase is a set of contiguous rounds where each of the k players performs at least one update operation.

- We generalize the above adversarial model to **general networks**. The network is modeled by a (hyper)graph $G = (V, E)$. Edges represent buses, routers, or other shared resources of limited bandwidth. The bandwidth capacity of edge e is denoted by $B(e)$. Each player comes with a set of edges constituting a *simple path* (i.e., a path in which every edge appears at most once). For player $i \in K$, let $path(i)$ denote the player's path, and for an edge $e \in E$ let $K(e) \subseteq K$ denote the set of those players whose paths contain e.

  As before, an adversary determines when players enter and leave the system and when they can update their rates. For the time being, we assume that update operations are performed *atomically*, i.e., an update operation is not performed by the adversary until the previous one has become effective on all edges of the respective path. We generalize the VPP as follows. Here, player j sets $r_j := \frac{\alpha}{\alpha+1}(r_j + \min_{e \in path(j)}(\bar{r}(e)))$ where $\alpha \geq 1$ denotes a global parameter.

  For every $\delta > 0$, the network is in a state of *δ-max-min fairness* if it is impossible to increase the rate r of any player by more than a factor of $(1+\delta)$ without exceeding the edge capacities in $path(i)$ or decreasing the rate of players whose rate is at most $(1+\delta)r$.

  The VPP converges against $\frac{1}{\alpha}$-max-min fairness, if for each member of a same finite set of players arbitrary often updates occur. We call such a phase a **closed system period**.

- It is an open question how fast the general VPP converges. Therefore we present a discrete variant of this protocol, where the convergence can be determined depending on the *dilation* D and the *congestion* C. The dilation is the maximum length of a path (of participating players) and the congestion is the maximum number of paths containing the same edge.

  We show that for every $\delta > 0$, there is a discrete variant of the VPP that approaches a δ-max-min fair state in any closed system phase. This state is reached after $O(DC^2 + DC(\log B)/\delta^2)$ phases.

- Furthermore, as a *lower bound* we prove for the **single bus** model that there is no protocol achieving full utilization and fairness in the limit, if an adversary determines the order of rate update events and one cannot distinguish between slow and stalled players.

2.7 Tree Network Design for the Cost-Distance-Model (Chapter 9)

Given n terminal points in the Euclidean space we investigate the problem of constructing a network with small cost and short distances. This research is motivated by a number of practical problems arising in network design for traffic in communication networks as well as real traffic in street or railway networks. If one minimizes only the network size, i.e. the sum of all edge lengths, some distances between terminals must be considerably increased. On the other hand if we minimize the distances between all terminals we face a complete network with large costs. These effects are described by the **weight cost-distance** measure, defined by

$$\text{WCD}_w(N) \quad := \quad \sum_{e \in E(N)} c(e) + \sum_{u,v \in V(N)} w(u,v) L_N(u,v) \ ,$$

where $c(e)$ denotes the length of an edge and $L_N(u,v)$ the length of the shortest path from u to v in the network N.

We investigate the following problems:

- WEIGHTED COST-DISTANCE NETWORK PROBLEM (**CDN**): Given a set of sites V in Euclidean space and a weighting $w : V \times V \mapsto \mathbb{R}^+$, find a network $N = (V, E)$ that optimizes the weighted cost-distance $\text{WCD}_w(N)$.

- WEIGHTED COST-DISTANCE TREE PROBLEM (**CDT**): Given V and $w : V \times V \mapsto \mathbb{R}^+$, find a tree $T = (V, E)$ that optimizes the weighted cost-distance $\text{WCD}_w(T)$.

It turns out that the optimal solution of CDN can contain Steiner points as well as cycles. Furthermore, there are instances where the optimal solution contains crossings. We present the following results.

- For a node set V in $\mathbb{R}^k$ CDN can be approximated by a constant factor within time $O(n \log n)$, where $|V| = n$.
- In $\mathbb{R}^2$ CDN can be approximated in polynomial time by a factor of $4.23\ldots$.
- In $\mathbb{R}^k$ a polynomial time algorithm approximates CDT by a factor of $O(\log n)$. Moreover, this tree approximates the optimal solution of CDN within the same factor $O(\log n)$.
- Trees cannot approximate optimal cost-distance networks better than $O(\log n)$. In particular, for every spanning tree T of the $n \times n$-grid, where $w(u, v) = 1$ if u and v are neighbored nodes and $w(u, v) = 0$ elsewhere, the weighted Cost-Distance is at least $\Omega(n^2 \log n)$, while the optimal Cost-Distance network has cost and weighted distance $O(n^2)$.

A preliminary version of this chapter was presented at the International Symposium on Algorithms and Computation (ISAAC'01) [SW01].

2.8 Energy, Congestion and Dilation in Wireless Networks (Chapter 10)

We investigate the problem of path selection in radio networks for a given set of sites in two-dimensional space. We consider a set $V \subseteq \mathbb{R}^2$ of n radio stations, featuring both transmitters and receivers, called sites or nodes, in 2-dimensional Euclidean space. Let $d = \max_{u,v \in V} |u, v|$ denote the geometric diameter of V. Each node $u \in V$ can adjust its transmission radius to some $r \geq 0$ for sending a packet to a neighbor $v \in V$ in range r. Then, the communication network $N = (V, E)$ has the edge $\{u, v\}$, where $|u, v| = r$. Note that for adjusting the transmission power nodes exchanging packets must interact during the transmission. In our model we simplify this interaction by assuming that the sending and acknowledging part of this interaction may interfere with any other such bi-directional connection if the distance is too small.

In particular, this means: To acknowledge this packet the receiving site adjusts its transmission radius to the same radius r as the sending radius. The transmission needs a unit time step and the area covered by sending and acknowledging a packet along $e = (u, v) \in E$ is $D(e) = D_r(u) \cup D_r(v)$, where $D_r(u)$ denotes a disk with center u and radius r. Of course edges only interfere when the routing protocol tries to send a packet at the same time and if $D(e')$ contains u or v. We expand the notion of interferences to edges: Edge (u, v) interferes with edge e' if u or v is in the area $D(e')$, which defines the set of interfering edges by $\text{Int}(e) := \{e' \in E(N) \mid e' \text{ interferes with } e\}$.

We contribute to modeling wireless communication networks with the following definitions.

- Note that sending a packet along e is successful only if no edge from $\text{Int}(e)$ sends concurrently. These interferences of network N describe the directed **interference graph** $G_{\text{Int}}(N)$. Its node set are all edges of N and its edges describe all interferences, i.e. $(c, e) \in E(G_{\text{Int}}(N))$ iff $c \in \text{Int}(e)$. The interference graph can be interpreted as an additional constraint for routing. An edge of the radio network can only be used for sending a packet in a time unit if all interfering edges remain silent.
- The number of this interfering edges is given by the in-degree of an edge in the interference graph and is called the **interference number** of a communication link. The maximum interference number of a site u is the maximum interference number of all edges with receiving site u. The interference number of the network is the maximum interference number of all edges.

Now consider a routing problem $w : V \times V \to \mathbb{N}$, where $w(u, v)$ packets have to be sent from u to v. The problem arises how to choose a **path system** $\mathcal{P}$ in the graph on V. This is a set of paths R_p from source to destination for the packets p in the graph on V. The union of all edges $E_{\mathcal{P}}$ of this path system gives the links of communication network $N = (V, E_{\mathcal{P}})$.

- We denote by the **dilation** $D_{\mathcal{P}}(V)$ the length of a longest path in $\mathcal{P}$, also known as the hop-distance.
- We distinguish two energy models. In the first model, called **unit energy model**, we assume that maintaining a communication link e is proportional to $O(|e|^2)$, where $|e|$ denotes its Euclidean length. Therefore the unit energy *U-Energy* used by radio network N is given by

$$\text{U-Energy}_{\mathcal{P}}(V) := \sum_{e \in E_{\mathcal{P}}(N)} |e|^2 \ .$$

- The **flow energy model** reflects the energy actually consumed by transmitting all packets. Here, the power consumption of a communication link is weighted by the actual load $\ell(e)$ on an edge e:

$$\text{F-Energy}_{\mathcal{P}}(V) := \sum_{e \in E_{\mathcal{P}}(N)} \ell(e)|e|^2 \ .$$

- We show that energy optimal path selection for radio networks can be computed in polynomial time. Particularly, the minimal spanning tree describes an optimal path system for the unit energy and a sub-graph of the Gabriel Graph contains an optimal path system for the flow energy.
- A main result of this chapter is that a spanner-graph construction as a communication network allows to approximate the congestion optimal communication network by a factor of $O(\log^2 |V|)$ (under the condition $\frac{\max_{u,v} ||u,v||_2}{\min_{u,v} ||u,v||_2} \leq |V|^{O(1)}$).

One major insight is the fact that trade-offs are unavoidable: Minimizing one measure is only possible at the cost of enlarging another one. We show trade-offs lower-bounding congestion $\times$ delay and delay $\times$ energy.

- There exists a node set G_n such that for every path system $\mathcal{P}$ the following trade-off between delay $D_{\mathcal{P}}(G_n)$ and congestion $C_{\mathcal{P}}(G_n)$ can be observed: $C_{\mathcal{P}}(G_n) \cdot D_{\mathcal{P}}(G_n) \geq \Omega(W)$.
- There exists a node set L_n with diameter d such that for every path system $\mathcal{P}$ the following trade-offs between delay D and unit energy *U-Energy* (resp. flow energy *F-Energy*) occurs:

$$\begin{aligned} D_{\mathcal{P}}(L_n) \cdot \textit{U-Energy}_{\mathcal{P}}(L_n) &\geq \Omega(d^2) \ , \\ D_{\mathcal{P}}(L_n) \cdot \textit{F-Energy}_{\mathcal{P}}(L_n) &\geq \Omega(d^2 W) \ . \end{aligned}$$

- For congestion and energy the situation is even worse. It is only possible to find a reasonable approximation for either congestion or energy minimization, while the other parameter is at least a polynomial factor worse than in the optimal network: There exists a node set V with minimal congestion C^*, minimal unit energy by U-Energy*, and minimal flow energy by F-Energy* such that for any path system $\mathcal{P}$ on this node set V it holds:

$$\begin{array}{lcl} C_{\mathcal{P}}(V) \geq \Omega(n^{1/3} C^*) & \text{or} & \text{U-Energy}_{\mathcal{P}}(V) \geq \Omega(n^{1/3} \text{U-Energy}^*) \ , \\ C_{\mathcal{P}}(V) \geq \Omega(n^{1/3} C^*) & \text{or} & \text{F-Energy}_{\mathcal{P}}(V) \geq \Omega(n^{1/3} \text{F-Energy}^*) \ . \end{array}$$

A preliminary version of this chapter has been published as a technical report [MSVG01].

Chapter 3

Notations

In this chapter, we introduce some basic notations and mathematical concepts used throughout this thesis. This is not an introduction, but a presentation of basic definitions and fundamental results relevant to the results and discussion presented in the other chapters of this thesis.

3.1 Set Theory

The empty set is denoted by $\emptyset$. We denote by $\mathbb{N} := \{1, 2, 3, \ldots, \}$ the set of integers, and $\mathbb{N}_0 := \mathbb{N} \cup \{0, \}$. For any $c \in \mathbb{N}$ the set $[c]$ represents the set $\{1, \ldots, c\}$. $\mathbb{R}$ denotes the set of reals, $\mathbb{R}^+$ is the set of all positive numbers, i.e. $\mathbb{R}^+ := \{x \in \mathbb{R} \mid x > 0\}$. $\mathbb{R}_0^+ := \mathbb{R}^+ \cup \{0\}$. For two real numbers $x \leq y$ we denote the intervals by

$$\begin{array}{rcl@{\qquad}rcl} [x, y] & := & \{z \in \mathbb{R} \mid x \leq z \leq y\} & [x, y) & := & \{z \in \mathbb{R} \mid x \leq z < y\} \\ (x, y) & := & \{z \in \mathbb{R} \mid x < z < y\} & (x, y] & := & \{z \in \mathbb{R} \mid x < z \leq y\} \end{array}$$

Given sets A, B in a domain D:

- $A \cup B := \{x \mid x \in A \text{ or } x \in B\}$ denotes the union of A and B,
- $A \cap B := \{x \mid x \in A \text{ and } x \in B\}$ denotes the intersection of A and B,
- $A \setminus B := \{x \mid x \in A \text{ and } x \notin B\}$ is called the difference of A and B,
- $\overline{A} := \{x \in D \mid x \notin A\} = D \setminus A$ is called the complement of A.
- If $A \subseteq B$, then A is a subset of B, i.e. for all $x \in A$ we have $x \in B$.
- By $A \subset B$ we denote that A is a proper subset of B, i.e. $A \subseteq B$ and $A \neq B$.
- We denote the power set of A by $\mathsf{P}(A) := \{C \mid C \subseteq A\}$.
- Two sets A, B are disjoint if $A \cap B = \emptyset$, where $\emptyset = \{\}$ denotes the empty set.

3.2 Combinatorics

We use the natural logarithm $\ln x = \log_e x$ as logarithm to the base of $e \approx 2.718281828\ldots$. By $\log x = \log_2 n$ we denote the logarithm to the base 2. $\log^k n$ means $(\log n)^k$; $\log^{[k]} n$ denotes the k-times iterated logarithm.

The factorial of n is written as $n! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot n$. By definition $0! = 1$. The binomial $\binom{n}{k}$ is defined for $0 \leq k \leq n$ as

$$\binom{n}{k} = \frac{n!}{(n-k)!k!} .$$

For functions $f, g : \mathbb{R}^+ \to \mathbb{R}^+$ and a class of functions $\mathcal{G}$ of such functions we define:

$$f \ \le_{ae} \ g \qquad :\Longleftrightarrow \qquad \exists n_0 \in \mathbb{R} \quad \forall n \ge n_0 : \quad f(n) \le g(n) \ .$$

$$\begin{array}{rcl} O(g) & := & \{f \mid \exists k \in \mathbb{R} \quad f \le_{ae} k \cdot g\} \ , \\ o(g) & := & \{f \mid \forall k \in \mathbb{R}^+ \quad k \cdot f \le_{ae} g\} \ , \\ \omega(g) & := & \{f \mid \forall k \in \mathbb{R}^+ \quad k \cdot g \le_{ae} f\} \ , \\ \Omega(g) & := & \{f \mid \exists k \in \mathbb{R} \quad g \le_{ae} k \cdot f\} \ , \\ \Theta(g) & := & O(g) \quad \cap \quad \Omega(g) \ . \end{array}$$

$$\begin{array}{rclcrcl} O(\mathcal{G}) & := & \bigcup_{g \in \mathcal{G}} O(g) & , & o(\mathcal{G}) & := & \bigcup_{g \in \mathcal{G}} o(g) \ , \\ \Omega(\mathcal{G}) & := & \bigcup_{g \in \mathcal{G}} \Omega(g) & , & \omega(\mathcal{G}) & := & \bigcup_{g \in \mathcal{G}} \omega(g) \ , \\ \Theta(\mathcal{G}) & := & \bigcup_{g \in \mathcal{G}} \Theta(g) & . & & & \end{array}$$

For a class $\mathcal{G}$ of functions we denote by $f \le \mathcal{G}$, that there is a function $g \in \mathcal{G}$ such that $f \le g$.

Table 3.1: The O-notations.

For every $x \in \mathbb{R}$ we have

$$e^x = \sum_{i=0}^{\infty} \frac{x^i}{i!} \qquad \text{and} \qquad \ln(1-x) = \sum_{i=1}^{\infty} \frac{(-1)^{i-1} x^i}{i} \ .$$

We will use the fact that $\lim_{n\to\infty}(1 - 1/n)^n = 1/e$ and particularly we have for all $n \in \mathbb{N}$:

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

We will use Stirling's formula:

$$n! \ = \ \alpha(n)\sqrt{2\pi n}\left(\frac{n}{e}\right)^n \qquad \text{where } \alpha(n) \in [1, e^{1/12n}] \ .$$

Asymptotics

To compare the asymptotic behavior of functions we use the so-called O-notations, see table 3.1.

We denote by pol the set of all polynomial functions, i.e. pol $:= \bigcup_c O(n^c)$. Similarly, we denote by polylog $:= \bigcup_c O(\log^c n)$.

3.3 Graph Theory

A graph G consists of a set of nodes (also known as vertices) V and an edge set E. The order of G is defined as the number of nodes $|V|$, while the size is given by $|E|$. We denote the node set of G by $V(G)$, and analogously the edges set of G by $E(G)$. We distinguish directed and undirected graphs. The edge set of a directed graphs consists of pairs (v, w) for vertices $v \neq w$, $v, w \in V$. In undirected graphs the edges set consists of subsets of V of cardinality 2, i.e. $E \subseteq \{\{v, w\} \mid v, w \in V\}$. The degree of an undirected graph is the maximum number of edges incident to a node.

A graph G' is a *sub-graph* of a graph G if $V(G') \subseteq V(G)$ and $E(G') \subseteq E(G)$. A path P from node u_1 to node u_m of a directed graph is a sub-graph with nodes $(u_1, \dots, u_m)$ such that $E(P) = \{(u_i, u_{i+1}) \mid i \in [m-1]\}$ and $E(P) = \{(\{u_i, u_{i+1}\} \mid i \in [m-1]\}$ in the case of an undirected graph. If $V' \subseteq V(G)$ then the *induced sub-graph* of V' of G is the sub-graph of G with node set V' and maximum number of edges.

A cycle is a path with at least one edge and the starting and ending node. A directed acyclic graph (DAG) is a directed graph, where no cycle can be found.

A *connected component* of an undirected graph is a sub-graph G' where for all nodes u, v in G' there is a path from u to v. All graphs can be separated into *connected components* of maximum size. The number of these sub-graphs is called the *number of connected components* of an undirected graph.

3.4 Probability Theory

We denote the probability of an event A by $\mathrm{P}[A]$ and the condition probability of an event by

$$\mathrm{P}[B \mid A] \ := \ \frac{\mathrm{P}[A \cap B]}{\mathrm{P}[A]} \ .$$

Two events A and B are independent if and only if

$$\mathrm{P}[B \mid A] = \mathrm{P}[B] \ .$$

If this is not the case, then A and B are *correlated*. A and B are called

- *negatively correlated*, if $\mathrm{P}[B \mid A] < \mathrm{P}[B]$ and
- *positively correlated*, if $\mathrm{P}[B \mid A] > \mathrm{P}[B]$.

and the expectation of a random variable X by

$$\mathbf{E}[X] \ := \ \sum_{x \in \mathbb{R}} x \cdot \mathrm{P}[X = x] \ .$$

Basic properties of the expectation are

- $\mathbf{E}[c \cdot X] = c \cdot \mathbf{E}[X]$.
- $\mathbf{E}[X + Y] = \mathbf{E}[X] + \mathbf{E}[Y]$.
- If two random variables X and Y are *independent*, i.e. for $x, y \in \mathbb{R}$: $\mathrm{P}[X = x \mid Y = y] = \mathrm{P}[X = x]$, then $\mathbf{E}[X \cdot Y] = \mathbf{E}[X] \cdot \mathbf{E}[Y]$.

The *conditional expectation* of a random variable Y with respect to an event A is defined by

$$\mathbf{E}[Y \mid A] \ := \ \sum_{y \in \mathbb{R}} y \cdot \mathrm{P}[Y = y \mid A] \ .$$

A fundamental property of the conditional expectation is that for any random variables X, Y:

$$\mathbf{E}[Y] = \mathbf{E}[\mathbf{E}[Y \mid X]] \ .$$

Variance The em variance of a random variable X, denoted by $V[X]$, is defined by

$$V[X] \ := \ \mathbf{E}[(X - E[X])^2] \ .$$

The number $\sigma = \sqrt{V[X]}$ is called the *standard deviation* of X.

Markov Inequality Let X be a non-negative random variable. Then for any $k > 0$,

$$\mathrm{P}[X \geq k] \ \leq \ \frac{\mathbf{E}[X]}{k} \ .$$

Chebyshev Inequality Let X be an arbitrary random variable. Then, for every $k > 0$,

$$\mathrm{P}[|X - \mathbf{E}[X]| \geq k] \;\leq\; \frac{V[X]}{k^2} \;.$$

Basic probability distributions A random variable X is *uniformly distributed* over a finite set $M \subset \mathbb{R}$ if for all $x \in M$:

$$\mathrm{P}[X = x] = \frac{1}{|M|} \;.$$

A random variable X is a *binomial distribution* $B_{p,n}$ if for a $p \in [0,1]$ and $n \in \mathbb{N}$ we have

$$\mathrm{P}[X = k] = \binom{n}{k} p^k (1-p)^{n-k} \;.$$

For its expectation and variance we have

$$\mathbf{E}[X] = n \cdot p \qquad V[X] = n \cdot p(1-p) \;.$$

A random variable X is *geometrically distributed* if there is a parameter $p \in [0,1]$ such that for all $k \in \mathbb{N}$

$$\mathrm{P}[X = k] = (1-p)^{k-1} p \;.$$

Chernoff Bound If $X_1, \ldots, X_n$ are independent binary random variables, then it holds for all $\delta \geq 0$ for $S_n = \sum_{i \in [n]} X_i$ and $\mu = \mathbf{E}[S_n]$ that

$$\mathrm{P}[S_n \geq (1+\delta)\mu] \;\leq\; e^{-\frac{\delta^2 \mu}{2(1+\delta/3)}} \;\leq\; e^{-\min\{\delta,\delta^2\}\cdot\mu/3}$$

and for $0 \leq \delta \leq 1$ that

$$\mathrm{P}[S_n \leq (1-\delta)\mu] \;\leq\; e^{-\frac{\delta^2 \mu}{2}} \;.$$

Martingales A sequence of random variables $X_0, X_1, \ldots$ is a *martingale* if for all $i \geq 1$

$$\mathbf{E}[X_i | X_0, X_1, \ldots, X_{i-1}] = X_{i-1} \;.$$

Azuma Inequality Let $X_0, X_1, \ldots$ be a martingale satisfying the property that $|X_i - X_{i-1}| \leq c_i$ for all $i \geq 1$. Then for any $t \geq 0$,

$$\mathrm{P}[X_n \geq X_0 + t] \;\leq\; e^{\frac{t^2}{2\sum_{i=1}^n}(c_i)^2}$$

and

$$\mathrm{P}[X_n \leq X_0 - t] \;\leq\; e^{\frac{t^2}{2\sum_{i=1}^n}(c_i)^2} \;.$$

3.5 Geometry

We use in the k-dimensional Euclidean space $\mathbb{R}^k$ the standard norm

$$||u,v||_2 = |u - v| = \left(\sum_{i=1}^{k} (u_i - v_i)^2\right)^{\frac{1}{2}}$$

and the absolute norm (Hamming distance)

$$||u,v||_1 = \sum_{i=1}^{k} |u_i - v_i| \;.$$

The β-skeleton [KR85, Vel92] of a set of points is a graph, defined to contain exactly those edges (a,b) such that no point c forms an angle $\measuredangle(c-a, c-b)$ greater than $\sin^{-1} 1/\beta$ (if $\beta \geq 1$) or $\pi - \sin^{-1} 1/\beta$ (if $\beta < 1$). Equivalently, if $\beta > 1$, the β-skeleton can be defined in terms of the union U of two circles, each having (a, b) as a chord and having diameter $\beta||a, b||_2$. Edge (a, b) is included in this graph exactly when U contains no points other than a and b.

If $\beta = 1$, an edge (a, b) is included in the β-skeleton exactly when the circle having ab as diameter contains no points other than a and b. The 1-skeleton is also known as the Gabriel graph [GS69].

For a node set V in $\mathbb{R}^k$ and a path $P = (u_1, \ldots, u_n)$ the stretch-factor of P is for $u_1 \neq u_n$ defined by $L_P(u_1, u_n)/||u_1, u_n||_2$, where $L_P(u_1, u_n) := \sum_{i=1}^{n-1} ||u_i, u_{i+1}||_2$. In a graph P is called the minimum path if for given u, v it minimizes $L_P(u, v)$. A graph has a stretch factor t if for all nodes u, v there is a connecting path with stretch factor of at most t. We call such graphs t-spanners.

The minimum spanning tree (MST) of a given graph G is a spanning graph of G where the cost, i.e. the sum of all edge lengths, is minimized. Such graphs can be very efficiently constructed by the algorithms of Prim and Kruskal.

3.6 Computational Complexity

We follow the notations of [Pap94] and [Rei90] for a short description of some important models of computational complexity.

3.6.1 Machine Models

We give a rough overview over machine models relevant to this dissertation. In general one distinguishes between **sequential machine models** and **parallel machine models**. The machine model describing modern computing devices most accurately is the random access machine. Whenever we describe an algorithm we measure time and space behavior by this sequential machine model. Apart from this machine model we will also use the (non-deterministic) Turing machine to describe an algorithm. We now give a short overview of these models.

The deterministic Turing machine (DTM)

This sequential machine model was introduced by Alan Turing for the investigation of the principle compatibility of problems. Every computing step of the Turing machine involves only some symbols of an alphabet, which are positioned on one or several constant number of tapes. A tape is a non-ending string of a discrete symbol set, called alphabet. The Turing machine may access only the symbol at a special position, called head position, and after performing a computational step the Turing machine may move its head one step to the left or the right.

Every state of the DTM determines the computation. The Turing machine is completely described by $(Q, \Sigma, \Delta, q_0, q_f)$. The DTM has a finite number of states Q. Special states are the starting state q_0 and the final state q_f. After changing the symbols at the head positions and moving the heads, the Turing machine may switch to a new state (depending on the symbols read on the tapes). More formally, the complete behavior of such a k-tape Turing machine can be described by a so-called transition function $\Delta : Q \times \Sigma^k \to Q \times \Sigma^k \times \{\leftarrow, \circ, \rightarrow\}^k$, where Q describes the set of states, Σ the alphabet, and $\leftarrow, \circ, \rightarrow$ the motion direction of the head on a tape. Inputs are given on the first tape. At the beginning all tapes are empty, i.e. they are filled with the same symbol.

The output is written on the last tape. If the Turing machine computes only a predicate, we call the machine an acceptor.

The non-deterministic Turing machine (NTM)

Here, we replace the transition function of a DTM with a transition relation $\Delta \subseteq Q \times \Sigma^k \times Q \times \Sigma^k \times \{\leftarrow, \circ, \rightarrow\}^k$. For each state-symbol combination, there may be more than one appropriate next step—or none at all.

A non-deterministic Turing machine accepts an input if there exists a valid sequence of transitions such that the Turing machine transits from the input configuration to accepting configuration. This interpretation of the output behavior of the NTM makes it difficult to predict the output using other models: In every round the number of possible configurations may grow exponentially.

The Random Access Machine (RAM)

The data structure of the random access machine (RAM) is an array of registers, each capable of containing an arbitrarily large integer. RAM instructions resemble the instruction set of actual computers, including direct and indirect addressing of the registers, addition, multiplication, and if-then-else. The program is a sequence of elementary commands. Loops can be implemented using `goto`s. Recursive programs are not available, but can be easily implemented in machine model without any additional time.

The RAM gives a more realistic model of a computer having random access to memory. However, if a RAM computes a solution in time T using the S contiguous memory cells having maximal bit length B, then a Turing can perform the same calculation in time $T \cdot S \cdot B$ and space $S \cdot B$. Hence, when we consider polynomial complexity classes for which it does not matter whether the underlying sequential machine model is a deterministic Turing machine or a RAM.

When we speak about the running time of algorithms we refer to the running time of an implementation on a RAM, if not stated otherwise.

The Parallel Random Access Machine (PRAM)

In this thesis we will also present some efficient parallel algorithms. The underlying machine model is a *parallel random access machine* (PRAM). It consists of an (unbounded) series of parallel processors based on the RAM model. Every processor can use local as well as global memory, which both are randomly accessible series of registers. Furthermore, each processor has access to its index number. All processors use the same program and work synchronously after being started at the same time.

Of course the access to the global memory can lead to conflicts if more than one processor tries to change the entry of a global register. Therefore the following models are considered:

1. *exclusive read* (**ER**): Only algorithms are valid where processors do not read same global register in the same round.

2. *concurrent read* (**CR**): Arbitrary many processors can simultaneously read a memory cell.

3. *owner write* (**OW**): For each memory cell there exists a dedicated owner, who alone may write into this cell. Other processors are allowed to read this cell.

4. *exclusive write* (**EW**): Only PRAM algorithms are valid where write-memory conflicts do not appear.

5. *concurrent write* (**CW**): No restrictions apply for the write-access to memory.

   The following models describe the conflict solution, if collisions occur, i.e. two processors try to write to the same memory cell.

   (a) *common*: Only simultaneous writes are valid, if all conflicting processors try to write the same value.
   (b) *collision*: If more than one processor writes to a register, a special conflict symbol will be written.
   (c) *priority*: According to some ranking, the processor with the highest priority writes its value into the register.

Whenever we refer to PRAMs in this thesis we talk about the CRCW-PRAM-model where write conflicts are solved in the common model (5a).

Boolean Circuits

In computational complexity Boolean circuits are also a method of describing parallel algorithms. Depth and size are the most often investigated complexity measures of Boolean circuits. Let $\mathcal{B}_n$ denote the set of Boolean functions $f : \{0,1\}^n \to \{0,1\}$.

A circuit is a directed acyclic graph (DAG) where the sources constitute the inputs $x_1, \ldots, x_n$ and the sinks represent the outputs $y_1, \ldots, y_n$.

Circuits will be defined over the standard basis of AND, OR and NOT gates. AND and OR-gates are nodes with an in-degree 2 while the NOT-gate has in-degree 1. The computation of an m-ary Boolean function follows straight-forward by applying the designated function of the node to the input values. The result of the output of the circuit C on input x is called $\mathbf{res}_C(x)$.

Because Boolean circuits have a fixed input size, one considers families of Boolean circuits. Such a family of circuits is called uniform, if a deterministic Turing machine can output the circuit for n input bits using only space $O(\log n)$ (The output tape cannot be read and thus is not accounted for the space complexity).

3.6.2 Complexity Classes

Traditionally, complexity classes are defined for decision problems, which solve predicates (e.g. given a graph does there exists a Hamiltonian cycle). Furthermore, these predicates are described by the set of inputs given a positive output, where the inputs are given as words over an alphabet Σ. Theses input sets are called languages $L \subseteq \Sigma^*$ and a machine that can decide whether a word of such a language is given as input, accepts the language. However, it just computes the binary output of a predicate defined over strings.

For a function $f : \mathbb{N} \to \mathbb{N}$ a machine decides a language with resource f, if there is a machine of that machine model for every input of length n, at most $f(n)$ of this resource is used. We need the following basic complexity classes:

The class of languages (predicates) that can be decided

- $L \in$ DTime(T): L can be decided by a deterministic Turing machine in time $T(n)$
- $L \in$ NTime(T): L can be decided by a non-deterministic Turing machine in time $T(n)$
- $L \in$ Space(T): L can be decided by a deterministic Turing machine in time $T(n)$
- $f \in \mathcal{NC}^k$: There exists a uniform circuit family $C_1, C_2, \ldots$, such that for all n and $x \in \{0,1\}^n$: $\text{res}_{C_n}(x) = f(x)$ and the depth of circuit C_n is bounded by $O(\log^k n)$ and the size of circuit C_n is polynomial.

Important complexity classes derivated from this definition are $\mathcal{P}$:= DTime(POL) and $\mathcal{NP}$:= NTime(POL).

For an intuition $\mathcal{P}$ can be seen as the class of problems efficiently solvable by a sequential machine, while the class $\mathcal{NC} := \bigcup_k \mathcal{NC}^k$ describes the class of problems which can be solved very efficiently by parallel computers. For this note that any problem in $\mathcal{NC}^k$ can be solved by a CRCW-PRAM in time $O(\log^k n)$ with a polynomial number of processors.

In the area of computational complexity very few problems are known where one can actually show that it is computationally infeasible to solve them. At least it is possible to show for a number of problems that they are $\mathcal{NP}$-hard: If an $\mathcal{NP}$-hard problem can be solved in polynomial time on a sequential machine model (like DTM or RAM), then $\mathcal{NP}$ collapses to $\mathcal{P}$ i.e. $\mathcal{NP} = \mathcal{P}$. This is one of the most important open question in computer science, and most of the researching community believes that such a collapse is not the case.

The technique to show such relative results relies on polynomial time reductions and is excellently presented in [GJ79]. A problem P_1 (also known as language, predicate) can be reduced to P_2 in polynomial time, denoted by $P_1 \leq_{\text{pol}} P_2$, if there is a function f computable in polynomial time such that for all inputs x: $P_1(x) = P_2(f(x))$, in language notation: $x \in P_1 \iff f(x) \in P_2$.

A problem is $\mathcal{NP}$-hard, if every problem of $\mathcal{NP}$ can be reduced to this problem. An $\mathcal{NP}$-hard problem is $\mathcal{NP}$-complete, if it is in $\mathcal{NP}$. There are numerous natural $\mathcal{NP}$-complete problems and if one of these problems is in $\mathcal{P}$, then $\mathcal{NP} = \mathcal{P}$, which is (if we recall the difference between the underlying machine models) absolutely contra-intuitive.

Chapter 4

Broadcasting in Planar and Decomposable Graphs

4.1 Introduction

Broadcasting in processor networks means disseminating a single piece of information, which is originally known only to some nodes, called the *sources*, to all members of the network. This is done in a sequence of rounds by pairwise message exchange over the communication lines of the network. In one round each processor can send a message to at most one of its neighbors. The goal is to inform everybody using as few rounds as possible. This number is called the *minimum broadcasting time* of the network. This short description constitutes the *telephone model* for broadcasting in undirected graphs.

Broadcasting is a basic task for multiprocessor systems that should be supported by the topology of the network. This problem has been studied extensively, mostly in the case of a single source – for an overview see [HHL88, HKMP96]. In several papers the broadcast capabilities of well known families of graphs like hypercubes, cube-connected-cycles, shuffle exchange graphs or de Bruijn graphs have been investigated and compared. In [MJ90] Hromkovič, Jeschke and Monien have studied the relation between the broadcasting time and the time for solving the related gossiping problem for special families of graphs.

On the other hand, one has tried to find optimal topologies for networks with a given number of nodes such that the broadcasting time is best possible. Here the worst case over all nodes as the single source should be minimized. The problem gets more complicated when restricting to graphs of bounded degree. In [LP88] Liestman and Peters have studied several classes of bounded degree graphs in this respect, see also [BHLP92]. Balanced binary trees already achieve a broadcasting time of logarithmic order, therefore the question is the optimal constant factor in front of the logarithm.

In this chapter we will investigate the optimization problem for arbitrary networks. That means, given a graph and a subset of nodes as sources, determine its specific broadcast time or more general find a broadcast schedule of minimal length. This problem in general is $\mathcal{NP}$-complete. We will show that this property remains even if one restricts to planar graphs of bounded degree or constant broadcast time. Furthermore, the problem cannot be solved approximately with an arbitrary precision unless $\mathcal{P} = \mathcal{NP}$.

On the other hand, we will investigate for which classes of graphs this problem can be solved efficiently. All that seems to be known is that broadcasting is easy for trees as shown by Slater, Cockayne, and Hedetniemi in [SCH81]. Many combinatorial optimization problems for graphs have been shown to be solvable in polynomial sequential time and even in poly-logarithmic parallel time for more general classes of graphs: graphs of bounded tree-width (see for example the paper by Arnborg, Lagergren and Seese [ALS91]) and graphs of small connectivity ([Rei91a]) – an overview can be found in [Rei91b].

The broadcasting problem seems to be more difficult in this respect since it does not have the finite-state-property or a bounded number of equivalence classes. Thus the methods of [ALS91] and [Rei91a] are not directly applicable. Still, modifying the framework developed in [Rei91a] we can show that broadcasting becomes easy for graphs with good decomposition properties. For this purpose we have to extend the notion of graph decomposition to measure its properties more exactly. A careful inspection of the possibilities how information can flow within a component and between different components of a graph will be required. For the internal flow components that are connected behave most favorably, but in general connectivity cannot always be achieved by a tree decomposition into small components. The algorithm even works for a more general version of the broadcasting problem. Furthermore, it can be parallelized efficiently to yield $\mathcal{NC}$-solutions.

As a conclusion we can say that combining these new negative and positive results the parameters that make broadcasting difficult are determined quite precisely. The complexity of this problem jumps from $\mathcal{P}$ to $\mathcal{NP}$ depending on the internal structure of the networks.

4.2 Definitions and Previous Results

We state a formal definition of the BROADCASTING DECISION PROBLEM [GJ79], which is also known as *Broadcasting in the Telephone Model* and the *single-port interconnection architecture* [Rav94].

Definition 1 *Let $G = (V, E)$ be a (directed) graph with a distinguished subset of vertices $V_0 \subseteq V$, the* **sources***, and $T^* \in \mathbb{N}$ be a deadline. The task is to decide whether there exists a* **broadcast schedule***, that is a sequence of subsets of edges*

$$E_1, E_2, \ldots, E_{T^*-1}, E_{T^*}$$

with the property $V_{T^*} = V$, where for $i > 0$ we define $V_i := V_{i-1} \cup \{ v \mid (u, v) \in E_i \text{ and } u \in V_{i-1} \}$ and require

$$E_i \subseteq \{ (u, v) \in E \mid u \in V_{i-1} \} \quad \text{and} \quad \forall u \in V_{i-1} : \mid E_i \cap (\{u\} \times V) \mid \leq 1 .$$

Let us distinguish between the MULTIPLE SOURCES BROADCASTING PROBLEM **MB** *and the restricted version with only a single source: the* SINGLE SOURCE BROADCASTING PROBLEM **SB***. □

The meaning of the sets E_i and V_i is the following: V_i denotes the set of nodes that have received the broadcast information by round i. For $i = 0$ this is just the set of sources. By the deadline T^* the set V_{T^*} should include all nodes of the network. E_i is the set of edges that are used to send information in round i, where each processor $u \in V_{i-1}$ can use at most one of its outgoing edges.

MB (denoted ND49 in [GJ79]) has shown to be $\mathcal{NP}$-complete.

Theorem 1 *[SCH81] MB for graphs with unbounded degree is $\mathcal{NP}$-complete, even if restricted to a fixed deadline $T^* \geq 4$.*

For a fixed deadline the number of sources obviously has to grow linearly in the size of the whole graph. But even the single source problem is difficult, in this case the deadline has to grow at least logarithmically.

Theorem 2 *[SCH81] SB for graphs with unbounded degree is $\mathcal{NP}$-complete.*

The proofs of both results were published by Slater, Cockayne, and Hedetniemi ([SCH81]). For the second result, their reduction of the 3-dimensional matching problem to SB requires a deadline of order $\sqrt[3]{|V|}$ for the broadcast problem. Furthermore, in the same paper it is shown:

Theorem 3 *[SCH81] SB can be solved in linear time for trees. This also holds for the constructive version of this problem finding an optimal broadcast schedule.*

Previous results concerning the approximation of the broadcasting problem can be found in the next chapter.

4.3 New Results

All theorems above can be improved significantly. For the lower bounds it suffices to consider undirected graphs, the upper bounds given below also hold for the more general case of directed graphs.

4.3.1 Lower Bounds

Designing more complicated reductions from the 3-dimensional matching problem and the 3-SAT problem we can show:

Theorem 4 *MB restricted to planar graphs with degree at most 4 and a fixed deadline T^* at least 3 is $\mathcal{NP}$-complete.*

After a preliminary representation of Theorem 4 Middendorf was able to improve it to degree 3 and deadline 2 [Mid93].

The broadcasting problem with a single source does not become substantially easier, even for bounded degree graphs with a logarithmic diameter.

Theorem 5 *SB restricted to graphs $G = (V, E)$ with degree at most 3 is $\mathcal{NP}$-complete, even if the deadline grows at most logarithmically in the size of the graph.*

Also planarity does not make things much simpler as the following result shows.

Theorem 6 *SB restricted to planar graphs $G = (V, E)$ of degree 3 is $\mathcal{NP}$-complete (in this case the deadline grows like $\sqrt{|V|}$).*

4.3.2 Upper Bounds

On the positive side, we will extend the classes of graphs for which the broadcasting problem can be solved fast. For this purpose, different ways on how a graph can be decomposed into smaller components will be considered: by removing edges (edge separators) or by removing nodes (node separators). The concept of graph decomposition based on the k-connected components of a graph is developed in [Hoh90] and [HR89] and is strongly related to the notion of tree-width introduced by Robertson and Seymour [RS83, RS86].

In [Hoh90] and [HR89] only node separators have been considered. For the broadcasting problem a slightly different notion of graph decomposition seems to be better suited. Furthermore, the weaker notion of edge separation is of interest because the analysis in this case is slightly less complicated and yields better bounds. For efficiency reasons an important point is to get good bounds on the round numbers, when nodes may receive the broadcast information. Things are easy if all components of the graph decomposition are connected, which in general cannot be assumed.

Here we restrict only to decompositions that generate a tree of components. Using more complicated techniques other decomposition graphs can also be handled. For the purpose of decomposing a graph G it suffices to consider only the case of undirected graphs. Thus, if G is directed in the following definition we simply mean the corresponding undirected graph.

Definition 2 *A graph $H = (V_H, E_H)$ is an* **edge decomposition graph** *of a graph $G = (V, E)$ if the following conditions hold:*

- *The nodes G_i of V_H represent induced subgraphs $G_i = (V_i, E_i)$ of G such that the V_i are pairwise disjoint and $V = \bigcup_{G_i \in V_H} V_i$.*
- *$\{G_i, G_j\} \in E_H$ iff there is an edge between a node of G_i and a node of G_j.*

H is called an **edge decomposition tree** *of G if H is a tree. Define the* **cut** *of an edge $\{G_i, G_j\}$, the cut of a node G_i, and the cut of H as those edges of G that connect G_i and G_j, resp. connect G_i to other components or connect any pair of components:*

$$\begin{aligned} \text{cut}(G_i, G_j) &:= \{ \{u, v\} \in E \mid u \in V_i \text{ and } v \in V_j\} \quad \textit{for } i \neq j \ , \\ \text{cut}(G_i) &:= \bigcup_{\{G_i, G_j\} \in E_H} \text{cut}(G_i, G_j) \qquad \textit{and} \qquad \text{cut}(H) := \bigcup_{G_i \in V_H} \text{cut}(G_i) \ . \end{aligned}$$

The **border** *of a node G_i are the nodes of other components that have connections to G_i:*

$$\text{border}(G_i) := \{ v \mid \{u, v\} \in \text{cut}(G_i) \text{ and } u \in V_i \} \ .$$

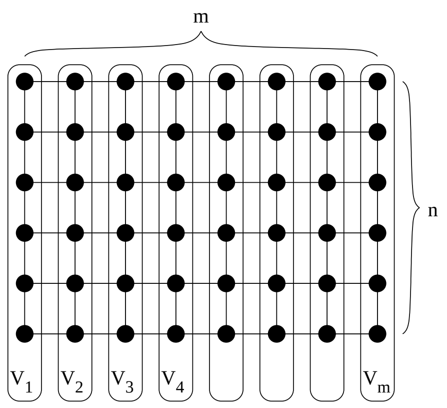

Figure 4.1: A (2n,n,1)-edge decomposition of the $m \times n$-grid

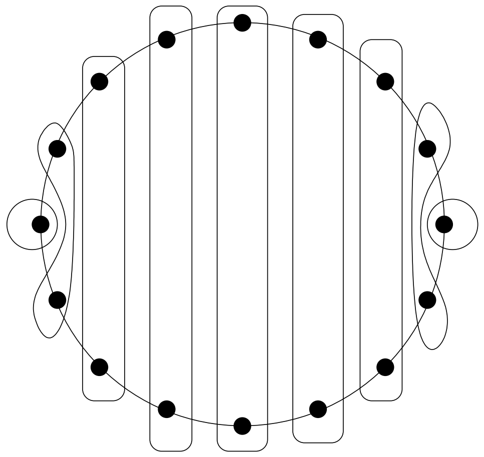

Figure 4.2: A (4,2,2)-edge decomposition of a cycle

A graph $G = (V, E)$ is $(\boldsymbol{\kappa}, \boldsymbol{\mu}, \boldsymbol{c})$**–edge decomposable** *if there exists an edge decomposition graph $H = (V_H, E_H)$ such that for all $G_i \in V_H$:*

$$|\operatorname{cut}(G_i)| \leq \kappa , \qquad |V_i| \leq \mu \qquad \textit{and} \qquad \mathtt{cc}(G_i) \leq c ,$$

where $\mathtt{cc}(G_i)$ denotes the number of connected components of G_i. In this case $\operatorname{cut}(H)$ *will be called a* $(\boldsymbol{\kappa}, \boldsymbol{\mu}, \boldsymbol{c})$**–edge separator** *of G.* □

Note that the decomposition process partitions a graph into different components. Each component G_i itself may be connected or fall into several connected components. For example, a $m \times n$-grid is $(2n, n, 1)$–edge decomposable into a tree, see Figure 4.1. For a cycle of length n the parameters are $(4, 2, 2)$, see Figure 4.2. Taking the number of connected components within each component into consideration will allow us to bound the algorithmic effort to solve the broadcasting problem in a nontrivial way.

Other approaches have been proposed how to decompose a graph into smaller components, based on the notions of tree-width ([RS83], [RS86]), see for example [ALS91],[BK91],[Lag90]. It is known that graphs with small tree-width allow the efficient solution of otherwise infeasible problems like Hamiltonian circuit or Independent set. For an overview see [Bod93]. However, it is an open problem whether this is also the case for the broadcasting problem.

In the following we assume that an edge decomposition of the network is given and are not bothered how to obtain such a decomposition.

Theorem 7 *For a graph $G = (V, E)$ of maximal degree d with a given (κ, μ, c)–edge decomposition tree MB can be solved in time*

$$\begin{aligned} & \mathrm{O}\left(2^{\kappa+c+\mu}|V|^{c+2}(\kappa+\mu)^{\kappa+c+2}\left(1+\frac{|E|}{|V|}\right)^{\mu}\right) \\ \leq\ & \exp \mathrm{O}\Big(c \cdot \log|V| \ +\ (\kappa+c)\cdot\log(\kappa+\mu) \ +\ \mu\cdot\log d\Big)\ . \end{aligned}$$

The algorithm we have designed actually works for a more general version of the broadcasting problem, in which the sources may receive the broadcast information in different rounds and each node of the network may have its individual deadline. Let us call this the *general broadcasting problem* **GB** (A formal definition can be found on page 43).

The time bound becomes polynomial for classes of graphs that can be decomposed into smaller components using not too large separators.

Corollary 1 *Restricted to graphs $G = (V, E)$ with*

- $(\mathrm{O}(\frac{\log n}{\log\log n}), \mathrm{O}(\frac{\log n}{\log\log n}), \mathrm{O}(1))$*–edge decomposition trees or*
- *to graphs with bounded degree and* $(\mathrm{O}(\frac{\log n}{\log\log n}), \mathrm{O}(\log n), \mathrm{O}(1))$*–edge decomposition trees*

MB (and even GB) can be solved in polynomial time.

So far, we have only considered the decision version of MB, resp. the task to determine the minimal length of a broadcast schedule. But applying ideas similar to the one in [Rei91a] one can also design an algorithm for constructing an optimal broadcast schedule by using the same techniques as for the decision problem.

Theorem 8 *Constructing an optimal broadcast schedule can be done in the same time bounds as stated for the decision problem in Theorem 7.*

Using the machinery developed in [Rei91a] we can also derive a fast and processor efficient parallel algorithm. Even if the decomposition tree is not nicely balanced using path compression techniques the problem can be solved with a logarithmic number of iterations (with respect to the number of components). The basic task one has to solve is how a chain of two components can be replaced by a single component that externally behaves identically with respect to broadcasting.

Theorem 9 *For a graph $G = (V, E)$ of maximal degree d with a given (κ, μ, c)–edge decomposition tree MB can be solved on the PRAM model in parallel time*

$$O\Big(\log|V| \cdot (c\cdot\log|V| \ +\ (\kappa+c)\cdot\log(\kappa+\mu) \ +\ \mu\cdot\log d)\Big)$$

with a processor bound of $\quad \exp O\Big(c\cdot\log|V| \ +\ (\kappa+c)\cdot\log(\kappa+\mu) \ +\ \mu\cdot\log d\Big)\ .$

For nicely decomposable classes of graphs these bounds put the MB-problem into $\mathcal{NC}$.

Corollary 2 *Restricted to graphs $G = (V, E)$ with*

- $(\mathrm{O}(\frac{\log n}{\log\log n}), \mathrm{O}(\frac{\log n}{\log\log n}), \mathrm{O}(1))$*–edge decomposition trees or*
- *to graphs with bounded degree and* $(\mathrm{O}(\frac{\log n}{\log\log n}), \mathrm{O}(\log n), \mathrm{O}(1))$*–edge decomposition trees*

MB is in $\mathcal{NC}^2$.

Furthermore, we consider a decomposition of graphs which is more closely related to the notion of tree-width.

Definition 3 *A graph $H = (V_H, E_H)$ is a* **node decomposition graph** *of a graph $G = (V, E)$ if*

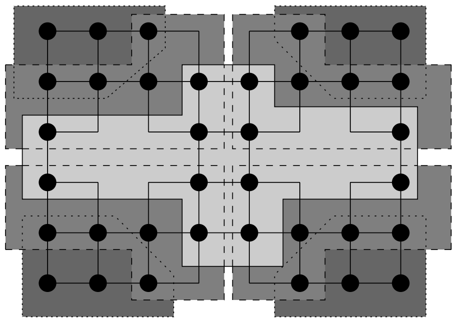

Figure 4.3: A (8,8,3)-node decomposition tree of an example graph

- *the nodes G_i of V_H represent subgraphs $G_i = (V_i, E_i)$ of G such that $V = \bigcup_{G_i \in V_H} V_i$ and $E = \bigcup_{G_i \in V_H} E_i$,*
- *for each node v holds: if $v \in V_i \cap V_j$ then H contains a path π from V_i to V_j such that v belongs to every node V_l in π.*

H is called a **node decomposition tree** *of G if H is a tree. Figure 4.3 gives an example of a node decomposition tree.*

Similarly to above, we define the **cut** *of an edge $\{G_i, G_j\}$, of a node G_i, and of H as* $\mathrm{cut}(G_i, G_j) := V_i \cap V_j$, *resp.*

$$\mathrm{cut}(G_i) \;:=\; \bigcup_{\{G_i,G_j\}\in E_H} \mathrm{cut}(G_i,G_j)\;, \qquad \mathrm{cut}(H) \;:=\; \bigcup_{G_i \in V_H} \mathrm{cut}(G_i)\;.$$

The **border** *of a node G_i are the nodes of other components G_j that are connected to* $\mathrm{cut}(G_i)$

$$\mathrm{border}(G_i) \;:=\; \{\; v \notin V_i \mid \exists\, u \in \mathrm{cut}(G_i) \text{ and } \{u,v\} \in E\}\;.$$

A graph $G = (V, E)$ is called **(κ, μ, c)–node decomposable** *if there exists a node decomposition graph $H = (V_H, E_H)$ such that for all $G_i \in V_H$ holds:*

$$|\;\mathrm{cut}(G_i)\;| \;\le\; \kappa, \qquad |\;V_i\;| \;\le\; \mu, \qquad \textit{and} \qquad \mathtt{cc}(G_i) \le c\;.$$

In this case $\mathrm{cut}(H)$ *is a* **(κ, μ, c)–node separator** *of G.* □

Theorem 10 *Given a graph $G = (V, E)$ of maximal degree d with a (κ, μ, c)–node decomposition tree MB can be solved in time*

$$\begin{aligned} &\mathrm{O}\Big(|V|^{c+1} \cdot ((d-1)\cdot\kappa + \mu)^{\kappa+2} \cdot (d+1)^{\mu + d\cdot\kappa} \cdot 2^{d\cdot\kappa}\Big) \\ \le\;\; &\exp \mathrm{O}\Big(c \cdot \log|V| \;+\; \kappa \cdot \log(d\cdot\kappa + \mu) \;+\; (\mu + \kappa\cdot d)\cdot \log d\Big)\;. \end{aligned}$$

Similarly, we get in the parallel case:

Theorem 11 *For graphs of maximal degree d with a (κ, μ, c)–node decomposition tree MB has a parallel solution of time complexity*

$$\mathrm{O}\Big(\log|V| \cdot (c \cdot \log|V| \;+\; \kappa\cdot\log(d\cdot\kappa+\mu) \;+\; (\mu+\kappa\cdot d)\cdot\log d)\Big)$$

and processor complexity $\exp \mathrm{O}\Big(c\cdot\log|V| \;+\; \kappa\cdot\log(d\cdot\kappa+\mu) \;+\; (\mu+\kappa\cdot d)\cdot\log d\Big)\;.$

As in the case of edge separators for nicely node-decomposable graphs we get:

Corollary 3 *Restricted to graphs $G = (V, E)$ with*

- $(\mathrm{O}(1), \mathrm{O}(\frac{\log n}{\log\log n}), O(1))$*–node decomposition trees, or*
- *with maximal degree $d \le \mathrm{O}(\frac{\log\log n}{\log\log\log n})$ and $(\mathrm{O}(\frac{\log n}{\log\log n}), \mathrm{O}(\frac{\log n}{\log\log\log n}), \mathrm{O}(1))$–node decomposition trees, or*
- *with constant degree and $(\mathrm{O}(\frac{\log n}{\log\log n}), \mathrm{O}(\log n), \mathrm{O}(1))$–node decomposition trees*

MB can be solved in polynomial time, even in $\mathcal{NC}^2$.

All these bounds apply to the General broadcasting problem (GB) as well as the constructive variant to determine a broadcasting schedule.

The remaining part of this chapter is organized as follows. In the next section the $\mathcal{NP}$-completeness of multiple source broadcasting in planar, bounded degree graphs is proven (Theorem 4). Section 4.5 describes a set of basic building blocks that are used in the lower bound proofs for single source broadcasting. In the following two sections we give the main ideas of the reductions that yield Theorem 5 and 6. Efficient algorithms for edge-, resp. node-decomposable graphs are described in the last two sections of this chapter.

4.4 MB with Deadline 4 is $\mathcal{NP}$–Complete

Let us first observe how a nondeterministic Turing machine can solve the Multiple source broadcasting problem (MB).

Lemma 1 *MB can be solved by a NTM in time* $\mathrm{O}(|V|^2 \cdot \log|V|)$.

Proof: For a graph $G = (V, E)$ with maximal degree d specified by adjacency lists, a set of sources V_0, and deadline $T^* \le |V|$ we can solve MB by the following nondeterministic strategy:

Step 1: For each node $v \notin V_0$ choose one edge $(v', v) \in E$ with the interpretation that v receives the broadcast information from its neighbor v' ($v = v'$ means that v does not receive the information from somebody else).

Step 2: Let F be the subgraph of G consisting of the edges chosen in step 1. Verify that F has no directed cycle. If this condition holds F is a forest of rooted trees with edges pointing away from their roots.

Step 3: Solve the broadcasting problem for the trees constructed in step 1. Analyzing the time complexity of the strategy in [SCH81] for broadcasting in trees one can show that a RAM can solve this step in time $\mathrm{O}(|V|)$. Hence, a Turing machine can solve it in time $\mathrm{O}(|V|^2 \log|V|)$.

The correctness follows from the fact that each broadcasting schedule can be described by a directed forest, in which the edges are labeled by the round, the broadcast information is sent across this edge. Step 1 guesses such a forest and step 3 checks whether it is possible to inform this forest within the deadline T^*. $\square$

The $\mathcal{NP}$-hardness of MB will be proved by a reduction of 3-DIMENSIONAL MATCHING (3DM [GJ79]):

Definition 4 3DM *Given a set $M \subseteq A \times B \times C$, and A, B and C are disjoint sets having the same number q of elements, decide whether M contains a matching, i. e., a subset $M' \subseteq M$ such that $|M'| = q$ and no two elements of M' agree in any coordinate.* $\square$

The graph $G' = G(A, B, C, M)$ of an instance (A, B, C, M) with $M \subseteq A \times B \times C$ of the 3DM problem is defined as follows: Each element of the sets A, B and C and each triple of M is represented by a vertex. The membership relation between set elements and triples defines the edges between these vertices.

$$\begin{aligned}
&G' := (V', E') \text{ with } \quad V_A := \{\,\alpha_x \mid x \in A\}, \quad V_B := \{\,\beta_x \mid x \in B\},\\
&V_C := \{\,\gamma_x \mid x \in C\}, \quad V_M := \{\,\mu_y \mid y \in M\}, \quad V' := V_A \cup V_B \cup V_C \cup V_M,\\
&E' := \{\,(\mu_y, v_x) \mid y \in M,\ v_x \in V_A \cup V_B \cup V_C \text{ and } x \in y\}
\end{aligned}$$

The reduction will use a restricted version of the 3DM problem, which is still $\mathcal{NP}$-complete [DF86]. For an instance (A, B, C, M) of RESTRICTED PLANAR 3-DIMENSIONAL MATCHING the following properties are required:

- $G(A, B, C, M)$ is planar.
- For each element x of $A \cup B \cup C$ there are at most 3 triples in M containing x (thus, $|M|$ is bounded by $3q$ where $q := |A| = |B| = |C|$).

Proof of Theorem 4: Let (A, B, C, M) be an instance of 3DM with $|A| = q$ and let $G' = G(A, B, C, M)$ be the matching graph. The corresponding broadcasting graph G is obtained by replacing each node $\alpha_i \in V_a$ of G' by a chain $\alpha_{i,1}$, $\alpha_{i,2}$ and $\alpha_{i,3}$ of length 3 (see Figure 4.4). The other nodes and edges remain unchanged. $V_{A,1}$ is chosen as the set of sources, and the deadline is set to 3.

$$G(A, B, C, M) \;:=\; (V, E) \qquad \text{with}$$

$$\begin{aligned}
V &:= V_{A,1} \cup V_{A,2} \cup V_{A,3} \cup V_B \cup V_C \cup V_M, \\
E &:= \{ (\mu_y, v_x) \mid y \in M,\ v_x \in V_A \cup V_B \cup V_C \text{ and } x \in y \} \\
&\quad\ \cup \ \{ (\alpha_{i,1}, \alpha_{i,2}), (\alpha_{i,2}, \alpha_{i,3}) \mid i \in A \} .
\end{aligned}$$

The node sets $V_B, V_C, V_{A,1}, V_{A,2}, V_{A,3}$, and V_M are defined as follows.

$$\begin{array}{rclrcl}
V_B &:=& \{ \beta_x \mid x \in B \}, & V_C &:=& \{ \gamma_x \mid x \in C \}, \\
V_{A,1} &:=& \{ \alpha_{i,1} \mid i \in A \}, & V_{A,2} &:=& \{ \alpha_{i,2} \mid i \in A \}, \\
V_{A,3} &:=& \{ \alpha_{i,3} \mid i \in A \}, & V_M &:=& \{ \mu_y \mid y \in M \} .
\end{array}$$

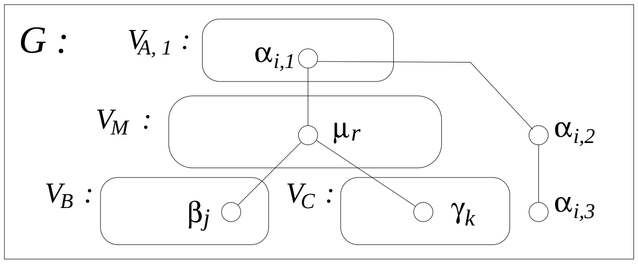

Figure 4.4: The broadcasting graph corresponding to an instance of the 3DM problem

Observe that $G(A, B, C, M)$ has degree 4 and is planar if G' is planar.

Lemma 2 *$G(A, B, C, M)$ has a broadcast schedule of length 3 iff M has a matching.*

Proof: Let $M' \subseteq M$ be a matching for A, B and C. Then the following strategy informs all nodes of $G(A, B, C, M)$ within 3 rounds:

Round 1: The q sources in $V_{A,1}$ send the information to the nodes of $V_{M'} := \{ \mu_r \mid r \in M' \}$ which represent the triples of M', hence $V_1 := V_0 \cup V_{M'}$.

Round 2: The q sources inform the nodes in $V_{A,2}$. The q nodes of $V_{M'}$ informed in round 1 inform the nodes of V_B, that means $V_2 := V_1 \cup V_{A,2} \cup V_B$.

Round 3: The nodes of $V_{A,2}$ send the information to the nodes in $V_{A,3}$, the nodes in $V_{M'}$ to the nodes of V_C, and the nodes in $V_{A,1}$ and V_B to the nodes of $V_M \setminus V_{M'}$, that is $V_3 := V_2 \cup V_{A,3} \cup V_C \cup (V_M \setminus V_{M'}) = V$.

Since M' is a matching for A, B and C, the nodes in $V_{M'}$ can inform all nodes in V_B in round 2, and all nodes in V_C in round 3.

It is also possible to inform the nodes in $V_M \setminus V_{M'}$ in round 3, because they can be matched with the nodes in $V_{A,1} \cup V_B$. This can be seen as follows: Each node of $V_M \setminus V_{M'}$ is connected to one node of $V_{A,1}$ and one node of V_B, whereas each node of $V_{A,1}$ and each node of V_B is connected to at most two nodes of $V_M \setminus V_{M'}$. Thus each subset V'' of $V_M \setminus V_{M'}$ is connected to a subset of $V_{A,1} \cup V_B$ of at least size $|V''|$. Therefore there exists a matching of $V_M \setminus V_{M'}$ with $V_{A,1} \cup V_B$.

For the other direction observe that each node $\alpha_{i,1} \in V_{A,1}$ has to inform $\alpha_{i,2}$ in the first or second round. Thus it is only possible to inform q nodes V' of V_M by round 1 or round 2. These q nodes have to send the information to the $2q$ nodes of V_B and V_C. Thus the neighborhood of V' contains all nodes of $V_{A,1}$, V_B and V_C. Since nodes in V' have degree 3, the triples corresponding to V' establish a matching M' of A, B, C. □

□

4.5 Modular Construction of Difficult Broadcast Networks

For the single source problem the reduction to show $\mathcal{NP}$–hardness is much more complicated. We will give a modular description by first constructing a series of some basic graphs with special broadcast properties.

Definition 5 *Let a graph $G = (V, E)$ and a broadcast schedule $\mathcal{E} = E_1, E_2, \ldots$ for G be given. The first round in which a node v gets the information is called its* **starting round** *$t_s(v)$. If v sends the information to a neighbor in round t we call v* **active** *in that round. Let $t_f(v)$ be the first round by which all neighbors of v are informed. A node v is* **busy** *in $\mathcal{E}$ if it is active in all rounds $t_s(v) + 1, \ldots, t_f(v)$. $\mathcal{E}$ is* **busy** *if all nodes are busy.* □

Observe that each broadcast schedule can easily be transformed into a busy broadcast schedule of the same or smaller length. Therefore we will only consider busy broadcast schedules in the following.

The proof of Theorem 6 is based on an intricate construction of a special broadcast network G. This section precedes with an analysis of some special subgraphs that will be used as basic building blocks. Each such subgraph G' has a designated set of input and output ports. Subgraphs will be connected over these ports only. If output ports of G' are connected to input ports of another subgraph G'' we call G'' a successor of G', and G' a predecessor of G'' .

If the broadcast information is sent over such a connecting edge we say that the edge is used in the corresponding round. Obviously, each edge does not have to be used more than once. The final network G will be built in such a way that in an optimal schedule the input and output edges of a subgraph G' have to be used at specific times.

Definition 6 *Let, for a given schedule of G, the input edges $e_1, e_2, \ldots, e_k$ of a subgraph G' be used in rounds $t_1, t_2, \ldots, t_k$. Then the vector $\tau = \langle t_1, t_2, \ldots, t_k \rangle$ is called an* **input time table** *of G'. The set of all possible time tables is called the* **input time sheet** *$\mathcal{T}(G')$ of G'. Analogously, we define* **output time tables** *and* **output time sheets**. □

The broadcast network we are going to construct has the property that in optimal schedule all input edges of a subgraph have to be used within a time interval of length at most 2, that means optimal input time tables are rather restricted.

Definition 7 *For a subgraph G' and an input time table $\tau = \langle t_1, t_2, \ldots, t_k \rangle$ of G' let $\mathbf{delay}(G', \tau)$ denote the minimal time that elapses between the round the broadcast information enters G' (that is the minimal t_i) and the first round an output edge of G' is used.* □

A lower bound for the time when an input edge e of G' can be used is obtained by adding up all delays on the shortest path from e to a source.

Definition 8 *Let v_0 be the unique source of the broadcast network G, and let* path(v_0, G') *be the set of all paths from v_0 to the subgraph G' of G. Then define*

$$\mathbf{dawn}(G') \;:=\; \min_{P \,\in\, \mathrm{path}(v_0,G')} \sum_{P \text{ crosses } G'' \,\neq\, G'} \min_{\tau \in \mathcal{T}(G'')} \textit{rm delay}(G'', \tau)\ .$$

If a node of a subgraph G' is informed in round t we call $t -$ dawn(G') *the* **relative round** *this node is informed.* □

Although edges between subgraphs are undirected and thus could be used in either direction we want to ensure that information enters a subgraph only at its input ports. A **ghost message** is a message that enters a subgraph G' through one of its output ports. To prevent ghost messages the following properties are helpful:

1. All successors of a subgraph G' have the same dawn.
2. All input ports of G' can be used in round dawn$(G) + 2$ at the latest.
3. Let the minimal number of rounds the information needs to reach an input port of G' starting at another input port of G' and using only edges of G' be the **ghost time** of G'. The ghost time of all subgraphs will be at least 3.

Let us call the mapping from the input time tables of a subgraph G' to its output time tables the *broadcast relation* $\mathcal{B}$ of G', or more formally:

Definition 9 *For the set of graphs $\mathcal{G}$ described below the broadcast relation*

$$\mathcal{B} \;:\; \mathcal{G} \times \mathbb{N} \times \mathbb{N}^m \;\rightarrow\; \mathsf{P}(\mathbb{N}^n) \cup \bot$$

is given by $(a_1, \ldots, a_n) \in \mathcal{B}(G', T^*, t_1 \ldots t_m)$ if the following two conditions hold:

- *$\tau \;=\; \langle t_1, \ldots, t_m \rangle$ is an input time table for G',*
- *if the m input edges of G' are used according to τ then the n output edges of G' can be used according to the output time table $\langle a_1, \ldots, a_n \rangle$ and all nodes of G' can be informed within the deadline T^*.*

$\mathcal{B}(G', T^*, t_1, \ldots, t_m) = \bot$ iff it is not possible to inform all nodes of G' within the deadline T^* using the input time table τ. □

Obviously, $\mathcal{B}(G', T^*, \tau)$ describes the information flow properties of G'.

4.5.1 Basic Broadcast Networks

Now we will analyze the functionality of the broadcast networks described in Figure 4.5, 4.7, 4.6 and 4.8.

The first subgraph Init$_{n,t}$ is called **initializer** (see Figure 4.5). If we choose the parameter $t = T^* -$ dawn(Init$_{n,t}$) with $t \geq 3\log n - 1$ the input node α has to send the information to β_1 in round dawn(Init$_{n,t}$) $+ 1$. Otherwise the last nodes of the chain cannot receive the information within the deadline. Hence, β_1 and β_2 can inform their successors using the edges Out$_1$ and Out$_2$ simultaneously in relative round 3. The initializer transmits the information simultaneously over all its n output edges, i.e. rm delay(Init$_{n,t}$, dawn(Init$_{n,t}$)) $= 3\log n$ and

$$\mathcal{B}(\mathrm{Init}_{n,t}, T^*, t') \;=\; \begin{cases} \{(\underbrace{3\log n, \ldots, 3\log n}_{n})\} & \text{if } t' = \mathrm{dawn}(\mathrm{Init}_{n,t})\ , \\ \bot & \text{if } t' > \mathrm{dawn}(\mathrm{Init}_{n,t})\ . \end{cases}$$

The following subgraphs model a binary coding system. The two possible values correspond to a receiving the broadcast information at relative rounds 0, resp. 2.

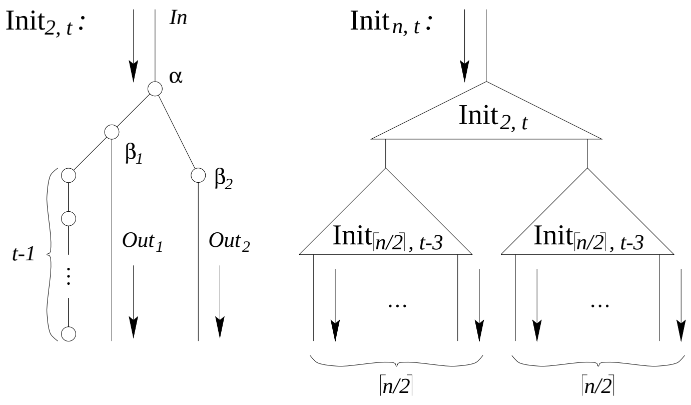

Figure 4.5: The recursive construction of the initializer $\mathrm{Init}_{n,t}$

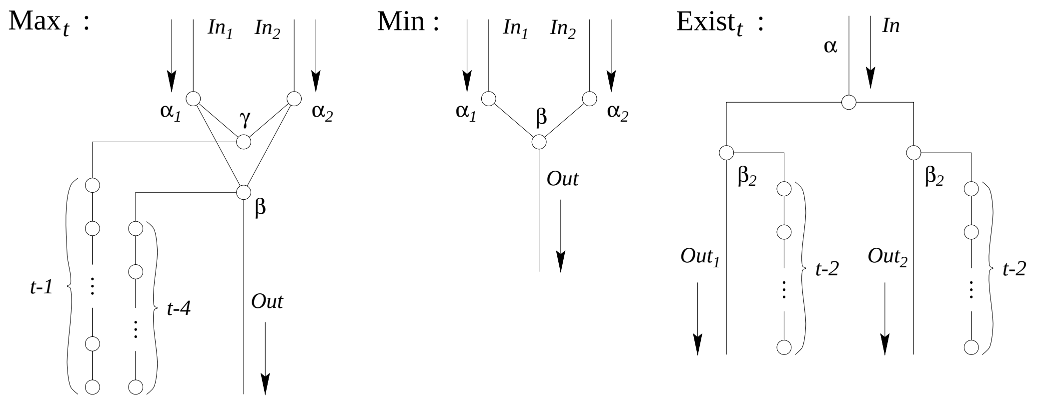

Figure 4.6: The max–graph Max_t, the min–graph Min and the guess–graph Exist_t

The **guess–graph** Exist_t (see Figure 4.6) with $t = T^* - \mathrm{dawn}(\mathrm{Exist}_t)$ and $t \geq 2$ is used to generate this encoding. It holds rm $\mathrm{delay}(\mathrm{Exist}_t, \delta) = 2$ and

$$\mathcal{B}(\mathrm{Exist}_t, t') = \begin{cases} \{(\delta+\mathbf{2}, \delta+\mathbf{4}), (\delta+\mathbf{4}, \delta+\mathbf{2})\} & \text{if } t' = \delta\ , \\ \perp & \text{if } t' > \delta\ , \end{cases}$$

with $\delta = \mathrm{dawn}(\mathrm{Exist}_t)$. Note that after informing α the broadcast strategy has to decide whether α sends the information to β_1 or β_2 first. We will interpret this decision as setting a Boolean variable.

The subgraph **duplicator** Dup_t (see Figure 4.7) with $t = T^* - \mathrm{dawn}(\mathrm{Dup}_t)$ and $t \geq 5$ will be used to duplicate this binary encoding. The input edges In_1 and In_3 inform α_1 and α_3 in round $\mathrm{dawn}(\mathrm{Dup}_t)$. For $\delta = \mathrm{dawn}(\mathrm{Dup_t})$ it holds rm $\mathrm{delay}(\mathrm{Dup_t}, \delta, \delta, \delta) = 3$ and

$$\mathcal{B}(\mathrm{Dup_t}, \mathrm{T^*}, \delta, \mathrm{t'}, \delta) = \begin{cases} \{(\delta+3, \delta+3)\} & \text{if } t' = \delta\ , \\ \{(\delta+5, \delta+5)\} & \text{if } t' \geq \delta+2\ . \end{cases}$$

To combine two binary encodings we use the **max–graph** (see 4.6) Max_t with $t = T^* - \mathrm{dawn}(\mathrm{Max}_t)$ and the **min–graph** Min. It is easy to see that rm $\mathrm{delay}(\mathrm{Min}, t') = 2$. If $\min(t_1, t_2) \leq T^* - 2$ we get $\mathcal{B}(\mathrm{Min}, T^*, t_1, t_2) = \{\min(t_1, t_2) + 2\}$, and else $\mathcal{B}(\mathrm{Min}, T^*, t_1, t_2) = \perp$. The max–graph does not simulate the computation of the maximum of two input rounds precisely. If both input edges are used later than $\mathrm{dawn}(\mathrm{Max}_t)$ at least one node $v \in \mathrm{Max}_t$ does not receive the information within the deadline. Note that we have to guarantee that γ receives the information before $\mathrm{dawn}(\mathrm{Max}_t) + 1$. So we get rm $\mathrm{delay}(\mathrm{Max}_t, \mathrm{dawn}(\mathrm{Max}_t), \mathrm{dawn}(\mathrm{Max}_t)) = 3$ and

$$\mathcal{B}(\mathrm{Max}_t, T^*, t_1, t_2) = \begin{cases} \{\mathrm{dawn}(\mathrm{Max}_t) + 3\} & \text{if } t_1 = t_2 = \mathrm{dawn}(\mathrm{Max}_t)\ , \\ \{\mathrm{dawn}(\mathrm{Max}_t) + 5\} & \text{if } |\{t_i | t_i = \mathrm{dawn}(\mathrm{Max}_t), i \in \{1,2\}\}| = 1\ , \\ \perp & \text{if } t_1, t_2 > \mathrm{dawn}(\mathrm{Max}_t)\ . \end{cases}$$

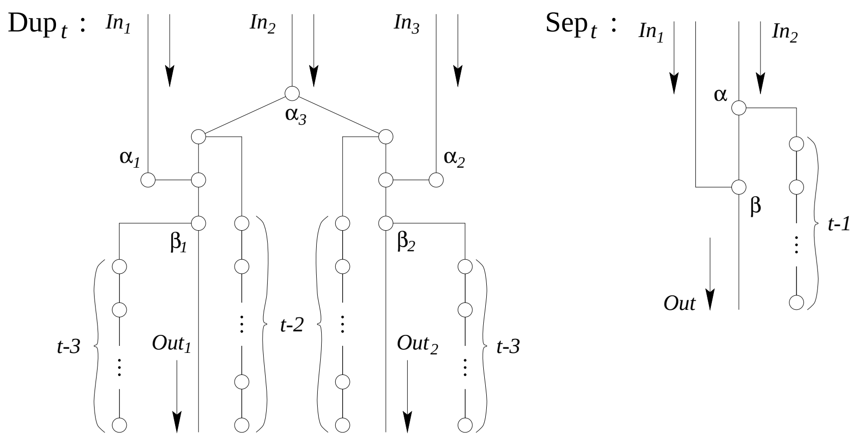

Figure 4.7: The subgraphs duplicator Dup_t and separator Sep_t

The subgraph **separator** (see Figure 4.7) Sep_t with $t = T^* - \mathrm{dawn}(\mathrm{Sep}_t)$ and $t \geq 2$ realizes a threshold function. It separates the set of all input constellations where In_1 is used at $\mathrm{dawn}(\mathrm{Sep}_t)$ into two groups:

$$\mathcal{B}(\mathrm{Sep}_t, T^*, \mathrm{dawn}(\mathrm{Sep}_t), t') = \begin{cases} \{\mathrm{dawn}(\mathrm{Sep}_t) + 1\} & \text{if } t' \leq \mathrm{dawn}(\mathrm{Sep}_t) + 1 \, , \\ \{\mathrm{dawn}(\mathrm{Sep}_t) + 2\} & \text{if } t' \geq \mathrm{dawn}(\mathrm{Sep}_t) + 2 \, . \end{cases}$$

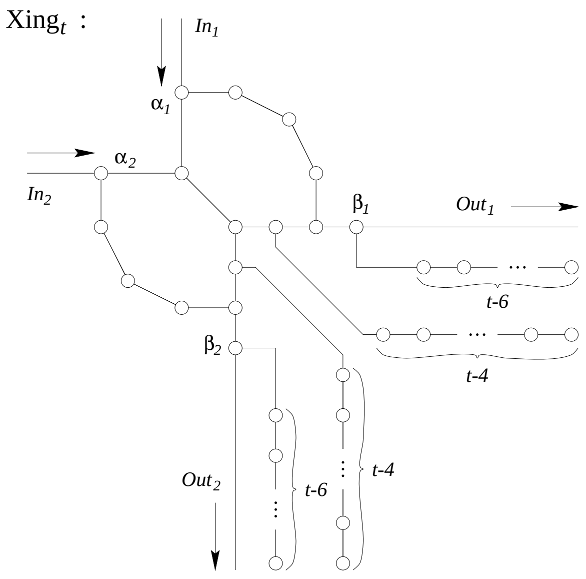

Figure 4.8: The planar crossing graph Xing_t

Furthermore, the **crossing** graph Xing_t with $t = T^* - \mathrm{dawn}(\mathrm{Xing}_t)$ and $t \geq 7$ realizes switching the location of incoming information. In particular, if both inputs are activated at dawn the same holds for the outputs. If one input is late, i.e. information arrives at $\mathrm{dawn}(\mathrm{Xing}_t) + 2$, only the opposite output is late. Observe that rm $\mathrm{delay}(\mathrm{Xing}_t, \mathrm{dawn}(\mathrm{Xing}_t), \mathrm{dawn}(\mathrm{Xing}_t)) = 6$ and

$$\mathcal{B}(\mathrm{Xing}_t, T^*, t_1, t_2) = \begin{cases} \{(t_2 + 6, t_1 + 6)\} & \text{if } t_1 = \mathrm{dawn}(X_l) \text{ or } t_2 = \mathrm{dawn}(X_l) \, , \\ \perp & \text{else,} \end{cases}$$

where $t_1, t_2 \in \{\mathrm{dawn}(\mathrm{Xing}_t), \mathrm{dawn}(\mathrm{Xing}_t) + 2\}$. An extended complete analysis of these graphs and a description of other elementary broadcasting subnetworks are given in [JRS94].

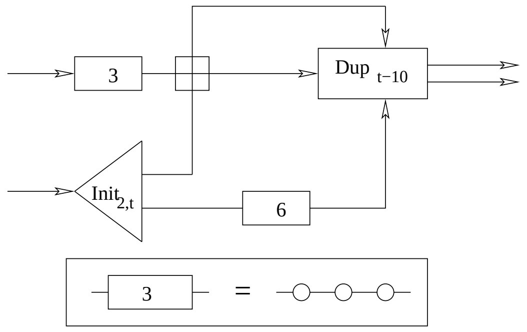

Figure 4.9: The construction of the planar duplicator

Using this crossing-graph we can substitute the duplicator by a **planar duplicator** as shown in Figure 4.9.

4.5.2 An Exact Encoding for Planar Crossings

The crossing graph Xing_t does not simulate a crossing of two broadcast signals exactly. If both input edges are used late, that means at dawn$(\mathrm{Xing}_t) + 2$, then there are some nodes $v \in \mathrm{Xing}_t$ which cannot received the information within the deadline. In the following we describe how to construct the crossing graph Cross_t that overcomes this difficulty. Two techniques are applied for this purpose.

1. The binary encoding of input rounds is made redundant by using pairs of inputs: dawn(Cross_t) and dawn$(\mathrm{Cross}_t) + 2$. Such a pair can be generated by a guess-graph $\mathrm{Exist}_{t'}$. We say a schedule uses an input edge (u, v) of a subgraph G' **in time (I)** if v receives the information from u in round dawn(G). If v receives the information from u in round dawn$(G) + 2$, the schedule uses (u, v) **late (L)**.

2. The crossing of two pairs $(x, x'), (y, y') \in \{(I, L), (L, I)\}$ will be realized by the following strategy: We first convert both pairs into an unary coding

$$[(x, x'), (y, y')] \longrightarrow [(\min\{u, v\})|_{u\in\{x,x'\} \text{ and } v\in\{y,y'\}}] .$$

This can be done by using several planar duplicators, min–graphs and crossings as shown in Figure 4.10. In a second step we decode this unary notation to the two binary exchanged pairs by using duplicators, crossings and max–graphs as shown in Figure 4.11. Note that the unary encoding contains always a permutation of $(1, 0, 0, 0)$. Hence, an arbitrary permutation of the positions of this unary encoding can be realized by using crossing–graphs $\mathrm{Xing}_{t''}$. The dawns of subgraphs used in the construction above are synchronized by additional chains.

The complete crossing graph Cross_t with $t = T^* -$ dawn(Cross_t) and $t \geq 230$ can be constructed such that rm delay$(\mathrm{Cross}_t, \delta) = 228$ and

$$\begin{aligned}\mathcal{B}(\mathrm{Cross}_t, T^*, \delta + x, \delta + 2 - x, \delta + y, \delta + 2 - y) \;=\; & \{(\delta + y + 228, \delta + 2 - y + 228,\\ & \delta + x + 228, \delta + 2 - x + 228)\}\end{aligned}$$

with $x, y \in \{0, 2\}$ and $d =$ dawn(Cross_t).

Combining some duplicators and some crossing-graphs Xing_t it is possible to construct a graph that duplicates the pairs (I, I, L) and (I, L, I) as shown in Figures 4.12. We call such a graph a **multiplicator** $\mathrm{Mult}_{n,t}$ where $t = T^* -$ dawn$(\mathrm{Mult}_{n,t})$ and n denotes the number of output pairs. This graph can be constructed such that rm delay$(\mathrm{Mult}_{n,t}) = 39 \log n$ and

$$B(\mathrm{Mult}_{n,t}, T^*, \delta + x, \delta + 2 - x) = \{([\delta + 39 \log n + x, \delta + 39 \log n + 2 - x]^n)\}$$

with $x \in \{0, 2\}$ and $\delta :=$ dawn$(\mathrm{Mult}_{n,t})$. All these graphs can easily be transformed into bipartite graphs with the same functionality.

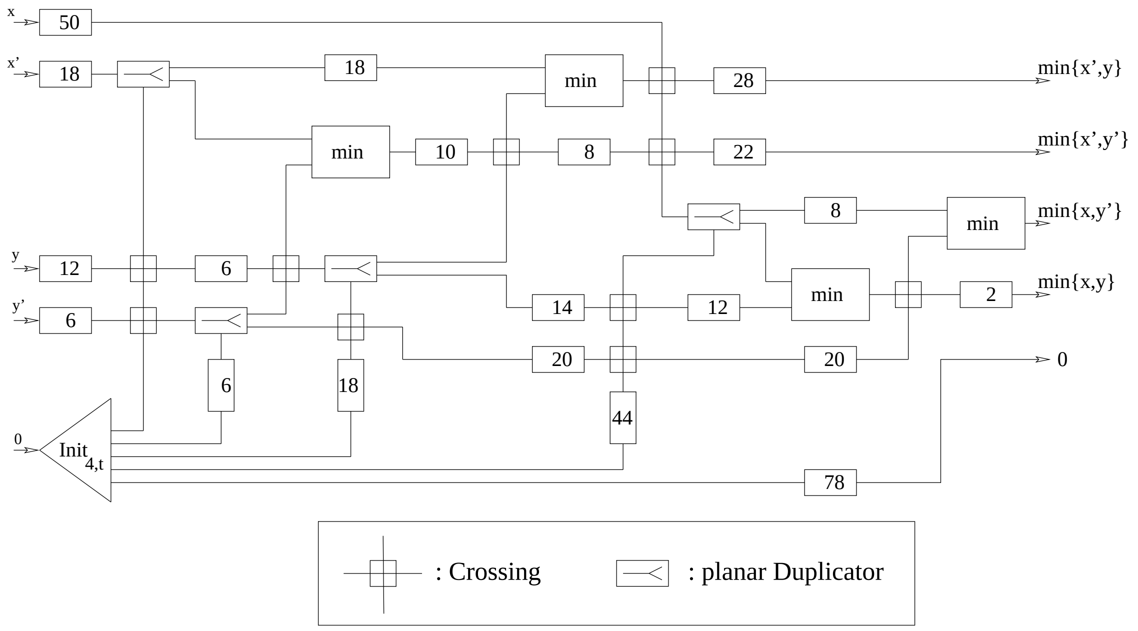

Figure 4.10: A schematic view at a binary-unary-converter

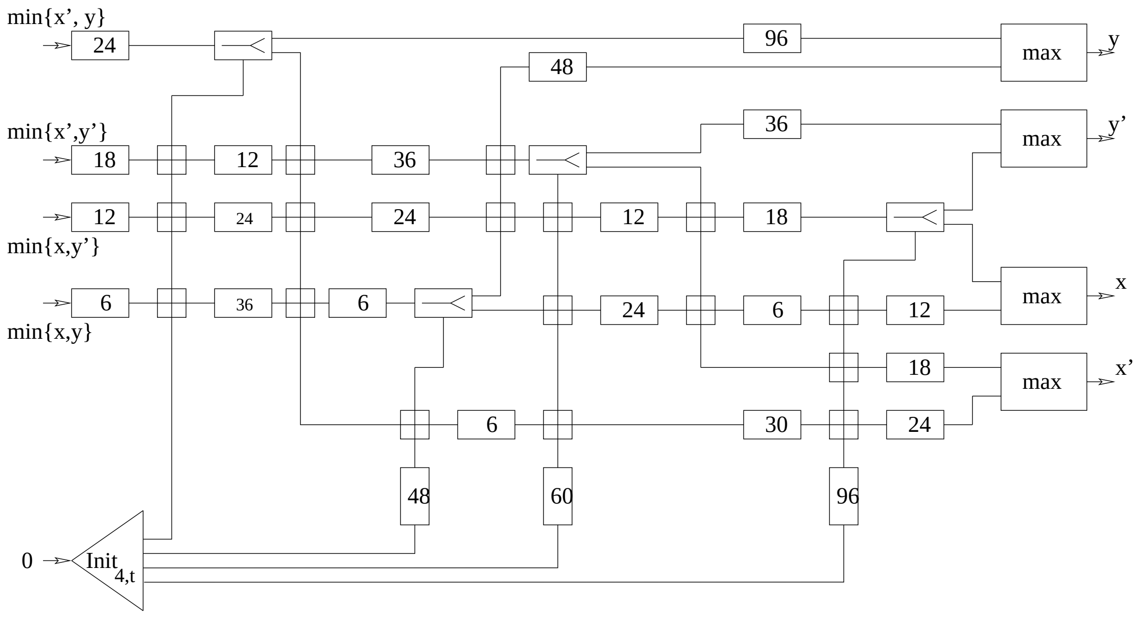

Figure 4.11: A schematic view at a unary-binary-converter

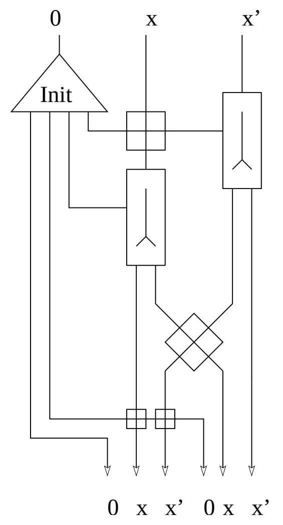

Figure 4.12: The multiplicator sub-graphs duplicates inputs $0, x, x'$

4.6 Single Source Broadcasting is $\mathcal{NP}$–Complete

The $\mathcal{NP}$–hardness of the SB problem for graphs with bounded degree will be proved by a reduction of a restricted version of 3DM problem, where for each element x of $A \cup B \cup C$ there are exactly three triples in M containing x [Bun84]. The main idea is similar to the reduction in the proof of Theorem 4.

Consider the tree $\text{Init}_{q,t}$ in Figure 4.5 with its root as the only source and q outgoing edges α_i, where $t \geq 3\lceil \log q \rceil$. It has the following properties:

- With a delay of $\delta := 3\lceil \log q \rceil - 1$ rounds this tree can reach a state such that in the next round $\delta + 1$ the information of the source can be propagated simultaneously over all outgoing edges α_i.
- If a broadcast schedule for $\text{Init}_{q,t}$ finishes by round t then none of these edges can propagate the information before round $\delta + 1$.

Connect each leaf of the tree with a node of $V_{A,1}$ of the graph G defined above. Let $t := 3\lceil \log q \rceil + 3$ and connect the root of $\text{Init}_{q,t}$ with the source v_0. Then this new graph G' has a broadcast schedule of length at most $3 \log q + 4$ iff M contains a matching. The resulting graph has degree 5, but is not necessarily planar since edges from $V_{A,1}$ to V_M may have to be crossed by the edges leaving $\text{Init}_{q,t}$. By additional effort G' can be modified to decrease the node degree to 3.

Observe that a graph $G = (V, E)$ with a single source cannot be informed within less then $\log |V|$ rounds. Our construction yields that the problem to find a minimal broadcast schedul is $\mathcal{NP}$-complete for logarithmic deadlines.

Proof of Theorem 5: Let A, B, C, M be an instance of 3DM with $|A| = q$. The corresponding graph G with unique source s consists of 4 levels (see Figure 4.13):

The first level consists of the source s connected with the root of an initializer that duplicates the number of sources. The second level consists of some subgraphs A_i simulate the nodes of $V_{A,1}$ and V_M in the proof of Theorem 4. The third level consists of the edges connecting a leaf of the subgraph A_i with an input node of the subgraph B_j and an input node of C_k iff $(i, j, k) \in M$. This means that the leaves of the subgraphs A_i simulate the nodes of V_M. The fourth level consists of some subgraphs B_j and C_k which simulate the nodes of V_B and V_C. The deadline is chosen as $T^* := 10 + 3 \cdot \log q$. Observe that G has maximum degree 3.

Lemma 3 *Let $(G = (V, E), V_0, T^*)$ represent an instance $M \subseteq A \times B \times C$ of the restricted 3DM-problem. Then G can be informed within T^* rounds iff M contains a matching.*

Proof: Let $V' := V \setminus (\bigcup_{i=1}^{p} (V_{B_i} \cup V_{C_i}))$ and $G' := (V', E')$ with $E' := \{(u, v) \in E \mid u, v \in V'\}$. Then for each schedule S' for G' with deadline T^* it holds: At most one leaf of each A_i receives the

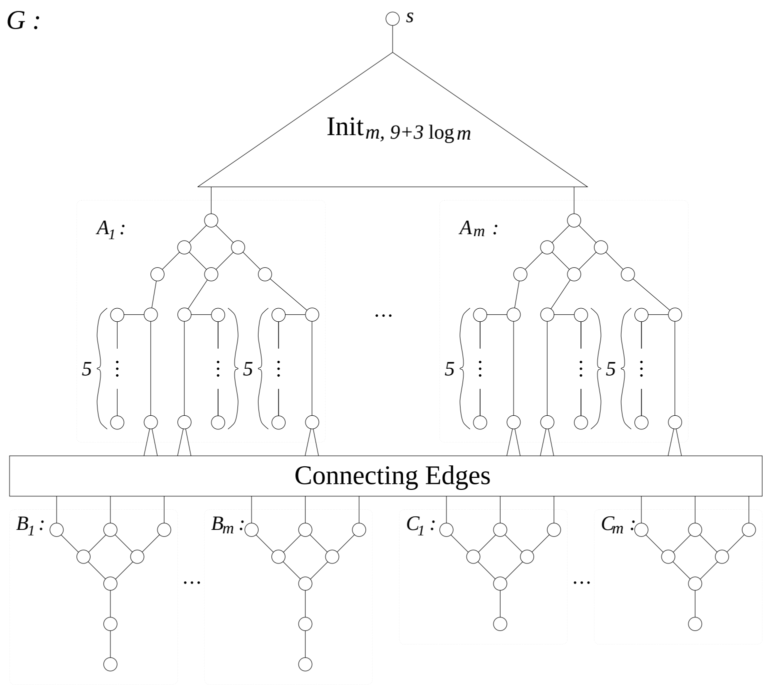

Figure 4.13: A broadcast graph corresponding to an instance of the 3DM problem

information in round $T^* - 5$ and at least two leaves in a round $T^* - 3$ or later. So the claim follows similar to the proof of Lemma 2. □

This proves Theorem 5. □

4.7 SB of Planar Graphs is $\mathcal{NP}$–Complete

To achieve planarity in the single source case we construct a direct reduction of 3SAT. The reduction will use the following restricted version of the satisfiability problem: Let F be a Boolean formula such that for each variable $x_i \in U$ there are at most 5 clauses in F that contain either x_i or $\overline{x_i}$ and each clause $C_i \in F$ satisfies $|C_i| = 3$. This restricted version of 3SAT remains $\mathcal{NP}$–complete [GJ79].

Proof of Theorem 6: We reduce a given instance F of the restricted version of 3SAT with clauses $C_1 \dots C_m$ and variables $x_1 \dots x_n$ to a graph consisting of 5 levels (Figure 4.14).

1. The first level consists of the source s connected with the root of an initializer $\mathrm{Init}_{n,t}$ with $t := T^* - 1$.

2. The leaves of the initializer are connected with the inputs of n parallel guess–graphs $\mathrm{Exist}_{t'}$ with $t' := T^* - 3\log n$. Let G_1 be the subgraph consisting of the source, the initializer and the guess–graphs. Let S be an arbitrary schedule for G_1 achieving the deadline such that the output edges Out_1 and Out_2 of the guess–graphs (Figure 4.6) can be used without artificial delay. Then one of the edges $\mathrm{Out}_1, \mathrm{Out}_2$ can be used in round $3\log n + 3$ and the other one in round $3\log n + 5$. We will denote this behavior by the time tuples $(0, 2)$ or $(2, 0)$ relative to the dawn of the successors, and interpret these pairs as codings for true and false setting of the corresponding variable.

3. The third level consists of n parallel multiplicators $\mathrm{Mult}_{5,t''}$, which are used to increase the number of binary encodings chosen in level 2.

4. On this level we send these binary encodings to the subgraphs of the last level which represent the clauses $C_1, \dots, C_m$. In this network we will use a special coding and decoding network to realize a crossing of the relative time tuples $(0, 2)$ and $(2, 0)$. We call a pair of coding and decoding networks

a double crossing. These components are combined in an allocation network depth $228 \cdot (5n)$ and size $O(n^2)$.

5. Finally, we connect the output nodes of the allocation network to the OR–graphs C_i.

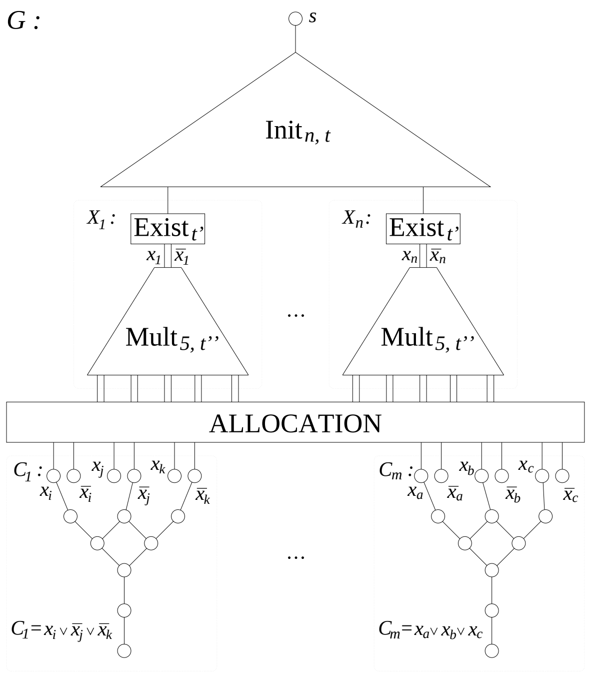
G :
s
Init$_{n, t}$
X_1 :
Exist$_{t'}$
x_1 $\overline{x}_1$
X_n:
Exist$_{t'}$
x_n $\overline{x}_n$
Mult$_{5, t''}$
Mult$_{5, t''}$
...
ALLOCATION
C_1 :
x_i $\overline{x}_i$ x_j $\overline{x}_j$ x_k $\overline{x}_k$
C_m :
x_a $\overline{x}_a$ x_b $\overline{x}_b$ x_c $\overline{x}_c$
...
$C_1 = x_i \vee \overline{x}_j \vee \overline{x}_k$
$C_m = x_a \vee x_b \vee x_c$

Figure 4.14: A planar broadcasting graph corresponding to an instance of the restricted version of 3SAT

For the resulting graph G with source s the deadline is set to dawn$(C_1) + 4$, i.e.

$$T^* := 3 \log n - 1 + 39 \cdot 3 + 2 + 228 \cdot d + 5.$$

Lemma 4 *Let (G, V_0, T^*) represent an instance F of the restricted version of 3SAT. Then G can be finished within T^* rounds iff there is a satisfying truth assignment for F.*

Proof: The claim follows from the fact that a schedule can only achieve the deadline iff for each subgraph C_i there exists at least one input node v that is connected to an inner node of C_i and v receives the information in round $T^* - 5$. □

This completes the proof of theorem 6. □

4.8 Efficient Algorithms for Decomposable Graphs

We start with a generalization of the broadcast problem. So far, each source node has got the broadcast information in round 0. In the more general case, a source v may get the information in an arbitrary round $\sigma(v) \geq 0$. Furthermore, for each node v there is an individual deadline $\rho(v)$ instead of a global deadline T^* identical for all nodes. This generalization may be of less interest with respect to practical applications. Nevertheless, it is necessary in order to apply an approach based on graph decompositions, as it has been for several other graph theoretical decision and optimization problems.

Definition 10 General Broadcast Problem **[GB]**:
Given a graph $G = (V, E)$ and two partial functions $\sigma, \rho : V \rightarrow \mathbb{N}$, decide whether there exists a broadcast schedule $E_1, E_2, \ldots,$ with

$$\begin{aligned} V_i &= V_{i-1} \cup \{ v \mid (u,v) \in E_i \text{ and } u \in V_{i-1} \} \cup \{ v \in V \mid \sigma(v) = i \}, \\ E_i &\subseteq \{ (u,v) \in E \mid u \in V_{i-1} \} \quad \textit{and} \quad \forall u \in V_{i-1} : |E_i \cap (\{u\} \times V)| \leq 1 \end{aligned}$$

such that $\forall\, v \in V \;:\; v \in V_{\rho(v)}$ *if* $\rho(v)$ *is defined.* □

The set of sources V_s is given by the domain of σ. The GB–problem can be solved similarly to the strategy of Lemma 1. Note that this problem is also $\mathcal{NP}$–complete. If we restrict the GB–problem to graphs $G = (V, E)$ with maximal degree d the number of different choices of step 1 is bounded by

$$\prod_{v \in V} d_v \leq \left(\frac{2 \cdot |E'|}{|V|}\right)^{|V|} \leq (d+1)^{|V|}$$

where d_v denotes the degree of $v \in V$ in G'. Thus for a graph $G = (V, E)$ the GB–problem can be solved in time $\mathrm{O}\Big(|V|^2 \cdot \Big(2 \cdot \frac{|E|+|V|}{|V|}\Big)^{|V|}\Big)$. Let V' denote the set of nodes of degree 1 in G, then restricted to a node $v \in V'$ we can simplify step 1 of the algorithm given in the proof of Lemma 1 as follows: if v is a source with $\sigma(v) \leq \rho(v)$ we choose the edge $\{v', v\}$, and $\{v, v'\} \in E$ else. Thus, there are at most

$$\prod_{v \in V \setminus V'} \delta'(v) \leq \left(2 \cdot \frac{|E| + |V| - |V'|}{|V| - |V'|}\right)^{|V|-|V'|} \leq (d+1)^{|V|-|V'|}$$

different choices.

Lemma 5 *Let V' denote the set of nodes of G of degree 1. Then the GB–problem for G can be solved in time*

$$\mathrm{O}\Big(|V|^2 \cdot \Big(2 \cdot \frac{|E|+|V|-|V'|}{|V|-|V'|}\Big)^{|V|-|V'|}\Big)\,.$$

The strategy above can be parallelized in a simple way.

Lemma 6 *The GB–problem restricted to graphs $G = (V, E)$ with maximum degree d can be solved by a CRCW–PRAM with $O((2 \cdot (|E| + |V|)/|V|)^{|V|})$ processors in time $O(|V|^2)$.*

These strategies will be used as basic routines for the components of a graph.

Proof of Theorem 7: Let $H = (V_H, E_H)$ be a (κ, μ, c)–edge decomposition tree of a graph $G = (V, E)$ with $V_H = \{G_1, \ldots, G_k\}$. Figure 4.15 shows such a component G_2 which is connected to three components G_1, G_3 and G_4. A component generated by H may fall into several connected subgraphs which we will call **subcomponents**.

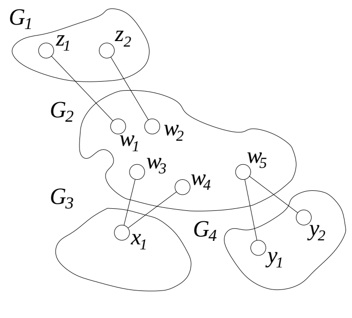

Figure 4.15: A node G_2 of an edge decomposition tree and its neighbors

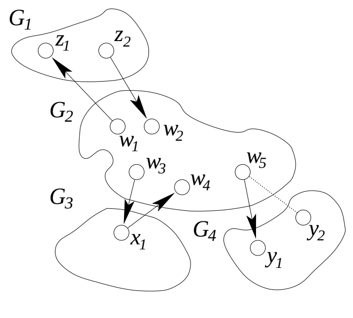

Figure 4.16: A possible information flow within a broadcast schedule: from G_2 to other components in one or in both directions; some edges may not be used.

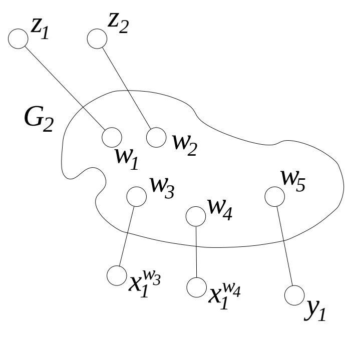

Figure 4.17: The minimum deadline of the general broadcast problem for this graph is used to calculate the minimum broadcast time for the graph above.

Let $\mathcal{G}_i := \{ G_i^1, \ldots, G_i^{c_i} \}$ with $\boldsymbol{G_i^a} = (V_i^a, E_i^a)$ be the set of subcomponents of the component G_i and define cut(G_i^a) as the set of edges of cut(G_i) with one endpoint in G_i^a. Define

$$\begin{aligned} \text{cut}(G_i^a, G_j) &:= \text{cut}(G_i, G_j) \cap \text{cut}(G_i^a) \qquad \text{and} \\ \text{cut}(G_i^a, G_j^b) &:= \text{cut}(G_i^a, G_j) \cap \text{cut}(G_j^b) \qquad \text{and} \\ \text{border}(G_i^a) &:= \{ v \mid \{u,v\} \in \text{cut}(G_i^a) \text{ and } u \in V_i^a \} . \end{aligned}$$

To describe a broadcasting schedule $\mathcal{E}$ of G, each edge $\{u, v\}$ of G is labeled by $(\tau(u, v), r(u, v))$. The first value $\tau(u, v)$ denotes the round this edge is used and the second $r(u, v)$ the direction ($[u \to v]$ or $[v \to u]$). If this edge is not used we set $\tau(u, v) := -1$.

If we restrict $\mathcal{E}$ to cut(G_i^a) we consider as the first round $\boldsymbol{\tau(G_i^a)}$ when a node of G_i^a gets the broadcast information. For each edge $\{u, v\}$ with $\tau(u, v) \geq 0$ the **relative round**

$$\boldsymbol{\widehat{\tau}(u,v)} := \tau(u,v) - \tau(G_i^a) .$$

If the edge $\{u, v\}$ is not used we set $\widehat{\tau}(u, v) := -1$. Let $\boldsymbol{\tau(G_i^a, G_j^b)}$ be the first round an edge of cut(G_i^a, G_j^b) is used. If no edge in cut(G_i^a, G_j^b) is used $\tau(G_i^a, G_j^b) := -1$. Similarly define

$$\boldsymbol{\widehat{\tau}(G_i^a, G_j^b)} := \tau(G_i^a, G_j^b) - \tau(G_i^a)$$

if $\tau(G_i^a, G_j^b) \geq 0$, else let $\widehat{\tau}(G_i^a, G_j^b) := -1$.

The following two lemmata show that with the help of the concept of relative rounds $\widehat{\tau}$ the number of possible protocols of information exchange between two components can be bounded quite substantially. This property will be basic for the time efficiency of the algorithm.

Lemma 7 *If $\mathcal{E}$ is a busy broadcast schedule then for all $\{u, v\} \in$ cut(G_i^a, G_j^b) holds: the numbers $\widehat{\tau}(u, v)$ and $\widehat{\tau}(G_i^a, G_j^b)$ are smaller than* $|\text{cut}(G_i^a)| + |V_i^a| - 1 \leq \kappa + \mu - 1$.

Proof: Let $G_i' = (V_i^a \cup \text{border}(G_i^a), E_i^a \cup \text{cut}(G_i^a))$ be the extended component of $G_i^a = (V_i^a, E_i^a)$. Note that the number of nodes of G_i' is bounded by $\kappa + \mu$. Hence, the minimal broadcasting time of G_i' is trivially bounded by $\kappa + \mu - 1$.

The claim follows from the fact that $\widehat{\tau}(G_i^a, G_j^b) + \widehat{\tau}(u, v)$ denotes the delay between the first node of G_i^a being informed and the round when the information is sent across $\{u, v\}$. □

For neighboring components G_i, G_j define a **relative state** as a tuple

$$\boldsymbol{\gamma_{i,j}} := \Big[(\widehat{\tau}(e), r(e)) \mid e \in \text{cut}(G_i, G_j) \Big]$$

Figure 4.16 illustrates a complex information flow between a component and its neighbors.

The **relative surface** $\boldsymbol{\Gamma_{i,j}^r}$ is the set of all possible relative states $\gamma_{i,j}$ of busy broadcast schedules. A **state** $S_{i,j}$ between two neighbors G_i and G_j is a vector consisting of a relative state $\gamma_{i,j}$ and a starting round $\tau(G_i^a)$ for all subcomponents of G_i with cut$(G_i^a, G_j) \neq \emptyset$. Let $\boldsymbol{\Gamma_{i,j}}$ be the set of all possible states $S_{i,j}$ that may appear in busy schedules. A **state** S_i of a component G_i is a vector consisting of the starting round $\tau(G_i^a)$ for all subcomponents of G_i and tuples $\gamma_{i,j}$ for all neighboring components of G_i. As above, let $\boldsymbol{\Gamma_i}$ be the set of all possible states S_i that may appear in busy schedules.

Lemma 8 *For a component G_i with cut-size $|\text{cut}(G_i)| \leq \kappa_i$, size $|V_i| \leq \mu_i$ and $c_i = \text{cc}(G_i)$ subcomponents, the size of Γ_i is bounded by*

$$\gamma(\kappa_i, \mu_i, c_i) := |V|^{c_i} \cdot (2(\kappa_i + \mu_i))^{\kappa_i} .$$

Proof: $|\Gamma_i| \leq \prod_{G_i^a \in \mathcal{G}_i} |V| \cdot \prod_{\text{cut}(G_i, G_j) \neq \emptyset} |\Gamma_{i,j}| \leq |V|^{c_i} \cdot (2(\kappa_i + \mu_i))^{\kappa_i}$. □

Note that $|\Gamma_{i,j}| \leq |\Gamma_i|$.

The following strategy solves the minimum broadcasting time problem for graphs $G = (V, E)$ with a given (κ, μ, c)–edge decomposition tree $H = (V_H, E_H)$. Let $\boldsymbol{\Delta(G_i, S_i)}$ denote the minimal schedule length of the local broadcast problem for the graph G_i and external information exchange as specified by state S_i ($= \infty$ if there is no schedule for state S_i). Observe that this value is independent of the structure of G outside of G_i.

Step 1: For each component $G_i = (V_i, E_i)$ and each state $S_i \in \Gamma_i$ determine $\Delta(G_i, S_i)$.

For each edge $\{v, w\} \in \text{cut}(G_i)$ with $w \notin G_i$ and $\widehat{\tau}(v, w) \geq 0$ generate a new node $w^{[v]}$ and a new edge $\{w^{[v]}, v\}$, where v is the only neighbor of $w^{[v]}$. Define $V(S_i)$ as the set of these new nodes $w^{[v]}$ and $E(S_i)$ as the set of new edges $\{w^{[v]}, v\}$. We define a local GB-problem with respect to G_i and S_i as follows:

$$\begin{aligned} G'_i &:= (V_i \cup V(S_i), E_i \cup E(S_i)) \\ \sigma^i(w^{[v]}) &:= \tau(v,w) - 1 \quad \text{if } S_i \;\; r(v,w) = [w \to v] \,, \\ \rho^i(w^{[v]}) &:= \tau(v,w) \qquad\quad \text{if } S_i \;\; r(v,w) = [v \to w] \,. \end{aligned}$$

For $v \in V_i$ define $\sigma^i(v) := \sigma(v)$ and $\rho^i(v) := \rho(v)$.

The construction of G'_i for the example given in Figure 4.16 is illustrated in Figure 4.17.

Step 2: Let $G_{i,0}, \ldots, G_{i,\ell_i}$ denote the neighbors of G_i.

Choose an arbitrary component G_r and declare G_r as the root of H. Let $G_{i,0}$ be the father of G_i in H according to the orientation with respect to G_r. Let G^*_i denote the subgraph of G containing G_i and all its descendents. Evaluate the function $\Delta(G^*_i, S_{i,0})$ for all G_i and $S_{i,0}$ starting with the leaf components of H.

Lemma 9 *Let $G_{i,1}, \ldots, G_{i,\ell_i}$ denote the sons of G_i and let $S_{i,j}$ be a state connecting G_i and $G_{i,j}$. The minimal deadline for the general broadcast problem for G^*_i with respect to external information exchange $S_{i,0}$ can be computed as*

$$\Delta(G^*_i, S_{i,0}) = \min_{\substack{S_i = (\tau_1, \ldots, \tau_{cc(G_i)},\ \gamma_{i,0}, \ldots, \gamma_{i,\ell_i}) \in \Gamma_i \\ \text{with } S_{i,0} \subseteq S_i}} \max\,(\{ \min_{S_{j,i} \in \Gamma_{j,i}(S_i)} \Delta(G^*_{i,j}, S_{j,i}) | j \in [1 \ldots l_i]\} \cup \{\Delta(G_i, S_i)\})$$

with

$$\begin{aligned} \Gamma_{j,i}(S_i) := \{(&\left[\tau(G^b_{j,i}) = \tau_k + \widehat{\tau}(G^k_i, G^b_{j,i}) - \widehat{\tau}(G^b_{j,i}, G^k_i) \mid \text{cut}(G^k_i, G^b_{j,i}) \neq \emptyset\right], \\ &\left[(\widehat{\tau}(e) - \widehat{\tau}(G^k_i, G^b_{j,i}) + \widehat{\tau}(G^b_{j,i}, G^k_i), r(e)) \mid e \in \text{cut}(G^k_i, G^b_{j,i}) \textit{ and} \right. \\ &\qquad\qquad \left. (\widehat{\tau}(e), r(e)) \in \gamma_{i,j}\right])\} \,. \end{aligned}$$

Proof: This property can be shown by induction on the depth of the subgraphs G_i.

If G_i is a leaf of H then $S_i = S_{i,0}$ thus $\Delta(G^*_i, S_{i,0}) = \Delta(G_i, S_i)$. The claim follows directly from the definition of $\Delta(G_i, S_i)$.

Let h_i be the depth of the subgraphs G_i. Assume that the claim holds for each subtree of depth less than h_i and each state of these subtrees, in particular for each son $G_{i,k}$ of G_i and each state $S_{k,i}$.

Given a state $S_i = (\tau_1, \ldots, \tau_{cc(G_i)},\ \gamma_{i,0}, \ldots, \gamma_{i,\ell_i}) \in \Gamma_i$ of G_i then $\Gamma_{j,i}(S_i)$ denotes the set of corresponding state $S_{j,i}$. Thus $\Delta(G^*_j, S_{j,i})$ denotes the minimal deadline of G^*_j with respect to external information exchange $S_{j,i}$ and

$$\max\,(\max_{j \in [1..l_i]} \{ \min_{S_{j,i} \in \Gamma_{j,i}(S_i)} \Delta(G^*_{i,j}, S_{j,i})\} \quad \cup \ \{\Delta(G_i, S_i)\}\ )$$

the minimal deadline of G^*_i with respect to external information exchange S_i.

The claim of the lemma follows since we minimize over all possible states of G_i that may appear in busy schedules. □

Therefore, $\Delta(G^*_r) = \Delta(G^*_r, \lambda)$ denotes the minimal schedule length for the graph G itself.

The correctness of step 1 follows directly from the definition of a surface and the definition of the general broadcast problem. The correctness of step 2 follows from Lemma 9.

According to Lemma 5 the computation of $\Delta(G_i, S_i)$ requires at most $\mathrm{O}((\mu_i + \kappa_i)^2 \cdot (2 \cdot (|E| + |V|)/|V|)^{\mu_i})$ steps. From Lemma 8 it follows that step 1 can be executed in time

$$\sum_{G_i} \gamma(\kappa_i, \mu_i, c_i) \cdot \mathrm{O}\Big((\mu+\kappa)^2 \cdot \Big(2 \cdot \frac{|E|+|V|}{|V|}\Big)^{\mu}\Big) \ \leq\ \mathrm{O}\Big(|V|^c \cdot (2(\kappa+\mu))^{\kappa} \cdot (\mu+\kappa)^2 \cdot \Big(2 \cdot \frac{|E|+|V|}{|V|}\Big)^{\mu}\Big) \,.$$

The computation of $\Delta(G_i^*, S_{i,0})$ is independent of the remaining structure of G. Note that given S_i $\widehat{\tau}(G_i^a, G_j^b) = \min_{e \in \text{cut}(G_i^a, G_j^b)} \widehat{\tau}(e)$, so $|\Gamma_{i,j}(S_i)| \le (\mu + \kappa)^c$. Thus given all values $\Delta(G_i, S_i)$ and $\Delta(G_{i,j}^*, S_{j,i})$, the computation of all $\Delta(G_i^*, S_{i,0})$ can be executed in time

$$\mathrm{O}(\gamma(\kappa_i, \mu_i, c_i) \cdot \ell_i \cdot (\mu + \kappa)^c \cdot c^2) \; .$$

Summing up over all G_i gives the bound

$$\begin{aligned} \sum_{G_i} \mathrm{O}(\gamma(\kappa_i, \mu_i, c_i) \cdot \ell_i \cdot (\mu + \kappa)^c \cdot c^2) &\le \mathrm{O}\Big(\gamma(\kappa, \mu, c) \cdot (\mu + \kappa)^c \cdot c^2 \cdot \sum_{G_i} \ell_i\Big) \\ &\le \mathrm{O}(|E| \cdot \gamma(\kappa, \mu, c) \cdot (\mu + \kappa)^c \cdot c^2) \; . \end{aligned}$$

This finishes the proof of Theorem 7 and 8. □

By using tree contraction methods the evaluation of the Δ-function can also be done in parallel requiring only a logarithmic number of iterations which yields Theorem 9. The details are described in [Rei91a].

4.9 Node Separation

The same technique with a slightly worse time bound due to a larger number of states also works for node decompositions of graphs.

Proof of Theorem 10: Let $H = (V_H, E_H)$ be a (κ, μ, c)–node decomposition tree of the graph $G = (V, E)$ with $V_H = \{G_1, \ldots, G_k\}$. Figure 4.18 shows a component G_2 which is connected to three other components G_1, G_3, and G_4.

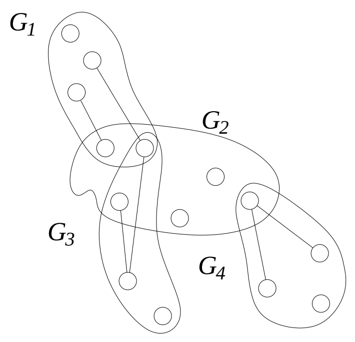

Figure 4.18: A node G_2 of an edge decomposition tree and its neighbors

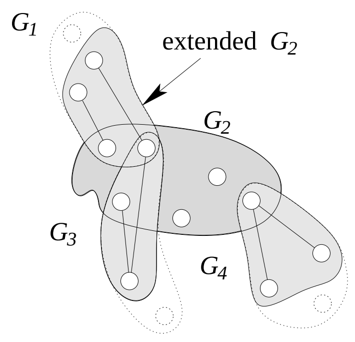

Figure 4.19: The extended component of G_2

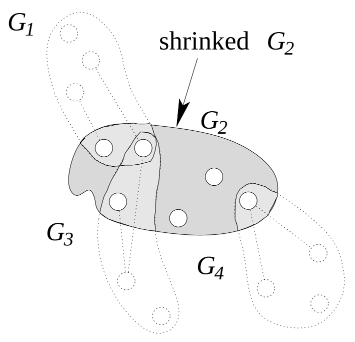

Figure 4.20: The minimum deadline of the general broadcast problem of this graph is used to calculate the minimum broadcast time for the graph above.

Let $\mathcal{G}_i := \{ G_i^1, \ldots, G_i^{c_i} \}$ with $\boldsymbol{G_i^a} = (V_i^a, E_i^a)$ be the set of subcomponents of the G_i and define cut(G_i^a) as the set of nodes of $V_i^a \cap \text{cut}(G_i)$.

$$\begin{aligned} \text{cut}(G_i^a, G_j) &:= \text{cut}(G_i, G_j) \cap \text{cut}(G_i^a) \qquad \text{and} \\ \text{cut}(G_i^a, G_j^b) &:= \text{cut}(G_i^a, G_j) \cap \text{cut}(G_j^b) \qquad \text{and} \\ \text{border}(G_i^a) &:= \{ v \notin V_i^a \mid \{v, u\} \in E \quad \text{and} \quad u \in \text{cut}(G_i^a) \} \; . \end{aligned}$$

For a node $u \in V$ let $\tau(u)$ be the round u gets the information, and for an edge $e_l \in E$ let $\tau(e_l)$ the round when e_l is used. To describe a broadcasting schedule of a graph G, each node u of G is labeled by a vector of rounds $\boldsymbol{\gamma_u} := (\tau(u), \widehat{\tau}(e_1), \ldots, \widehat{\tau}(e_{\eta(u)}))$, where $e_1, \ldots, e_{\eta(u)}$ denote the edges which are incident to u and $\boldsymbol{\widehat{\tau}(e_l)} := \tau(e_l) - \tau(u)$. Note that the values $\widehat{\tau}(e_l)$ are bounded by d. γ_u will be called the **state** of u. The **surface** $\boldsymbol{\Gamma_u}$ of a node $u \in V$ is the set of all possible states of u that may appear in busy broadcast schedules. Note that for a fixed $\tau(u)$ $|\Gamma_u|$ is bounded by $(d+1)^d$.

For a node $u \in V_i^a$ let $\boldsymbol{\widehat{\tau}(u)} := \tau(u) - \tau(G_i^a)$ where $\tau(G_i^a)$ denotes the first round a node of V_i^a receives the information.

Lemma 10 *For a busy broadcast schedule $\widehat{\tau}(u)$ is bounded by $(d-1) \cdot \kappa + \mu - 1$.*

Proof: For a component $G_i^a = (V_i^a, E_i^a)$ with $|V_i^a| \geq 2$ let $G_i' = (V_i^a \cup \text{border}(G_i^a), E_i^a \cup \{\{u,v\} \in E \mid u \in \text{cut}(G_i^a)\})$ be the extended component of G_i^a (see Figure 4.19). Note that the number of nodes of $G_i' = (V_i', E_i')$ is bounded by $(d-1) \cdot \kappa + \mu$. Thus the minimum broadcast time of G_i' is bounded by $(d-1) \cdot \kappa + \mu - 1$.

Let u and v be two nodes of a subcomponent of a component of G. Then the difference between the rounds when the two nodes u and v are informed is at most $(d-1) \cdot \kappa + \mu - 1$. □

A **state $\boldsymbol{\gamma_{i,j}}$** between two neighbors G_i and G_j is a vector of states γ_u, one for each node of $\text{cut}(G_i, G_j)$. The **surface $\boldsymbol{\Gamma_{i,j}}$** of a $\text{cut}(G_i, G_j)$ is the set of all possible states $\gamma_{i,j}$ that may appear in busy broadcast schedules. A **state $\boldsymbol{S_i}$** of a component G_i is a vector of states γ_u, one for each node of $\text{cut}(G_i)$. The **surface $\boldsymbol{\Gamma_i}$** of a component G_i is the set of all possible states S_i that may appear in busy schedules.

Lemma 11 *For a component G_i with cut-size $|\text{cut}(G_i)| \leq \kappa_i$, size $|V_i| \leq \mu_i$ and $c_i = \texttt{cc}(G_i)$ subcomponents, the size of Γ_i is bounded by $\gamma(\kappa_i, \mu_i, c_i) := |V|^{c_i} \cdot (((d-1) \cdot \kappa_i + \mu_i) \cdot (d+1)^d)^{\kappa_i}$.*

Proof: Define the **relative state $\boldsymbol{\gamma_u^r}$** $:= (\widehat{\tau}(u), \widehat{\tau}(e_1), \ldots, \widehat{\tau}(e_{\eta(u)}))$ and the **relative surface $\boldsymbol{\Gamma_u^r}$** as the set of all possible relative states γ_u^r that may appear in busy broadcast schedules. Then the **state $\boldsymbol{\overline{\gamma}_i}$** between two neighbors G_i and G_j is a vector

$$\overline{\gamma}_i := [\tau(G_i^a) \mid G_i^a \in \mathcal{G}_i], [\gamma_u^r \mid u \in \text{cut}(G_i)] \ .$$

The **surface $\boldsymbol{\overline{\Gamma}_i}$** of $\text{cut}(G_i)$ is the set of all possible states $\overline{\gamma}_i$ that may appear in busy broadcast schedules. Note that $|\overline{\Gamma}_i| = |\Gamma_i|$ and $|\Gamma_{i,j}| \leq |\Gamma_i|$. Hence,

$$|\Gamma_i| \leq \prod_{G_i^a \in \mathcal{G}_i} |V| \cdot \prod_{u \in \text{cut}(G_i^a)} |\Gamma_u^r| \leq |V|^{c_i} \cdot (((d-1) \cdot \kappa_i + \mu_i) \cdot (d+1)^d)^{\kappa_i} \ .$$

□

The following strategy solves the minimum broadcasting time problem for graphs $G = (V, E)$ with a given (κ, μ, c)–node decomposition tree $H = (V_H, E_H)$. Let $\boldsymbol{\Delta(G_i, S_i)}$ denote the minimal schedule length of the local broadcast problem for the graph G_i and external information exchange as specified by state S_i ($= \infty$ if there is no schedule for state S_i). Again this value is independent of the structure of G outside of G_i.

Step 1: For each component $G_i = (V_i, E_i)$ and each state $S_i \in \Gamma_i$ determine $\Delta(G_i, S_i)$.

Let $G_i' := (V_i \setminus \text{cut}(G_i),\ E_i \setminus \{\{u,v\} | u \in \text{cut}(G_i)\})$ be the shrunken component of G_i.

For each edge $\{v, w\} \in E_i$ with $v \in \text{cut}(G_i)$ generate a new node $w^{[v]}$ and a new edge $\{w^{[v]}, v\}$, such that v is the only neighbor of $w^{[v]}$. Define $V(S_i)$ as the set of these new nodes $w^{[v]}$, and $E(S_i)$ the set of new edges $\{w^{[v]}, v\}$. We define a local GB–problem with respect to G_i and S_i as follows:

$$\begin{aligned} G_i'' &:= (V_i' \cup V(S_i), E_i' \cup E(S_i)) \ , \\ \sigma^i(w^{[v]}) &:= \tau(v,w) - 1 \quad \text{if } \gamma_w \in S_i\ \widehat{\tau}(v,w) > 0 \ , \\ \rho^i(w^{[v]}) &:= \tau(w) \qquad\quad\ \text{if } \gamma_u \in S_i\ \widehat{\tau}(v,w) = 0 \ . \end{aligned}$$

For $v \in V_i'$ define $\sigma^i(v) := \sigma(v)$ and $\rho^i(v) := \rho(v)$.

Note that $|V_i''| \leq \mu_i + (d-2) \cdot \kappa_i$. The construction of G_i'' is illustrated in Figure 4.20.

Step 2: Let $G_{i,0}, \ldots, G_{i,\ell_i}$ denote the neighbors of G_i.

Choose an arbitrary component G_r and declare G_r as the root of H. Let $G_{i,0}$ be the father of G_i in H according to the orientation with respect to G_r. Let G_i^* denote the subgraph of G containing G_i and all its descendants. Evaluate the function $\Delta(G_i^*, S_{i,0})$ for all G_i and $S_{i,0}$ starting with the leaf components of H.

Lemma 12 *Let $G_{i,1}, \ldots, G_{i,\ell_i}$ denote the sons of G_i and let $S_{i,j}$ be a state connecting G_i and $G_{i,j}$. The minimal deadline for the general broadcast problem for G_i^* with respect to external information exchange $S_{i,0}$ can be computed as*

$$\Delta(G_i^*, S_{i,0}) \;=\; \min_{\substack{S_i = (\gamma_u \;\mid\; u \in \text{cut}(G_i)\;) \;\in\; \Gamma_i \\ \text{with } S_{i,0} \subseteq S_i}} \; \max \; \left(\left\{\, \Delta(G_{i,j}^*, S_{j,i}) \,\middle|\, j \in [1 \ldots l_i] \right\} \;\cup\; \{\Delta(G_i, S_i)\}\right)$$

with $S_{j,i} \;:=\; (\gamma_u | u \in \text{cut}(G_i, G_{i,j})) \;\subseteq\; S_i$.

The proof is almost identical to the one of Lemma 9. □

As in the case of edge-decomposition $\Delta(G_r^*) \;=\; \Delta(G_r^*, \lambda)$ denotes the minimal schedule length for the graph G itself.

The correctness of step 1 follows directly from the definition of a surface and the correctness of step 2 from Lemma 12. According to Lemma 5 the computation of $\Delta(G_i, S_i)$ requires at most $\mathrm{O}(((d-2)\cdot\kappa_i + \mu_i)^2 \cdot (d+1)^{\mu_i})$ steps. From Lemma 11 follows that step 1 can be executed in time

$$\sum_{G_i} \gamma(\kappa_i, \mu_i, c_i) \cdot \mathrm{O}\Big(((d-1)\cdot\kappa + \mu)^2 \cdot (d+1)^{\mu}\Big) \;\leq\; \mathrm{O}\Big(|V|^c \cdot ((d-1)\cdot\kappa + \mu)^{\kappa+2} \cdot (d+1)^{\mu + d\cdot\kappa}\Big) \,.$$

The computation of $\Delta(G_i^*, S_{i,0})$ is independent of the remaining structure of G. Note that for fixed $S_{i,0}$ the number for $S_{i,j}$ that may appear in a busy broadcast schedule is bounded by $2^{(d-1)\cdot\kappa}$. Thus given all values $\Delta(G_i, S_i)$ and $\Delta(G_{i,j}^*, S_{j,i})$ the computation of all $\Delta(G_i^*, S_{i,0})$ can be executed in time $\mathrm{O}(\gamma(\kappa_i, \mu_i, c_i) \cdot \ell_i \cdot 2^{(d-1)\cdot\kappa})$. Summing up over all G_i gives the bound

$$\sum_{G_i} \mathrm{O}(\gamma(\kappa_i, \mu_i, c_i) \cdot \ell_i) \;\leq\; \mathrm{O}\Big(\gamma(\kappa, \mu, c) \cdot \sum_{G_i} \ell_i\Big) \;\leq\; \mathrm{O}(|V| \cdot 2^{(d-1)\cdot\kappa} \cdot \gamma(\kappa, \mu, c)).$$

All together, we get a total time of

$$\mathrm{O}\Big(|V|^{c+1} \cdot ((d-1)\cdot\kappa + \mu)^{\kappa+2} \cdot (d+1)^{\mu + d\cdot\kappa} \cdot 2^{d\cdot\kappa}\Big) \,.$$

□

Again the evaluation of the Δ-function can also be done in parallel with a logarithmic number of iterations, which gives Theorem 11.

4.10 Conclusions

We have shown that the single source broadcasting problem remains hard for planar networks of bounded degree if the internal connectivity is high, that means there is no edge- or node-decomposition with components of small size. On the other hand, even a much more general version with many sinks and individual deadlines can be solved efficiently on graphs that can be decomposed nicely.

Thus one can conclude that generating optimal broadcast schedules is a difficult task in general. The intuition that this must be due to a complex structure of the network which gives a lot of freedom designing a schedule has been verified by a rigorous proof. Most interestingly, such structures can already occur in bounded degree planar graphs.

Chapter 5

On the Inapproximability of Broadcasting

5.1 Introduction

We have seen that the exact solution of the Broadcasting problem is combinatorically infeasible, if $\mathcal{P} \neq \mathcal{NP}$. In this chapter we discuss the computational complexity of approximating broadcasting time. An approximation algorithm for broadcasting is an algorithm that on input (G, V_0) of a graph G and a set of sources outputs a broadcasting schedule S. For the performance quality we distinguish additive and multiplicative approximation schemes.

If an polynomial time algorithm approximates a problem by an **additive term** $A(n)$, then the performance $P(x)$ (broadcasting time $b_S(G, V)$) of the output of an instance x (a graph G and a source set V) of size n (the number of nodes in G) differs from the optimal solution Opt$(x) = b(G, v)$ at most by $A(n)$, i.e. $|P(x) - \text{Opt}(x)| \leq n$. The approximation algorithm achieves a **multiplicative approximation** by $M(n)$, if $M(x) \leq M(n)\text{Opt}(x)$ for all n and all instances of size n.

Thus, the $\mathcal{NP}$-hardness results of Theorem 6 can be translated to an inapproximability result: Consider the set of graphs resulting from the reduction described in Theorem 6. The broadcasting time of these instances is either 4 or 3. The Theorem proves that it is $\mathcal{NP}$-hard to distinguish between these two sets of instances. Hence, if an polynomial time approximation algorithm for broadcasting with multiple sources exists with multiplicative approximation ratio $\frac{4}{3} - \epsilon$ for $\epsilon > 0$, then it could make this distinction and $\mathcal{P} = \mathcal{NP}$ would follow. This inapproximability factor increases to $\frac{3}{2} - \epsilon$ when we apply the result of [Mid93].

Such a direct transfer of lower bounds from decision problems to inapproximability results is an unusual proof technique. Recall that this trick does not work with single source broadcasting, since here broadcasting time increases by the size of the graph. For a comparison the same technique only implies a trivial constant additive inapproximability bound of $1 - \epsilon$ for $\epsilon > 0$.

In this chapter we will concentrate on additive and multiplicative inapproximability bounds for single source broadcasting (SB) and additive bounds for multiple source broadcasting (MB). This chapter is organized as follows. In the next section we will present previous work and state the main results of this chapter. In section 5.3 we repeat some notation of the last chapter and introduce the sub-graphs used in this chapter. In section 5.4 we prove the multiplicative inapproximability of $\frac{3}{2} - \epsilon$ for SB. Then, in section 5.5 we present a tight lower additive approximation bound for SB. We will show in section 5.6 additive inapproximability bounds for multiple and single source broadcasting in ternary graphs. In the last section of this chapter we will conclude these results.

5.2 Previous Work

Broadcasting is the task of disseminating information from one or many *sources* to all members of a communication network. We envisage a static network where it makes sense to compute broadcast strategies offline. We already introduced the telephone model as one of the simplest timing models in Definition 1. A straight-forward generalization, the *postal model* [BNGNS98, BNK94], modifies the *edge delay* of an edge and *switching time* of a node. The edge delay delays the information of a node along this edge. The switching time is the time span a node needs to start the the next transmission to a neighbor. Again nodes can participate in only one call at each time. In *heterogeneous* networks every node and edge may have different timing behavior unlike in *homogeneous* networks where the ratio between the network's edge delay and switching time describes the *latency* of the network. Choosing a latency of 1 leads back to the telephone model. An interesting special case is the *open-path model* where the edge delay is zero and the switching time is 1. Clearly, if broadcasting with respect to one of the simplest communication models, the telephone model, is infeasible then the situation for the heterogeneous model cannot be better.

5.2.1 Approximation algorithms

In [Rav94] it is shown that broadcast time in the telephone model can be approximated within a factor of $O(\frac{\log^2 n}{\log\log n})$ given a network of n nodes. Bar-Noy et al. [BNGNS98] improve these techniques and present a polynomial-time approximation algorithm for the single source broadcasting problem with an approximation factor of $O(\log n)$ for the more general postal model.

For graphs with bounded tree-width with respect to the standard tree decomposition the broadcast time can be even approximated within $O(\frac{\log n}{\log\log n})$ [MRS$^+$98].

For broadcast time in the telephone model there exists an additive $O(\sqrt{n})$-approximation algorithm [KP95]. In particular there is a polynomial time bounded algorithm that for a graph with n nodes and broadcast time $b(G)$ constructs a broadcast schedule of length $b(G) + O(\sqrt{n})$.

5.2.2 Inapproximability results

In [BNGNS98] heterogeneous networks in the postal model were considered. For this timing model it is not possible to approximate the broadcast time within a factor of $3 - \epsilon$ for any $\epsilon > 0$ unless $\mathcal{P} \neq \mathcal{NP}$. The proof uses a reduction from set-cover using a latency varying from 0 to $O(1/\epsilon)$.

Allowing multiple sources can reduce the broadcast time to a constant. Yet for deadline 2 the decision problem is still $\mathcal{NP}$-complete even for planar graphs of small degree [Mid93]. As we already have discussed this $\mathcal{NP}$-completeness result for broadcasting in the telephone model implies an inapproximability factor of $\frac{3}{2}$. Note that this does not imply that graphs of higher broadcast time $b > 2$ cannot be distinguished from graphs with broadcast time $\frac{3}{2}b$.

5.2.3 New results

In this paper we investigate inapproximability bounds for the broadcast time in the telephone model for undirected communication networks. The lower bounds refer to the length of the optimal broadcast schedule and not necessarily to its computation. Clearly, computing good broadcast schedules is at least as hard as approximating the optimal length of valid broadcast schedules.

We solve the open problem recently addressed by [BNGNS98] whether there is a polynomial time algorithm approximating broadcasting by an additive constant. We state an inapproximability factor of $\frac{3}{2} - \epsilon$ for any $\epsilon > 0$ using a polynomial time reduction from set-cover.

Theorem 12 *For every $\epsilon > 0$ there exist graphs with n nodes with broadcast time at most b such that it is $\mathcal{NP}$-hard to distinguish those from graphs with broadcast time of at least $(\frac{3}{2} - \epsilon)b$.*

Then, we concentrate on the lower additive approximability bound. We prove that the approximation algorithm of Kortsartz et al. [KP95] is best possible up to a constant factor of the additive term. Unless $\mathcal{P} = \mathcal{NP}$ broadcasting cannot be approximated by a polynomial time bounded algorithm within an additive term of $c\sqrt{n}$ for some constant c. In particular we prove:

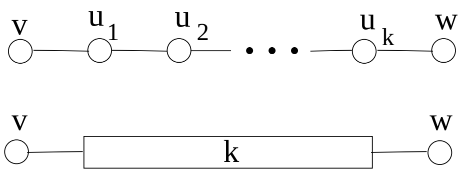

Figure 5.1: The chain and its symbol.

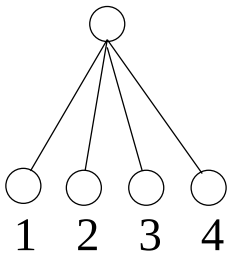

Figure 5.2: The star and a broadcast schedule.

Theorem 13 *For every $\epsilon > 0$ there exist graphs G with n nodes and broadcast time at most $b \in \Theta(\sqrt{n})$ such that it is $\mathcal{NP}$-hard to distinguish those from graphs with broadcast time at least $(\frac{57}{56} - \epsilon)b$.*

The proofs of these theorems use graphs with large degree. In the last chapter we have seen that there are situations where low degree simplifies the complexity of broadcasting to some extent. We show how to transfer these techniques to ternary graphs, which are undirected graphs with a degree of at most three. For ternary graphs and multiple sources we prove an additive lower bound of $\Theta(\log n)$.

Theorem 14 *It is $\mathcal{NP} - hard$ to distinguish ternary graphs $G = (V, E)$ with multiple sources and broadcast time $b \in \Theta(\log |V|)$ from those with broadcast time $b + c\sqrt{b}$ for any constant c.*

These indistinguishable ternary graphs have a polynomial number of information sources. We are also interested in the case of a single information source and ternary graphs. Modifying the proof of Theorem 14 we can state a lower additive inapproximability bound of $\Omega(\sqrt{\log n})$:

Theorem 15 *It is $\mathcal{NP}$-hard to distinguish ternary graphs $G = (V, E)$ with* single sources *and broadcast time $b \in \Theta(\sqrt{|V|})$ from those with broadcast time $b + c\sqrt{\log b}$ for some constant c.*

5.3 Notations and Basic Techniques

For a self-contained representation within this chapter we repeat Definition 1. Let $G = (V, E)$ be an undirected graph with a set of nodes $V_0 \subseteq V$, called the *sources*. The task is to compute the *broadcast time* $\boldsymbol{b(G, V_0)}$, the minimum length T of a *broadcast schedule* S. This is a sequence of sets of directed edges $S = (E_0, E_1, E_2, \ldots, E_T)$. Their nodes are in the sets $V_0, V_1, V_2 \ldots, V_T = V$, where for $i > 0$ we define $V_i := V_{i-1} \cup \{v \mid (u, v) \in E_i\}$. A broadcast schedule S induces a directed spanning forest with the sources as roots, and directed edges describing the information flow. S fulfills the properties

1. $E_i \subseteq \{(u, v) \mid u \in V_{i-1}, \{u, v\} \in E\}$ and
2. $\forall u \in V, \forall t \, : \, |E_i \cap (\{u\} \times V)| \leq 1$.

We define the *broadcast time of a graph G with sources V_0*, $\boldsymbol{b(G, V_0)}$, to be the minimum time required to complete broadcasting from nodes V_0. Let S be a broadcast schedule for (G, V_0), where $G = (V, E)$. The *broadcast time of a node $v \in V$* is defined as $b_S(v) = \min\{i \mid v \in V_i\}$.

Furthermore, as we have already discussed in the last chapter we can restrict our considerations to busy broadcast schedules. Here every processor tries to inform a neighbor in every step starting from the moment it is informed. When this fails it stops. By this time, all its neighbors are informed. Furthermore, every node is informed only once. Every broadcast schedule can be transformed into a busy schedule within polynomial time without increasing the broadcast time of any node. For a detailed proof we refer to [Sch00a]. From now on, every schedule is considered to be busy. In [BNGNS98] this argument is generalized to the postal model (the authors call busy schedules *not lazy*).

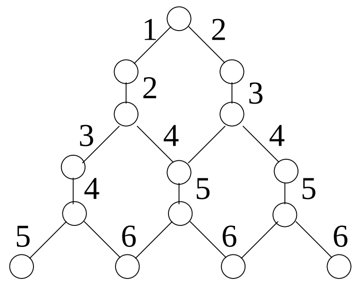

Figure 5.3: The ternary pyramid and a broadcast schedule.

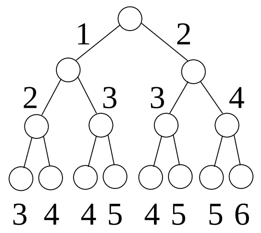

Figure 5.4: A complete binary tree and a broadcast schedule.

5.3.1 Sub-Graphs

As a basic tool we consider a *chain* (in the preceding chapter called timer, see Figure 5.1)

$$C_m(v,w) = (\{v = u_0, u_1 \ldots, u_m, w = u_{m+1}\}, \{\{v_i, v_{i+1}\}\})$$

starting at node v and ending at w with p interior nodes that are not incident to any other edge of the super-graph. A star consists of a central node c and n ray nodes v_i (Figure 5.2)

$$S_n = (\{c, v_1, \ldots, v_n\}, \{\{c, v_i\}\}) \ .$$

In Figure 5.3 a pyramid is shown. A pyramid consists of a honeycomb structure where the top node has equal distance to the base nodes $v_1, \ldots, v_n$.

The broadcasting behavior of these sub-graphs is the following (For simplicity we assume that the sources inform nodes of the sub-graph at first).

Lemma 13 *Let the ends v and w of a chain $C_k(v,w)$ be informed in time b_v and b_w, where $b_w - k \leq b_v \leq b_w$. Then, the total chain is informed in time $\lceil (b_v + b_w + k)/2 \rceil$.*

The center of star S_n can use $n!$ different busy broadcast schedules to inform all ray nodes. Their broadcast times can describe any permutations of $\{1, \ldots, n\}$.

The root of a complete binary tree of depth d informs $\binom{d}{i}$ leaves in time $d + i$.

For a pyramid with n base nodes the top node informs all but one base nodes in time $2n - 2$. The last base node is informed in time $2n - 3$.

5.4 Inapproximability by a Factor of $\frac{3}{2} - \epsilon$

In this section we show that the lower multiplicative inapproximability bound of $\frac{3}{2} - \epsilon$. The proof consists of a reduction to the set-cover problem.

Definition 11 (Set-cover) *Let S be a set of n elements and $F = \{S_1, \ldots, S_s\}$ a collection of subsets of S.* Set-cover *is the problem of selecting as few as possible subsets from $\mathcal{F}$ such that every point in S is contained in at least one of the selected subsets.*

Feige [Fei98] proved the following hardness result.

Theorem 16 ([Fei98]) *Unless $\mathcal{NP} \subseteq DTime(n^{O(\log\log n)})$, the set-cover problem cannot be approximated by a factor which is better than $\ln n$.*

Here, we will only use a constant lower multiplicative inapproximability bound for set-cover.

Theorem 17 ([Hoc97]) *Unless $\mathcal{P} = \mathcal{NP}$, the set-cover problem cannot be approximated by any constant factor.*

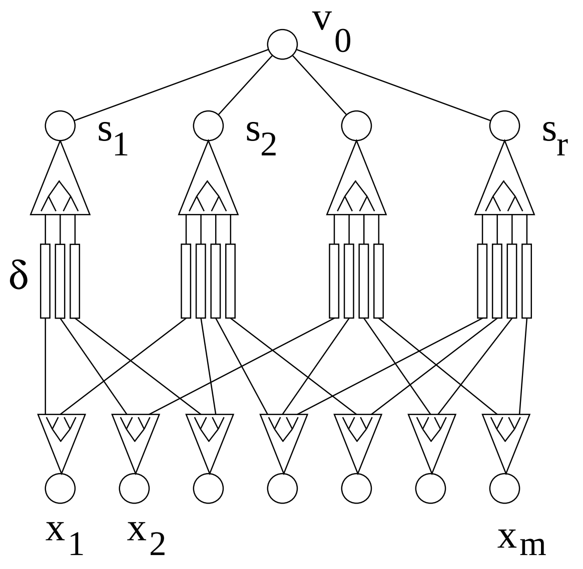

Figure 5.5: The reduction graph $G(F, \delta)$.

Proof of Theorem 12:

Given an instance of set-cover with $F = (S_1, \dots, S_r)$ and m elements the reduction constructs a graph $G(F, \delta)$ ($\delta \in \mathbb{N}$ will be chosen later on):

- The single source v_0 is the center of a star with ray nodes $s_1, \dots, s_r$.
- For all i the node s_i is the root of a binary balanced tree with $|S_i|$ leaves. Each leaf is the starting point of a chain of length δ which ends at a leaf of a binary balanced tree with a root named x_j if $j \in S_i$ for $j \in \{1, \dots, m\}$.
- These nodes x_j represent the ground set $\{1, \dots, m\}$. Their trees have $b(j) := |\{i \mid x_j \in S_i\}|$ leaves, each leaf is connected to only one of the above mentioned chains.

An example of this graph is shown in Figure 5.5. Note that the number of nodes of this graph is bounded by $O(\delta(m + r))$. Let $\ell := \log \max_i\{|S_i|, b(i)\}$ be the maximum depth of the balanced binary sub-trees.

Lemma 14 *A set-cover for F of size s implies $b(G(F, \delta)) \leq s + 2\delta + 7\ell$.*

Proof: We present a broadcast schedule for $G(F, \delta)$. The source v_0 informs the s set nodes s_i corresponding to the minimum set-cover. According to Lemma 13 all element nodes x_j are informed in time $s + 3\ell + \delta$.

Now all nodes will be informed by nodes $x_1, \dots, x_m$. They inform their trees in time 2ℓ and all set nodes s_i in additional time $\delta + \ell$. Then all nodes are informed. □

Lemma 15 *If any set-cover of F has at least size $2\delta + 1$ then $b(G(F, \delta)) > 3\delta$.*

Proof: We will show that there is an element node x_j which cannot be informed in time 3δ. Fix a broadcast schedule S. In every step the source can only inform one neighbor s_i. Let $V := \{s_{i_1}, \dots, s_{i_{2\delta}}\}$ be those neighbors informed in the first 2δ steps. Note that every path disjoint from the source between s_i and s_k is longer than 2δ. Therefore no further set node can be informed by this time.

We pick an element j from the set $\{1, \dots, n\} \setminus (S_{i_1} \cup \dots, \cup S_{i_{2\delta}})$ which is non-empty since there is no set-cover of size 2δ. The distance from the corresponding node x_j to any node in V is larger than 2δ, since none of the leaves of the V's trees is directly connected to the leaves of x_j's tree.

If x_j was informed by a node in V then $b_S(x_j) > 3\delta$. On the other hand, if x_j was informed by a set node not in V, then this set node was informed in time $2\delta + 1$ at the earliest. This set node has distance of at least δ to x_j, which results in a broadcast time $b_S(x_j) > 3\delta$. □

We continue the proof of Theorem 12. For any $k > 1$ it is possible to describe a set I of set-cover instances either having a set-cover of size at most s or having one of at least size ks. Since this proof is constructive, the decision of this property in I is $\mathcal{NP}$-hard even with the knowledge of s [Hoc97].

For our reduction we want to ensure that $\ell \leq s/7$. This property can be easily guaranteed: Consider the ground set (i, j) for $i \in \{1, \dots, n\}$ and $j \in \{1, \dots, 7 \log n\}$ and for $k \in \{1, \dots, r\}$ let $(i, j) \in S'_{k,j}$ iff $i \in S_k$. This padding increases every set-cover of size s to $s' = 7s \log n$, but does not increase ℓ. Since $\ell \leq \log n$, we have then $\ell \leq s'/7 = s \log n$.

We reduce this set I to a graph G by the construction above with the choice $\delta = \frac{ks}{2}$. Lemma 14 implies for a set-cover of at most s a broadcast time of at most $(2 + \frac{4}{k})\delta$, while Lemma 15 implies for a set-cover instance of at least ks a minimum broadcast time of 3δ.

This completes the proof. For $\epsilon = \frac{3}{2k+4}$ we have constructed a set of graphs for which it is $\mathcal{NP}$-hard to distinguish whether the broadcast time is smaller than $b = (2 + \frac{4}{k})\delta$ or larger than $(\frac{3}{2} - \epsilon)b$. □

5.5 A Tight Additive Bound

Now we present the proof for an lower additive bound that matches the bound of a polynomial time approximation algorithm presented in [KP95]. We use a polynomial time reduction from E3-SAT which denotes the satisfiability problem of Boolean CNF-formulas with exactly three literals in each clause.

Theorem 18 *[Hås97] For any $\epsilon > 0$ it is $\mathcal{NP}$-hard to distinguish satisfiable E3-SAT formulas from E3-SAT formulas for which only a fraction $7/8 + \epsilon$ of the clauses can be satisfied, unless $\mathcal{P} = \mathcal{NP}$.*

Let F be a 3-CNF with m clauses $c_1, \dots, c_m$ and variables $x_1, \dots, x_n$. Let $a(i)$ denote the number of occurrences of the positive literal x_i in F. In the proof of Theorem 18 Håstad uses a variant of E3-SAT where every variable occurs equally often as positive and negative clause in F. Let $\delta := 2\ell m'$, where $m' := \sum_{i=1}^{n} a(i)$ with ℓ being a large number to be chosen later on. Note that $m = \frac{2}{3}m'$.

The formula F is reduced to an undirected graph $G_{F,\ell}$ (see Figure 5.6) as follows.

- The source v_0 is the center of a star S_δ with δ rays $x^b_{i,j,k}$, for $b \in \{0, 1\}$, $i \in \{1, \dots, n\}$, $j \in \{1, \dots, a(i)\}$, and $k \in \{1, \dots, \ell\}$.
- We call the nodes $x^b_{i,j,k}$ *literal nodes*. They belong to ℓ disjoint isomorphic sub-graphs $G_1, \dots, G_\ell$. A sub-graph G_k contains literal nodes $x^b_{i,j,k}$, representing the literal x^b_i ($x^1_i = x_i$, $x^0_i = \overline{x_i}$).
- Between the literal nodes corresponding with a variable x_i in G_k we insert chains $C_\delta(x^0_{i,j,k}, x^1_{i,j',k})$ for all $i \in \{1, \dots, n\}$ and $j, j' \in \{1, \dots, a(i)\}$.
- For every clause $c_\nu = x^{b_1}_{i_1} \vee x^{b_2}_{i_2} \vee x^{b_3}_{i_3}$ we insert *clause nodes* $c_{\nu,k}$ which we connect via the three chains $C_{\delta/2}(c_{\nu,k}, x^{b_\rho}_{i_\rho,j_\rho,k})$ for $\rho \in \{1, 2, 3\}$ of length $\delta/2$ to their corresponding literal nodes $x^{b_1}_{i_1,j_1,k}, x^{b_2}_{i_2,j_2,k}, x^{b_3}_{i_3,j_3,k}$. This way, every literal node is connected to one clause node.

The underlying idea of this construction is that the assignment of a variable x_i corresponds to the time when the corresponding literal nodes will be informed. For the upper bound we can use the satisfying assignment of F to construct a fast broadcast schedule.

Lemma 16 *If F is satisfiable, then $b(G_{F,\ell}, v_0) \leq \delta + 2m' + 2$.*

Proof: The schedule S informs all literal nodes directly by v_0. Let $\alpha_1, \dots, \alpha_n$ be a satisfying assignment of F. The literal nodes $x^{\alpha_i}_{i,j,k}$ of graph G_k are informed within the time period $(k-1)m' + 1, \dots, km'$. The literal nodes $x^{\overline{\alpha_i}}_{i,j,k}$ are informed within the time period $\delta - km' + 1, \dots, \delta - (k-1)m'$.

Note that m' is a trivial upper bound for the degree at a literal node. So, the chains between two literal nodes can be informed in time $\delta + 2m' + 1$. A clause node can be informed in time $km' + \delta/2 + 1$ by an assigned literal node of the first type, which always exists since $\alpha_1, \dots, \alpha_n$ satisfies F. Note that all literal nodes corresponding to the second type are informed within $\delta - (k-1)m'$. So the chains between those and the clause node are informed in time $\delta + 2m' + 2$. □

$$F = \underbrace{(x_1 \vee \overline{x}_2 \vee x_3)}_{C_1} \wedge \underbrace{(x_2 \vee \overline{x}_3 \vee x_4)}_{C_2} \wedge \underbrace{(\overline{x}_1 \vee x_3 \vee \overline{x}_4)}_{C_3} \wedge \circ \circ \circ$$

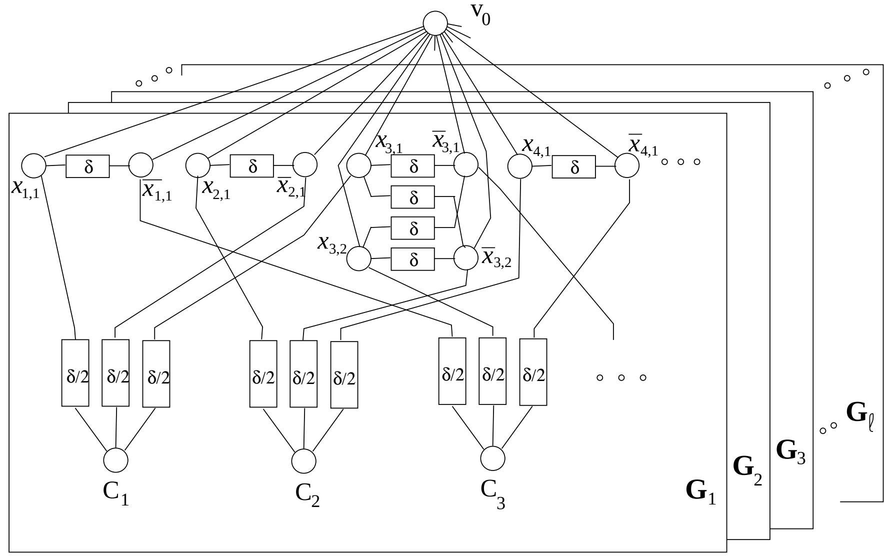

Figure 5.6: The reduction graph $G_{F,\ell}$.

Lemma 17 *Let S be a busy broadcast schedule for $G_{F,\ell}$. Then,*

1. *every literal node will be informed directly from the source v_0, and*

2. *for $c_{\nu,k} = x^{\alpha_1}_{i_1,j_1,k} \vee x^{\alpha_2}_{i_2,j_2,k} \vee x^{\alpha_3}_{i_3,j_3,k}$: $b_S(c_{\nu,k}) > \frac{\delta}{2} + \min_\rho\{b_S(x^{\alpha_\rho}_{i_\rho,j_\rho,k})\}$.*

Proof:

1. Every path between two literal nodes that avoids v_0 has at least length $\delta + 1$. By Lemma 13 even the first informed literal node has no way to inform any other literal node before time point δ, which is the last time a literal node is going to be informed by v_0.

2. follows by 1.

□

If only one clause per Boolean formula is not satisfied, this lemma implies that if F is not satisfiable, then $b(G_{F,\ell}, \{v_0\}) > \delta + \ell$. A better bound can be achieved if the inapproximability result of Theorem 18 is applied. A busy schedule S for graph $G_{F,\ell}$ defines an assignment for F. Then, we will categorize every literal node as *high*, *low* or *neutral*, describing whether its broadcast time corresponds to the assignment and whether it is delayed. Clause nodes are classified either as *high* or *neutral*. Every unsatisfied clause of the E3-SAT-formula F will increase the number of high literals. Besides this, high and low literal nodes come in pairs, yet possibly in different subgraphs G_k and $G_{k'}$. The overall number of the high nodes will be larger than those of the low nodes.

Proof of Theorem 13: Consider an unsatisfiable E3-SAT-formula F, the above described graph $G_{F,\ell}$ and a busy broadcast schedule S on it. The schedule defines for each subgraph G_k an assignment $x_{1,k}, \ldots, x_{n,k} \in \{0,1\}^n$ as follows. Assign the variable $x_{i,k} = \alpha$ if the number of delayed literal nodes with $b_S(x^{\alpha}_{i,j,k}) > \delta/2$ is smaller than those with $b_S(x^{\overline{\alpha}}_{i,j,k}) > \delta/2$. If both numbers are equal, w.l.o.g. let $x_{i,k} = 0$.

1. A literal node $x^{\alpha}_{i,j,k}$ is *coherently assigned*, iff $b_S(x^{\alpha}_{i,j,k}) \leq \delta/2 \;\Leftrightarrow\; x_{i,k} = \alpha$. We also call coherently assigned literal nodes *neutral*.

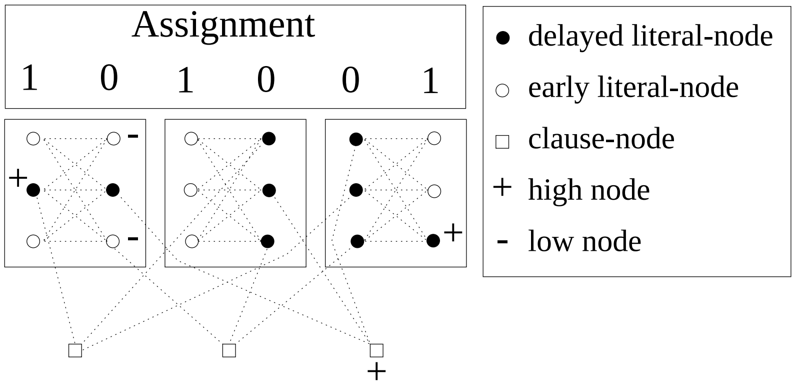

Figure 5.7: The circle denote literal nodes. Literal nodes in a rectangle belong to the same variable. A given broadcast schedule informs literal nodes early or delayed. This timing defines the assignment of variables and whether nodes are low, high or neutral

2. A literal node $x^{\alpha}_{i,j,k}$ is *high* if it is delayed and not coherently assigned, i.e., $x_{i,k} = \alpha$ and $b_S(x^{\alpha}_{i,j,k}) > \delta/2$.
3. A literal node $x^{\alpha}_{i,j,k}$ is *low* if it is not delayed and not coherently assigned, i.e., $x_{i,k} = \overline{\alpha}$ and $b_S(x^{\alpha}_{i,j,k}) \leq \delta/2$.
4. A clause node $c_{\nu,k}$ is *high*, if all of its three connected literal nodes are neutral and delayed, i.e., $\forall \rho \in \{1,2,3\}\ b_S(x^{\alpha_i}_{i_\rho,j_\rho,k}) > \delta/2$.
5. All other clause nodes are *neutral*.

Every high literal node $x^{\alpha}_{i,j,k}$ with broadcast time $\delta/2 + \epsilon_1$ for $\epsilon_1 > 0$ can be matched to a neutral delayed literal node $x^{\overline{\alpha}}_{i,j',k}$ with broadcast time $b_S(x^{\overline{\alpha}}_{i,j',k}) = \delta/2 + \epsilon_2$ for $\epsilon_2 > 0$. Lemma 13 shows that the chain between both of them can be informed in time $\delta + \frac{\epsilon_1+\epsilon_2}{2}$ at the earliest.

For a high clause node with literal nodes $x^{\alpha_i}_{i_\rho,j_\rho,k}$ and broadcast times $b_S(x^{\alpha_i}_{i_\rho,j_\rho,k}) = \delta/2 + \epsilon_\rho$ with $\epsilon_1, \epsilon_2, \epsilon_3 > 0$, Lemma 17 shows that this high clause node gets the information not earlier than $\delta + \min\{\epsilon_1, \epsilon_2, \epsilon_3\}$. So, the chain to the most delayed literal node will be informed at $\delta + (\min\{\epsilon_1, \epsilon_2, \epsilon_3\} + \max\{\epsilon_1, \epsilon_2, \epsilon_3\})/2$ at the earliest.

Lemma 18 *Let q be the number of low literal nodes, p the number of high literal nodes, and p' the number of high clause nodes. Then the following holds:*

1. $p = q$,
2. $b_S(G_{F,\ell}, v_0) \ \geq\ \delta + p$,
3. $b_S(G_{F,\ell}, v_0) \ \geq\ \delta + (p + 3p')/2$.

Proof:

1. Consider the set of nodes $x^{\alpha}_{i,j,k}$, for $j \in \{1, \ldots, a(i)\}$ and $\alpha \in \{0,1\}$. For this set let $p_{i,k}$ be the number of high nodes, $q_{i,k}$ the number of low nodes and $r_{i,k}$ the number of nodes with time greater than $\delta/2$. By the definition of high and low nodes the following holds for all $i \in \{1, \ldots, n\}$, $k \in \{1, \ldots, \ell\}$:
$$r_{i,k} - p_{i,k} + q_{i,k} = a(i)\ .$$
Lemma 13 and Lemma 17 show that half of the literal nodes are informed within $\delta/2$ and the rest later on:
$$\sum_{i,k} r_{i,k} = \delta/2 = \sum_{i,k} a(i)\ ,$$

It then follows that:

$$q - p = \sum_{i,k} (r_{i,k} - p_{i,k} + q_{i,k} - a(i)) = 0 \; .$$

2. Note that we can match each of the p high (delayed) literal node $x^{\alpha}_{i,j,k}$ to a neutral delayed literal node $x^{\overline{\alpha}}_{i,j',k}$. Furthermore, these nodes have to inform a chain of length δ. If the latest of the high nodes and its partners is informed at time $\delta/2 + \epsilon$, then Lemma 13 shows that the chain cannot be informed earlier than $\delta + \epsilon/2$.

   The broadcast times of all literal nodes are pairwise distinct. Therefore it holds $\epsilon \geq 2p$, proving $b_S(G_{F,\ell}, v_0) \geq \delta + p$.

3. Every high clause node is connected to three neutral delayed literal nodes. The task to inform all chains to the three literal nodes is done at time $\delta + \epsilon'/2$ at the earliest, if $\delta/2 + \epsilon'$ is the broadcast time of the latest literal node. For p' high clause nodes, there are $3p'$ corresponding neutral delayed literal nodes. Furthermore, there are p delayed high literal nodes (whose matched partners may intersect with the $3p'$ neutral literal nodes). Nevertheless, the latest high literal node with broadcast time $\delta/2 + \epsilon''$ causes a broadcast time on the chain to a neutral delayed literal node of at least $\delta + \epsilon''/2$.

   From both groups consider the most delayed literal node $v_{\max}$. Since every literal node has a different broadcast time it holds that $\epsilon'' \geq 3p' + p$, and thus $b_S(v_{\max}) \geq \delta + (3p' + p)/2$.

□

Suppose all clauses are satisfiable. Then Lemma 16 gives an upper bound for the optimal broadcast time of $b(G_{F,\ell}, v_0) \leq \delta + 2m' + 2$.

Let us assume that at least κm of the m clauses are unsatisfied for every assignment. Consider a clause node that represents an unsatisfied clause with respect to the assignment which is induced by the broadcast schedule. Then at least one of the following cases can be observed:

- The clause node is high, i.e., its three literal nodes are coherently assigned.
- The clause node is neutral and one of its three literal nodes is low.
- The clause node is neutral and one of its three literal nodes is high.

Since each literal node is chained to one clause node only, this implies

$$\kappa \ell m \;\leq\; p' + p + q \;=\; p' + 2p \; .$$

The case $p \geq 3p'$ implies $p \geq \frac{3}{7}(2p + p')$. Then it holds for the broadcast time of any busy schedule S:

$$b_S(G_{F,\ell}, v_0) \;\geq\; \delta + p \;\geq\; \delta + \tfrac{3}{7}(2p + p') \; .$$

Otherwise, if $p < 3p'$, then $\frac{1}{2}(p + 3p') \geq \frac{3}{7}(2p + p')$ and

$$b_S(G_{F,\ell}, v_0) \;\geq\; \delta + \tfrac{1}{2}(p + 3p') \;\geq\; \delta + \tfrac{3}{7}(2p + p') \; .$$

Note that $\delta = 3m\ell$. Combining both cases, it follows that

$$b_S(G_{F,\ell}, v_0) \;\geq\; \delta + \tfrac{3}{7}\kappa \ell m \;=\; \delta \left(1 + \tfrac{1}{7}\kappa\right) \; .$$

For any $\epsilon > 0$ this gives, choosing $\ell \in \Theta(m)$ for sufficient large m

$$\frac{b_S(G_{F,\ell}, v_0)}{b(G_{F,\ell}, v_0)} \;\geq\; \frac{1 + \frac{1}{7}\kappa}{1 + \frac{2m'+2}{\delta}} \;\geq\; 1 + \tfrac{1}{7}\kappa - \epsilon \; .$$

Theorem 18 states $\kappa = \frac{1}{8} - \epsilon''$ for any $\epsilon'' > 0$ which implies claimed lower bound of $\frac{57}{56} - \tilde{\epsilon}$ for any $\tilde{\epsilon} > 0$. Note that the number of nodes of $G_{F,\ell}$ is in $\Theta(m^4)$ and $\delta \in \Theta(m^2)$. □

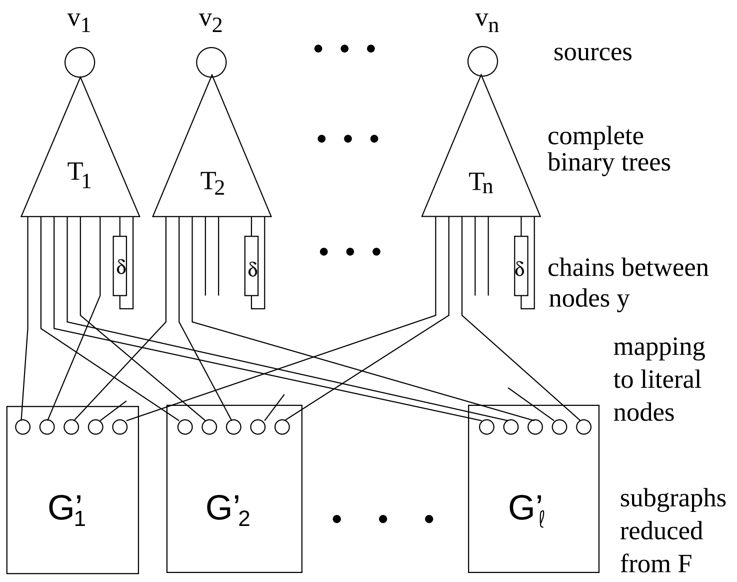

Figure 5.8: The reduction graph $G'_{F,\ell}$.

5.6 Lower Bounds for Ternary Graphs

The previous reduction uses graphs $G_{F,\ell}$ with a large degree at the source node. To address ternary graphs with multiple sources we modify this reduction as follows.

The proof uses a reduction from the E3-SAT-6 problem: a CNF formula with n variables and $m = n/2$ clauses is given. Every clause contains exactly three literals and every variable appears three times positive and three times negative, but does not appear in a clause more than once. The output is the maximum number of clauses that can be satisfied simultaneously by some assignment to the variables.

Lemma 19 *For some $\epsilon > 0$, it is $\mathcal{NP}$-hard to distinguish between satisfiable 3CNF-6 formulas, and 3CNF-6 formulas in which at most a $(1-\epsilon)$-fraction of the clauses can be satisfied simultaneously.*

Proof: Similar as Proposition 2.1.2 in [Fei98]. Here, every second occurrence of a variable is replaced with a fresh variable when reducing from E3-SAT. This way the number of positive and negative literals remains equally high. $\square$

How can the star at the source be replaced by a ternary sub-graph that produces high differences between the broadcast times of the literal nodes? It turns out that a good way to generate such differences in a very symmetric setting is a complete binary tree. Using trees instead of a star complicates the situation. A busy broadcast schedule informs $\binom{d}{t}$ leaves in time $d+t$ where in the star graph only one was informed in time t. This is the reason for the decrease of the inapproximability bound.

The ternary reduction graph $G'_{F,\ell}$, given a 3CNF-6-formula F and a number ℓ to be chosen later, consists of the following sub-graphs (see Figure 5.8).

1. The sources $v_1,\dots,v_n$ are roots of *complete binary trees* $B_1,\dots,B_n$ with depth $\delta = \log(12\ell)$ and leaves $v^i_1,\dots,v^i_{2^\delta}$. The number ℓ will be chosen such that δ is an even number.

A constant fraction of the leaves of B_i are the literal nodes $x^\alpha_{i,j,k}$ of a subgraph G_k. The rest of them, $y^\alpha_{i,j}$ is connected in pairs via δ-chains. For an accurate description we introduce the functions

$$f(p) \;:=\; \sum_{i=\delta/2+1}^{\delta/2+p} \binom{\delta}{i}$$

and $g(x) \;:=\; \min\{p \mid f(p) \geq x\}$.

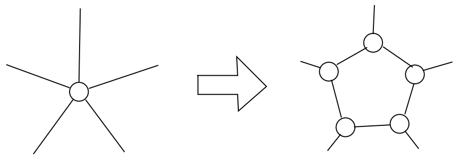
Figure 5.9: A ring of n nodes as replacement of a star-sub-graph.

Lemma 20 *1. For $p \in \{1, \dots, \sqrt{\delta}\}$:*

$$\frac{p}{10}\frac{2^\delta}{\sqrt{\delta}} \leq f(p) \leq p\frac{2^\delta}{\sqrt{\delta}} .$$

2. For $x \in [0, \frac{2^\delta}{10}]$:

$$x\frac{\sqrt{\delta}}{2^\delta} \leq g(x) \leq 10x\frac{\sqrt{\delta}}{2^\delta} .$$

Proof: Note that Stirling's formula implies $\binom{\delta}{\delta/2} \leq \frac{2^\delta}{\sqrt{\delta}}$ and $\binom{\delta}{\delta/2+\sqrt{\delta}} \geq \frac{2^\delta}{10\sqrt{\delta}}$. Since the series $\binom{\delta}{i}, \binom{\delta}{i+1}, \dots$ decreases for $i \geq \delta/2$ this implies the claim. □

Every node of B_i is labeled by a binary string given by the path from the root, i.e., for the root r **label(r)** is the empty string λ; two successing nodes v_1, v_2 of a node w are labeled by **label$(w)0$** and **label$(w)1$**. We call two leaves x, y are *opposite* if label(x) can be derived from label(y) by negating every bit. For a binary string let $\Delta(s) := |\#_1(s) - \#_0(s)|$ be the difference of occurrences of 1 and 0 in s. Consider an *indexing* $v_1^i, \dots, v_{2^\delta}^i$ of the leaves of B_i such that for all $j \in \{1, \dots, 2^\delta - 1\}$: $\Delta(\text{label}(v_j^i)) \leq \Delta(\text{label}(v_{j+1}^i))$, and v_j^i and $v_{2^\delta-j+1}^i$ have opposite labels for all $j \in \{1, \dots, 2^\delta\}$.

2. For every binary tree B_i according to these indices the literal nodes of G_k are defined by $x_{i,j,k}^0 = v_{2^{\delta-1}+3(k-1)+j}^i$ and $x_{i,j,k}^1 = v_{2^{\delta-1}-3(k-1)-j+1}^i$ for $j \in \{1, \dots, 3\}$, and $k \in \{1, \dots, \ell\}$.
3. The other leaves of B_i are connected pairwise by chains of length δ such that opposite leaves of a tree represent free literal nodes $y_{i,j}^0$ and $y_{i,j}^1$. These nodes are not part of any sub-graph G_k.
4. The sub-graphs G_k for $k \in \{1, \dots, k\}$ described in the previous section have a degree 5 at the literal nodes. These nodes are replaced with rings of size 5 to achieve degree 3 (see Figure 5.9).

Proof of Theorem 14: If F is satisfiable, then there is a coherently assigning broadcast schedule with $b(G'_{F,\ell}) \leq 2\delta + 4$.

An analogous observation to Lemma 17 for a busy broadcast schedule S for $G'_{F,\ell}$ is the following

1. Every literal node will be informed directly from the source of its tree;
2. For all $i \in \{1, \dots, n\}$ and for all $t \in \{0, \dots, \delta\}$ it holds

$$|\{j \in \{1, \dots, 2^\delta\} \mid b_S(v_j^i) = t + \delta\}| = \binom{\delta}{t} ;$$

3. For $c_{\nu,k} = x_{i_1,j_1,k}^{\alpha_1} \vee x_{i_2,j_2,k}^{\alpha_2} \vee x_{i_3,j_3,k}^{\alpha_3}$:

$$b_S(c_{\nu,k}) = \frac{\delta}{2} + \min_\rho\{b_S(x_{i_\rho,j_\rho,k}^{\alpha_\rho})\} + O(1) .$$

Again literal nodes are defined to be either low, high, or neutral. Clause nodes are either high or neutral. For the number q of low literals, p of high literals, and p' the number of high clauses it holds $p = q$. There are $2p$, resp. $3p'$ nodes in different chains that are informed later than $2\delta - 1$. Therefore there is a tree B_k

that informs at least $2p/n$, resp. $3p'/n$ delayed nodes. Using g_δ it is possible to describe a lower bound of the time delay caused by B_k as follows:

$$b_S(G_{F,\ell}) \ \geq \ 2\delta - 1 + \frac{1}{2}\max\left\{g_\delta\left(\frac{2p}{n}\right), g_\delta\left(\frac{3p'}{n}\right)\right\} .$$

Let us assume that at least κm clauses are unsatisfied for every assignment. The constant fraction of y-leaves of trees T_i can be seen as an additional set of unused literal nodes. Now consider a clause node that represents an unsatisfied clause with respect to the assignment which is induced by the broadcast schedule. Then there is at least a high clause node, a neutral clause node connected to a low literal node, or a neutral clause node connected to a high literal node.

Since each literal node is chained to at most one clause node, this implies

$$\kappa\ell m \ \leq \ p' + p + q \ = \ p' + 2p .$$

Note that $24\ell \geq 2^\delta$. These observation combind with Lemma 20 now imply

$$b_S(G'_{F,\ell}, \{v_1, \ldots, v_n\}) \ \geq \ 2\delta - 1 + \frac{1}{2}\, g_\delta\!\left(\frac{\kappa\ell m}{2n}\right) \ \geq \ 2\delta + \epsilon\sqrt{\delta}$$

for some $\epsilon > 0$. Since for the set of nodes V of $G'_{F,\ell}$ it holds $|V| \in \Theta(\ell m \log \ell)$ it is sufficient to choose ℓ as a non constant polynomial of m. □

Proof of Theorem 15: We start to combine the reduction graph of the preceding theorem with a ternary pyramid (see Fig 5.3). The single source v_0 is the top of the pyramid. The n base nodes have been previously the sources $v_1, \ldots, v_n$. Note that the additional amount of broadcast time in a pyramid is $2n-2$ for $n-1$ nodes and $2n-1$ for one node for any busy broadcast schedule. Thus, the former sources are informed nearly at the same time.

For the choice $\ell \in \Theta(\frac{m}{\log m})$ the number of nodes of the new graph is bounded by $\Theta(m^2)$. The broadcast time increases from $\Theta(\log m)$ of $G'_{F,\ell}$ to $\Theta(m)$ and the indistinguishable difference remains $\Theta(\sqrt{\log m})$. □

5.7 Conclusions

The complexity of broadcast time is a key for understanding the obstacles to efficient communication in networks. This article answers the open question stated recently by [BNGNS98], whether single source broadcasting in the Telephone model can be approximated within any constant factor. At the moment, the best upper bound approximation ratio for broadcast time is known $O(\log |V|)$ [BNGNS98] and as a lower bound we state a factor of $\frac{3}{2} - \epsilon$. For this problem of the multiplicative approximation is still wide open and there is a new unpublished result of Elken and Kortsartz [EK01] which improves this bound to $3 - \epsilon$ under the assumption $\mathcal{NP} \subseteq \text{DTime}(n^{O(\log n)})$.

Theorem 13 closes the gap for the additive approximation. In [KP95] an additive $O(\sqrt{n})$-approximation algorithm was presented for general graphs with n nodes. Here, we present a lower additive bound of $\Omega(\sqrt{n})$.

It is possible to transfer this result to bounded degree graphs. But the reconstruction of sub-graphs with large degree decreases the lower bound dramatically. Nevertheless, for ternary graphs with a single source this paper improves the inapproximability difference from 1 [JRS98] up to $\sqrt{\log n}$. For the approximation factor of such graphs little is known so far. The upper bound is a constant factor and Theorem 15 implies a lower bound of $1 + \Theta\left(\frac{\log n}{\sqrt{n}}\right)$. So matching upper and lower bounds remain unknown.

From a practical point of view, network structures are often uncertain because of dynamic and unpredictable changes. And if the network is static, it is hardly ever possible to determine the ratio between switching time on a single processor and the delay on communication links. But even if these parameters are known as constants like in telephone model of broadcasting, these results show that developing a good broadcast strategy is a computationally infeasible task.

Chapter 6

Randomized Rumor Spreading

6.1 Introduction

In this chapter we investigate the problem of broadcasting information in a processor network from a different view. Unlike as in the preceding chapters an interconnection network allows arbitrary point-to-point communication. The message to be passed will be called rumor to emphasize the specifics of this communication model. We investigate the problem of spreading rumors in a distributed environment using randomized communication. Suppose n players exchange information in parallel communication rounds over an indefinite time. In each round t, the players are connected by a communication graph G_t generated by *random phone calls* as follows: each player u selects a communication partner v at random and u *calls* v; two players u and v are connected by an edge in G_t if u calls v in round t. Rumors can be started in any round by any player and can be transmitted in both directions along the edges in the graph G_t in round t. The goal is to spread the rumor among all participating players using a small number of rounds and a small number of transmissions.

The motivation for using randomized communication is that it naturally provides robustness, simplicity, and scalability. For example, consider the following so-called *push algorithm*. Starting with the round in which a rumor is generated, each player that holds the rumor forwards it to a communication partner selected independently and uniformly at random. The distribution of the rumor is terminated after some fixed number of $O(\log n)$ rounds. At this time all players are informed with high probability. The term with high probability (w.h.p.) means with probability at least $1 - O(n^{-\alpha})$ for an arbitrary constant $\alpha > 0$.

Clearly, one can also inform all players in $O(\log n)$ rounds using a deterministic interconnection of constant degree, e.g., a shuffle network (For an overview of deterministic information dissemination we refer to [HHL88] or [HKMP96]). The advantage of the randomized push algorithm, however, is its inherent robustness against several kinds of failures compared to deterministic schemes that either need substantially more time [GP96] or can tolerate only a relatively small number of faults [LMS92]. For example, consider node failures in which a player (different from the player starting the rumor) fails to communicate or simply crashes and forgets its rumors. Obviously, when using a sparse deterministic network, even a single node failure can result in a large fraction of players not receiving the rumor. When using the randomized push algorithm, however, the effects of node failures are very limited. In fact, it is not difficult to prove that F node failures (specified by an oblivious adversary) result in only $O(F)$ uninformed players with high probability.

Unfortunately, the push algorithm produces a large communication overhead. In fact, it needs to forward each individual rumor $\Theta(n \log n)$ times before all players are informed, in comparison to a deterministic scheme which requires only $n - 1$ transmissions. It seems that the large number of transmissions is the price for the robustness. This gives rise to the question whether this additional communication effort is a special property of the above push algorithm or is inherent to rumor spreading using random phone calls in general.

6.1.1 Background

The rumor spreading model is originated in the mathematical modeling to the spread of an infectious disease. In 1926 McKendrick developed the first stochastic theory in 1926. An overview over such models is presented in [Bai75]. Standard mathematical models of viral infection characterize individuals by a small number of states, e.g. *infected*, *uninfected*, *immune*, etc. The *push model* considers uninfected and infected individuals. In the first round one individual is infected. Every infected individual contacts a random partner of the whole group (possibly being already infected) and transmits the disease.In 1987 Pittel [Pit87] analyzed for this model that the expected number of rounds T_n to infect all n participants is bounded by

$$T_n \;=\; \log n + \ln n + O(1)$$

with probability converging to 1.

Demers et al. [DGH$^+$87] introduced the idea of using so-called epidemic algorithms for the lazy update of data objects in a data base replicated at many sites, e.g., yellow pages, name servers, or server directories. In particular, they propose the following two concepts:

- *Anti-entropy:* Every site regularly chooses another site at random and resolves all differences by exchanging the complete data base contents.

- *Rumor mongering:* When a site receives a new update it becomes a “hot rumor”. While a site holds a “hot rumor”, it periodically chooses another site at random and sends the rumor to the other site.

It turns out that anti-entropy is extremely reliable but produces such an enormous amount of communication that it cannot be used too frequently. The idea of rumor mongering is to exchange only recent updates, thereby reducing the communication overhead significantly. In practice one might use a combination of both concepts, that is, using rumor mongering frequently and anti-entropy very rarely in order to ensure that all updates are recognized by all sites. In this paper, we solely investigate algorithms implementing the rumor mongering concept.

The original idea for rumor spreading was to send rumors only from the caller to the called player (*push transmission*) [DGH$^+$87]. Several termination mechanisms deciding when a rumor becomes “cold” so that it transmission is stopped were investigated. All these algorithms share the same phenomenon: the fraction u of players that do not know a particular rumor decreases exponentially with the number of transmissions t (i.e., messages that contain this rumor). So-called *mean field equations* (implicitly assuming that u is sharply concentrated around its mean value $\mathbf{E}\left[u\right]$) lead to the conjecture that $u \approx \exp(-t/n)$ for all variants of the push algorithm that have been investigated. In other words, a push algorithm needs $\Theta(n \log n)$ transmissions for sending a rumor to all players.

A further idea introduced in [DGH$^+$87] is to send rumors from the called to the calling player (*pull transmission).* It was observed that the number of uninformed players decreases much faster using a pull scheme instead of a push scheme. This kind of transmission makes sense if updates occur frequently so that (almost) every player places a random call in each round anyway. Mean field equations lead to the conjecture that $u \approx \exp(-2^t)$ for pull schemes. Clearly, this double exponential behavior implies that only $\Theta(n \;\log\log n)$ transmissions are needed if the distribution of the rumor can be stopped at the right time. Such a termination mechanism, however, is not presented. Instead, the authors predict that $\Theta(n \sqrt[3]{\log n})$ transmissions are sufficient for some other specific termination mechanisms.

The work of Demers et al. initiated an enormous amount of experimental and conceptual study of epidemic algorithms. For example, there is a variety of research issues like consistency, correctness, data structures, and efficiency [AAS97, GL91, GPP93, LLSG92, RGK96]. Recent theoretical work concentrates on the robustness against Byzantine failures [MMR99]. In this paper, we concentrate only on the efficiency of these randomized algorithms. In particular, we study their time and communication complexity using a simple model for the underlying randomized communication.

6.1.2 The Random Phone Call Model

Let V denote the set of players. The communication graph $G_t = (V, E_t \subseteq V \times V)$ of round $t \geq 1$ is obtained by a distributed, randomized process. In each round, each player u chooses a communication partner v from V at random and u *calls* v. Unless otherwise stated, we assume that all players choose their communication partners independently and uniformly at random from V.

Even though we envisage an application (such as the lazy transmission of updates to distributed copies of a database) in which rumors are constantly generated by different players, our analysis is concerned with the distribution of a single rumor only. We focus on the lifetime of the rumor and the number of transmissions rather than the number of connections established because the latter cost is amortized over all the rumors using that connection.

In round t, the rumor and other information can be exchanged in both directions along the edges of G_t. Whenever a connection is established between two players, each one of them (if holding the rumor) has to decide whether to transmit the rumor to the other player, typically without knowing whether this player has received the rumor already. Regarding the flow of information, we distinguish between push and pull transmissions. Assume player u calls player v.

- The rumor is *pushed* If u tells v the rumor.
- The rumor is *pulled* if v tells u the rumor.

We do not limit the size of the information exchanged in any way. Each information exchange between neighboring players in a round is counted as a single transmission (We point out that our algorithms only add small counter values to rumors, whereas our lower bounds hold even for algorithms in which players exchange their complete history whenever the rumor is sent in either direction). Communication inside each round, however, is assumed to proceed in parallel, that is, any information received in a round cannot be forwarded to another player in the same round.

The major issue that has to be specified by a rumor spreading algorithm is how players decide whether the rumor shall be forwarded to a communication partner. An algorithm is called *distributed* if players make these decisions using only local knowledge. In other words, the decision whether a player sends a message to a communication partner in round t depends only on the player's *state* in that round. The initial state of a player is defined by the player's address, the number of players, and possibly a random bit string. The state of a player in round $t \geq 1$ is a function of its initial state, the addresses of its neighbors in the communication graphs $G_1, \ldots, G_t$, and the information received in rounds 1 to $t-1$. (For our lower bounds we allow the state to depend in addition on a globally known round number and the birth date of the rumor considered.)

Finally, an algorithm is called *address-oblivious* if a player's state in round t does not depend on the addresses of the neighbors in G_t but only on the number of neighbors in G_t. (The state can still depend on the addresses of neighbors in $G_1, \ldots, G_{t-1}$.) We point out that all rumor spreading algorithms proposed by Demers et al. [DGH$^+$87] are address-oblivious.

6.1.3 New Results

We prove that the number of transmissions can be reduced significantly when the rumor is sent in both directions, that is, when using push and pull rather than only push operations. We introduce a *simple push&pull algorithm* spreading the rumor to all players in $O(\log n)$ rounds using only $O(n \ \log\log n)$ transmissions in comparison to $\Theta(n \log n)$ as used by the push algorithm. For this analysis it is necessary to analyze the performance of the plain push algorithm and pull algorithm step by step. It turns out that these algorithms need $\Theta(n \log n)$ message to inform all players with high probability, i.e., with probability $1 - n^{-c}$ for some constant $c > 1$. Both algorithms need time $\Theta(\log n)$.

Theorem 19 *The simple push&pull-scheme informs all players in time* $\log_3 n + O(\log\log n)$ *using* $O(n \ \log\log n)$ *messages with high probability.*

The drawback of the simple push&pull-algorithm is that its success heavily relies on a very exact, global estimation of the right termination time. This mechanism is very sensitive to any kind of errors that influence the expansion of the set of informed players.

Scott Shenker proposed a distributed termination mechanism using a counter indicating indirectly the spread of the rumor. We show that this *min-counter algorithm* performs as well as the push&-pull algorithm:

Theorem 20 *The min-counter algorithm informs all players in time $\log_3 n + O(\log\log n)$ using $O(n\ \log\log n)$ messages with high probability.*

In order to improve the robustness, we devise a distributed termination scheme, called the *median-counter algorithm*, that is provably robust against adversarial node failures as well as stochastic inaccuracies in establishing the random connections.

In particular, we show that the efficiency of the algorithm does not rely on the fact that players choose their communication partners uniformly from the set of all players. We show that the median-counter algorithm takes $O(\log n)$ rounds and needs only $O(n\ \log\log n)$ transmissions regardless of the probability distribution used for establishing the random connections as long as all players act independently and each player uses the same distribution $\mathcal{D} : V \to [0, 1]$ to select its communication partner. For example, this allows sampling from an arbitrary address directory (possibly with redundant addresses and some non-listed players as in a telephone book). In other words, the algorithm can be executed even without global knowledge about the set of players.

Theorem 21 *Assuming an arbitrary distribution $\mathcal{D}$ and up to F node failures as described above, the median-counter algorithms informs all but $O(F)$ players in $O(\ln n)$ rounds using $O(n\ \log\log n)$ transmissions with high probability.*

In addition, we provide lower bounds assuming that the communication partners are selected using the uniform probability distribution. Both the simple push&pull algorithm as well as the median-counter algorithm are address-oblivious and use only $O(n\ \log\log n)$ transmissions. We prove a corresponding lower bound showing that any address-oblivious algorithm needs to perform $\Omega(n\ \log\log n)$ transmissions in order to inform all players. We point out that this bound holds independently of the number of rounds executed.

Theorem 22 *Any address-oblivious algorithm guaranteeing that all but a fraction f of the players receive the rumor with constant probability needs to perform $\Omega(n\ \log\log \frac{1}{f})$ transmissions in expectation.*

The situation changes substantially when considering general (i.e., possibly non-address-oblivious) algorithms. Allowing $\Theta(n \log n)$ rounds, an algorithm that exploits the addresses of communication partners can spread the rumor using only $n - 1$ transmissions. Here is a simple example. The player initiating the rumor simply waits until each other player appears as communication partner for the first time and then forwards the rumor to this player. Clearly, this is not a practical algorithm as it takes too many rounds. Nevertheless, it illustrates the additional possibilities when the addresses of communication partners can be exploited.

The above example leads to the question of whether general epidemic algorithms can spread a rumor in a small number of rounds while using only a linear number of transmissions. We give a lower bound answering this question negatively. In particular, we show that any randomized rumor spreading algorithm running for $O(\log n)$ rounds requires $\omega(n)$ transmissions.

Theorem 23 *Any distributed rumor spreading algorithm guaranteeing that all but a fraction $o(1)$ of the players receive the rumor within $O(\ln n)$ rounds with constant probability needs to perform $\omega(n)$ transmissions in expectation.*

This lower bound holds regardless of the amount of information that can be attached to the rumors. For example, players might always exchange their complete communication history whenever the rumor is transmitted in either direction. Thus, there is a fundamental gap between rumor spreading algorithms based on random interconnections and deterministic broadcasting schemes.

The rest of this chapter is organized as follows. We start in the next section with an extensive analysis of the push, pull, push&pull-algorithm. In section 6.3 we present the min-counter and in section 6.4 the med-counter algorithms. Then, in section 6.5 we prove the lower bound for the number of transmission in the case of oblivious communication. In the last section of this chapter we present the general lower bound for number of transmissions in this communication model.

```
Push algorithm for player $p$ and rumor $R$
begin
    for each round $t$ do
        if $p$ knows $r$ and $\texttt{age}(r) \leq c \log n$ then
            $\texttt{age}(r) \rightarrow \texttt{age}(r) + 1$
            $r_0 \rightarrow \texttt{call}(p, t)$
            Send the rumor to $r_0$    (i.e., $p$ pushes the rumor)
        fi
    od
end.
```

Figure 6.1: The push algorithm.

6.2 The Advantage of Push&Pull

First, let us explain the differences in the propagation of the rumor obtained by push transmissions on the one hand and pull transmissions on the other hand.

6.2.1 The Push Scheme

Consider a *push scheme* in which every informed player, in every round, forwards the rumor to the player it calls until all players are informed. We identify the players by the numbers $[n] := \{1, \ldots, n\}$. Let $\texttt{call}(P, t)$ be the player that P calls in round t. Wlog. player 1 is informed at the beginning. We attached to the rumor the age of the rumor and denote this by $\texttt{age}(R)$. When the rumor emerges its age is set to 0. Figure 6.1 exemplifies the push algorithm and Figure 6.2 shows how the push scheme spreads a rumor.

Pittel [Pit87] proved that for $c \geq 1 + \ln 2$ with probability 1 all players will be informed. We will show that a constant choice of c suffices to ensure high probability, i.e., probability $1 - n^{-c'}$ for some constant c'.

Let the random variable s_t denote the number of informed players in round t. Before the first round we have $s_0 = 1$. Formally the random variables $s_0, s_1, s_2, \ldots$ describe a Markov process: there is a finite set of states $[n]$ and the transition probability $P_{\text{push}}(a, b)$ is the same for each round. $P_{\text{push}}(a, b)$ denotes the probability that q players are informed in the next round under the assumption p players are informed in this round. We can compute this probability $P_{\text{push}}(a, b)$ using the equality

$$P_{\text{push}}(a, b) = \begin{cases} p(a, b-a, a) & \text{if } a \leq b \\ 0 & \text{else .} \end{cases}$$

The term $p(s, a, h)$ denotes the probability that s informed player hit a uninformed players with h calls. The following recursion for $s \in [n]$, $b \geq a \geq 1$ and $h \geq 1$ describes this term.

$$\begin{aligned} p(s,0,0) &= 1 , \qquad p(s,a,0) \;=\; 0 , \qquad p(s,0,h) \;=\; \frac{s}{n} \cdot p(s,0,h-1) , \\ p(s,a,h) &= \frac{n-a-s+1}{n} \cdot p(s,a-1,h-1) + \frac{a+s}{n} \cdot p(s,0,h-1) . \end{aligned}$$

Given this Markov process for each given n the speed of convergence can be computed accurately using standard techniques as described in [Sin92]. But this approach does not have an obvious generalization for all n and in fact we use a different approach.

It turns out that the behavior of s_t can rather accurately described if we divide the information spreading process into the following phases (An overview can be found in Figure 6.3).

1. **Startup:** $s_t \leq k \log n$

   We want to estimate the number of rounds necessary to leave this phase with high probability. We will use the following Lemma.

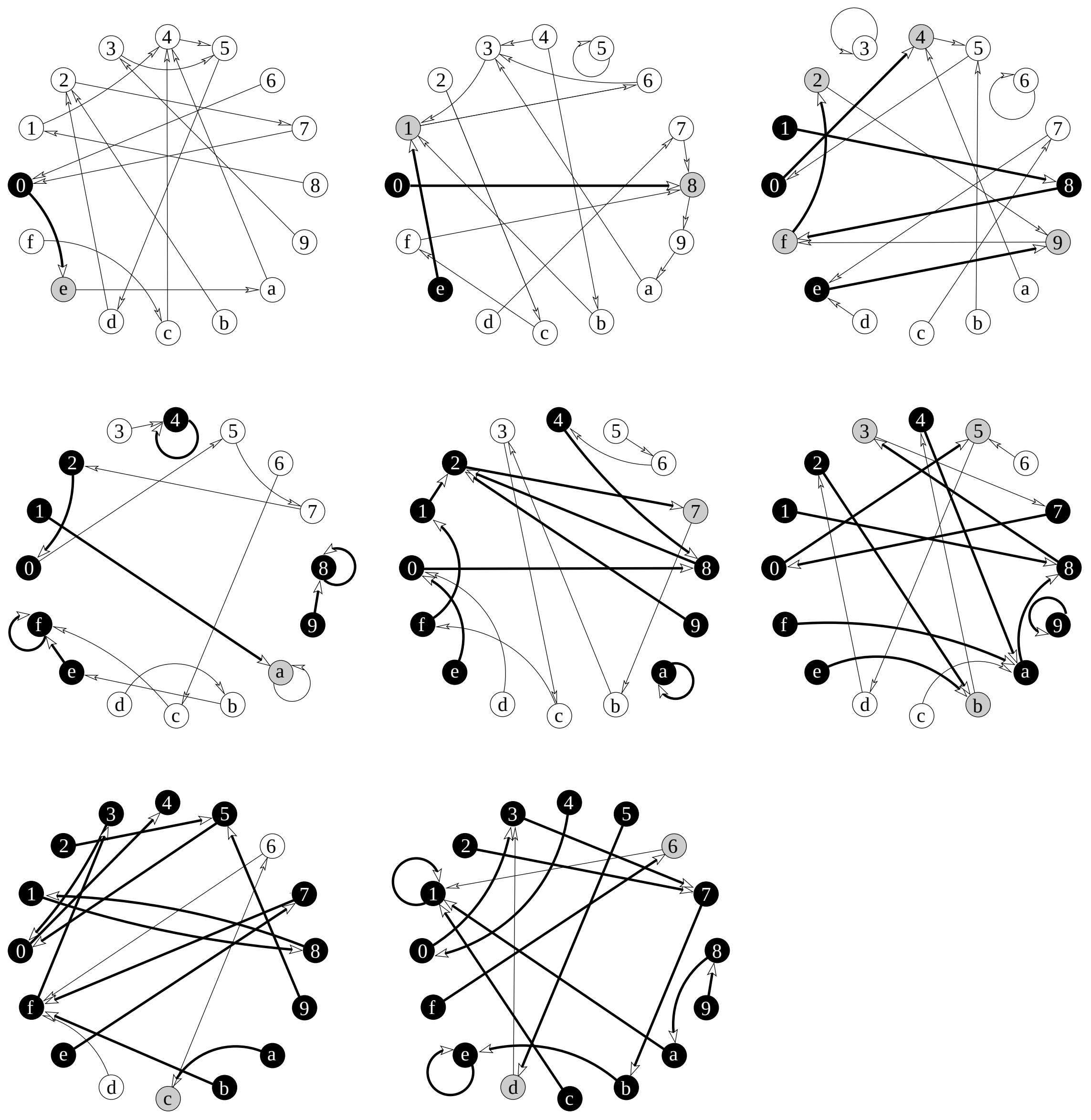
Figure 6.2: A 16 processor network spreading a rumor using the push scheme.

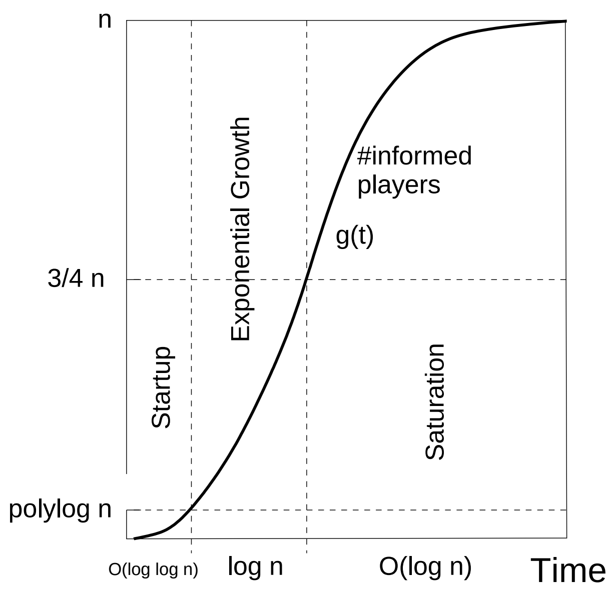

Figure 6.3: The growth of the number of informed players in the push scheme.

Lemma 21 *If all $s \leq k \log n$ informed players* **only push** *their information for c subsequent rounds, then there are at least s new informed players after with probability $1 - O(n^{-c'})$, for $c \geq \frac{s+c'+1}{s-k}$.*

Proof: Each of the s players calls c addresses. We neglect the information spreading impact of newly informed players during these c rounds.

For each round we observe one of the following possibilities for each player.

(a) He can address one of the s originally informed player. This occurs with probability s/n.

(b) He tries to push to the same player as another informed player. The probability of this event is at most s/n.

(c) He hits a player that was informed in a preceding round. Since the phase ends with s additional informed players, this probability is bounded by s/n, too.

Therefore it holds, that in every round a player successfully informs a new one with probability of at least $1 - 3s/n$. The probability that at least $(c-1)s$ pushes fail is therefore at most

$$\begin{aligned}
\sum_{i=cs-s}^{cs} \binom{cs}{i} \left(\frac{3s}{n}\right)^i \left(1 - \frac{3s}{n}\right)^{cs-i} &\leq (s+1)2^{cs} \left(\frac{3s}{n}\right)^{cs-s} \\
&\leq (k \log n + 1)2^{cs} \left(\frac{3s}{n}\right)^{cs-s} \\
&\leq \frac{(k \log n + 1)(3k \log n)^{cs-s}}{n^{cs-s-ck}} \\
&\leq \frac{(k \log n + 1)(3k \log n)^{c'-c+1}}{n^{c'+1}} \\
&\leq O\left(\frac{1}{n^{c'}}\right)
\end{aligned}$$

□

Hence, the number of informed players doubles after every c rounds with probability $1 - O(n^{-c'})$. Therefore after $c \log \log n + c \log k$ rounds the phase is passed with probability $1 - O(n^{-c'+1})$.

2. **Exponential growth:** $k \log n \leq s_t \leq \frac{n}{8}$

   It turns out that in every round the number of informed players increases nearly by a factor two with high probability. In particular we can prove the following lemma.

   Lemma 22 *If* $s \in \{k \log n, \ldots, n/8\}$ *players are informed in a round of the* **push algorithm**, *then for all* $\alpha > 0$ *there are at least* ℓ *new informed players in the next round with probability* $1 - O(n^{-c})$ *if* $k > 2c + 1$*, where*

   (a) if $s \leq n^{1/2}$ *we have* $\ell := s(1 - \frac{3}{\log n})$

   (b) and otherwise we have $\ell := s(1 - \frac{4s}{n+2s}) - c \log n \ \geq \ \frac{3}{5}s - c \log n$.

   Proof: The probability that one of the s players will push to an uninformed player and no other player pushes to this player in the same round is at least $1 - \frac{2s}{n}$. Hence, the probability that a push succeeds independently from the behavior of all other players is at least this probability $1 - \frac{2s}{n}$. Hence, we can use a binomial distribution to estimate the probability that more than $s - \ell$ players fail to inform new players. Therefore an upper bound of this probability is given by the following term.

$$\sum_{i=s-\ell}^{s} \binom{s}{i} \left(\frac{2s}{n}\right)^{i} \left(1 - \frac{2s}{n}\right)^{s-i}$$

   (a) If $s \leq \sqrt{n}$, let $\ell = s(1 - \frac{3}{\log n})$ it holds

$$\begin{aligned}
\sum_{i=s-\ell}^{s} \binom{s}{i} \left(\frac{2s}{n}\right)^{i} \left(1 - \frac{2s}{n}\right)^{s-i} &\leq \sum_{i=s-\ell}^{s} \binom{s}{i} \left(\frac{2s}{n}\right)^{i} \\
&\leq \ell 2^{s} \left(\frac{2s}{n}\right)^{s-\ell} \\
&\leq \ell 2^{s - \frac{1}{2}\log n (s-\ell)} \\
&\leq s 2^{-\frac{1}{2}s} \\
&\leq O(n^{-k/2+1})
\end{aligned}$$

   (b) For $s \geq \sqrt{n}$, let $T(i) := \binom{s}{i}\left(\frac{2s}{n}\right)^{i}\left(1 - \frac{2s}{n}\right)^{s-i}$. Let $\ell' := \ell + c \log n = s - \frac{4s^2}{n+2s}$. Now observe for all $i \geq s - \ell' = \frac{4s}{n+2s}$:

$$\begin{aligned}
\frac{T(i+1)}{T(i)} &= \frac{(s-i)2s}{(i+1)(n-2s)} \\
&\leq \frac{(s-i)2s}{i(n-2s)} \\
&\leq \frac{(s - \frac{4s^2}{n+2s})2s}{\frac{4s^2}{n+2s}(n-2s)} \\
&\leq \frac{(ns - 2s^2)2s}{4s^2(n-2s)} \quad \leq \ \frac{1}{2}
\end{aligned}$$

   Note that $T(s - \ell') \leq 1$ and therefore $T(s - \ell + j) \leq \frac{2^{-j}}{n^c}$.

$$\sum_{i=s-\ell}^{s} \binom{s}{i} \left(\frac{2s}{n}\right)^{i} \left(1 - \frac{2s}{n}\right)^{s-i} \ \leq \ \sum_{i=s-\ell}^{\infty} T(i) \ \leq \ 2n^{-c}$$

□

Until $s \leq \frac{n}{\log n}$ players are informed, the basis of the exponential growth is two. For this note that for $s_t \geq \sqrt{n}$:

$$\begin{array}{rcl} s_{t+1} & \geq & s_t\left(2 - \dfrac{4s_t}{n+2s_t}\right) - c\log n \\ & \geq & s_t\left(2 - \dfrac{4}{\log n + \frac{2}{\log n}} - \dfrac{c\log n}{s_t}\right) \\ & \geq & s_t\left(2 - \dfrac{4}{\log n + \frac{2}{\log n}} - \dfrac{c\log^2 n}{n}\right) \\ & \geq_{ae} & s_t\left(2 - \dfrac{5}{\log n}\right) \end{array}$$

Here the first inequality holds with high probability, while the last inequality holds almost everywhere, i.e., for all $n \geq n_0$ for some n_0. Now observe that applying this recursion for $\log n$ rounds yields:

$$\begin{array}{rcl} s_{t+\log n} & \geq & \left(2 - \dfrac{5}{\log n}\right)^{\log n} s_t \\ & \geq & \left(1 - \dfrac{5}{2\log n}\right)^{\log n} n s_t \\ & \geq & 4^{-\frac{2}{5}} n s_t \end{array}$$

Of course this number of informed players is larger than the precondition allows for the inequality. However it is guaranteed that after $\log n$ steps at least $\frac{n}{\log n}$ players are informed with high probability.

In the following rounds we can the second bound of Lemma 22

$$s_{t+1} \;\geq\; \frac{8}{5}\cdot s_t - c\log n \;\geq_{ae}\; \frac{7}{5}\cdot s_t$$

and after some $O(\log\log n)$ rounds at least $\frac{n}{8}$ players are informed with high probability.

3. **Saturation phase**: $\frac{n}{8} \leq s_t \leq n$

   At about this time the exponential growth of the set of informed players stops. Let us consider the set of uninformed players $u_t := n - s_t$. Once a constant portion of the players are informed, this set shrinks by a constant factor in each round. At the end of the rumor spreading process this factor is about $1 - 1/\mathrm{e}$ since the fraction of players that do not receive a call in a round is given by $\left(1 - \frac{1}{n}\right)^n$ and thus quickly converging to $1/\mathrm{e}$. Thus, the shrinking of the uninformed players takes at least $\ln n$ rounds. Pittel [Pit87] showed that this bound can be achieved with probability 1. But if we want to ensure a higher probability of $1 - n^{-c}$ for some $c > 1$ and consider the case that all but one players are informed, then a straight-forward calculation shows that we need at least $c\ln n$ rounds for this simple task.

   For an upper bound, note that if $\frac{n}{8}$ players are informed every uninformed player is informed in a round with probability $\frac{1}{8}$. Hence, the probability of remaining uninformed after $c\ln n$ rounds is bounded by $\left(\frac{7}{8}\right)^{c\log n} \leq n^{-c\log(8/7)}$. Clearly, all players are then informed with probability $1 - n^{-c\log(8/7)-1}$.

```
Pull algorithm for player $p$ and rumor $r$
begin
    for each round $t$ do
        if $p$ knows $r$ and $\texttt{age}(r) \leq c \log n$ then
            $\texttt{age}(r) \to \texttt{age}(r) + 1$
            for all $p' \in \texttt{called}(p, t)$ do
                Send the rumor to $p'$    (i.e., $p'$ pulls the rumor)
            od
        fi
    od
end.
```

Figure 6.4: The pull algorithm.

6.2.2 The Pull Scheme

Now consider a *pull scheme* in which only called players send the rumor towards the calling players. Let $\texttt{called}(P, t)$ be the set of player that call P in round t. Again player 1 is informed at the beginning and we attach the rumor's age to the transmitted message starting with 0 when the rumor emerges. The pull algorithm is shown in Figure 6.4 and in Figure 6.5 a rumor is spread by this algorithm.

Again we denote by s_t the number of informed players in round t. Again we describe the corresponding Markov process with its transition probability $P_{\text{pull}}(a, b)$, i.e., the probability that if b players are informed in round $t + 1$ under the condition a were informed in round t. We have

$$P_{\text{pull}}(a, b) = \begin{cases} \binom{n-b}{b-a} \cdot \left(\frac{a}{n}\right)^{b-a} \cdot \left(1 - \frac{a}{n}\right)^{n-a} & \text{if } a \leq b \\ 0 & \text{else .} \end{cases}$$

We can derive the following Lemmas directly from this equation.

Lemma 23 *If s players are informed and all $u := n - s$ uninformed players* **only pull** *their information in a round, then for all $d \in \mathbb{N}$, $\epsilon \geq 0$ there are at least $\frac{su}{n}(1 - \epsilon) - d$ new informed players in the next round with probability $1 - \frac{(1-\epsilon)^d}{\epsilon}$.*

Proof: An uninformed player calls an informed player with probability s/n. The number of new informed players can be described by a binomial distribution $B(u, s/n)$. Therefore the probability that at most ℓ players will be informed is exactly

$$\sum_{i=0}^{\ell} \binom{u}{i} \left(\frac{u}{n}\right)^{u-i} \left(\frac{s}{n}\right)^{i}$$

Let $T(i) := \binom{u}{i} \left(\frac{u}{n}\right)^{u-i} \left(\frac{s}{n}\right)^{i}$. Then it holds

$$\frac{T(i+1)}{T(i)} = \frac{s(u-i)}{u(i+1)}$$

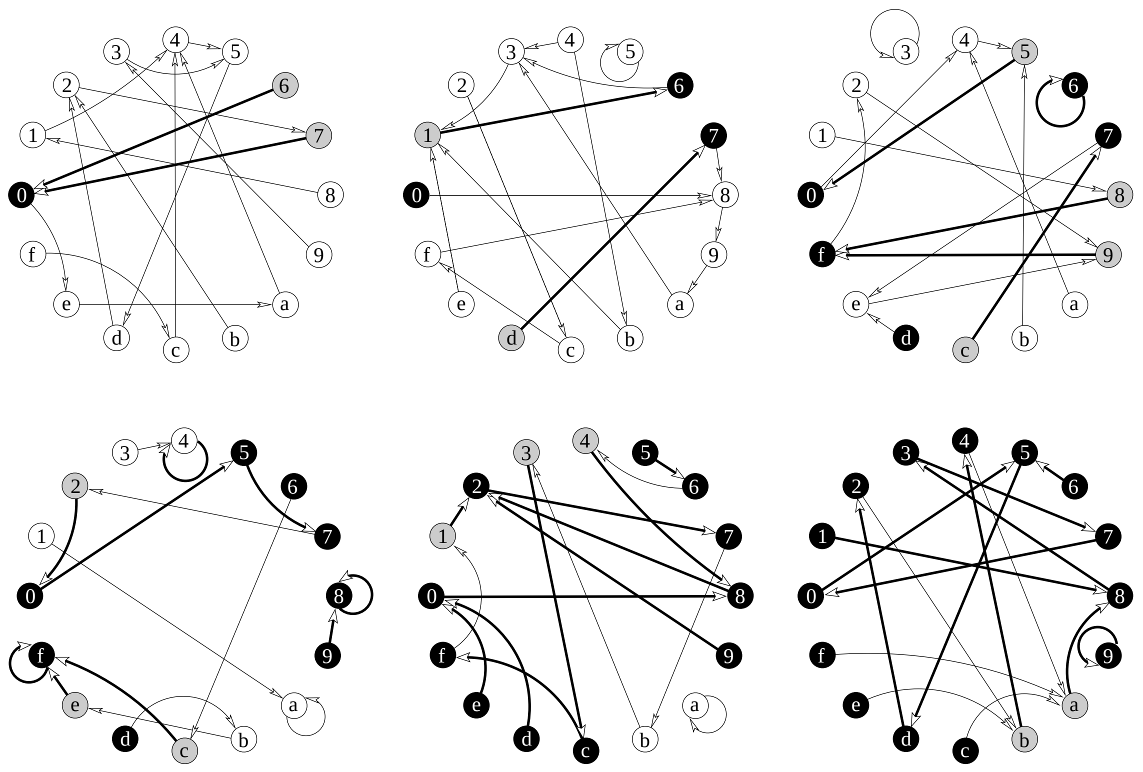

Figure 6.5: A 16 processor network spreading a rumor using the pull scheme.

Let $i \leq \ell := \frac{su}{n}(1-\epsilon) - 1$. Then it follows

$$\begin{aligned}
\frac{T(i+1)}{T(i)} &\leq \frac{s(u-(\ell+1))}{u(\ell+1)} \\
&= \frac{s(nu - su(1-\epsilon))}{su^2(1-\epsilon)} \\
&= \frac{n - s(1-\epsilon)}{u(1-\epsilon)} \\
&= \frac{1+\epsilon\frac{s}{u}}{1-\epsilon} \\
&\geq \frac{1}{1-\epsilon}
\end{aligned}$$

Therefore $T(\ell - j) \leq (1-\epsilon)^j$ and we get:

$$\sum_{i=0}^{\ell-d} T(i) \leq \frac{(1-\epsilon)^d}{\epsilon}$$

□

Lemma 24 *For $k > 0$, if all $u \leq n^{-k}$ uninformed players* **only pull** *their information for some c rounds, then all players are informed with probability $1 - n^{-(c-1)k}$.*

Proof: The probability that an uninformed player makes c continuous calls to uninformed players is n^{-ck}. So the overall probability that any player remains uninformed after c rounds is at most $n^k n^{-ck}$ □

Again we observe that the rumor spreading process can be accurately described by three phases.

1. **Startup:** $s_t \leq \log^3 n$.

   Note that there is a constant probability of $(1-\frac{1}{n})^n \approx \frac{1}{e}$ that a player receives no call by another player in a round. Therefore, we need some $\Theta(\log n)$ rounds to guarantee that a second player receives the rumor. Further note that the standard deviation is at least $\Theta(\log n)$, too. The Chebyshev inequality implies that the startup phase must last at least some $c \log n$ rounds.

   Now we prove that $O(\log n)$ rounds suffice to leave this phase with high probability.

   If $g \leq \log^3 n$ players are informed, then the probability that an uninformed player calls this player is g/n. This happens independently from the calling patterns of the other informed players. Therefore the probability that at most g players fail to call this set S of g informed players within some $\delta \geq \left\lceil c\frac{\ln n}{g}\right\rceil$ rounds is given by the following term.

$$\begin{aligned}
\sum_{i=0}^{g-1}\left(1-\frac{g}{n}\right)^{n\delta - i}\left(\frac{g}{n}\right)^i \binom{i}{n\delta} &\leq g\cdot\left(1-\frac{g}{n}\right)^{n\delta-g}\left(\frac{g}{n}\right)^g\binom{g}{n\delta} \\
&\leq g\cdot\left(1-\frac{g}{n}\right)^{\frac{n}{g}(\delta g - \frac{g^2}{n})}\left(\frac{g}{n}\right)^g (n\delta)^g \\
&\leq e^{\ln g - g\delta + \frac{g^2}{n} + g\ln g - g\ln n + g\ln n + g\ln\delta} \\
&\leq e^{3\ln g - g\delta + \frac{g^2}{n} + g\ln g + g\ln\delta} \\
&\leq e^{-g\delta} e^{3\ln g + \frac{g^2}{n} + g\ln g - g\ln n + g\ln n + g\ln\delta} \\
&\leq e^{-\frac{c\ln n}{2}} \leq n^{-c/2}
\end{aligned}$$

   The last inequality follows since $3\ln g + \frac{g^2}{n} + g\ln g - g\ln n + g\ln n + g\ln\delta \leq c\ln n$ and $g\delta \geq 2c\ln n$.

   The reason why $O(\log n)$ rounds suffice to leave this phase is the following. To inform two players we need $c\ln n$ rounds with high probability. The next four players are informed within $\frac{c\ln n}{2}$ rounds, then eight players within $\frac{c \ln n}{4}$ rounds, and so forth. Hence the overall number of rounds is $2c \ln n$ with probability $1 - n^{-c/2+1}$.

2. **Exponential growth:** $\log^3 n \leq s_t \leq \frac{3}{4} n$

   Until some $n/\log n$ players are informed the growth is exponential to the basis 2. Then it slows down a little bit. From Lemma 23 it follows that in every round additional $s_t \frac{(n-s_t)}{n}(1-\epsilon) - d$ players are informed with probability $1 - \frac{(1-\epsilon)^d}{\epsilon}$. We choose $\epsilon = \frac{1}{\log n}$ and $d = c \log^2 n$ yielding the probability

$$1 - \frac{(1-\epsilon)^d}{\epsilon} = 1 - \frac{(1-\frac{1}{\log n})^{c \log^2 n}}{\frac{1}{\log n}} \geq 1 - e^{-c \log n} \log n \geq 1 - n^{-c/\ln 2 + 1} .$$

   With this probability the number s_t of informed players increases as follows.

   - If $s_t \leq \frac{n}{\log n}$

$$s_{t+1} \ \geq \ s_t + s_t \left(1 - \frac{s_t}{n}\right)(1-\epsilon) - d \ \geq \ 2 s_t \left(1 - \frac{3}{\log n}\right) .$$

     From the same argument as in the exponential growth phase of the push algorithm it follows that this sub-phase lasts at most $\log n + O(1)$ rounds.

   - If $\frac{g}{\log n} \leq s_t \frac{3}{4} n$

$$s_{t+1} \ \geq \ s_t + s_t \left(1 - \frac{s_t}{n}\right)(1-\epsilon) - d \ \geq \ \frac{5}{4} s_t \left(1 - \frac{2}{\log n}\right) .$$

     Hence, this sub-phase requires at most $O(\log \log n)$ rounds.

3. **Quadratic shrinking:** $s_t \geq \frac{3}{4} n$

   From this time on, the pull algorithm has an advantage against the push algorithm as the fraction of uninformed players roughly squares from round to round. This is because in a round starting with $u_t = n - s_t$ uninformed players, each individual player has probability $1 - u_t/n$ to receive the rumor. Here the probability of staying uninformed is u_t/n, resulting in an expected number of $(u_t/n)^2 n$ uninformed players at the end of the round. Thus, we can expect that the shrinking phase only takes $\Theta(\log \log n)$ rounds so that only $\Theta(n \log \log n)$ messages are sent during this phase.

   We can prove this estimation in two steps:

   - $u_t \geq \sqrt{n} \log^2 n$:
     We apply Lemma 23 and choose $\epsilon = \frac{1}{\sqrt{n}}$ and $d = c\sqrt{n} \log n$ and get with probability

$$1 - \frac{(1-\epsilon)^d}{\epsilon} = 1 - \sqrt{n}\left(1 - \frac{1}{\sqrt{n}}\right)^{c\sqrt{n}\log n} \geq 1 - n^{-c/\ln 2 + \frac{1}{2}}$$

     the following inequality

$$\begin{aligned} u_{t+1} &\leq u_t - u_t(1 - \frac{u_t}{n})(1-\epsilon) + d \\ &= u_t \left(\frac{u_t}{n} + \epsilon - \epsilon \frac{u_t}{n}\right) + d \\ &\leq \frac{(u_t)^2}{n}\left(1 + \frac{1}{\log^2 n} + \frac{c}{\log n}\right) \\ &\leq_{ae} 2 \frac{(u_t)^2}{n} \end{aligned}$$

     Note that $\frac{2u_{t+1}}{n} \leq \left(\frac{2u_t}{n}\right)^2$. Hence, after $\log \log n$ rounds the number of uninformed players is smaller than $\sqrt{n} \log^2 n$.

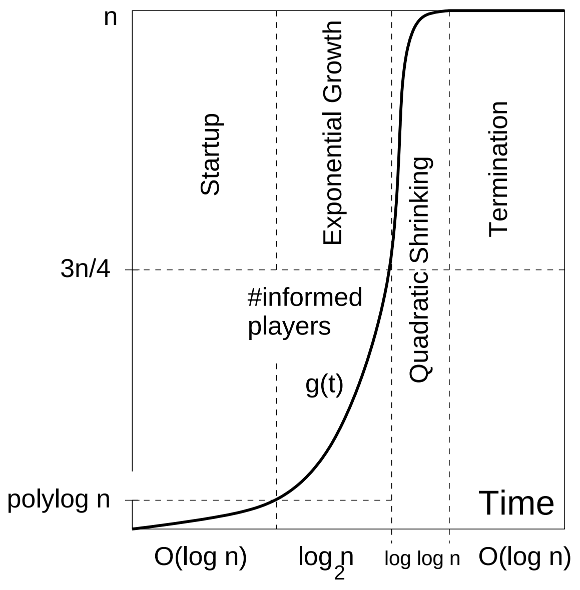

Figure 6.6: The growth of the number of informed players in the pull scheme

```
**Push&Pull algorithm** for player $p$ and rumor $r$
begin
    for each round $t$ do
        if $p$ knows $r$ and $\mathtt{age}(r) \leq t_{\max}$ then
            $\mathtt{age}(r) \rightarrow \mathtt{age}(r) + 1$
            for all $p' \in \mathtt{called}(p,t) \cup \{\mathtt{call}(p,t)\}$ do
                Send the rumor to $p'$    (i.e., $p'$ pushes and pulls the rumor)
            od
        fi
    od
end.
```

Figure 6.7: The push&pull algorithm.

- $u_t \leq \sqrt{n} \log^2 n$:
  Lemma 24 shows that here some constant rounds suffice to inform all players with high probability.

Informing all players does not necessarily mean that the pull algorithm stops. Recall that because of the uncertain startup the exponential growth starts within a time range $\pm\Theta(\log n)$. If we want to ensure that all players are informed after the third phase we have to leave with a fourth phase of length $\Theta(\log n)$ where possibly all players are informed and rumors are still being told.

6.2.3 Push and Pull

In order to combine the predictability of the push scheme with the quadratic-shrinking property of the pull scheme, we simply send the rumor in both directions whenever possible. In detail, our *push&pull scheme* works as follows. The creator of the rumor initiates a time-counter with 0 representing the *age* of the rumor. The age is incremented in each round and distributed with the rumor. In every round every informed player pushes and pulls unless the age of the rumor is higher than $t_{\max} = \log_3 n + O(\log \log n)$. Figure 6.7 shows the algorithm.

To prove Theorem 19 we need the following Lemma.

Lemma 25 *If g informed players* **push and pull** *their information in a round, then for all $\alpha > 0$ with probability $1 - O(n^{-c})$ there are at most $2\frac{v \cdot g}{n} + c \log n$ new informed players who are informed via push and pull, if v was the number of uninformed players getting information from a pull.*

Proof: The probability that an informed player pushes to one of the v players is $\frac{v}{n}$. The probability that i players out of the v players get a call from the g informed players is smaller than the probability that g informed players can call i addresses of v, since some calls can produce the same address. The probability that more ℓ of the v players get a push is smaller than

$$\sum_{i=\ell}^{g} \binom{g}{i} \left(\frac{v}{n}\right)^i \left(1 - \frac{v}{n}\right)^{g-i} .$$

Let $T(i) := \binom{g}{i} \left(\frac{v}{n}\right)^i \left(1 - \frac{v}{n}\right)^{g-i}$ and $i \geq \ell := 2\frac{gv}{n} + 1$. Then it holds

$$\begin{aligned}
\frac{T(i+1)}{T(i)} &\leq \frac{(g-i+1)v}{(i+1)(n-v)} \\
&\leq \frac{(g-\ell-1)v}{(\ell+1)(n-v)} \\
&\leq \frac{(g-2\frac{gv}{n})v}{2\frac{gv}{n}(n-v)} \\
&\leq \frac{n-2v}{2(n-v)} \leq \frac{1}{2} .
\end{aligned}$$

Therefore $T(\ell + j) \leq 2^{-j}$ and thus

$$\sum_{i=\ell+c\log n}^{g} \binom{g}{i} \left(\frac{v}{n}\right)^i \left(1 - \frac{v}{n}\right)^{g-i} \leq 2n^{-c} .$$

□

Now we prove Theorem 19 which states:

Theorem 19 *The simple push&pull-scheme informs all players in time* $\log_3 n + O(\log\log n)$ *using* $O(n \ \log\log n)$ *messages with high probability.*

Proof: We combine the arguments used for the push-algorithm and the pull-algorithm. Let S_t be the set of informed players and U_t the set of uninformed players at the end of round t. Define $s_t = |S_t|$ and $u_t = |U_t|$. We distinguish the following phases.

1. The *startup phase* starts in the round in which the rumor is created and ends with the first round after the execution of which there are at least $\log^3 n$ informed players for the first time. At the beginning of the first round only one player holds the rumor. Alone the push-communication ensures that we need at most some $O(\log\log n)$ rounds to inform $\log^3 n$ players with high probability. This follows directly from the analysis of the startup phase and the exponential growth phase of the push algorithm.

2. The *exponential-growth phase* ends when at least $\frac{n}{\log^2 n}$ players are informed, using the effect of pushing and pulling. From Lemma 23 it follows that in each round additional $\frac{s_t u_t}{n}(1-\epsilon) - d$ players are informed with probability $1 - \frac{(1-\epsilon)^d}{\epsilon}$ from a pull only. We choose $\epsilon := \frac{1}{\log n}$ and $d := (c+1)\log^2 n$. So, the number of informed players by a pull is at least $\frac{s_t u_t}{n}(1-\epsilon) - d \geq s_t(1 - \frac{3}{\log n})$. Let v_t be the number of players informed by pulls. If $v_t \geq 3s_t$ then the number of players has tripled and the performance of this phase follows easily. We will now assume $v_t \leq 3s_t$:

   (a) $s_t^2 \leq n$
   Lemma 22 predicts that the number of players informed by **pull**-operations is at least $s_t(1 - \frac{1}{\log n})$ with sufficiently high probability. From Lemma 25 it follows that the number s_{t+1} of

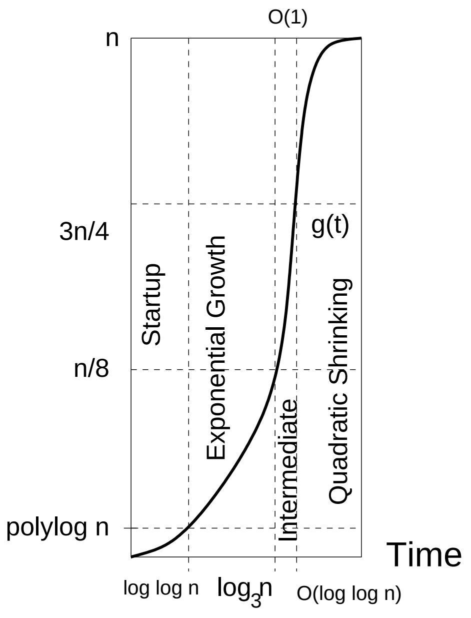

Figure 6.8: The growth of the number of informed players in the push&pull scheme

players informed by a push **and** a pull-operation can be bounded by is $2\frac{v_t s_t}{n} + c\log n \leq \frac{2v_t}{\sqrt{n}} + \frac{cv_t}{\log^2 n} \leq \frac{6s_t}{\sqrt{n}} + \frac{6cs_t}{\log^2 n}$. Hence, the overall number of informed players in the next round can be estimated by

$$s_{t+1} \geq s_t + s_t\left(1 - \frac{3}{\log n}\right) + s_t\left(1 - \frac{1}{\log n}\right) - \frac{2v_t}{\sqrt{n}} - \frac{cv_t}{\log^2 n} \geq 3s_t\left(1 - \frac{5}{\log n}\right) .$$

(b) $s_t^2 \geq n$

From Lemma 23 it follows that the number of players informed by **pull**-operations is at least $s_t\left(1 - \frac{4s_t}{n+2s_t}\right) - c\log n$ with sufficient high probability. From Lemma 25 the following bound holds for the number nodes informed by push **and** pull: $2\frac{v_t s_t}{n} + c\log n \leq \frac{6s_t^2}{n} + c\log n$ and therefore

$$s_{t+1} \geq s_t + s_t\left(1 - \frac{3}{\log n}\right) + s_t\left(1 - \frac{4s_t}{n+2s_t}\right) - \frac{6s_t^2}{n} + c\log n \geq s_t\left(3 - \frac{5}{\log n}\right) .$$

For termination mechanism based on the counter it is important to ensure that this growth phase is not faster than $\log_3 n - \Theta(\log\log n)$. Clearly, in every round at most s_t additional players can be informed by push-communication. Now we estimate the number z_t of players calling a player in S_t. The probability is independently given by s_t/n, describing a binomial distribution $B(u_t, s_t/n)$. Therefore we can use the Chernoff bound to estimate the following probability for $s_t \geq \log^4 n$:

$$\mathrm{P}\left[z_t \geq \left(1 + \frac{1}{\log n}\right) s_t\right] \leq e^{-\frac{s_t}{2\log^2 n(1+\frac{1}{3\log n})}} \leq e^{-\frac{1}{2}\log^2 n} \leq n^{-c}.$$

The upper bound of $\log_3 n - O(\log\log n)$ on the number of rounds follows immediately.

3. Between the exponential growth and the quadratic shrinkage phase we introduce a short **intermediate** phase which ends when at least $\frac{3}{4}n$ are informed. We consider only the influence of pulls. Since the pull algorithm is still in its exponential growth phase this phase lasts some $O(\log\log n)$ rounds with high probability.

4. The *quadratic-shrinking phase* ends with the round after the execution of which there are at least $\sqrt{n}\log^2 n$ uninformed players for the last time. Even if we only take into account pull transmissions we obtain (by following the arguments explaining the general properties of pull algorithms) that
$$\mathbf{E}\left[\frac{u_t}{n}\right] \leq \left(\frac{u_{t-1}}{n}\right)^2 .$$
Now from the analysis of the quadratic-shrinking phase of the pull algorithm we already know that we need $O(\log\log n)$ rounds until the number of uninformed players drops from $\frac{1}{4}n$ to $\sqrt{n}\log^2 n$ with high probability.

5. In the *final phase*, we inform the few remaining uninformed players. Since the number of informed players in this phase is guaranteed to be larger than $n - \sqrt{n}\log^2 n$, each uninformed player has at least probability
$$\frac{n-\sqrt{n}\log^2 n}{n} \;=\; 1-\frac{\log^2 n}{\sqrt{n}}$$
to receive a rumor due to a pull transmission in each round of this phase. Consequently, we need only a constant number of rounds until all players are informed with high probability.

For a round t a simple upper bound for the number of messages caused by push-communication is s_t, while a straight-forward upper bound for the number caused by pull-communication is s_{t+1}. Note that in phases 2 and 3 the number of informed players s_t increases exponentially, bounding the number of messages by $O(n)$. The quadratic shrinking phase starts in round $\log_3 \pm\Theta(\log\log n)$. In this phase and in the final phase the maximum number of messages of $\Theta(n)$ per round needs to be sent. Because we can accurately pin down the starting point of these phases we can ensure with high probability that these last two phases need at most $O(\log\log n)$ rounds and therefore $O(n\log\log n)$ messages. This completes the proof of Theorem 19. □

6.3 The Min-Counter Algorithm

The push&pull algorithm relies heavily on a very exact estimation of the expansion of the set of informed players. The algorithm has to be executed for exactly $\log_3 n + \Theta(\log\log n)$ rounds. For example, a constant fraction of players remains uninformed if the algorithm terminates after $(1-\epsilon)\log_3 n$ rounds, and the algorithm uses $\Theta(n\ln n)$ transmissions when terminating after $(1+\epsilon)\log_3 n$ rounds, for any constant $\epsilon > 0$. A robust algorithm requires a more flexible, distributed termination mechanism that recognizes when all players have been informed.

Shenker proposed a distributed termination mechanism which is not based on the age of the rumor. Let r denote the rumor being considered. Each player holds a counter $\texttt{ctr}(p,r)$ starting at 0 if the player does not know the rumor and we have $\texttt{ctr}(p,r) = 1$ if the rumor r emerges at player p. In Figure 6.9 we show the min-counter algorithm.

We will know prove

Theorem 20: *The min-counter algorithm informs all players in time* $\log_3 n + O(\log\log n)$ *using* $O(n\ \log\log n)$ *messages with high probability.*

Proof: The information spreading process is the same as in the push&pull algorithm, see Table 6.1. We show that there is a choice of c such that no counter exceeds $c\log\log n$ before all players are informed and that after having informed all players all counters will be larger than $c\log\log n$ after some $\Theta(\log\log n)$ rounds.

For this we prove that until at most $\frac{n}{\log^2 n}$ players are informed no player has a counter larger than $\log\log n$ with probability $1-n^{-c}$.

Let $H(t,z)$ denote the number of players who have a counter of at least z for t rounds at the end of this phase. It follows from the strict exponential growth that
$$H(t,1) \leq \frac{n}{b^t\log^2 n}$$

```
**Min-counter algorithm** for player $p$ and rumor $r$
begin
    for each round $t$ do
        if $p$ knows $r$ and $\mathtt{ctr}(p,r) \le c\log\log n$ then
            $\mathcal{Q} \leftarrow \mathtt{called}(p,t) \cup \{\mathtt{call}(p,t)\}$
            for all $p' \in \mathcal{Q}$ do
                Send the rumor to $p'$     (i.e., $p'$ pushes and pulls the rumor)
            od
        if $p$ learns the rumor $r$ then
            $\mathtt{ctr}(p,r) \leftarrow 1$
        fi
        if $p$ knows $r$ and $\mathtt{ctr}(p,r) \le c\log\log n$ then
            if $\exists p' \in \mathcal{Q}$ with $\mathtt{ctr}(p',r) \ge c\log\log n$ then
                $\mathtt{ctr}(p,r) \leftarrow c\log\log n$
            else if $\exists p' \in \mathcal{Q}$ with $\mathtt{ctr}(p',r) \ge \mathtt{ctr}(p,r)$ then
                $\mathtt{ctr}(p,r) \rightarrow \mathtt{ctr}(p,r) + 1$
            fi
        fi
    od
end.
```

Figure 6.9: The min-counter algorithm.

Phase	Informed players	Time	Messages	Growth	Method
Start-up	$[1, \log^3 n]$	$O(\log\log n)$	$O(\log^3 n)$	$s_{t+c} \ge 2s_t$	push
Exponential Growth	$[\log^3 n, \frac{n}{\log n}]$	$\log_3 n$	$O(\frac{n}{\log n})$	$s_{t+c} \ge 3(1 - \frac{5}{\log n})s_t$	push& pull
Intermediate Phase	$[\frac{n}{\log n}, \frac{3}{4}n]$	$O(\log\log n)$	$O(n)$	$s_{t+c} \ge \frac{5}{4}(1 - \frac{2}{\log n})s_t$	pull
Quadratic Shrinking	$[\frac{3}{4}n, n - \sqrt{n}\log^2 n]$	$O(\log\log n)$	$O(n\log\log n)$	$u(t+1) \le 2\frac{(u_t)^2}{n}$	pull
Final Phase	$[n - \sqrt{n}\log^2 n, n]$	$O(1)$	$O(n)$	—	pull

Table 6.1: The phases of the push&pull and min-counter algorithm.

for some basis $b \leq 3 - \frac{5}{\log n}$. Further, the following the arguments used for the analysis of the push&pull algorithm holds for this recurrency with high probability.

$$\frac{H_{t,z+1}}{n} \leq \left(3 + \frac{1}{\log n}\right) \sum_{i \geq t+1} \left(\frac{H_{i,z}}{n}\right)^2 .$$

A straight-forward calculation shows that $H_{1,\log\log n} < 1$ with high probability.

The other phases last some $O(\log\log n)$ rounds, which will be reflected in the appropriate choice of c in the min-counter algorithm. It remains to show that after all players have been informed all counters exceed $c \log\log n$ after some $O(\log\log n)$ rounds. Consider the time point at which all players are informed. Clearly, all counters are at least 1. Then, in every following round each counter is incremented. Clearly, after $c \log\log n$ rounds all counters exceed $c \log\log n$.

Note that this algorithm uses only $O(n \log\log n)$ messages, since it runs at most some $O(\log\log n)$ rounds longer than the push&pull algorithm. $\square$

6.4 The Median-Counter Algorithm

In order to improve the robustness, we devise a distributed termination scheme, called the **median-counter algorithm**, that is provably robust against adversarial node failures as well as stochastic inaccuracies in establishing the random connections. In particular, we show that the efficiency of the algorithm does not rely on the fact that players choose their communication partners uniformly from the set of all players.

Let r denote the rumor being considered. During the course of the algorithm each player v can be in one out of four states A, B, C, or D (with respect to r). State A means the player has not yet received the rumor. In all other states, the player knows the rumor. When a player is in one of the states B or C it pushes and pulls the rumor r along every established connection. In state D the player does not propagate the rumor anymore. Each player in state B holds a counter $\texttt{ctr}(v,r)$. We say a player v is in state B-m if $\texttt{ctr}(v,r) = m$. These counters are irrelevant in other states. The transitions between different states are defined as follows.

We first give an informal description of the median-counter algorithm shown in Figure 6.10.

- State A: The player v does not know r. (For the purpose of analysis, we assume that $\texttt{ctr}(v,r) = 0$ in this state.) If a player v in state A receives r only from players in state B then it switches to state B-1. If a player in state A receives r from a player in state C then it switches to state C.

- State B-m: The player v knows r and $\texttt{ctr}(v,r) = m$. (The player injecting the rumor starts in state B-1.)

  Median rule: If during a round a player v in state B-m receives r from more players in state B-m' with $m' \geq m$ than from players in state A and B-m'' with $m'' < m$ then it switches to state B-$(m+1)$, i.e., increases its counter. There is one exception to this rule. If $\texttt{ctr}(v,r)$ is increased to $\texttt{ctr}_{\max}$ (where $\texttt{ctr}_{\max} = O(\log\log n)$ is a suitable integer) then v switches to state C. Furthermore, if a player in state B receives the rumor from a player in state C then it switches to state C, too.

- State C: Every player stays in this phase for at most $O(\log\log n)$ rounds, and then switches to state D, i.e. it terminates the rumor spreading.

Roughly speaking, the counters in state B are used in order to determine the point in time when the algorithm switches from the exponential-growth phase into the quadratic-shrinking phase. A counter value of $\texttt{ctr}_{\max}$ indicates that $n/\text{polylog}(n)$ players are informed so that it is sufficient to continue the propagation for only $O(\log\log n)$ rounds (which is done in state C). In order to make sure that the median-counter algorithm terminates even in the very unlikely event that the counter mechanism fails, we determine that every player stops propagating the rumor after some fixed number of $O(\ln n)$ rounds, regardless of its current state.

We investigate the robustness of the median-counter algorithm against different sources of errors and inaccuracies.

```
**Median-counter algorithm** for player $p$ and rumor $r$
begin
    for each round $t$ do
        if $p$ is in state $\{B, C\}$ then
            $\mathcal{Q} \leftarrow \texttt{called}(p,t) \cup \{\texttt{call}(p,t)\}$
            for all $p' \in \mathcal{Q}$ do
                Send the rumor to $p'$     (i.e., $p'$ pushes and pulls the rumor)
            od
        fi
        if $p$ learns the rumor then
            $\texttt{state}(p) \leftarrow B$
            $\texttt{ctr}(p) \leftarrow 1$
        fi
        if $p$ is in state $\{B, C\}$ then
            if $\texttt{state}(p) = C$ then
                $\texttt{ctr}(p) \leftarrow \texttt{ctr}(p) + 1$
                if $\texttt{ctr}(p) \geq \texttt{ctr}_{\max}$ then
                    $\texttt{state}(p) = D$
                fi
            else if $\exists p' \in \mathcal{Q}$ with $\texttt{state}(p') = C$ then
                $\texttt{state}(p) \leftarrow C$
                $\texttt{ctr}(p) \leftarrow 1$
            else if $|\{i \in \mathcal{Q} \mid \texttt{ctr}(i) \geq \texttt{ctr}(p)\}| > |\{i \in \mathcal{Q} \mid \texttt{ctr}(i) < \texttt{ctr}(p)\}|$ then
                $\texttt{ctr}(p) \leftarrow \texttt{ctr}(p) + 1$
                if $\texttt{ctr}(p) \geq \texttt{ctr}_{\max}$ then
                    $\texttt{state}(p) \leftarrow C$
                    $\texttt{ctr}(p) \leftarrow 1$
                fi
            fi
        od
    end.
```

Figure 6.10: The median-counter algorithm.

- First, we assume the random connections in each round are established using an arbitrary (possibly non-uniform) probability distribution $\mathcal{D} : V \to [0, 1]$.
- Second, we assume that an oblivious adversary can specify up to F node failures occurring during the execution of the algorithm. The adversary specifies a set $\mathcal{F}$ of players (not containing the player starting the rumor) who fail to exchange information in some of the rounds (as specified by the adversary). We assume $|\mathcal{F}| \leq F$ and $n \sum_{v \in \mathcal{F}} \mathcal{D}(v) \leq F$.

Clearly, we cannot hope to inform all players when allowing adversarial node failures. Therefore, we are satisfied if the algorithm informs all but $O(F)$ players. (Alternatively, one may assume stochastic rather than adversarial failures, e.g., each random phone call fails with probability F/n. In this case, staying for $\tau = \Theta(\log\log n + \ln_{n/F} F)$ rounds in stage C ensures that all players are informed within $O(\ln n + \tau)$ rounds using $O(\tau n)$ transmissions with high probability.)

Theorem 21 *Assuming an arbitrary distribution $\mathcal{D}$ and up to F node failures as described above, the median-counter algorithms informs all but $O(F)$ players in $O(\ln n)$ rounds using $O(n \ \log\log n)$ transmissions with high probability.*

Proof: We start with investigating the errorless case. Let w_i be the probability that a player calls player i, let S_t, s_t, U_t, and u_t be defined as above and let s_t be the weight of all informed players: $g_t := \sum_{i \in S_t} w_i$. Consider the following three phases.

1. **Startup:** We want to ensure that at least $s_t \in \Omega(\ln n)$ informed players with weight $g_t \geq \frac{\log n}{n}$ are established.

   A straightforward analysis shows that $\Theta(\log\log n)$ rounds of push communication suffice to achieve this with high probability. Then, $\Theta(\log\log n)$ rounds of pull-communication establish the wanted number of informed players with high probability.

2. **Exponential growth:** This phase ends when the weight g_t is greater than $\frac{1}{\log n}$.

   In this phase the weight h_t of the set of uninformed players H_t with larger weight than $\frac{1}{g_t}$ is of particular interest:
$$h_t := \sum_{i \in U_t : w_i \geq 1/g_t} w_i \ .$$
   Note that $|H_t| \leq s_t$ and that the probability of a member of H_t being called by an informed player in S_t is larger than the constant $1 - 1/e$. Therefore, push operations cause an increase of the weight of informed players by the amount of $(1-\epsilon)(1-1/e)h_t$ for some constant $\epsilon > 0$ with high probability.

   In $U_t \setminus H_t$ the fraction of players which get only one call in this round is at least $1/e - \epsilon$ for an arbitrary small constant $\epsilon > 0$ with high probability. The probability that one of these players gets the rumor pushed from S_t is $\frac{s_t}{n}$. The expected number of informed players in the next round is therefore
$$\begin{aligned} \mathbf{E}[s_{t+1}] &\geq s_t + \frac{s_t}{n}(1/e - \epsilon)(n - s_t - |H_t|) \\ &\geq s_t \left(1 + (1/e - \epsilon)\left(1 - \frac{2s_t}{n}\right)\right) \ . \end{aligned}$$
   If $s_t \leq \frac{n}{\log n}$ for $h_t \leq \frac{1}{2}$ this implies $s_{t+1} \geq s_t(1+\frac{1}{e}-\epsilon')$ and in the other case $g_{t+1} \geq g_t(\frac{3}{2}-\frac{1}{2e}-\epsilon')$ for some arbitrary small $\epsilon' > 0$.

   So after some $O(\log n)$ rounds we have either $g_t \geq \frac{n}{\log n}$ or $s_t \geq \frac{2n}{\log n}$. In the second case every player with weight larger than $\frac{c \log^2 n}{n}$ is informed in the next round with high probability. Furthermore, the expected weight of all informed players is
$$\mathbf{E}[g_{t+1}] \quad \geq \quad \sum_{i=1}^{n} w_i^2 s_t \ .$$

It turns out that this sum is minimal for the uniform probability distribution. Hence, $\mathbf{E}[g_{t+1}] \geq \frac{s_t}{n}$. Because the weights are upper-bounded we can apply Chernoff bounds and get $g_{t+1} \geq \frac{s_t}{2n} \geq \frac{1}{\log n}$.

For the number of messages note that in all but one round $s_t \leq \frac{2n}{\log n}$. Therefore, the number of messages is bounded by $O(n)$.

Now we discuss how often a counter of a player will be increased during this phase. We consider a player i with weight w_i who is informed during this phase.

(a) $w_i \geq \frac{3 \log n}{n}$
In each round at least $2 \log n$ uninformed players call i, while i receives a call only from at most $\log n$ informed players ($s_t \leq \frac{2n}{\log n}$); i's push call can be neglected. So, this player will communicate with more uninformed than informed players in each round and the median rule prevents an increment of i's counter.

(b) $w_i \leq \frac{3 \log n}{n}$
We allow that during the time interval $\{a, \ldots, b\}$ for which we have $\frac{1}{\log^2 n} \leq w_i s_t \leq c \log n$ the counter of P_i is increased in every round t.

In every round g_t or s_t (but possibly not both) grow by a factor $\alpha > 1$. Nevertheless they interact pairwise, since the expected number of uninformed nodes informed by a pull is $u_t g_t$. Therefore we have $s_{t+1} \geq (1-\epsilon)u_t g_t \geq n g_t(1-\epsilon')$ for $\epsilon, \epsilon' > 0$ with high probability. On the other hand, every informed node pushes in every round such that $g_{t+1} \geq \frac{s_t}{2n \log n}$ with high probability. So, this time interval is bounded by $O(\log \log n)$.

At any time step after b the number of uninformed players calling P_i is higher than the number of informed players calling P_i for the same reasons as in 1.

In every round t before a we concentrate on weights w_i with $w_i \leq \frac{1}{s_t \log^2 n}$. The probability that a player with such a weight is called by an informed player is smaller than $1-(1-\frac{1}{s_t \log^2 n})^{s_t} \leq \frac{1}{\log^2 n}$. Let q_i be the number of the players who increase their counter at least i times before round a and let $q_0 = s_t$. In the worst case all players stay in this situation for the whole phase. Only q_i players can cause an increase for a counter larger than i. The probability that such a player calls another is $\frac{q_i}{s_t \log^2 n}$. Therefore, we have $\mathbf{E}[q_{i+1}] \leq \frac{q_i^2}{s_t \log^2 n}$. It follows $\frac{q_{i+1}}{s_t} \leq c\frac{q_i^2}{s_t^2 \log^2 n}$ if $q_i \in \Omega(\log n)$; and if $q_i \leq O(\log n)$, then $q_{i+c'} = 0$ for some constants c, c' with high probability. This proves $q_{O(\log \log n)} = 0$. So, there are no players whose counters will be increased more than some $c \log \log n$ time during this phase.

3. **Quadratic-shrinking:** This phase ends, when all players have left states A or B.

The probability for each uninformed player to remain uninformed is at most $1 - g_t$, if we consider only pull-communication. Therefore, we have $\mathbf{E}[u_{t+1}] \leq u_t(1-g_t)$, which implies

$$u_{t+1} \leq u_t(1-g_t)\left(1 + \frac{\log n}{\sqrt{n}}\right) \quad \text{with high probability .}$$

The expected weight of the uninformed player of the next round is $\mathbf{E}[1 - g_{t+1}] = (1-g_t)^2$. Note that $\max_{i \in U_t} w_i \leq \frac{c \log^2 n}{n}$. Therefore, applying Chernoff bounds it follows that

$$1 - g_{t+1} \leq (1-g_t)^2 \left(1 + \frac{\log^2 n}{\sqrt{n}}\right) \quad \text{with high probability .}$$

It is clear that after some $O(\log \log n)$ rounds we have $1 - g_{t+1} \leq \frac{2 \log^2 n}{\sqrt{n}}$. Then, some constant number of rounds of pull will sufficiently decrease the probability of an uninformed player remaining in state A.

Since in every round each counter may be incremented only once, it suffices to choose $\mathtt{ctr}_{\max} \geq c \ \log \log n$ for some constant c independent of $\mathcal{D}$.

It remains to show that after some additional $O(\log\log n)$ rounds all counters reach $\mathtt{ctr}_{\max}$. Consider the time point at which all players are informed. Clearly, all counters are at least 1. Then, in every step i each counter is at least $i+1$. Therefore, the distributional algorithm ends after $O(\log\log n)$ rounds.

Since every player produces only one random call in each round the overall number of messages in this phase is bounded by $O(n \ \log\log n)$.

We now focus on the case of $F \le \frac{1}{4}n$ node failures with weight F/n. We assume that if a node failure occurs on v, that v terminates, i.e., switches to state D without learning the rumor. The analysis of the startup and exponential phases can be easily adapted to this case, since the growth of informed nodes proceeds more slowly but still exponentially. We now investigate the situation in the double exponential shrinkage phase.

Let $\mathcal{F}$ be the set of nodes which may be disconnected in some rounds. Then S_t and U_t are defined as the set of informed and uninformed nodes, excluding the nodes in $\mathcal{F}$; u_t, s_t, and g_t are defined as before. The probability that a node remains uninformed is at most $1-g_t$ per round. Therefore, we can conclude that with high probability $u_{t+1} \le (1-g_t)u_t$. Similarly to the error-free case, we can conclude that

$$1-g_{t+1} \le \frac{F}{n} + \left(1+\frac{\log^2 n}{n}\right)(1-g_t)^2 \quad \text{with high probability .}$$

This recursion converges in $O(\log\log n)$ rounds to $1-g_{t'} \in O(\frac{F}{n})$. This implies a maximum number of $O(F)$ uninformed nodes within the next round.

The main problem for the error case is to verify that the number of messages does not grow beyond $O(n \ \log\log n)$. We prove this by showing that at least $O(n/\log n)$ players have reached state C or D, by the time the first error-free players reach state D. The remaining error-free players can only cause $O(\log n)$ messages each, where F faulty players do not add further messages. We start our analysis in the moment when only $F' \in O(F)$ nodes with weight F'/n have remained uninformed. Let us assume that all informed players are in the state B-1.

Let $Z_{t,m}$ be the set and $y_{t,m}$ the weight of error-free nodes in round t with $\mathtt{ctr}(v) = m$. The probability that a node in $Z_{t,m}$ is increased is at least $\sum_{i=m}^{\mathtt{ctr}_{\max}} y_{t,i}$. We want to prove that in the triangular section where $t \le km$ for some constant k, $y_{t,m}$ decreases exponentially in t. For the analysis we allow that some of the counters may be decreased. The aim of this modification is that the series $y_{t,1},\ldots,y_{t,m_t}$ is exponentially increasing, the series $y_{m_t,t}, y_{m_t+1,t},\ldots$ is exponentially decreasing, and the weight $y_{t,m_t+1} \ge \frac{1}{2}$ contains the rest of the weight. More formally, $\forall i \le m_t \ : \ y_{t,i} \le \alpha y_{t,i+1}$ and $y_{t,m_t+1} = 1 - F'/n - \sum_{i=0}^{m_t} y_{t,i}$ for some $\alpha > 1$.

By decreasing some of the counters it can be ensured that in the next round we have $\forall i \le m_t \ : \ y_{t,i} \le \alpha\, y_{t,i+1}$ and $y_{t+1,i} \le \frac{1+\alpha}{2}\, y_{t,i}$. This follows by the fact that $\sum_{i=j}^{m_t} y_{t,j} \ge \frac{1}{2}$ and by reducing the number of players increasing their counter to a fraction of $\frac{1}{2}$ each. After some constant number of rounds c we have $y_{t+c,m_t+1} \ge \alpha y_{t+c,m_t}$. Then, we increase $m_{t+c} := m_t + 1$ and get the claimed triangular section.

Therefore, after some $O(\log\log n)$ rounds only a fraction of $O(n/\log n)$ players has a smaller counter than $c \ \log\log n$. □

6.5 Lower Bound for Address-Oblivious Algorithms

Our first lower bound shows that the two presented push&pull algorithms achieve the best possible results for the class of address-oblivious algorithms. Clearly, any algorithm requires $\Omega(\ln n)$ rounds in order to inform all players. In addition, we show that any address-oblivious algorithm requires $\Omega(n \ \log\log n)$ transmissions, regardless of the number of rounds. We assume the random phone call model using the uniform distribution.

Theorem 22 *Any address-oblivious algorithm guaranteeing that all but a fraction f of the players receive the rumor with constant probability needs to perform $\Omega(n \ \log\log \frac{1}{f})$ transmissions in expectation.*

Proof: Let us fix an address-oblivious algorithm $\mathcal{A}$. Depending on the execution of $\mathcal{A}$, we will partition the rounds into contiguous phases such that the number of transmissions during the first i phases is at least $(i-1)n/4 = \Omega(in)$. (The actual length of the phases depends possibly on the outcome of random experiments influencing the execution of $\mathcal{A}$. Thus, the length of the phases might give some evidence about the outcome of some random experiments. The following statement, however, holds regardless of this evidence.) Let U_i denote the number of uninformed players at the end of phase i, and define $u(i) = n\exp(-2^i + \frac{3}{2})$. We will show by induction that $U_i \geq u(i)$ with high probability. Consequently, $\mathcal{A}$ needs $\Omega(\log\log\frac{1}{f})$ phases and, hence, $\Omega(n \ \log\log f)$ transmissions in order to inform all but a fraction f of the players. Clearly this yields the Theorem.

Phases are defined as follows. Phase 1 starts with the round in which the rumor is generated. If phase i ends in round t then phase $i+1$ starts in round $t+1$. Thus, it remains to describe when a phase ends. We distinguish sparse and dense phases. A *sparse phase* contains at most $n/2$ transmissions. The length of these phases is maximized, that is, a sparse phase ends in round t if adding round $t+1$ to the phase resulted in more than $n/2$ transmissions. A *dense phase* consists of only one round containing more than $n/2$ transmissions. Observe that the number of transmissions during the phases 0 to i is at least $(i-1)n/4$ because any pair of consecutive phases contains at least $n/2$ transmissions by construction.

Now assume by induction that the number of uninformed players at the beginning of phase i is at least $u(i-1)$. We have to show that the number of uninformed players at the end of phase i is at least $u(i)$ with high probability.

For $1 \leq k \leq u(i-1)$, let x_k denote a 0-1 random variable indicating whether the kth of those players who are uninformed at the beginning of round i receives a message containing the rumor during the round. We claim

$$\mathbf{Pr}\left[x_k = 0\right] \geq \frac{u(i-1)}{\mathrm{e}n} \ .$$

The arguments leading to this inequality are different for sparse and dense rounds.

- Suppose phase i is sparse. Then $\mathcal{A}$ sends maximum $\frac{n}{2}$ messages during this phase. Each of these messages is initiated without knowing the receiver because decisions are placed in an address-oblivious fashion. As connections are chosen uniformly at random, the probability that any particular message reaches player k is $\frac{1}{n}$. Consequently, $\mathbf{Pr}\left[x_k = 1\right] \leq \frac{n}{2}\cdot\frac{1}{n} \leq \frac{1}{2}$ so that $\mathbf{Pr}\left[x_k = 0\right] \geq \frac{1}{2} \geq \frac{u(i-1)}{\mathrm{e}n}$.
- Now suppose phase i is dense. Then the phase consists of only one round. In this case, the probability p_1 that player k does not call an informed player is at least $\frac{u(i-1)}{n}$. Furthermore, the probability p_2 that player k is not called by any other player is at least $\frac{1}{\mathrm{e}}$. As these two probabilities are independent, $\mathbf{Pr}\left[x_k = 0\right] = p_1 p_2 \geq \frac{u(i-1)}{\mathrm{e}n}$.

Since $U(i) = \sum_{k=1}^{u(i-1)}(1-x_k)$, we obtain

$$\begin{aligned}
\mathbf{E}\left[U(i)\right] &= \sum_{k=1}^{u(i-1)} \mathbf{Pr}\left[x_k = 0\right] \\
&\geq \frac{u(i-1)^2}{\mathrm{e}n} = \frac{(n\exp(-2^{i-1}+\frac{3}{2}))^2}{\mathrm{e}n} \\
&= n\exp(-2^i + 2) = \sqrt{\mathrm{e}}u(i) \ .
\end{aligned}$$

In particular, $u(i) \leq (1-\frac{1}{3})\mathbf{E}\left[U(i)\right]$. Observe that the random variables x_k are slightly dependent since the random interconnections used for transmissions in phase i form partial permutations on the caller sites. This dependence, however, is negative so that we can apply a Chernoff bound [DR98]. Assuming $u(i) \geq (\ln n)^2$, we obtain

$$\begin{aligned}
\mathbf{Pr}\left[U_i < u(i)\right] &\leq \mathbf{Pr}\left[U_i < (1-\tfrac{1}{3})\mathbf{E}\left[U(i)\right]\right] \\
&\leq \exp(-\tfrac{1}{18}\mathbf{E}\left[U(i)\right]) \\
&\leq \exp(-\tfrac{1}{12}u(i)) = O(n^{-\alpha}) \ ,
\end{aligned}$$

for any positive constant α. This completes the proof of Theorem 22. □

6.6 Lower Bound for General Algorithms

The above lower bound for address-oblivious algorithms does not hold for those rumor spreading algorithms which can base their decisions on the addresses of communication partners. In the introduction, we give an example showing how all players can be informed using only $O(n)$ transmissions. This unrealistic algorithm, however, requires $\Theta(n \ln n)$ rounds. Now we investigate whether there is an algorithm that is both time-optimal (i.e., using only $O(\log n)$ rounds) and communication-optimal (i.e., using only $O(n)$ transmissions). The following lower bound answers this question negatively. Again, we assume the random phone call model using the uniform distribution.

Theorem 23 *Any distributed rumor spreading algorithm guaranteeing that all but a fraction $o(1)$ of the players receive the rumor within $O(\ln n)$ rounds with constant probability needs to perform $\omega(n)$ transmissions in expectation.*

Proof: The difficulty in analyzing arbitrary distributed rumor spreading algorithms is that the distribution of the rumor can be a highly dependent process although the underlying random calling mechanism is generated by n independent experiments in each round. For example, if player 1 is the only player with an odd address sending the rumor to players with even addresses, then the success of the algorithm is highly dependent on the event that player 1 receives the rumor. This small example (not even involving any additional communication) shows that the analysis needs more than simply applying martingales or Chernoff bounds.

Our basic trick in the following analysis is that we choose a random sample of the players who can be guaranteed to act independently. This independence, however, can be guaranteed only for about $\frac{1}{8}\log n$ rounds. Of course, this number of rounds is not enough to inform all players about a rumor initiated by a single player. Therefore, let us assume for the time being that the rumor has already been spread to at least half of the players and we consider the next $T = \lfloor \frac{1}{8}\log n \rfloor$ rounds.

Consider an arbitrary rumor spreading algorithm $\mathcal{A}$. Let $U_V \leq n/2$ denote the number of initially uninformed players. (In order to be able to extend our result to more than T rounds later, we assume that the initially uninformed players are known by all players in the system, e.g., assume that $\{1, \ldots, U_V\}$ is the set of initially uninformed players.) Let X_V denote a random variable describing the number of messages sent during the T rounds. Furthermore, let U'_V denote a random variable describing the number of uninformed players after round T (These randomness of the variable refers to the random phone calls and any kind of other random decisions made by $\mathcal{A}$.)

Let S denote a set of $m = \lfloor n^{1/8} \rfloor$ players chosen randomly from V. The set S will be our random sample. Let U_S denote the random variable describing the number of initially uninformed players in S (with respect to the random choice of S). Let X_S denote a random variable describing the number of messages received by the players in S, and let U'_S denote the random variable describing the number of uninformed players in the set S after the last round. (These random variables are with respect to random decisions of $\mathcal{A}$ and the random choice of S.)

Recall that the communication graph G_t in round t is obtained by a distributed random process, i.e., each player v chooses a player u from V at random and v calls u. This random process generates a probability distribution $\mathcal{D}$ on the set $\mathcal{G}$ of possible communication graphs. Repeating this random process for T rounds extends the probability distribution $\mathcal{D}$ to the sample space $\mathcal{G}^T$.

In many parts of the following analysis we will assume a slightly different probability distribution $\mathcal{D}'$ on $\mathcal{G}$ which is easier to handle than $\mathcal{D}$. Instead of letting each player call a random other player, we assume that the connections are established as follows. In each round t,

- we choose uniformly at random a collection of m disjoint subsets $B_t(v)$ ($v \in S$), each containing m players from $V \setminus S$ (once these sets are chosen, the players in S can act fully independently);
- each player $v \in S$ randomly chooses an integer $\delta(v) \geq 0$ with $\mathbf{Pr}[\delta(v) = i] = \dfrac{1}{\mathrm{e}i!}$ (in the very unlikely case that $\delta(v) \geq m$, set $\delta(v) = m - 1$);
- each player $v \in S$ independently and uniformly randomly chooses a set of $\delta(v) + 1$ different players $u_0(v), \ldots, u_{\delta(v)}(v)$ from $B_t(v)$.

We determine that every player $v \in S$ calls player $u_0(v)$, and the players $u_1(v), \ldots, u_{\delta(v)}(v)$ call v. Every player for whom we have not yet specified whom to call simply chooses a communication partner from $V \setminus S$ independently and uniformly at random. Clearly, $\mathcal{D}$ and $\mathcal{D}'$ are different distributions. The following lemma, however, shows that these distributions are closely related.

Lemma 26 *The total variation distance between $\mathcal{D}$ and $\mathcal{D}'$ on $\mathcal{G}^T$ is $O(n^{-1/4})$.*

Proof:

We show that the total variation distance between $\mathcal{D}$ and $\mathcal{D}'$ on $\mathcal{G}$ (that is, for one round) is $O(n^{-5/8})$ for each round. Consequently, the total variation distance between $\mathcal{D}$ and $\mathcal{D}'$ on $\mathcal{G}^T$ (that is, over all rounds) is $T \cdot O(n^{-5/8}) = O(n^{-1/4})$.

Fix a round t. We start our analysis on the variation distance with proving some useful properties about distribution $\mathcal{D}$. Let Δ denote the set of persons being called by the persons in A, and let Δ_v denote the set of persons calling the person $v \in A$. Each of the following properties holds with probability $1 - O(n^{-1/2})$.

P1) $|\Delta| = m$

P2) $|\Delta(v)| < m, v \in A$

P3) $\Delta \cap A = \emptyset$

P4) $\Delta \cap \Delta(v) = \emptyset, v \in A$

Property P1 follows because the persons being called by the persons in A are selected independently, uniformly at random from V so that the probability that two persons from A call the same person is $m(m-1)/(2n) = O(n^{-1/2})$. P2 follows by applying Chernoff bounds. In particular, $\mathbf{E}[|\Delta(v)|] = 1$ so that

$$\mathbf{Pr}[|\Delta(v)| \geq m] \leq \mathbf{Pr}[|\Delta(v)| \geq m \cdot \mathbf{E}[|\Delta(v)|]] = \exp\left(-\frac{m-1}{3}\right) = O(n^{-1/2}) \ .$$

P3 follows from the fact that Δ and A are randomly selected sets of size at most m so that the probability for a joint element is at most $m^2/n = O(n^{-1/2})$. P4 follows analogously to P3 replacing A with $\Delta(v)$ and assuming P2. All of the properties together, for every $v \in A$, are satisfied with probability $O(m \cdot n^{-1/2}) = O(n^{-5/8})$.

Let $\mathcal{D}^* = \mathcal{D} \mid \text{P1} \cap \text{P3} \cap \text{P4}$ denote the distribution obtained by enforcing the properties P1 , P3, and P4, i.e., restricting the state space $\mathcal{G}$ of $\mathcal{D}$ to communication graphs satisfying these properties and rescaling all probabilities. Let $p = O(n^{-5/8})$ denote the probability for violating one of these properties. Then the rescaling factor is $\frac{1}{1-p} = 1 + O(n^{-5/8})$. Consequently, the variation distance between $\mathcal{D}$ and $\mathcal{D}^*$ is $O(n^{-5/8})$.

The distribution $\mathcal{D}^*$ can be generated by the following process. W.l.o.g., assume $A = \{1, \ldots m\}$. For $1 \leq v \leq m$, let $\Gamma(v)$ denote the set of persons connected to v in the considered round, that is, if $u(v)$ denotes the person called by v, then $\Gamma(v) = \Delta(v) + \{u(v)\}$. The properties P1, P3, and P4 state that the sets $\Gamma(v)$ $(v \in A)$ are disjoint from each other and disjoint from A. Therefore, we can select these sets as follows. First, we select at random the set of nodes called by the persons in A, that is, we choose $\Delta = \{u(v) | v \in A\}$ from $V \setminus A$. Second, we determine the disjoint $\Delta(v)$ sets one by one. This we do by simulating the corresponding probabilities. Define $\delta(v)$ to be a random variable corresponding to the number of persons calling v, i.e., $\delta(v) = |\Delta(v)|$. We choose these variables using the following probabilities:

$$\mathbf{Pr}[\delta(v) = i] = \binom{n - 2m - \sum_{w=1}^{v-1} \delta_w}{i} \left(\frac{1}{n-v+1}\right)^i \left(1 - \frac{1}{n-v+1}\right)^{n-i} \ .$$

After setting $\delta(v)$, we choose at random the set $\Delta(v)$ of size $\delta(v)$ from $V \setminus A \setminus \bigcup_{w=1}^{v-1} \Delta(w)$. Finally, it remains only to choose the persons being called by the persons in $V \setminus \bigcup_{v=1}^{m} \Delta(v)$. For each person w in this set we choose i.u.r. a person in u from $V \setminus A$.

Comparing $\mathcal{D}^*$ with $\mathcal{D}'$, we observe that these distributions deviate only in the following two aspects. The first difference is that we use different probabilities for the random variables $\delta(v)$. Assuming $i \leq m^2$, we can estimate the difference in these probabilities by

$$\begin{aligned}
\mathbf{Pr}\left[\delta(v) = i | \mathcal{D}^*\right] &\geq \binom{n-2m-m^2}{i}\left(\frac{1}{n}\right)^i\left(1-\frac{1}{n-m}\right)^n \\
&\geq \left(\frac{n-2m-m^2}{n}\right)^i\left(1-\frac{1}{n-m}\right)^{m+1}\frac{1}{\mathrm{e}i!} \\
&\geq \left(1-\frac{i(2m-m^2)}{n}\right)\left(1-\frac{m+1}{n-m}\right)\frac{1}{\mathrm{e}i!} \\
&= \mathbf{Pr}\left[\delta(v) = i|\mathcal{D}'\right] - O\left(\frac{m^3}{n}\right) \quad ,
\end{aligned}$$

and

$$\begin{aligned}
\mathbf{Pr}\left[\delta(v) = i | \mathcal{D}^*\right] &\leq \binom{n}{i}\left(\frac{1}{n-m}\right)^i\left(1-\frac{1}{n}\right)^n \\
&\leq \left(\frac{n}{n-m}\right)^i\frac{1}{\mathrm{e}i!} \\
&\leq \left(1+\frac{m}{n-m}\right)\frac{1}{\mathrm{e}i!} \\
&= \mathbf{Pr}\left[\delta(v) = i|\mathcal{D}'\right] + O\left(\frac{m}{n}\right) \quad .
\end{aligned}$$

As a consequence, replacing the probabilities $\mathbf{Pr}\left[\delta(v) = i|\mathcal{D}^*\right]$ by $\mathbf{Pr}\left[\delta(v) = i|\mathcal{D}'\right]$, for $1 \leq v \leq m$, $0 \leq i \leq m-1$ yields a variation distance of $m^2 \cdot O\left(m^3/n\right) = O(n^{-5/8})$.

The second difference between $\mathcal{D}^*$ and $\mathcal{D}'$ is that we map all probabilities for the events $\delta(v) \geq m$ to the event $\delta(v) = m-1$. This change in the distribution makes it possible to choose first the disjoint $B_t(v)$ sets at random and then to select $\Gamma(v)$ at random from $B_t(v)$. The remapping of the probabilities for $\delta(v) \geq m$, however, is covered by a variation distance of $O(n^{-5/8})$ in case of both distributions because

$$\mathbf{Pr}\left[\exists v \in A : \delta(v) \geq m|\mathcal{D}'\right] = m \cdot \sum_{i \geq m}\frac{1}{\mathrm{e}i!} \leq m^{-m+2} = O(n^{-5/8}) \quad ,$$

and $\mathbf{Pr}\left[\exists v \in A : \delta(v) \geq m|\mathcal{D}^*\right] = O(n^{-5/8})$ applying property P2. (Observe that we have stated this property originally for $\mathcal{D}$ but it holds for $\mathcal{D}^*$ with probability $1 - O(n^{-1/4})$ because of our bound on the variation distance between these two distributions.)

Putting it all together, the variation distance between $\mathcal{D}$ and $\mathcal{D}'$ is bounded above by $O(n^{-5/8})$ in each round and $O(n^{-1/4})$ over all rounds. This completes the proof of Lemma 26. □

Based on this bound, we are able to give the following lemma comparing the behavior of the complete system $V|\mathcal{D}$ with that of the small system $S|\mathcal{D}'$.

Lemma 27 *For $\beta \geq 0$, $u \geq n^{-1/16}$, $0 \leq p \leq 1$,*

a) $\mathbf{E}\left[X_V|\mathcal{D}\right] \leq \beta n \quad \Longrightarrow \quad \mathbf{Pr}\left[X_S > \frac{\beta m}{p}|\mathcal{D}'\right] \leq p + O(n^{-1/4})$,

b) $U_V \geq un \quad \Longrightarrow \quad \mathbf{Pr}\left[U_S < \frac{um}{2}\right] = O(n^{-1/4})$, *and*

c) $\mathbf{Pr}\left[U'_S \geq um|\mathcal{D}'\right] < p \quad \Longrightarrow \quad \mathbf{Pr}\left[U'_V < \frac{un}{2}|\mathcal{D}\right] \leq p + O(n^{-1/4})$.

Proof: Result a) is seen easily as follows. The set A is chosen at random. As a consequence, $\mathbf{E}\left[X_A\right] = \mathbf{E}\left[X_V\right] \cdot m/n$, so that $\mathbf{E}\left[X_V|\mathcal{D}\right] \leq \beta n$ implies $\mathbf{E}\left[X_A|\mathcal{D}\right] \leq \beta m$. Now we apply first Lemma 26 and then the Markov inequality which yields

$$\mathbf{Pr}\left[X_A > \frac{\beta m}{p}\,\middle|\,\mathcal{D}'\right] = \mathbf{Pr}\left[X_A > \frac{\beta m}{p}\,\middle|\,\mathcal{D}\right] + O(n^{-1/4}) = p + O(n^{-1/4}) \quad .$$

Hence, result a) is shown. The results b) and c) follow from the following lemma.

Lemma 28 *Suppose V contains $un \geq n^{-15/16}$ uninformed persons. Let $Y \subseteq V$ denote a randomly chosen subset of size $m = \lfloor n^{-1/8} \rfloor$. Then the expected number of uninformed persons in Y is $um \geq n^{-1/16}/2$. Furthermore, for any constant $\delta > 1$, the probability that Y contains less than um/δ or more than δum uninformed persons is $O(n^{-1})$.*

This Lemma is a straightforward consequence of Chernoff bounds. Applying the lemma with $\delta = 2$ yields

- $U_V \geq un \Rightarrow$ **Pr** $\left[U_A < \frac{um}{2}\right] = O(n^{-1})$, and
- **Pr** $[U'_A \geq um|\mathcal{D}] < p \Rightarrow$ **Pr** $\left[U'_V < \frac{un}{2}|\mathcal{D}\right] = p + O(n^{-1})$.

The first of these results corresponds directly to result b), and the second result combined with Lemma 26 yields c). This completes the proof of Lemma 27. □

Informally, this lemma states that it is sufficient to analyze $S|\mathcal{D}'$ in order to estimate $V|\mathcal{D}$. In fact, restricting to the smaller and simpler system $S|\mathcal{D}'$ will enable us to deal with the complex dependencies in the original system $V|\mathcal{D}$. The following lemma summarizes our analysis for $S|\mathcal{D}'$.

Lemma 29 *Let c denote a suitable constant. Suppose $U_S \geq m/\alpha$ and $X_S \leq \beta m$ with $\alpha \geq 4$ and $\beta \geq 1$. Then*

$$U'_S \geq m\alpha^{-\exp(c\alpha\beta)} \quad ,$$

with probability $1 - O(n^{-1/4})$, provided that α, β, c are not too large so that $\alpha^{-\exp(c\alpha\beta)} \geq n^{-1/16}$.

Combining Lemma 27 and 29, we obtain the following result for $V|\mathcal{D}$. Suppose $U_V \leq n/\alpha$ and **E** $[X_V \leq \beta n]$ with $2 \leq \alpha \leq n^{1/16}$ and $\beta \geq 0$. Applying Lemma 27 a) and b) yields

$$X_S \leq \kappa\beta m \quad \text{and} \quad U_S \geq \frac{m}{2\alpha} \quad ,$$

with probability at least $1 - \frac{1}{\kappa} - O(n^{-1/4})$, for any $\kappa > 0$. Now applying Lemma 29 yields

$$U'_S \quad \geq \quad m\alpha^{-\exp(c\alpha(\kappa\beta+1))} \quad ,$$

with probability $1 - \frac{1}{\kappa} - O(n^{-1/4})$. Finally, we can conclude from Lemma 27 c) that

$$U'_V \quad \geq \quad \frac{n}{2}\alpha^{-\exp(c\alpha(\kappa\beta+1))} \quad , \tag{6.1}$$

with probability $1 - \frac{1}{\kappa} - O(n^{-1/4})$. Assuming $n \gg \kappa$, this probability is lower-bounded by $1 - \frac{2}{\kappa}$.

For the time being, let us assume α and β to be constants. Then equation 6.1 can be interpreted as follows. Starting with n/α uninformed players (possibly known by all players), performing $X_v \leq \beta n$ transmissions in $\lceil \frac{1}{8} \log\log n \rceil$ rounds reduces the number of uninformed players only by some constant factor with probability at least $1 - \frac{2}{\kappa}$. Now let us consider the execution of c phases of length at most $\lceil \frac{1}{8} \log\log n \rceil$ each, for any constant $c \geq 1$. Suppose we spend at most βn transmissions in each of these phases. Then the number of uninformed players after all c phases is $\Theta(n)$ with probability $1 - \frac{2c}{\kappa}$. Let us set $\kappa \geq 2c/\epsilon$, for any constant $\epsilon > 0$. Then spending $\Theta(n)$ transmissions in $O(\ln n)$ rounds leaves $\Theta(n)$ uninformed players with probability $1 - \epsilon$. (A rigorous analysis based on inequality 6.1 shows that informing all but a fraction f of the players with constant probability requires **E** $[X_V] = \Omega(\ln^{[2k]} \frac{1}{f})$, where $\ln^{[x]}$ denotes the natural logarithm iterated for x times.) Hence, Theorem 23 is shown. □

Chapter 7

Online Prediction with Partial Feedback

7.1 Introduction

Bandwidth allocation in the Internet is managed by the *Transport Control Protocol (TCP)*. According to this protocol it is the task of the traffic inducing party to regulate the transmission rate of its packets. The only information available to this party is the flow of acknowledgment packets showing whether the message was correctly delivered. If the traffic in the network is larger than the capacity then these acknowledgments will no longer return to the sender of the message and the sender has to cut back on the transmission rate (if he behaves according to TCP requirements). Based on this simple model Karp et al. [KKPS00] investigate protocols optimizing the time needed to find the optimal transmission rate. We follow their approach and consider the following simple game.

On the link there is the problem of the rate of unicast flow from host A to host B. The bandwidth y_t available fluctuates according to varying requirements for bandwidths of other competing flows. Host A determines its packet rate g_t, also called allocated bandwidth, for some unit time period and receives only limited feedback. TCP uses only the fact that some packet drops have occurred. We assume that such a packet drop indicates that in a period t the acquired packet rate g_t is too high, i.e. $g_t > y_t$. Furthermore we assume that this knowledge is available before choosing the packet rate g_{t+1} of the next round. We model this feedback by $f(y, g)$, where $f(y, g) = 1$ if $g \leq y$ and $f(y, g) = 0$ else (see Figure 7.1).

A cost function $\ell(y, g)$ is given and reflects two major components: *opportunity costs* due to sending less than the available bandwidth when $g < y$, and *retransmission delay and overhead* due to dropped packets when $g > y$. The goal of the host A is to minimize the total cost incurred over all periods. In [KKPS00] the following cost models are considered:

1. The *gentle cost function,* $C_\alpha(y, g) = y - g$ when $g \leq y$ and $C_\alpha(y, g) = \alpha(g - y)$ when $g > y$, see Figure 7.2.

   This function models the case where the protocol does not need to wait for lost packets to time out (e.g., the so-called *fast-retransmit* in TCP) so u packets get through to the receiver, but still there is an overhead for detecting and retransmitting some extra packets that are dropped.

2. The *severe cost function,* $S(y, g) = y - g$ when $g \leq y$ and $S(y, g) = y$ when $g > y$, see Figure 7.3.

   This function models the case where the protocol must wait for the first dropped packet to *time out* before resuming transmission. If in a period the first packet is lost then essentially no packets are transmitted during that period and the lost bandwidth can be approximated as y.

In [KKPS00] the static case has been considered when $y_t \in [K]$ is constant in time. The authors show an upper bound of $O(K \log \log K)$ of the total loss for the severe cost function and a lower bound of

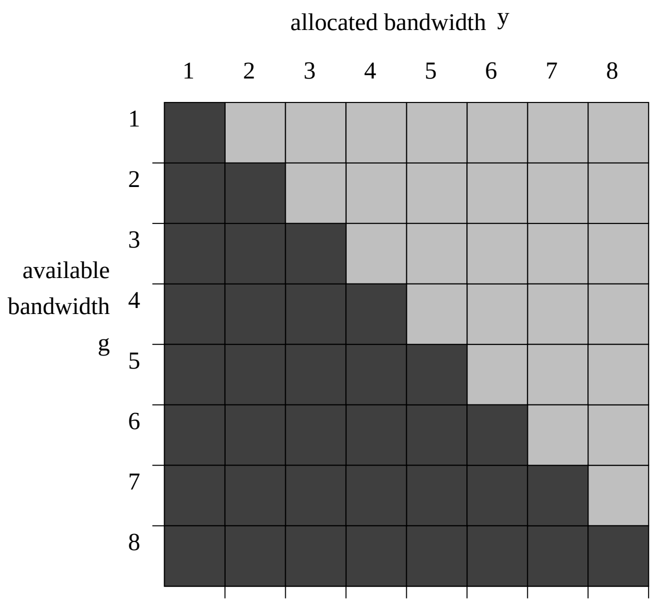

Figure 7.1: The feedback used by TCP.

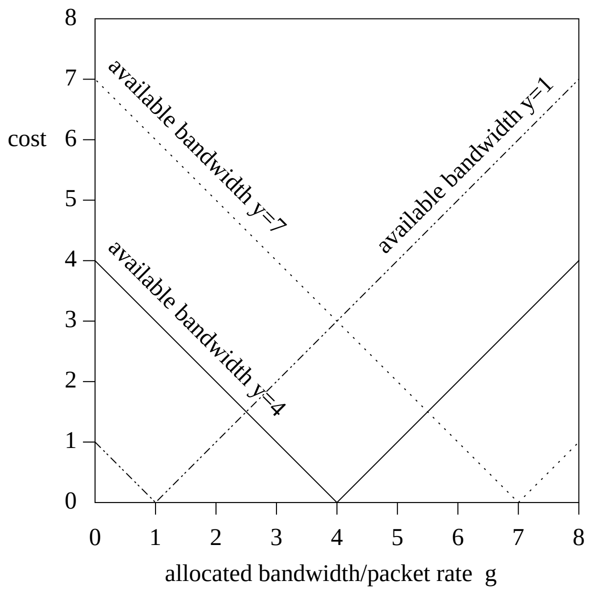

Figure 7.2: The gentle cost function for $\alpha = 1$.

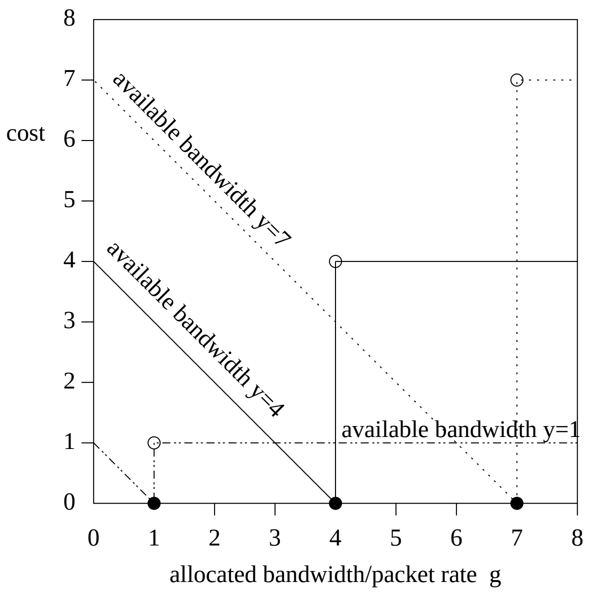

Figure 7.3: The severe cost function.

$\Omega(\frac{K \log\log K}{\log\log\log K})$. For the gentle cost function they show an algorithm with expected cost $\frac{\sqrt{a}n}{2} + O(\log n)$ and worst case cost $\sqrt{a}n + O(\log n)$.

Furthermore they investigate the dynamic case where the change of y_t is restricted. There, the absolute change $|y_{t+1} - y_t|$, the relative change $\max\{\frac{y_{t+1}}{y_t}, \frac{y_t}{y_{t+1}}\}$, or the range of $y_t \in [a, b]$ is bounded. For all these cases there are algorithms bounding the absolute cost. In all these dynamic scenarios the analysis is competitive, i.e. the quality of an algorithm is compared to the cost of the best off-line strategy, while the bandwidth may be chosen by an adversary following the general framework of [BEY98]. In [KKPS00] the following results are shown:

- If $|y_{t+1} - y_t| \leq \alpha$ then there is an algorithm with competitive ratio $4 + \alpha$, while a lower bound of $1 + \alpha$ can be shown;
- if $\max\{\frac{y_{t+1}}{y_t}, \frac{y_t}{y_{t+1}}\} \leq \mu$, then there a variant of TCP with competitive ratio $4\mu - 2$ and there exists a lower bound of μ;
- if $y_t \in [a, b]$, then there is an optimal deterministic algorithm with competitive ratio a/b and the optimal randomized competitive ratio against an oblivious adversary is $1 + \ln(a/b)$.

In this chapter we will discuss the general dynamic case with $y_t \in \{1, \ldots, K\}$ for a randomized algorithm and no restriction on the adversary. For the competitive ratio there is no hope to prove any reasonable result: For these cost functions the best off-line strategy is to choose y_t, causing total cost 0. Now if the adversary chooses K and 1 with equal probability, then the expected cost of any algorithm in the severe cost model is at least $\frac{K-1}{2}$, giving the *competitive difference* (the competitive ratio is not defined since the denominator is 0).

Following an idea of Karp that if the adversarial choice is too good to compete with, one should at least try to compete with the best constant strategy. The interpretation is that the algorithm shall try to perform as well as if the algorithm had acquired the constant bandwidth $g^* \in [K]$, which performs best with the available bandwidth. Although the situation does not seem much better it is now possible to show that in the long run the difference between the cost of the algorithm and the cost of the best constant choice, called *relative cost* or *regret*, is substantially smaller than the KT, i.e. we will show that the *average regret* per round converges against 0. During the investigation of this problem it turned out that our approach generalizes to an arbitrary cost function $\ell(y, g)$ and any function $f(y, g)$ giving sufficient feedback.

An algorithm choosing the bandwidth is fronting a trade-off similar to the one that is the most distinctive trait of the multi-armed bandit problem: on one hand, trying to match the maximum bandwidth at any time step; on the other, choosing the bandwidth in order to collect more information about the available bandwidth.

Another, even simpler, instance of this general setting arises from a simple quality control problem also known as the apple tasting problem. In a manufacturing operation, the items produced can be either working or defective. Unfortunately, to assess the quality of an item it is necessary to destroy it. Both delivering a defective item and destroying a working one are undesirable events. Suppose that customer feedback is unavailable, late or unreliable. The only information available about the sequence of items produced so far is the one the destructive testing procedure provides, but we want to apply it as little as possible. When the plant is working properly, defective items are extremely rare so that little testing seems optimal, but a failure would be detected with a worrisome delay.

The goal we set for ourselves was to make these two examples, together with the multi-armed bandit problem and others, fit a general framework that encompasses different sequence prediction games where the prediction is based only on some "clues" about the past rounds of the game and good predictions are rewarded according to some weighting scheme. We model the available feedback on the sequence as a function of two arguments. One is the current sequence value itself, as it is common in system theory and the other is the prediction itself. In system theory the classic problem is that of observability: Is the feedback sufficient to find out the initial state of the system whose transition function is assumed to be known? More closely related to our problem is that of learning from noisy observations, where the sequence is obfuscated by some noise process as opposed to a deterministic transformation. The presence of the second argument, the prediction, makes our approach consistent with a large body of work in the sequence prediction literature, where the feedback is the reward. Decoupling the feedback and reward functions is the most notable feature of our approach.

Following a relatively recent trend in sequence prediction research (e.g. see [LW94, HKW95, Vov98, Sch99, Sch01, CBFH$^+$97, CBFHW94, HKW98, CBL99, FS99, ACBFS95, Vov99]) we make no assumptions whatsoever concerning the sequence to be predicted, meaning that we do not require, for instance, a statistical model of the sequence. For lack of a model, we need to assume that the sequence is arbitrary and therefore generated by an all-powerful device or adversary, which, among other things, is aware of the strategy a prediction algorithm is using. It might seem that competing with such a powerful opponent is hopeless. This is why, instead of the absolute performance of a prediction algorithm, it is customary to consider the regret w.r.t. the best predictor in some class. In this paper we make the choice of comparing our algorithm against the best constant predictor. Even if it seems a very restrictive setting, let us remind the reader that the best constant prediction is picked after the whole sequence is known, that is with a much better knowledge than any prediction algorithm has available and even more so in the incomplete feedback setting. Moreover, a constant predictor can output a mixed strategy, that is, not a constant outcome but a constant distribution on all possible outcomes. Finally, constant predictors are the focus of an important line of research on iterated games [Han57, FS99, ACBFS95].

Our research is closely related to the one presented in [FS99] where the subject is, indeed, the problem of learning a repeated game from the point of view of one of the players —which can be thought of, indeed, as a predictor, once we accept that prediction can be rewarded in general ways and not according to a metric. In that work the authors designed the Multiplicative Weighting algorithm and proved that it has regret $O(\sqrt{T})$ when compared against the optimal constant strategy. This algorithm is used as a subroutine of ours. In their setting the predictor receives as input not the sequence at past rounds but the rewards every alternate prediction (not only the one made) would have received. Since this is all that matters to their algorithm, this setting is called *full information game* in [ACBFS95], even if, according to our definitions, the sequence and not the reward is the primary information. In the latter paper, a *partial information game* corresponds to the multi-armed bandit problem, in which only the reward relative to the prediction made is known to the predictor. What would have happened picking any of the other choices remains totally unknown. The best bound on the regret for this problem has been recently improved to $O(T^{1/2})$ [Aue00].

The apple tasting problem as the simplest special case of our prediction problem and the multi-armed bandit problem has been very well investigated by Helmbold, Littlestone and Long [HLL92, HLL00]. They analyze the same prediction strategies and show that there is a prediction algorithm within an expected regret of at most $O(\frac{T^{1/2}}{\log n})$. Furthermore they show a matching asymptotic lower bound.

In this chapter we extend this result to our more general setting, provided that the feedback and loss functions jointly satisfy a simple but non-trivial condition. This case includes relevant special cases such as the bandwidth allocation and quality control problems mentioned at the beginning of the present section, as well as the classic multi-armed bandit problem and others. In this case it is possible to prove a bound of $O(T^{3/4})$ on the regret. The aforementioned condition is not specific to our algorithm: Indeed we proved that, when it is not satisfied, any algorithm incurred a regret $\Omega(T)$, just as a prediction with no feedback at all.

Also closely related is the work presented in [WM01] where the same worst case approach to sequence prediction is assumed, but the sequence is available to a prediction algorithm only through noisy observations. Albeit very general, their results make some assumptions on the noise process, such as statistical independence between the noise components affecting observations at different time steps. Our feedback model encompasses also the situation of noisy observations, but gives up any statistical assumptions on the noise process, too, in analogy with the notion of “malicious errors” in the context of PAC learning [KL93]. That is we claim our work can be seen also as a worst case approach to the prediction of a noisy sequences.

The chapter is structured as follows. In Section 7.2 we formally describe the problem. In Section 7.3 we describe the basic algorithm and prove bounds on its performance. In Section 7.4 we review some examples and highlight some shortcomings of the basic algorithm and show how to overcome them. In Section 7.5 we present a general algorithm and prove that the algorithm is essentially the most general. In Section 7.6 we discuss our results.

7.2 The Model

We describe the problem as a game between a player choosing an action g_t and an adversary choosing the action y_t at time t. There are K possible actions available to the player, without loss of generality from the set $[K] = \{1, \dots, K\}$, and R actions in the set $[R]$ from which the adversary can pick from. At every time step the player suffers a loss equal to $\ell(y_t, g_t) \in [0, 1]$.

The game is played in a sequence of trials $t = 1, 2, \dots, T$. The adversary has full information about the history of the game, whereas the player only gets a feedback according to the function $f(y, g)$. Hence the $R \times K$-matrices L and F, with $L_{ij} = \ell(i, j)$ and $F_{ij} = f(i, j)$ completely describe an instance of the problem. At each round t the following events take place.

1. The adversary selects an integer $y_t \in [R]$.

2. Without knowledge of the adversary’s choice, the player chooses an action by picking $g_t \in [K]$ and suffers a loss $x_{g_t}(t) = \ell(y_t, g_t)$.

3. The player observes $f_t = f(y_t, g_t)$.

Note that due to the introduction of the feedback function this is a generalization of the partial information game of [ACBFS95].

Let $W(T) := \sum_{t=1}^{T} x_{g_t}(t) = \sum_{t=1}^{T} \ell(y_t, g_t)$ be the total loss of player A choosing $g_1, \dots, g_T$. We measure the performance of the player by the expected *regret* R_A, which is the difference between the total loss of A and the total loss of the best constant choice g^*, that is $\sum_{t=1}^{T} x_{g_*}(t)$.

$$R_A := \max_{y_1,\dots,y_T} \mathbf{E}\left[\sum_{t=1}^{T} \ell(y_t, g_t) - \min_j \sum_{t=1}^{T} \ell(y_t, j)\right]$$

where each y_i is a function of $g_1, \dots, g_{t-1}$. In some works the corresponding min-max problem is investigated, transforming the loss into a reward. The two settings are equivalent, as it is easy to check.

```
Multiplicative Weighting Algorithm (MW)
input $K$
constant $\eta \in (0,1)$
begin
  Initialize $p_i(1) := \frac{1}{K}$ for all $i \in \{1,\dots,K\}$.
  for $t$ from $1$ to $T$ do
    Choose $g_t$ according to probabilities $p(t)$.
    Receive the loss vector $x(t)$
    $Z_t := \sum_{i=1}^{K} \frac{p_i(t)}{\exp(\eta x_i(t))}$
    for $i$ from $1$ to $K$ do
      $p_i(t+1) := \frac{p_i(t)}{\exp(\eta x_i(t)) Z_t}$
    od
  od
end
```

Figure 7.4: The multiplicative weighting algorithm.

7.3 The Basic Algorithm

For the full information case the following *Multiplicative Weighting Algorithm* (see Figure 7.4) has been used in different settings and has recently been analyzed in [ACBFS95]. Figure 7.5 shows the *Hedge Algorithm* which is their setting. The following Lemma shows the equivalence of both algorithms.

Lemma 30 *Hedge and MW are equivalent algorithms.*

Proof: By induction, we prove that in each round the probability $p_i(t)$ computed by MW is identical to the probability $\tilde{p}_i(t)$ of the Hedge algorithm.

Note that for all $i, j \in [K]$ we have for the MW algorithm

$$\frac{p_i(t)}{p_j(t)} = \frac{p_i(t-1)}{p_j(t-1)} \frac{\exp(\eta x_j(t))}{\exp(\eta x_i(t))} = \frac{\exp(\eta L_j(t))}{\exp(\eta L_i(t))} \; .$$

We have for the hedge algorithm

$$\frac{\tilde{p}_i(t)}{\tilde{p}_j(t)} = \frac{\exp(\eta L_j(t))}{\exp(\eta L_i(t))} \; .$$

□

This Lemma implies $\prod_{i=1}^{t} Z_i = W_t$, since MW defines $p_i(t+1) = \frac{\exp(-\eta L_i(t))}{\prod_{i=1}^{t} Z_i}$ and hedge defines $p_i(t+1) = \frac{\exp(-\eta L_i(t))}{W_t}$.

The analysis of [FS99] leads to a tight result for the full knowledge model. We will base our analysis on an adaption of their main theorem. Let us define

$$\Phi(x) \; := \; e^x - 1 - x .$$

Lemma 31 *For all $z \in [-1,1]$:*

$$\frac{z^2}{e} \; \leq \; \Phi(z) \; \leq \; (e-2)z^2 \; .$$

Proof: This Lemma follows by applying elementary methods of analysis. By considering the first and second derivative it turns out that for the function $g(x) := \Phi(x) - (e-2)x^2$ for all $x < 0$ we have $g(x) < 0$. Furthermore it is easy to see that there exists an interval $[0, x_0]$ such that $g(x) \leq 0$ and if $x > x_0$ we observe $g(x) > 0$. It turns out that $x_0 = 1$ since $g(0) = g(1) = 0$, which proves the right inequality.

By a similar consideration it turns out that for $h(x) := \Phi(x) - \frac{x^2}{e}$ there exists an interval $[x_0, 0]$ where $h(x) \geq 0$, for $x > 0$ we have $h(x) > 0$, and for $x < x_0$ we have $h(x) < 0$. We have $h(-1) = h(0) = 0$ proving the left inequality. □

```
Hedge Algorithm
input $K$
constant $\eta \in (0,1)$
begin
    $p_i(1) := \frac{1}{K}$, for all $i \in [K]$.
    $L_i(0) := 0$, for all $i \in [K]$.
    for $t$ from $1$ to $T$ do
        Choose $g_t$ according to probabilities $p_i(t)$.
        Receive the loss vector $x(t)$
        for $i$ from $1$ to $K$ do
            $L_i(t) := L_i(t-1) + x_i(t)$
        od
        $W_t := \sum_{i=1}^{K} \frac{1}{\exp(\eta L_i(t-1))}$
        for $i$ from $1$ to $K$ do
            $p_i(t+1) := \frac{1}{\exp(\eta L_i(t)) W_t}$
        od
    od
end
```

Figure 7.5: The hedge algorithm.

Lemma 32 *For all $\eta \in [-1,1]$, and for all $x \in [-1,1]$:*

$$1 + \eta x + \frac{1}{2}\Phi(\eta)x^2 \quad \le \quad e^{\eta x} \quad \le \quad 1 + \eta x + 2\Phi(\eta)x^2 \ .$$

Proof: The proof uses Lemma 31 for $z \in [-1,1]$:

$$\frac{z^2}{e} \ \le \ e^z - 1 - z \ \le \ (e-2)z^2 \ .$$

Note that $\eta x \in [-1,1]$. Then, we have

$$\frac{1}{2}\Phi(\eta)x^2 \ \le \ \frac{1}{e(e-2)}x^2(e^\eta - 1 - \eta) \ \le \ \frac{1}{e}x^2\eta^2 \ \le \ e^{\eta x} - 1 - \eta x$$

and

$$e^{\eta x} - 1 - \eta x \ \le \ (e-2)x^2\eta^2 \ \le \ e(e-2)x^2(e^\eta - 1 - \eta) \ \le \ 2\Phi(\eta)x^2 \ .$$

□

The following theorem establishes a bound on the performance of MW that holds for any loss function ℓ.

Theorem 24 *[FS99] For $\eta \in (0,1)$, for any loss matrix L with R rows and K columns with entries in the range $[0,1]$ and for any sequence $y_1, \ldots, y_T$ the sequence of $p(1), \ldots, p(T)$ produced by algorithm MW satisfies*

$$\sum_{t=1}^{T}\sum_{i=1}^{K} \ell(y_t, i)p_i(t) \ \le \ \min_j \sum_{t=1}^{T} \ell(y_t, j) + \frac{2\Phi(\eta)}{\eta}\sum_{t=1}^{T}\sum_{i=1}^{K} p_i(t)\ell(y_t, i)^2 + \frac{\ln K}{\eta} \ .$$

Proof: We will prove that, for any choice of $q(1), \ldots, q(T)$ where $q(t)$ represents a probability distribution over $[R]$ and where $q_i(t)$ is the probability of action i in step t as well as for any vector p representing a probability distribution over $[K]$, we have:

$$\sum_{t=1}^{T} q_t^T \, L \, p(t) \le \sum_{t=1}^{T} q_t^T \, L \, p + \frac{2\Phi(\eta)}{\eta}\sum_{t=1}^{T}\sum_{i=1}^{K} p_i(t)(q_t^T \, L \, e_i)^2 + \frac{\ln K}{\eta} \ .$$

We use the *Kullback-Leibler divergence*, also called *relative entropy*, which is defined for probability distributions over $\{1, \ldots, K\}$ by

$$\mathrm{RE}(P \,||\, Q) \;:=\; \sum_{i=1}^{K} P_i \ln\left(\frac{P_i}{Q_i}\right) .$$

Lemma 33 *For any iteration t where MW is used with parameter $\eta > 0$ and for any probability vector p*

$$\begin{aligned}\mathrm{RE}(p \,||\, p(t+1)) - \mathrm{RE}(p \,||\, p(t)) \quad &\geq \quad \eta\, q^T(t) L p - \eta\, q^T(t) L p(t) \\ &\quad + 2\Phi(\eta) \sum_{i=1}^{K} p_i(t)\ \left(q(t)^T L e_i\right)^2 .\end{aligned}$$

Proof:

$$\begin{aligned}&\mathrm{RE}(p \,||\, p_{t+1}) - \mathrm{RE}(p \,||\, p(t)) \\ &\quad= \sum_{i=1}^{K} p_i \ln \frac{p_i}{p_i(t+1)} - \sum_{i=1}^{K} p_i \ln \frac{p_i}{p_i(t)} \\ &\quad= \sum_{i=1}^{K} p_i \ln \frac{p_i(t)}{p_i(t+1)} \\ &\quad= \sum_{i=1}^{K} p_i \ln Z_t \exp(\eta\, q(t)^T L e_i) \\ &\quad= \eta \left(\sum_{i=1}^{K} p_i\, q(t)^T L e_i\right) + \ln Z_t \\ &\quad= \eta\, q(t)^T L p + \ln\left(\sum_{i=1}^{K} \frac{p_i(t)}{\exp(\eta x_i(t))}\right) \\ &\quad\leq \eta\, q(t)^T L p + \ln\left(\sum_{i=1}^{K} p_i(t) \left(1 - \eta x_i(t) + 2\Phi(\eta) x_i(t)^2\right)\right) \\ &\quad\leq \eta\, q(t)^T L p + \ln\left(1 - \sum_{i=1}^{K} \eta p_i(t) x_i(t) + \sum_{i=1}^{K} 2 p_i(t) \Phi(\eta) x_i(t)^2\right) \\ &\quad\leq \eta\, q(t)^T L p - \eta q(t)^T L p(t) + \sum_{i=1}^{K} 2 p_i(t)\ \Phi(\eta)\ \left(q(t)^T L e_i\right)^2\end{aligned}$$

The last lines use the fact that $\ln(1+x) \leq x$ for any $x > -1$ and Lemma 32. □

We sum over t:

$$\begin{aligned}\eta \sum_{t=1}^{T} q(t)^T L p(t) - \eta \sum_{t=1}^{T} q(t)^T L p - 2\Phi(\eta) \sum_{t=1}^{T} \sum_{i=1}^{K} p_i(t) \left(q(t)^T L e_i\right)^2 \\ \leq \quad \mathrm{RE}(p \,||\, p(1)) - \mathrm{RE}(p \,||\, p(T+1))\end{aligned}$$

Noting that $\eta > 0$, $\mathrm{RE}(p \,||\, p(t+1) \geq 0$, and $0 \leq \mathrm{RE}(p \,||\, p(1)) \leq \ln K$ for all p gives the claim. □

Our algorithm relies on the existence of a $K \times K$ matrix G satisfying the following equation:

$$F\,G = L\ .$$

For instance, if F is non singular, this property can be fulfilled by choosing $G = F^{-1}L$. If such a G does not exist the basic algorithm fails, i.e. it cannot compute a strategy at all.

```
FeedExp3
input $F, L$
begin
    Compute $G$ such that $F\,G = L$.
    Choose $\eta, \gamma \in (0,1)$ according to $G$.
    Initialize $p(1)$ with $p_i(1) := \frac{1}{K}$ for all $i \in \{1,\ldots,K\}$.
    for $t$ from $1$ to $T$ do
        Select action $g_t$ to be $j$ with probability
            $\hat{p}_j(t) := (1-\gamma)p_j(t) + \frac{\gamma}{K}$.
        Receive as feedback the number $f_t$.
        for $i$ from $1$ to $K$ do
            $\hat{x}_i(t) := \dfrac{f_t\, G_{g_t,i}}{\hat{p}_{g_t}(t)}$
        od
        $Z_t := \sum_{i=1}^{K} p_i(t)\exp(\eta\hat{x}_i(t))$
        for $i$ from $1$ to $K$ do
            $p_i(t+1) := p_i(t)\,\dfrac{\exp(\eta\hat{x}_i(t))}{Z_t}$
        od
    od
end
```

Figure 7.6: The feedback exponential exploration and exploitation algorithm.

The algorithm can be described as follows. First, it estimates the loss vector using the matrix G and the feedback. This estimate is fed into the MW algorithm which returns a probability distribution on the player's actions. MW tends to choose an action with very low probability if the associated loss over the past history of the game is high. This is not acceptable in the partial information case, because actions are useful also from the point of view of the feedback. Therefore, and again in analogy with [FS99], the algorithm adjusts the distribution $p(t)$, output by the MW algorithm, to a new distribution $\hat{p}(t)$ such that $\hat{p}_i(t) \geq \frac{\gamma}{K}$ for each i. We will give an appropriate choice of γ and other parameters affecting the algorithm later on. What is new to this algorithm and what makes it much more general, is the way the quantities $x_i(t)$ are estimated. More in detail, given F and L and assuming there is a G such that $FG = L$, our basic algorithm works as shown in Figure 7.6.

The following lemma shows that $\hat{x}(t)$ is an unbiased estimator of the loss vector $x(t)$.

Lemma 34 *For all i, t we have*

$$\mathbf{E}[\hat{x}_i(t)|g_1,\ldots,g_{t-1}] = x_i(t) \quad \textit{and} \quad \mathbf{E}[\hat{x}_i(t)] = \mathbf{E}[x_i(t)]\ .$$

Proof: Note that

$$\mathbf{E}[\hat{x}_i(t)|g_1,\ldots,g_{t-1}] = \sum_{j=1}^{K} \hat{p}_j(x)\frac{F_{y_t,j}}{\hat{p}_j(x)}G_{j,i} = \sum_{j=1}^{K} F_{y_t,j}G_{j,i} = L_{y_t,i} = x_i(t)\ .$$

This implies $\mathbf{E}[\hat{x}_i(t)] = \mathbf{E}[\mathbf{E}[\hat{x}_i(t)|g_1,\ldots,g_{t-1}]] = \mathbf{E}[x_i(t)]$. □

Let $S_{y,i}(g) := F_{y,g}G_{g,i}$, for all $y, i, g \in \{1,\ldots,K\}$, $S^+ := \max_{y,g,i}\{S_{y,i}(g)\}$, $S^- := \min_{y,g,i}\{S_{y,i}(g)\}$, $\sigma := \max 0, -S^-$ and $\rho := S^+ - S^-$.

Lemma 35 *For any sequence $y_1,\ldots,y_T$ the sequence $\hat{p}(1),\ldots,\hat{p}(T)$ produced by FeedExp3 satisfies for all j:*

$$\sum_{t=1}^{T}\sum_{i=1}^{K}\hat{x}_i(t)\hat{p}_i(t) \ \leq\ \sum_{t=1}^{T}\hat{x}_j(t) + \frac{2\Phi(\eta)\rho KT}{\eta\gamma} + \frac{\rho K\ln K}{\gamma\eta} + \frac{\gamma}{K}\sum_{t=1}^{T}\sum_{i=1}^{K}\hat{x}_i(t)\ .$$

Proof: Consider a game where $p(t)$ denotes the probability distribution and the estimated loss $\hat{x}(t)$ is the real loss. Then, the FeedExp3-algorithm above reduces to a MW-algorithm, where $x(t)$ is replaced by $\hat{x}(t)$. Note that the range of the estimation vector now is $[KS^-/\gamma, KS^+/\gamma]$. So, we normalize the loss by defining $\hat{x}'_i(t) := \frac{\gamma}{K\rho}\hat{x}_i(t) - S^-/\rho$ to $[0,1]$. Theorem 24 now implies

$$\begin{aligned}\sum_{t=1}^{T}\sum_{i=1}^{K} p_i(t)\hat{x}'_i(t) &\le \sum_{t=1}^{T}\hat{x}'_j(t) + \frac{2\Phi(\eta)}{\eta}\sum_{t=1}^{T}\sum_{i=1}^{K} p_i(t)\hat{x}'_i(t)^2 + \frac{\ln K}{\eta}\\ &\le \sum_{t=1}^{T}\hat{x'}_j(t) + \frac{2T\Phi(\eta)}{\eta} + \frac{\ln K}{\eta}\ .\end{aligned}$$

Rescaling leads to this inequality:

$$\sum_{t=1}^{T}\sum_{i=1}^{K} p_i(t)\hat{x}_i(t) \le \sum_{t=1}^{T}\hat{x}_j(t) + \frac{2\Phi(\eta)\rho KT}{\gamma\eta} + \frac{\rho K\ln K}{\gamma\eta}\ .$$

In algorithm FeedExp3 we have defined $\hat{p}_i(t) := (1-\gamma)p_i(t) + \frac{\gamma}{K}$. Hence, we can apply

$$\sum_{t=1}^{T}\sum_{i=1}^{K}\hat{p}_i(t)\hat{x}_i(t) = (1-\gamma)\left(\sum_{t=1}^{T}\sum_{i=1}^{K} p_i(t)\hat{x}_i(t)\right) + \frac{\gamma}{K}\sum_{t=1}^{T}\sum_{i=1}^{K}\hat{x}_i(t)\ .$$

So, a choice $\gamma > 0$ implies the claim. □

Lemma 36 *Let $\lambda > 0$ and $\delta > 0$. Then with probability at least $1-\delta$, for every action i, we have*

$$\sum_{t=1}^{T}\hat{x}_i(t) \ge \left(1 - \frac{\Phi(\lambda)}{\lambda}\right)\sum_{t=1}^{T} x_i(t) - \frac{\rho K\ln(K/\delta)}{\gamma\lambda} - \frac{\Phi(\lambda)\sigma TK}{\gamma\lambda\rho}\ , \tag{7.1}$$

$$\sum_{t=1}^{T}\hat{x}_i(t) \le \left(1 + \frac{\Phi(\lambda)}{\lambda}\right)\sum_{t=1}^{T} x_i(t) + \frac{\rho K\ln(K/\delta)}{\gamma\lambda} + \frac{\Phi(\lambda)\sigma TK}{\gamma\lambda\rho}\ . \tag{7.2}$$

Proof: We use a martingale argument in close analogy to the proof of Lemma 5.1 in [ACBFS95].

(7.1): Let us define the random variable

$$Z_t = \exp\left(\lambda\sum_{t'=1}^{t}(x'_i(t') - \hat{x}'_i(t')) - \Phi(\lambda)\sum_{t'=1}^{t} x'_i(t')\right)$$

where $x'_i(t) := \frac{\gamma}{K\rho}x_i(t) - S^-/\rho$ and $x'_i(t) := \frac{\gamma}{K\rho}\hat{x}_i(t) - S^-/\rho$ are normalized replacements of $x_i(t)$ and $\hat{x}_i(t)$. The main claim of the proof is that $\mathbf{E}[Z_T] \le 1$. Given this claim, we have by Markov's inequality that

$$P[Z_T > K/\delta] \le \delta/K$$

which, by some algebra, is equivalent to (7.1). We prove that $\mathbf{E}[Z_t] \le 1$ for $t = 0,\ldots,T$ by induction on t using a method given by Neveu ([Nev75], Lemma VII-2-9). For $t = 0$, $Z_0 = 1$. To prove the inductive step for $t > 0$, we have

$$\mathbf{E}[Z_t \mid g_1,\ldots,g_{t-1}] = Z_{t-1}\exp\left(-\Phi(\lambda)x'_i(t)\right)\mathbf{E}[\exp(\lambda(x'_i(t) - \hat{x}'_i(t))) \mid g_1,\ldots,g_{t-1}]\ .$$

Due to normalization we have $x'_i(t) \in [0,1]$ and $\hat{x}'_i(t) \in [0,1]$. Therefore $x'_i(t) - \hat{x}'_i(t) \le 1$ and it follows by Lemma 32

$$\begin{aligned}&\mathbf{E}[\exp(\lambda(x'_i(t) - \hat{x}'_i(t))) \mid g_1,\ldots,g_{t-1}]\\ &\qquad\le \mathbf{E}[1 + \lambda(x'_i(t) - \hat{x}'_i(t)) + \Phi(\lambda)(x'_i(t) - \hat{x}'_i(t))^2 \mid g_1,\ldots,g_{t-1}]\\ &\qquad\le 1 + \Phi(\lambda)x'_i(t)\\ &\qquad\le \exp(\Phi(\lambda)x'_i(t))\ .\end{aligned}$$

The second inequality follows from Lemma 34 and the following chain of inequalities:

$$\begin{aligned}\mathbf{E}[(x'_i(t)-\hat{x}'_i(t))^2 \mid g_1,\ldots,g_{t-1}] &= \mathbf{E}[\hat{x}'_i(t)^2 \mid g_1,\ldots,g_{t-1}] - x'_i(t)^2 \\ &\le \mathbf{E}[\hat{x}'_i(t)^2 \mid g_1,\ldots,g_{t-1}] \\ &\le \mathbf{E}[\hat{x}'_i(t) \mid g_1,\ldots,g_{t-1}] \\ &= x'_i(t)\ .\end{aligned}$$

(7.2): We define the random variable

$$Z_t = \exp\left(\lambda \sum_{t'=1}^{t} (\hat{x}'_i(t') - x'_i(t')) - \ \Phi(\lambda) \sum_{t'=1}^{t} x'_i(t')\right)\ .$$

Again the main claim of the proof is that $\mathbf{E}[Z_T] \le 1$. We have by Markov's inequality with $P[Z_T > K/\delta] \le \delta/K$, which, by some algebra, is equivalent to (7.2). The rest of this proof is analogous to the first part.

□

We single out a special case of Lemma 36 for further reference.

Corollary 4 *For $\gamma, \delta > 0$ and for the random variable $g^* = \arg\max_j \sum_{t=1}^{T} x_j(t)$ we have with probability $1-\delta$:*

$$\sum_{t=1}^{T} \hat{x}_{g^*}(t) \quad \le \quad \left(1 + \frac{\Phi(\lambda)}{\lambda}\right) \sum_{t=1}^{T} x_{g^*}(t) + \frac{\rho K \ln(K/\delta)}{\gamma\lambda} + \frac{\Phi(\lambda)\sigma T K}{\gamma\lambda\rho}\ .$$

Theorem 25 *If there exists G such that $F\ G = L$ then the expected regret $\mathbf{E}[R_{\textit{FeedExp3}}(T)]$ of algorithm FeedExp3 after T steps is bounded by*

$$\mathbf{E}[R_{\textit{FeedExp3}}(T)] \ = \ O(T^{3/4}(\ln T)^{1/2} K^{1/2})$$

with a constant factor linear in $\rho + \frac{\sigma}{\rho}$.

Proof: We first rewrite the expected loss $\mathbf{E}[W(T)]$ of algorithm FeedExp3 in a different way:

$$\begin{aligned}\mathbf{E}[W(T)] &= \sum_{t=1}^{T} \mathbf{E}[x_{g_t}] \\ &= \sum_{t=1}^{T} \mathbf{E}[\mathbf{E}[x_{g_t} | g_1 ... g_{t-1}]] \\ &= \sum_{t=1}^{T} \mathbf{E}\left[\sum_{i=1}^{K} x_i(t)\hat{p}_i(t)\right] \\ &= \mathbf{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{K} x_i(t)\hat{p}_i(t)\right] \qquad (7.3)\end{aligned}$$

We then apply Lemma 36(7.2) and choose $\gamma = \frac{K^{1/2}(\ln T)^{1/2}}{T^{1/4}}$, $\lambda = \frac{1}{T^{1/2}}$ and $\delta = \frac{1}{T^2 K}$. Lemma 31 states for $z \in [-1,1]$ that $\frac{z^2}{e} \le \Phi(z) \le (e-2)z^2$ and Lemma 32 for $z \in \ (0,1]\ :\ (1-z)^{-1} \le 1 + 2z$. This implies for $\gamma \in]0,1[$:

$$\begin{aligned}
\sum_{t=1}^{T}\sum_{i=1}^{K}\hat{p}_i(t)x_i(t) &\leq \frac{\sum_{t=1}^{T}\sum_{i=1}^{K}\hat{p}_i(t)\hat{x}_i(t)+\frac{\rho K\ln(K/\delta)}{\gamma\lambda}+\frac{\Phi(\lambda)\sigma TK}{\gamma\lambda\rho}}{1-\frac{\Phi(\lambda)}{\lambda}} \\
&\leq \left(1+2\frac{\Phi(\lambda)}{\lambda}\right) \\
&\quad \left(\sum_{t=1}^{T}\sum_{i=1}^{K}\hat{p}_i(t)\hat{x}_i(t)+\frac{\rho K\ln(K/\delta)}{\gamma\lambda}+\frac{\Phi(\lambda)\sigma TK}{\gamma\lambda\rho}\right) \\
&\leq \left(1+\frac{2}{T^{1/2}}\right)\sum_{t=1}^{T}\sum_{i=1}^{K}\hat{p}_i(t)\hat{x}_i(t)+ \\
&\quad \left(\rho+\tfrac{\sigma}{\rho}\right) T^{3/4}(\ln T)^{1/2}K^{1/2}+o(T^{1/2}K^{1/2})\ . \qquad (7.4)
\end{aligned}$$

Choose $\eta = T^{-1/2}$. Then Lemma 35 implies, for all j:

$$\begin{aligned}
\sum_{t=1}^{T}\sum_{i=1}^{K}\hat{p}_i(t)\hat{x}_i(t) &\leq \sum_{t=1}^{T}\hat{x}_j(t)+\frac{2\Phi(\eta)\rho KT}{\eta\gamma}+\frac{\rho K\ln K}{\gamma\eta}+\frac{\gamma}{K}\sum_{t=1}^{T}\sum_{i=1}^{K}\hat{x}_i(t) \\
&\leq \sum_{t=1}^{T}\hat{x}_j(t)+\frac{3\rho T^{3/4}K^{1/2}}{(\ln T)^{1/2}}+\frac{\gamma}{K}\sum_{t=1}^{T}\sum_{i=1}^{K}\hat{x}_i(t)\ . \qquad (7.5)
\end{aligned}$$

Note that from Lemma 36(7.1) it follows for the rightmost additional term with probability $1-\delta := 1-\frac{1}{T^2K}$.

$$\begin{aligned}
\frac{\gamma}{K}\sum_{t=1}^{T}\sum_{i=1}^{K}\hat{x}_i(t) &\leq \frac{\gamma}{K}\left(1+\frac{\Phi(\lambda)}{\lambda}\right)\sum_{t=1}^{T}\sum_{i=1}^{K}x_i(t)+\frac{\rho K\ln(K/\delta)}{\lambda}+\frac{\Phi(\lambda)\sigma TK}{\lambda\rho} \\
&\leq O(T^{3/4}(\ln T)^{1/2}K^{1/2})+O((\rho+\tfrac{\sigma}{\rho})T^{1/2}(\ln T)K)\ . \qquad (7.6)
\end{aligned}$$

At last we will use Corollary 4. Then, we have with probability $1-\delta$

$$\begin{aligned}
\sum_{t=1}^{T}\hat{x}_{g^*}(t) &\leq \left(1+\frac{\Phi(\lambda)}{\lambda}\right)\sum_{t=1}^{T}x_{g^*}(T)+\frac{\rho K\ln(K/\delta')}{\gamma\lambda}+\frac{S^{-}TK\Phi(\lambda)}{\rho\gamma\lambda} \\
&\leq \left(1+\frac{1}{T^{-1/2}}\right)\sum_{t=1}^{T}x_{g^*}(T)+O((\rho+\tfrac{-S^{-}}{\rho})T^{3/4}(\ln T)^{1/2}K^{1/2})\ .
\end{aligned}$$

For the expectation we add error term $3\delta T$ for the combined error probabilities 3δ and combine (7.3), (7.4), (7.5), (7.6), and (7.7).

$$\begin{aligned}
\mathbf{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{K}\hat{p}_i(t)x_i(t)\right] &\leq \mathbf{E}\left[\sum_{t=1}^{T}x_{g^*}(t)\right]\left(1+\frac{4}{T^{1/2}}\right)+3\delta T \\
&\quad +O((\rho+\tfrac{-S^{-}}{\rho})T^{3/4}(\ln T)^{1/2}K^{1/2}) \\
&\leq \mathbf{E}\left[\sum_{t=1}^{T}x_{g^*}(t)\right]+\ O((\rho+\tfrac{-S^{-}}{\rho})T^{3/4}(\ln T)^{1/2}K^{1/2})
\end{aligned}$$

□

7.4 Applications, Limits, and Extensions

We are now equipped to show how the bandwidth allocation problem which initially prompted this research, as well as other important examples, can be solved using this algorithm, but we will also see that only some tweaking allows to solve even more prediction problems. We will see in the next section that these "tricks" lead to a general algorithm, that, after some preprocessing, uses the basic algorithm to achieve sub-linear regret whenever this is feasible.

7.4.1 Bandwidth Allocation

In the bandwidth allocation problem the feedback function is defined as follows (**threshold feedback**):

$$f(y,g) = \begin{cases} 0 & \text{if } y < g \\ 1 & \text{otherwise .} \end{cases}$$

The feedback matrix F is therefore a lower triangular matrix with only 1's on the diagonal and below.

$$F = \begin{pmatrix} 1 & & & \mathbf{0} \\ & 1 & & \\ & & \ddots & \\ \mathbf{1} & & & 1 \end{pmatrix}$$

This matrix is invertible and therefore the condition $FG = L$ can be satisfied by defining $G = F^{-1}L$ where

$$F^{-1} = \begin{pmatrix} 1 & & & & \mathbf{0} \\ -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ \mathbf{0} & & & -1 & 1 \end{pmatrix}$$

We now consider the **severe cost function**

$$S(y,g) = \begin{cases} y-g & \text{if } g \geq y \\ y & \text{otherwise .} \end{cases}$$

We rescale the matrix to $S'_{y,g} := \frac{1}{K-1}S(y,g)$ such that $S'_{y,g} \in [0,1]$ and apply Theorem 25. Then the matrix G is given by

$$G \quad = \quad F^{-1}S' \quad = \quad \frac{1}{K-1}\begin{pmatrix} 0 & & & & \mathbf{1} \\ & -1 & & & \\ & & -2 & & \\ & & & \ddots & \\ \mathbf{1} & & & & 1-K \end{pmatrix} .$$

This gives for the severe cost function $\rho = S^+ - S^- = \max_{y,g,i}\{F_{y,g}G_{g,i}\} - \min_{y,g,i}\{F_{y,g}G_{g,i}\} \in [1,2]$ and $\sigma = -\min_{y,g,i}\{F_{y,g}G_{g,i}\} = 1$. Now Theorem 25 implies an expected regret for the cost function S' of $O(T^{3/4}(\ln T)^{1/2}K)$. Rescaling to the severe cost function S leads to the following corollary:

Corollary 5 *For the treshold feedback function and the severe cost function the FeedExp3 algorithm suffers the expected regret*

$$\mathbf{E}[R_{\mathit{FeedExp3}}(T)] \;=\; O(T^{3/4}(\ln T)^{1/2}K^2)$$

with respect to the best constant choice of bandwidth.

In other words the FeedExp3 algorithm bounds the average regret of $O\left(K^2\frac{(\ln T)^{1/2}}{T^{1/4}}\right)$ per round.

For the **gentle cost function** for $\alpha \geq 1$ defined as

$$C_\alpha = \begin{cases} y-g & \text{if } g \geq y \\ \alpha(g-y) & \text{otherwise} \end{cases}$$

in its rescaled setting $C'_\alpha := \frac{1}{\alpha(K-1)} C_\alpha$ we get

$$G = F^{-1}C_\alpha = \frac{1}{\alpha(K-1)} \begin{pmatrix} 0 & \alpha & 2\alpha & 3\alpha & \cdots & (K-1)\alpha \\ 1 & -\alpha & -\alpha & -\alpha & \cdots & -\alpha \\ 1 & 1 & -\alpha & -\alpha & \cdots & -\alpha \\ 1 & 1 & 1 & -\alpha & \cdots & -\alpha \\ \vdots & \vdots & \vdots & \ddots & \ddots & -\alpha \\ 1 & 1 & 1 & \cdots & 1 & -\alpha \end{pmatrix} .$$

This implies $1 \le \rho \le 2$ and $\sigma \le 1$ and the following corollary.

Corollary 6 *For the threshold feedback function and the gentle cost function C_α the FeedExp3 algorithm suffers the expected regret*

$$\mathbf{E}[R_{\mathit{FeedExp3}}(T)] = O(\alpha T^{3/4}(\ln T)^{1/2}K^2)$$

with respect to the best constant choice of bandwidth.

We now want to investigate a **continuous setting**: Instead of discrete choices $\{1,\dots,K\}$ the allocated bandwidth and available bandwidth are real numbers in the range $[0,1]$. This models applies if K is very large, i.e. $K \ge T^c$. Since the regret depends on the number of constant experts, i.e. possible discrete choices of K, the bound on the regret of FeedExp3 becomes too large. For a solution we consider the $K+1$ discrete choices for allocating the bandwidth $g_t \in \{0, \frac{1}{K}, \frac{2}{K}, \dots, 1\}$. Of course the available bandwidth y_t is still continuous. Because of the discrete threshold feedback, our algorithm underestimates y_t: if $f(y,g) = 0$ then the algorithm interpretes $g > y$ as by $y' := \frac{\lfloor yK \rfloor}{K}$.

Now note that the real costs and the costs refering to y' differ not too much for the severe cost function $S(y,g)$.

Lemma 37 *For the continuous severe cost function $S(y,g)$ with $g, y \in [0,1]$ and $g' \in \{0, \frac{1}{K}, \frac{2}{K}, \dots, 1\}$ we have*

$$S(y,g') - S\left(\frac{\lfloor yK \rfloor}{K}, g'\right) \in \left[0, \frac{1}{K}\right) ,$$
$$S\left(y, \frac{\lfloor gK \rfloor}{K}\right) \le S(y,g) + \frac{1}{K} .$$

Proof: follows straight-forward from the definition of S. □

This Lemma has two implications.

1. If we consider a game using $y'_t := \frac{\lfloor yK \rfloor}{K}$ instead of y_t in each round, then we get additional loss of at most $\frac{1}{K}$ per round.

2. The best constant choice g^* reduces the total loss by at most $\frac{T}{K}$ over all rounds compared to the choice $\frac{\lfloor g^* K \rfloor}{K}$.

These considerations immediately imply the following Theorem.

Theorem 26 *In the continuous case of $g_t, y_t \in [0,1]$, for the bandwidth allocation problem with threshold feedback and severe cost function there exists an algorithm A with expected regret of at most*

$$\mathbf{E}[R_A(T)] = O(T^{7/8}(\ln T)^{1/4})$$

with respect to the best constant choice of bandwidth $g^* \in [0,1]$.

Proof: Use the FeedExp3 algorithm in the discretized world $g'_t, y'_t \in \{0, \frac{1}{K}, \frac{2}{K}, \dots, 1\}$ where $y'_t := \frac{\lfloor yK \rfloor}{K}$ for $K := T^{1/8}(\ln T)^{-1/4}$. Theorem 25 shows that in this discretized world this algorithm suffers an expected regret of

$$\mathbf{E}[R_{\text{FeedExp3}}(T)] = O(T^{7/8}(\ln T)^{1/4}) .$$

Lemma 37 shows that the best discrete constant choice performs over all T rounds at most $\frac{T}{K}$ worse than the optimal continuous choice. Furthermore, Lemma 37 shows that a continuous choice of the available bandwidth also increases the cost by at most $\frac{T}{K}$.

Hence, the regret of the algorithm FeedExp3 increases in the continuous setting by at most $\frac{T}{K} = T^{7/8}(\ln T)^{1/4}$ □

For the gentle cost function $C_\alpha(y,g)$ for $\alpha \geq 1$ in the continuous setting the same observations apply.

Lemma 38 *For the continuous gentle cost function $C_\alpha(y,g)$ with $g, y \in [0,1]$ and $\alpha \geq 1$ we have*

$$\left| S(y,g) - S\left(\frac{\lfloor yK \rfloor}{K}, \frac{\lfloor gK \rfloor}{K}\right) \right| \quad < \quad \frac{\alpha}{K} .$$

Proof: follows straight-forward from the definition of C_α. □

Theorem 27 *In the continuous case of $g_t, y_t \in [0,1]$, for the bandwidth allocation problem with threshold feedback and gentle cost function C_α for $\alpha \geq 1$ there exists an algorithm A with expected regret of at most*

$$\mathbf{E}[R_A(T)] \;=\; O(\alpha T^{7/8}(\ln T)^{1/4})$$

with respect to the best constant choice of bandwidth $g^* \in [0,1]$.

Proof: For the choice $K := T^{-1/4}(\ln T)^{-1/4}$ the proof is analogous to the proof of Theorem 26 and follows from Lemma 38 and Theorem 25. □

7.4.2 Loss Feedback and Full Information

The multi-armed bandit problem with partial information of Freund *et al.* [ACBFS95] corresponds to the case $F = L$. Under this condition, $G = I$ is a suitable choice. A somehow dual situation arises when $F = I$, that is when the feedback is a binary "hit or miss" information. Then $G = L$ is a suitable choice for G.

A more troublesome situation is the full feedback case. Even if in this case the machinery presented in this paper is not necessary, since an expected regret of $O(T^{1/2} \log K)$ can be achieved by the MW algorithm [FS99], it is clear that a general algorithm for this class of problems must be able to solve this special case, too. A natural choice for F is $F_{ij} = i$, which implies $f_t = y_t$. Unfortunately, such a matrix has rank 1 and therefore the condition $FG = L$ can be satisfied only when L has a very special, and rather trivial, form. But more than the specific values of the entries of F, what defines "full feedback" is the fact that no two entries in every column of F have the same value, that is there is a bijection between the values in F_i and the range of y_t. If F satisfies this property, it is possible to compute y_t from f_t and hence we can say we are still in the full information case. Therefore, we are interested in finding a full rank matrix within the set of matrices just described, which all represent the full feedback case.

One possible solution is to replace every diagonal entry with a number large enough to satisfy Hadamard's theorem, that is:

$$|F_{ii}| > \sum_{j=1, j\neq i}^{K} |F_{ij}| .$$

implies that $\det(F) \neq 0$. But this solution is specific to the full feedback case, whereas the problem of singular or low rank F arises in many contexts.

For instance, consider the threshold feedback and modify slightly the definition to be $f(y,g) = 0$, if $y \leq g$ and 1 otherwise. Then F becomes singular, but it is enough to reverse the arbitrary roles of 0 and 1 to get an equivalent problem, where this time F is invertible.

7.4.3 Extensions

An acceptable transformation of F can be detailed as a set of functions for each column of F, from the range of the elements of F into some other range. The goal is to obtain a new matrix F', where each column is obtained applying one of the functions to the elements of a column of F, for which there is a G such that $F'G = L'$. It is clear that F' can have more columns than F, because each column can be transformed in different ways, but no fewer, since every action has to be represented. This corresponds to introducing new actions that are essentially replicas, but for each of which the feedback undergoes a different transformation. From the point of view of the loss, these additional actions are totally equivalent and therefore we need to extend L into a larger matrix L' by duplicating the appropriate columns. What we seek is a general way to expand F' so as to keep the number of columns reasonably small but making the linear span of F' all-inclusive, that is such that it cannot be enlarged by adding more columns obtained in a feasible way. This can be accomplished as follows. For every column F_i containing r_i distinct values (w.l.o.g. from the set $[r_i]$) we define r_i columns $F'_{R_i+1} \dots F'_{R_i+r_i}$, where $R_i = \sum_{j=1}^{i-1} r_i$, as follows: $F'_{R_i+j,k} = [j = F_{i,k}]$, for $1 \leq j \leq r_i$, where $[P] = 1$ if P is true and 0 otherwise. As to L', we set $L'_j = L_i$ if and only if $R_i < j \leq R_i + r_i$. It is straightforward to check that the matrix F' obtained this way has the largest possible linear span among all the ones that can be obtained from F via the transformations detailed above. Also, since F is $R \times K$, F' is at most $R \times KR$. These are more columns than we need and would impact negatively the bounds on the regret: Therefore we will pick the smallest subset of columns S which is still good for our purposes, that is, it satisfies the following conditions:

- All the columns of L are represented in L' or, equivalently, all the actions in the original instance are represented, that is for every $i \in [K]$ there is a $j \in S$ such that $R_i < j \leq R_i + r_i$;
- $\mathcal{L}(\{F'_i : i \in S\}) = \text{range}(F')$.

The final feedback and distance matrices can be obtained by dropping all the columns not in S from F' and L', and we will continue to use the same symbols for the submatrices defined this way. In the next section we will present a greedy algorithm which solves this problem.

Let us see how this helps in the full feedback case. Recall that a natural choice for F is $F_{ij} = i$. Therefore, the corresponding F' has maximum rank (some columns of F' form an $R \times R$ identity matrix), $F'G = L$ can be solved for G and the general algorithm can be applied successfully.

A further complication arises from *non-exploitable* actions. These are actions which for any adversarial strategy do not turn out to be optimal. The problem here is that the condition $FG = L$ might be impossible to be satisfied because of some columns related to non-exploitable actions. Consider, for instance,

$$F = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix} \qquad L = \begin{pmatrix} 1 & 1 & 0 \\ 2 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} .$$

Here column 1 of L is not in the linear span of F, but it is easy to see that actions 2 and 3 can be always preferred to the first. Therefore, it might seem reasonable to simply drop the first column as it is related to a non-exploitable action. It turns out, though, it is just action 1 which provides the necessary feedback to estimate the loss. It is clear that simply omitting non-exploitable actions is not a good strategy.

As with the feedback matrix F, the solution for these problems is to transform the loss matrix L into a new L' in a way that does not lower the regret.

If we add the same vector x to every column of L, we are not changing the problem instance in any substantial way, since the regret, our performance measure, is invariant w.r.t. this transformation. Therefore, we are interested in those transformations that help fulfilling the condition $FG = L$. This time, it makes sense to try to obtain a matrix L' from L of minimum rank. Rank minimization is a difficult problem in general, but this special case turns out to be rather trivial.

Lemma 39 *Given three matrices L, L' and L'' such that for every i $L'_i = L_i - L_j$ and $L''_i = L_i - x$, we have that, for any vector x and index j, $\mathcal{L}(L') \subseteq \mathcal{L}(L'')$.*

Proof: Since $L_i - L_j = L_i - x - (L_j - x)$, the lemma follows. □

Therefore, choosing x equal to one of the columns of L minimizes the linear span of L'. In the following we will assume $L_1 = (0, \dots, 0)$ w.l.o.g.

As to non-exploitable actions, we first need to formally define them. Let us define a partition[1] of the set of mixed strategies (for the adversary) as follows. Every element of the partition is centered around a column of L' and is defined as:

$$N(L_i) = \{v \in \mathcal{A} \mid \forall j : \ L_i \neq L_j \Rightarrow vL_i \leq vL_j\}$$

where the set $\mathcal{A} := \{v \in [0,1]^R \mid \ \sum_i v_i = 1\}$ denotes all possible mixed strategies of the adversary.

That is, an element of this partition is the set of mixed adversarial strategies such that a certain prediction is preferred to any other. If $N(L_i)$ is empty, then i is a non-exploitable action. The rationale behind this definition is that no sensible algorithm will ever try this action for exploitation purposes (that is often), since there are other actions which bear a smaller loss. The interior of $N(L_i)$ is defined as follows:

$$S(L_i) = \{v \in \mathcal{A} \mid \forall j : \ L_i \neq L_j \Rightarrow vL_i < vL_j\}$$

The following lemma shows that we can replace every mixed adversarial strategy on the surface of some element of the partition by another strategy not on the surface, with no penalty in performance.

Lemma 40 *For all mixed adversarial strategies $q \in \mathcal{A}$ there exists a column L_i with $S(L_i) \neq \emptyset$ such that $q \in N(L_i)$.*

Proof: We concentrate on elements in the set $\mathcal{F} := \bigcup_i N(L_i) \setminus S(L_i)$. Note that we have

$$\mathcal{F} \ \subseteq \ \bigcup_{i,j} \{v \in \mathcal{A} \mid v(L_i - L_j) = 0\} \ .$$

Therefore, $\mathcal{F}$ is a subset of a union of at most K^2 subspaces of dimension $R-2$. Since $\mathcal{A}$ is a $R-1$ dimensional polytope, any ϵ-ball centered on a point $v \in N(L_i)$ contains elements not in $\mathcal{F}$. Such an element $v' \notin \mathcal{F}$ is contained in a set $L_{i'}$ with $S(L_{i'}) \neq \emptyset$. Since this is true for any $\epsilon - ball$, then v belongs to the surface of $N(L_{i'})$ too, that is $vL_i = vL_{i'}$. □

Hence, we can extend the definition of non-exploitable actions to columns with $S(L_i) = \emptyset$, since their choice gives no improvement over actions with $S(L_i) \neq \emptyset$.

In order to extend the applicability of the basic algorithm, we set in L all the entries in the columns corresponding to non-exploitable actions equal to the size of the maximum element in its column. This can only increase the regret w.r.t. the best constant strategy, because none of the actions associated to these columns can be part of any optimal strategy. Furthermore, it is easy to check that the columns obtained this way are in the linear span of F' for every F.

7.5 The General Algorithm

In Figure 7.7 we show how to implement the construction of F' and L'. Let $[F_{i,j} = v]_{i=1,\dots,R}$ denote the vector obtained replacing, in the jth column of F, every entry equal to v by 1 and all others by 0. The algorithm constructs F' and L' by appending columns derived from F and L to their right sides.

Augmented with this kind of preprocessing for the loss and feedback matrices, our algorithm covers all the examples we considered. A natural question is therefore whether the condition $F'G = L'$ is not only necessary for our algorithm to apply, but in general for any useful algorithm. The answer is positive, meaning that if the condition cannot be fulfilled, then any algorithm will undergo a loss $\Omega(T)$.

Theorem 28 *For any prediction game (F, L) we have either one of the following situations:*

- *The General Algorithm solves it with an expected regret of*

$$\mathbf{E}[R_{\textit{General}}] \ \leq \ O(T^{3/4} (\ln T)^{1/2} \max(K, R^{1/2})) \ .$$

[1] Strictly speaking, it is not a partition, but the idea helps the intuition.

```
The General Algorithm
  input $R \times K$-matrices $F$, $L$
  begin
    $z := 0$
    for $j$ from $1$ to $K$ do
      $a := 0$
      for all values $v$ in $F_i$ do
        if $[F_{i,j} = v]_{i=1,\ldots,R} \notin \mathcal{L}(F'_1, \ldots, F'_z)$ then
          $a := 1$
          $z := z + 1$
          $F'_z := [F_{i,j} = v]_{i=1,\ldots,R}$
          $L'_z := L_j$
          $h(z) := j$
        fi
      od
      if $a = 0$ then
        $z := z + 1$
        $F'_z := (0, \ldots, 0)$
        $L'_z := L_j$
        $h(z) := j$
      fi
    od
    b := 0
    for $i$ from $1$ to $z$ do
      if $b = 0$ and $S(L'_i) \neq \emptyset$ then
        $b := i$
      fi
    od
    for $i$ from $1$ to $z$ do
      $L'_i := L'_i - L'_b$
    od
    for $i$ from $1$ to $z$ do
      if $S(L'_i) = \emptyset$ then
        $L'_i := (\max_{j'}\{L'_{j'i}\}, \ldots, \max_{j'}\{L'_{j'i}\})$
      fi
    od
    Perform $\mathbf{FeedExp3}(F', L')$ and replace each guess $g_t$ by $h(g_t)$
  end
```

Figure 7.7: The General Algorithm

- *There is an adversarial strategy which causes any algorithm A to produce a regret of $\Omega(T)$ with probability* $1/2$.

Proof: In the previous section, we have already seen that we can map a sequence of actions for the prediction game (F', L') to the instance F, L in a way that does not essentially increase the regret. This proves the first part of the theorem. We can rephrase the second part as follows:

> Given an instance of the prediction game (F, L) let F' and L' be the matrices obtained through the transformations detailed in the previous section. If there is no G such that $F'G = L'$, then any prediction algorithm will undergo a loss $\Omega(T)$.

We associate a graph $H = (V, E)$ to the partition $\{N(L'_1), \ldots, N(L'_k)\}$ by defining $V = \{L_i : S(L'_i) \neq \emptyset\}$ and $(L'_i, L'_j) \in E$ if and only if $L'_i = L'_j$ or the sets $N(L'_i)$ and $N(L'_j)$ share a facet, i.e. a face of dimension $R-2$. Note that for all i the set $N(L'_i)$ describes a polytope of dimension $R-1$, or its interior $S(L'_i)$ is empty.

Let $\mathcal{L}(E)$ be the linear span of the set of differences between vectors at the endpoints of each edge in E. We have the following

Lemma 41 $\mathcal{L}(E) = \mathcal{L}(\{L'_i : L'_i \in V\})$.

Proof: For each $L'_i \in V$, let $L'_i = L'_i - L'_{i_1} + L'_{i_1} - L'_{i_2} + \ldots + L'_{i_p} - L'_1$, where $(L'_i, L'_{i_1}, \ldots, L'_{i_p}, L'_1)$ is a path connecting L'_i to L'_1, if such a path exists.

We need only to prove that H is connected. Given the two vertices L'_i and L'_j, we seek a path joining them. Consider the segment joining a point in the interior of $N(L'_i)$ to one in the interior of $N(L'_j)$. Since the set of mixed strategies is convex, every point in the segment is a mixed strategy. Let us pick an arbitrary orientation for this segment and consider the sequence of polytopes that share with the segment some interior point, and specifically two consecutive entries in the sequence, $N(L'_h)$ and $N(L'_k)$. If the segment goes from the first to the second through a facet, then the two corresponding vertices in the graph are joined by an edge. If not, that means that the two polytopes share only a face of dimension $R-3$ or lower, e.g. a vertex or an edge. In that case we need to pick a different point in, say, $N(L_j)$. This is always possible because $N(L_1)$ has dimension $R-1$ whereas the set of points collinear with the designated point in $N(L_i)$ and any point in any face of dimension $R-3$ or lower has dimension at most $R-2$. □

Now, let us assume that there is no G such that $F'G = L'$. This implies that there is L'_i such that $L'_i \notin \mathcal{L}(F')$. Let us assume $S(L'_i) = \emptyset$. By definition of L', $L'_i = \alpha(1, \ldots, 1)$ for some α. This implies, by definition of F', $L'_i \in \mathcal{L}(F')$, a contradiction. Therefore, $S(L'_i) \neq \emptyset$ and, by lemma 41, $\mathcal{L}(E) \not\subseteq \mathcal{L}(F')$. Hence, for some $(L'_i, L'_j) \in E$, we have that $L'_i - L'_j \notin \mathcal{L}(F')$. Since the range of F' is the orthogonal complement to the null space of F'^T we have that, for some non-zero vector $n \in \mathrm{Ker}(F'^T)$, $n(L'_i - L'_j) \neq 0$. Let y be a point in the interior of the facet shared by $N(L'_i)$ and $N(L'_j)$. We have that $y + \alpha n$ and $y - \alpha n$ are both mixed strategies for some α. They are indistinguishable from the point of view of any algorithm because $(y + \alpha n)F' = (y - \alpha n)F' = yF'$, but they correspond to different optimal actions, and the regret implied by making the wrong choice is $|\alpha n(L'_i - L'_j)|$. □

7.6 Conclusion and Open Problems

We solve the problem of discrete loss and feedback online prediction games in its general setting, presenting an algorithm which, on average, has sub-linear regret against the best constant choice, whenever this is achievable.

In the full knowledge case, it is well known that the average per step regret is bounded by $O(T^{-1/2})$. In [ACBFS95] it is shown that, if the feedback is identical to the loss, there is an algorithm the average regret of which is bounded by $O(T^{-1/3})$ (omitting polylogarithmic terms), recently improved to $O(T^{-1/2})$ [Aue00]. In this chapter, we show that, for every "reasonable" feedback, the average per step regret is at most $O(T^{-1/4})$. Otherwise, no algorithm can do better than $\Omega(T)$.

If the number of rounds T is unknown in advance the general algorithm can be modified to work in epochs, a technique shown in [ACBFS95]. This modification does not change the asymptotical bounds on the regret.

We applied the FeedExp3 algorithm to bandwidth allocation problems and showed a sub-linear regret for the severe and the gentle cost function under threshold feedback. We will discuss in the next chapter whether this results can be actually applied to the original setting.

While we proved that no algorithm can attain sub-linear regret on a larger class of problems than ours does, it is an open problem whether such general prediction games can be solved with a bound on the regret as good as the one obtained for the multi-armed bandit problem, in the most general setting or under some additional assumptions.

It is straightforward to transfer the upper bounds shown for the worst case regret against constant predictors to the finite pool of general predictors (a.k.a. “expert”) model, in analogy with the argument of [ACBFS95], Section 7. However, the lower bound is not readily applicable to this case and, therefore, it is an open question whether our general algorithm achieves sub-linear regret whenever it is possible in this context.

Another interesting question is whether a uniform algorithm exists that works for any feedback and loss function and achieves the best known performance for each feedback. Note that the algorithms presented in this work, even when given as an input a feedback function corresponding to the “full knowledge” case, guarantees only an average per step regret of $O(T^{-1/4})$, whereas $O(T^{-1/2})$ is the best bound known.

Chapter 8

Bandwidth Allocation under Adversarial Timing

8.1 Introduction

In this chapter, we investigate distributed and cooperative bandwidth allocation protocols. A well-known example for such a protocol is the Transport Control Protocol (TCP) in the Internet. This protocol was modified when the Internet experienced a severe service degradation or “Internet Meltdown” during the early growth phase of the mid 1980s [Nag84]. The dynamics of packet forwarding were underestimated which resulted in a “congestion collapse”. The fix for the Internet meltdown is the “back off” behavior of TCP [Jac88]. In simplified form, when TCP suffers a packet loss, it decreases its sending rate (by decreasing its window size by a factor of two), and when a packet is successfully delivered, it increases its sending rate (by increasing its window size by one). This *additively increasing and multiplicatively decreasing (AIMD)* behavior implements social interaction between the allocation patterns of concurring host-to-host connections.

Consider two connections P and Q sharing a link of capacity B, see Figure 8.1. Suppose the algorithm allocating each connection bandwidth uses AIMD behavior like in TCP and assume that if the sum of the chosen packet rate of P and Q is larger than B, then packets are dropped. Let P be the first established connection and after some time Q will join in. We observe that as long P is the only active process, its bandwidth of P oscillates: P increases its bandwidth until it is larger than B, then packets are dropped and thus P decreases its bandwidth by a constant factor, then P’s bandwidth increases and so on. Now Q joins and since P does not use the complete bandwidth B for most of the time, there is some bandwidth left for Q such that Q has a chance to get a constant fraction of the bandwidth. Clearly, the process *does not converge* and *does not reach full utilization*, i.e. in an average round only a constant fraction of the available bandwidth is used.

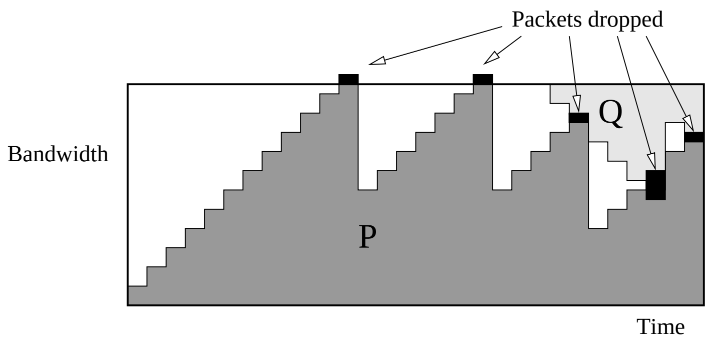

Figure 8.1: Allocation behavior of pure AIMD.

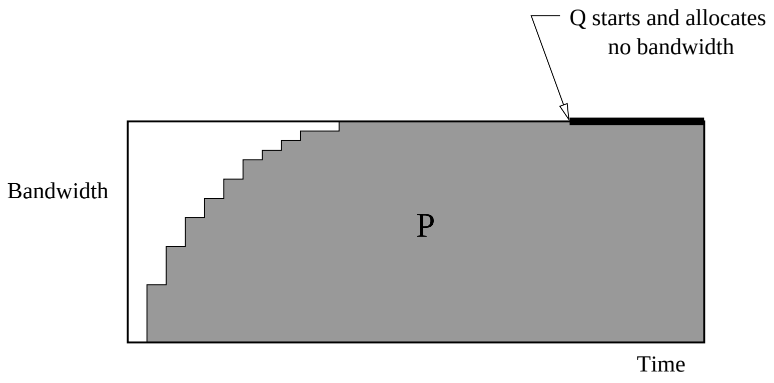

Figure 8.2: Unfair allocation behavior of an online prediction algorithm.

Now let us replace the AIMD behavior by an online prediction algorithm introduced in the previous chapter (E.g. we use FeedExp3 algorithm and neglect the influence of the uniformly distributed testing pattern or we use the Hedge/Multiplicative Weighting algorithm). Figure 8.2 shows that when P starts it quickly allocates the whole bandwidth B if it minimizes the gentle or the severe costs. Then Q starts with minimal bandwidth 0. Now observe that P does not suffer any loss in the gentle and severe cost model, since the available bandwidth for P is B. The available bandwidth for Q is 0, hence the cost for Q is 0, too. Since both allocation algorithms suffer no loss, the allocated bandwidths of both algorithms will not be changed anymore. We notice that the system *converges* and reaches *full utilization*. However, the situation is undesirable, since it is *unfair*.

But also for TCP fairness is not guaranteed, since its behavior depends heavily on the speed at which individual players increase their rate. It is known that TCP is inherently unfair to connections with long-round trip times [FJ92] and the unfairness can sometimes be as bad as the inverse square of round-trip times [LM97].

In this chapter we concentrate on fairness and full bandwidth utilization. For this, we consider an asynchronous distributed network in a very simplified setting. In contrast to TCP and the feedback model in the preceding chapter (but along some concurrent concepts), we allow the protocols to see the residual bandwidth, while other information like the allocated bandwidths (or even the number) of competing protocols is not used. Following the ideas of [BEY98, KKPS00] we challenge our protocols by an adversary to ensure robustness and reliability. In [KKPS00] this is modeled in form of the choice of the bandwidth by an adversarial strategy. In our new approach the link bandwidth is fixed and fluctuations in the available bandwidth for individual players are modeled using an adversary who determines when player enter and leave the system and, in particular, controls the timing of rate update operations of individual players. Let us describe this in more detail.

Consider a set of k players who share a single bus of bandwidth B. Each participating player i holds a rate variable r_i describing how much bandwidth the player currently occupies. From time to time new players arrive and claim a fair share of bandwidth while other players leave the system and release allocated bandwidth. Clearly, such a dynamic environment requires a resource management that adapts the bandwidth allocation continuously to the varying circumstances. For example, if several players share a single bus and a new player arrives then the established players have to release parts of their bandwidth so that the newly arrived player can receive a fair amount of bandwidth. Similarly, if some players leave the system then the remaining players can divide up the released bandwidth.

Let us transfer TCP into this model: A player increases its rate by one unit when he observes that its current rate value can actually be realized since $\sum_i r_i \leq B$. Eventually, the rates will be increased by such an amount that the sum of the individual rates exceeds the available bandwidth and the system collapses. This collapse is observed by the individual players and as a response all players halve their rate values. Then players continue with the linear increase and so on.

Recently, some TCP implementations which use more aggressive congestion strategies and increase their rates at higher speed have been suggested. In fact, already today the speed at which players increase their rates depends on many different aspects, especially on the so-called round-trip times, which again depend on the bandwidth utilization and, hence, on the rates chosen by the players.

In order to study the influence of different speeds in our toy model, consider two players X and Y who interact on a bus. Suppose player X increases his bandwidth s times as fast as player Y. Then, on the long run, the average rate of player X will be s times higher than the average rate of player Y. (This is because the ratio between the sum of rate increments and the sum of rate decrements converges against one with time so that the average loss of player X in case of a collapse must be s times higher than the one of player Y, which in turn implies that also the average rate of X must be s times the average rate of Y.)

We summarize that different speeds for updating the bandwidths can result in an unfair bandwidth allocation in practice as in our toy model (see also [CJ89, MSM97]). In the following, we will have a closer look at this kind of problems in an adversarial model of time. We start with upper and lower bounds for a very simple model in which players interact on a single bus. Afterwards we generalize our model to general networks.

8.1.1 Model 1: Fair bandwidth allocation on a single bus

Consider a single bus of bandwidth B. We assume an *open system* in which players can enter and leave the bus continuously. Let K denote the possibly infinite set of players. When players from K *enter* the bus they request a share of its bandwidth, and when they *leave* the bus they release the allocated bandwidth. *Active players* (i.e., players who entered but did not leave the bus) need to agree on the share of bandwidth they receive. This is done by so-called "rate update operations" that active players can perform in order to adjust their individual share of bandwidth. We formalize this as follows.

We model the open system by an adversary that specifies a sequence of events $\sigma = \sigma_1\sigma_2\sigma_3\ldots$, where each event σ_t is a tuple (i, x) with $i \in K$ and $x \in \{\text{enter}, \text{leave}, \text{update}\}$. With each player $i \in K$, we associate a positive rate variable r_i the value of which is zero if the player is inactive, that is, the initial value of r_i is zero and r_i is reset to zero whenever the adversary calls (i, leave). The adversary calls update operations only for active player. In particular, if the adversary calls (i, update) then player i can set r_i to any positive value. In other words, the adversary determines how often and when players can redefine their rate. At any given time, we define the share of bandwidth b_i player i receives by

$$b_i = \begin{cases} r_i & \text{if } \sum_{i\in K} r_i \leq B \ , \\ 0 & \text{otherwise} \ . \end{cases}$$

Thus, the share of bandwidth of all players is zero when the system is overloaded. (For analogous models see, e.g., [KKPS00].) A fair and efficient allocation protocol aims to set the rates in such a way that all players in the system get almost the same share of bandwidth and the unused bandwidth is as small as possible.

Clearly, when the adversary frequently changes the set of active players or does not allow to perform a reasonable number of update operations for all active players, then it is impossible to achieve a fair and efficient allocation of bandwidths among the active players. Therefore, we focus on periods of times in which the system is closed. A *closed system period* $(\mathcal{I}, \mathcal{K})$ is defined by a possibly infinite interval of time $\mathcal{I}$ and a finite set of players $\mathcal{K} \subseteq K$. During $\mathcal{I}$ there are no players entering and leaving the system and the adversary only allows the players in $\mathcal{K}$ to perform update operations. Our goal is to rapidly approach a fair and efficient allocation of bandwidth in closed system periods. For this purpose, we investigate the following simple protocol which is also known as the Phantom Protocol [AMO00]. Let $\alpha \geq 1$ denote a global parameter. Figure 8.3 shows the allocation of three players using this protocol.

The Virtual Player Protocol (VPP)
Suppose player j performs an update operation. Let $\bar{r} = \max\{B - \sum_{i\in K} r_i, 0\}$ denote the unused bandwidth immediately before the update operation. Then player j sets

$$r_j := \frac{\alpha}{\alpha+1}(r_j + \bar{r}) \ .$$

In order to describe the behavior of the virtual player protocol (VPP) in a closed system period $(\mathcal{I}, \mathcal{K})$, let us partition $\mathcal{I}$ into contiguous phases in such a way that each phase contains at least one update operation for each player.

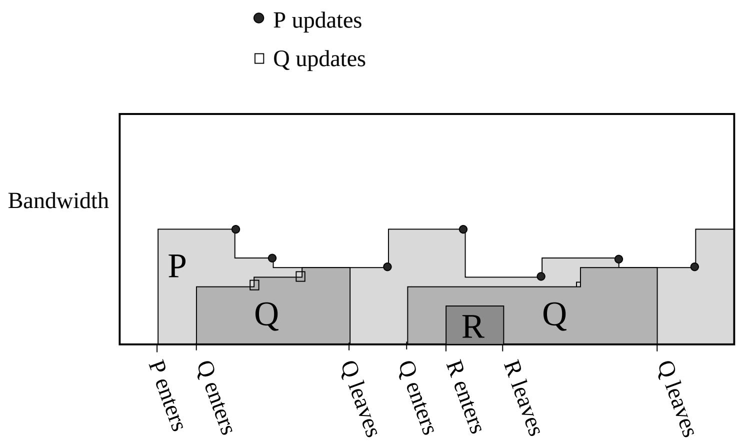

Figure 8.3: Allocation behavior of the Virtual Player Protocol.

Theorem 29 *Let c_0 denote a suitable absolute constant. Consider any closed system period $(\mathcal{I}, \mathcal{K})$. Define $k = |\mathcal{K}|$ and let $R = \frac{B}{1+\alpha k}$. Within the first $c_0(\alpha k^2)\log(\alpha k/\epsilon)$ phases of the interval $\mathcal{I}$ the VPP reduces the unused bandwidth to at most $(1+\epsilon)R$ and yields*

$$b_i \in \left[(1-\epsilon)\frac{B-R}{k}, (1+\epsilon)\frac{B-R}{k}\right]$$

for all all $i \in \mathcal{K}$, regardless of the initial rates.

In other words, the VPP utilizes the available bandwidth almost completely and distributes it in a fair way among the players in $\mathcal{K}$. In fact, one can interpret the unused bandwidth $\bar{r}$ (which is the only feedback used by the VPP) as the rate of an additional virtual player. Suppose $\alpha = 1$. Then an update operation of player j simply brings r_j into line with $\bar{r}$. This way, the bandwidth will finally be divided up in a fair way among all players in $\mathcal{K}$ and the virtual player. By increasing α, the share of the virtual player can be made arbitrarily small. A formal proof of the theorem is given in Section 8.2.

Let us measure the *length* of closed system periods in the number of phases they define. Then the theorem implies that the VPP converges against a completely fair bandwidth allocation in closed system periods of infinite length. In other words,

$$\lim_{\tau \to \infty} \frac{b_i(\tau)}{b_j(\tau)} = 1 \ ,$$

for all players i, j from $\mathcal{K}$. Observe, however, that the VPP does not utilize the full bandwidth. In fact, the wasted bandwidth is $R = \frac{B}{1+\alpha k}$ in the limit. This gives rise to the question whether it is possible to obtain fairness and full utilization simultaneously. The following theorem answers this question negatively and, hence, gives a strong motivation for leaving a small fraction of the bandwidth unused.

Theorem 30 *For any bandwidth allocation protocol $\mathcal{P}$ converging against full utilization in closed system periods of infinite length, there is an adversarial sequence σ that defines a closed system period $(\mathcal{I}, \mathcal{K})$ of infinite length with $|\mathcal{K}| \leq 2$ that enforces a bandwidth assignment of at most ϵB ($\epsilon > 0$) for one of the players in $\mathcal{K}$.*

This surprising impossibility result follows from a simple, elegant lower bound argument. The corresponding proof is given in Section 8.3.

Note that the incompatibility of fairness and full utilization also holds if all players know the complete current status, e.g. for explicit rate based algorithms like in [CRL96, Rob96].

8.1.2 Model 2: Bandwidth allocation in general networks.

We generalize the above adversarial model to general networks. The network is modeled by a (hyper)graph $G = (V, E)$. Edges represent buses, routers, or other shared resources of limited bandwidth. The bandwidth capacity of edge e is denoted by $B(e)$. Each player comes with a set of edges constituting a *simple path* (i.e., a path in which every edge appears at most once). For player $i \in K$, let $path(i)$ denote the player's path, and for an edge $e \in E$ let $K(e) \subseteq K$ denote the set of those players whose paths contain e.

As before, an adversary determines when players enter and leave the system and when they can update their rates. For the time being, we assume that update operations are performed *atomically*, i.e., an update operation is not performed by the adversary until the previous one has become effective on all edges of the respective path. We generalize the VPP as follows. Let $\alpha \geq 1$ denote a global parameter.

The Virtual Player Protocol for General Networks
Suppose player j performs an update operation. For every edge e, let $\bar{r}(e) = B(e) - \sum_{i \in \mathcal{K}(e)} r_i(e)$ denote the free bandwidth on edge e. Then player j sets

$$r_j := \frac{\alpha}{\alpha + 1}(r_j + \min_{e \in path(j)}(\bar{r}(e))) \ ,$$

where $\alpha \geq 1$ denotes a global parameter.

The most widely accepted criterion for a fair and efficient bandwidth allocation in networks is the concept of "max-min fairness" [Jaf81, KRT99]. The network is considered to be in a state of *max-min fairness* if it is impossible to infinitesimally increase the rate of any player without exceeding the edge capacities or decreasing the rate of players whose rate is equal or smaller. Our impossibility result for a single edge implies that one cannot converge against max-min fairness in closed system periods. Therefore, we relax the concept of max-min fairness as follows.

For every $\delta > 0$, the network is in a state of *δ-max-min fairness* if it is impossible to increase the rate r of any player by more than a factor of $(1 + \delta)$ without exceeding the edge capacities in $path(i)$ or decreasing the rate of players whose rate is at most $(1 + \delta)r$. We define that a protocol *converges against δ-max-min fairness* if, given any closed system period $(\mathcal{I}, \mathcal{K})$ of infinite length, the rates converge against a state in which the above criterion is fulfilled among the players in $\mathcal{K}$.

Theorem 31 *The VPP converges against $\frac{1}{\alpha}$-max-min fairness.*

The proof of the theorem can be found in Section 8.4. If $\alpha = 1$ then we can describe the state against which the protocol converges as follows. For every edge e, we define a virtual player whose path contains only the edge e. The rate of this player is defined to be the unused bandwidth on edge e. Then the system converges against a state of max-min fairness among all participating players including the virtual players. Increasing α simply decreases the share of the virtual player and, hence, the wasted bandwidth.

Unfortunately, the analysis showing the convergence does not also prove a fast convergence. For this purpose, we investigate a *discrete variant* of the VPP adopting some ideas of [AS98b], that is, the rate values of active players are of the form $(1 + \epsilon)^q$, for fixed $\epsilon > 0$ and $q \in \mathcal{Z}$. Fix any closed system period $(\mathcal{I}, \mathcal{K})$. Let the *congestion* $C = C(\mathcal{I}, \mathcal{K})$ denote the maximum number of paths (of participating players) which contain the same edge, and let the *dilation* $D = D(\mathcal{I}, \mathcal{K})$ denote the maximum length of a path. Furthermore, let $B = B(\mathcal{I}, \mathcal{K})$ denote the ratio between the bandwidth that is available for the participating players on the widest and the narrowest edge.

Theorem 32 *For every $\delta > 0$, there is a discrete variant of the VPP that approaches a δ-max-min fair state in any closed system phase. This state is reached after $O(DC^2 + DC(\log B)/\delta^2)$ phases.*

The proof of this theorem is given in Section 8.5. Observe that the performance of the protocol depends only on local parameters such as the congestion or the dilation but not on global parameters like the total number of players or the size of the network. Furthermore, the protocol does not need to be parametrized with any other parameter than δ, and the only feedback a player needs in order to perform an update operation is the unused bandwidth on the narrowest edge on its path.

8.2 The Virtual Player Protocol

We will now prove that the Virtual Player Protocol (VPP) converges against fairness.
Proof of Theorem 29: W.l.o.g., assume $\mathcal{K} = \{1, \ldots, k\}$. We add a virtual player 0 whose rate is defined by $r_0 := \alpha\bar{r}$. This way, the set of all participating players is $[k+1] := \{0, \ldots, k\}$. Furthermore, we assume that the closed system period starts with σ_1. We want to show that the maximum distance between any pair of rates (including the rate of the virtual player) is at most $(B-R)\epsilon/k^2$ after $T = O(\alpha k^2 \log(\alpha k))$ phases, which implies the theorem.

For every $i \in [k+1]$ and $t \geq 1$, let $r_i(t)$ denote the rate of player i after step t and let $r_i(0)$ denote the initial rate. For $t \geq 0$, let

$$\Delta_t \; = \; \max_{i,j \in [k+1]} (r_i(t) - r_j(t))$$

denote the maximum distance after t. We define the following potential function

$$P_t \; = \; \alpha \sum_{\{i,j\} \in \mathcal{K}} |r_i(t) - r_j(t)| + \sum_{i \in \mathcal{K}} |r_i(t) - r_0(t)| \; .$$

Observe that $k\Delta_t \leq P_t \leq \alpha k^2 \Delta_t$, for every $t \geq 0$. Hence, we only have to show that the value of the potential function drops below $(B-R)\epsilon/k$ after $T = O(\alpha k^2 \log(\alpha k))$ phases.

For ≥ 1, define $\delta_t = |r_{i_t}(t) - r_0(t)|$, i.e., the distance between the virtual player and the activated player i_t. We observe $P_t \leq P_t - \delta_t$ because

$$|r_{i_t}(t) - r_0(t)| \; = \; |r_{i_t}(t-1) - r_0(t-1)| - \delta_t$$

and, for every $j \in \mathcal{K} \setminus \{i_t\}$,

$$\alpha|r_j(t) - r_{i_t}(t)| + |r_j(t) - r_0(t)| \; \leq \; \alpha|r_j(t-1) - r_{i_t}(t-1)| + |r_j(t-1) - r_0(t-1)| \; .$$

Thus, the rate of the virtual player changes by $\frac{\alpha}{\alpha+1}\delta_t$ during step t. In other words, the potential decreases by the distance that the virtual player moves times $\frac{\alpha+1}{\alpha}$.

Now, for $T \geq 1$, let $P_{(T)}$ and $\Delta_{(T)}$ denote the potential and the maximum distance, resp., at the end of phase T, and let $P_{(0)}$ and $\Delta_{(0)}$ denote the corresponding initial values. Observe that the distance traveled by the virtual player in phase T is at least $\frac{\Delta_{(T-1)}\alpha}{\alpha+1}$ because its rate is averaged with the smallest and the largest rate in every phase. As a consequence,

$$P_{(T)} \; \leq \; P_{(T-1)} - \frac{\alpha+1}{\alpha} \cdot \frac{\Delta_{(T-1)}\alpha}{\alpha+1} \; = \; P_{(T-1)} - \Delta_{(T-1)} \; .$$

Applying $\Delta_{(T-1)} \geq \frac{P_{(T-1)}}{\alpha k^2}$ and $P_{(0)} \leq \alpha k^2 B$ gives

$$P_{(T)} \; \leq \; \alpha k^2 B \left(1 - \frac{1}{\alpha k^2}\right)^T \; .$$

Finally, we observe that

$$P_{(T)} \; \leq \; \frac{\epsilon B}{2k} \; \leq \; \frac{(B-R)\epsilon}{k}$$

for $T = O(\alpha k^2 \log(\alpha k/\epsilon))$. This completes the proof of Theorem 29.

Technical remarks. On the first view it might seem that the speed of convergence should be polylogarithmic rather than polynomial in k. In fact, under a randomized sequence of activations of players the rates would converge within $O(\log k)$ phases. A simple counterexample, however, shows that the adversary can force the process to take a linear number of phases until all players come close. This counterexample is given in Section 8.6.

The system of rates can also be interpreted as a simple physical system in which we are given $k+1$ perfectly isolated rooms that initially have different temperatures. The rooms 1 to k have a door leading to room 0. If such a door is opened then the temperatures in both rooms are averaged. Clearly, if all doors are used frequently then the temperatures in all rooms will come closer and closer. In other words, the entropy of the physical system decreases.

This metaphor suggests to consider the entropy as a potential function, e.g., in form of the sum of the squares of the rates or in form of the relative entropy (Kullback Leibler divergence). In fact, both of these potential functions can also be used in order to show the convergence. However, these functions do not decrease as fast as the potential function P (e.g. on the initial instances of the counter example given in the Section 8.6) and, hence, lead to slightly weaker upper bounds on the performance. $\square$

8.3 Fairness versus Full Utilization

We will now show that under adversarial timing fairness and full utilization cannot be both satisfied.

Proof of Theorem 30:

Assume that such a protocol $\mathcal{P}$ exists. We start with two players Tom and Tina. At the beginning Tom allocates all the bandwidth and Tina none at all. The adversary activates Tina only if the free bandwidth is smaller than $\frac{\epsilon}{(k+1)^2}$, where k denotes the number of Tinas active rounds. Particularly this implies that Tina is activated again if she allocates more than the free bandwidth (system overload). Since the protocol has to resolve this blockade, we consider only the last allocation of Tina in this sequence.

If the protocol converges to full utilization, Tina is activated infinitely often. If not, Tom would remain alone in a closed system period where the wasted bandwidth never falls below a constant value which contradicts our assumption.

So, Tina can allocate additional bandwidth of at most $\frac{\epsilon}{k^2}$ in her k-th active round. Hence, her overall bandwidth is bounded by $\sum_{i=1}^{\infty} \frac{\epsilon}{k^2} = \frac{\pi^2}{6}\epsilon$. $\square$

8.4 VPP Converges against $\frac{1}{\alpha}$-max-min Fairness

Proof of Theorem 31:

Fix a closed system phase $(\mathcal{I}, \mathcal{K})$. W.l.o.g., we assume $\mathcal{K} = \{1, \ldots, k\}$ and all other players have rate zero. We show that the VPP converges against a particular state S which we describe in the following paragraph.

For every edge e, we define an additional, *virtual player* whose path contains only edge e. The rate of this player is defined by the unused bandwidth of edge e times α. The set of virtual players is called $\mathcal{K}'$. Now let us imagine for a moment that virtual players have a rate independent from the unused bandwidth of the respective edge, that is we want to treat virtual players like original players, except that the bandwidth used by a virtual player is only $\frac{1}{\alpha}$ times its rate. Suppose we increment all rates including the rates of the virtual players in round-robin fashion with infinitesimal increments, starting with all rates being zero, until the bandwidth capacities of the narrowest edges are reached. At this point, we stop to increase the rates for all paths using one of these edges and continue with the remaining paths in the same fashion until all rates are settled. Let us denote the final state of this process by S.

We observe that S utilizes the bandwidth of all edges if we take into account also the bandwidth occupied by the virtual players. From now on, we consider the bandwidths occupied by the virtual players again as unused bandwidth. For player i, let $e(i)$ denote one of its *bottleneck edge*, i.e., an edge because of which it stopped increasing the bandwidth. By our incremental construction, the rate of player i in state S is equal to the final rate of the virtual player of $e(i)$. In other words, the rate of every player i in S is α times the unused bandwidth on its bottleneck edge $e(i)$. Furthermore, the values of the unused bandwidth on all other edges on $path(i)$ are not smaller than this value. This implies that S is a *fixed point*, i.e., the VPP does not diverge from state S once it reaches this state. Furthermore, we can observe that S satisfies $\frac{1}{\alpha}$-max-min fairness since increasing the rate of a player by more than $1 + \frac{1}{\alpha}$ would exceed the capacity on its bottleneck edge. (In fact, S yields min-max fairness if we take into consideration also the rates of the virtual players.) Therefore, it remains only to show that the VPP converges against the fixed point S.

For an edge $e \in E$, let $r_0^*(e)$ denote the value of the rate of the virtual player on e in the steady state S. Define $R^* = \{r_0^*(e) | e \in E\}$. Define $m = |R^*|$. (Observe that possibly $m < |E|$.) Let $r[1], \ldots, r[m]$

denote the elements from R^* in increasing order, and define $E[\ell] = \{e \in E | r_0^*[e] = r(\ell)\}$ for $1 \leq \ell \leq m$. Furthermore, let $\mathcal{K}[\ell] \subseteq \mathcal{K}$ denote those players whose bottleneck edge is in $E[\ell]$, i.e., the set of players whose steady state rate is equal to $r[\ell]$. We will show by induction on ℓ that the rates of the players in $K[\ell]$ will converge against $r[\ell]$.

Claim 1 *Let $\gamma > 0$ denote any positive real number. For every $\ell \in \{1, \dots, m\}$, there exists $\tau \geq 1$ such that, after phase τ, the rates of all players in $K[\ell]$ are in the interval $[r[\ell] - \gamma, r[\ell] + \gamma]$.*

In the rest of the remaining analysis we will show this claim using induction. Let $B^*(e)$ denote the unused bandwidth on edge e if we assume that the players $\mathcal{K}[1] \cup \cdots \cup \mathcal{K}[\ell - 1]$ have bandwidths as described by S and all other players have rate zero. In fact, we can assume by induction that the rates of all players in $\mathcal{K}[1] \cup \cdots \cup \mathcal{K}[\ell - 1]$ deviate at most by $\pm\beta/k$ from their values in S for any $\beta > 0$. Under this assumption, the bandwidth available for the players in $\mathcal{K}^* = K[\ell] \cup \cdots \cup K[m]$ on edge e fluctuates only within the interval $[B^*(e) - \beta, B^*(e) + \beta]$. Observe that we can choose β arbitrary small. Nevertheless, we need to take into account these fluctuations explicitly because phases can have arbitrarily length so that a small change in the bandwidth at any given time potentially has vast consequences on the system of rates in later time steps that might be even in the same phase.

In the following, we consider only the players in $\mathcal{K}^*$, that is, we ignore the players from $\mathcal{K}[1] \cup \cdots \cup \mathcal{K}[\ell - 1]$ but we take into account the small fluctuations they cause as follows. We define that the maximal available bandwidth on edge e is $B'(e) = B^*(e) + \beta$ but, in each step t, players may observe a slightly disturbed bandwidth $B'_t(e) \in [B'(e) - 2\beta, B'(e)]$. By our construction, none of the players in $\mathcal{K}^*$ uses an edge from $E[1] \cup \cdots \cup E[\ell - 1]$. Therefore, we can restrict our attention to the set of edges $E^* = E \setminus (E[1] \cup \cdots \cup E[\ell - 1])$. Let $\mathcal{K}''$ denote the set of virtual players of edges in E^*.

Now fix an edge e. Let C denote the number of players on this edge. Let $B = B'(e)$ denote the maximal bandwidth of this edge, and C the number of players whose paths contain e. If an external observer only sees the behavior of the rates on edge e without knowing any details about the rest of the network then he can observe a behavior which is covered by the following protocol.

Adversarial VPP
Suppose player j performs update operation σ_t. Let $\bar{r}(t-1) = B - \sum_{i=1}^{C} r_i(t-1)$. Then player j sets

$$r_j(t) := \frac{\alpha}{\alpha + 1}(r_j(t-1) + \bar{r}(t-1) - X_t) \ ,$$

where $X_t \in [0, \bar{r} + 2\beta]$ is selected by an adversary.

The adversarial sequence X models the disturbing influence due to other edges and bandwidth fluctuations simultaneously. Let $r_0 = \alpha\bar{r}$ denote the bandwidth of the virtual player, also called player 0. Furthermore, let Z denote the fix point of the protocol under the assumption that $X_t = 0$, for all t, that is,

$$Z \ = \ \frac{B\alpha}{\alpha k + 1} \ = \ \frac{(B^*(e) + \beta)\alpha}{\alpha k + 1} \ \geq \ r[\ell] \ .$$

Observe that at least one player $i \in [C + 1]$ satisfies $r_i \geq Z$ at any given time. Define $r_{\min} = \min_{i \in [C+1]}(r_i(0))$, i.e., the smallest initial rate.

Lemma 42 *Assume $r_{\min} \leq Z - \epsilon$, for any $\epsilon > 0$. Then in every time step after performing one phase, $r_0 \geq r_{\min} + \epsilon 2^{-C-1}$.*

Proof: The lemma follows because of the following monotonicity property of the VPP on single edges: Given an adversarial sequence X, increasing X_t, for any t, increases $r_0(t')$ and does not increase $r_i(t')$, for every $t' \geq t$, $1 \leq i \leq C$. (This property can be shown easily by induction. Observe that monotonicity against adversarial bandwidth fluctuations holds only for single edges. In networks with several edges, reducing the bandwidth of a single edge can decrease and increase rates on other edges. In fact, a small local change in bandwidth can have strong influence on the rates of remote edges. For an example showing exponential effects in a similar context see [AMO96]. Here we cover these vast inter-dependencies among different edges by worst-case assumptions based on the adversarial sequence X.) Because of this monotonicity property, we can assume in the following that $X_t = 0$, for all t, without increasing the rate of the virtual player.

Next we observe that either the initial value of r_0 is at least Z or there is another player with at least rate Z that performs an update during the first executed phase. Consequently, there is a step t^* in the first phase yielding

$$r_0(t^*) \;\geq\; \frac{\alpha(Z + r_{\min})}{\alpha + 1} \;\geq\; r_{\min} + \frac{\epsilon}{2} \;.$$

Once more, we apply monotonicity and assume, w.l.o.g., that all updates σ_t $(t > t^*)$ moving the virtual player upward are skipped, i.e., all updates with $r_j(t-1) \geq r_0(t-1)$. Under this assumption, each player is called at most once after t^* because $r_j \geq r_0$ in all time steps after its first update. Now a straightforward induction shows that

$$r_{\min} + \frac{\epsilon}{2^{i+1}} \;,$$

after the ith of at most C updates. Clearly, this proves the lemma. □

Now let us take into account all edges again. We consider *double phases*, i.e., pairs of contiguous phases. Let $R_{\min}$ denote the minimal rate over all players in $\mathcal{K}^* \cup \mathcal{K}''$ at the beginning of a double phase. Suppose $R_{\min} \leq r[\ell] - \epsilon$. Then Lemma 42 gives a lower bound on the rates of the virtual players after the first phase, namely $r_0^*(e) \geq R_{\min} + \epsilon 2^{-C-1}$, for every $e \in E^*$, where C denotes the maximum number of players on the same edge. Thus, in the second phase, each player is averaged with a virtual player of value at least $R_{\min} + \epsilon 2^{-C-1}$ so that, after the execution of one double phase, the minimum rate over all players increases to

$$\frac{\alpha R_{\min} + (R_{\min} + \epsilon 2^{-C-1} - 2\beta)}{\alpha + 1} \geq R_{\min} + \frac{\epsilon 2^{-C-2}}{\alpha + 1} \;,$$

provided $\beta \leq \epsilon 2^{-C-4}$. Consequently, all rates will have value at least $r[\ell] - \epsilon$ after a finite number of phases.

Finally, we observe that this lower bound on the minimal rates also upper-bounds the maximal rate for edges from $E[\ell]$. For $\alpha = 1$ the maximal rate is $r[\ell] + k(\epsilon + \beta)$ as $r[\ell]$ denotes the average rate over all players. For general $\alpha > 1$ a small calculation shows the maximal rate is $r[\ell] + \gamma$ with $\gamma = O(\alpha k \epsilon)$. This proves Claim 1 and, hence, completes the proof of Theorem 31. □

8.5 The Discrete Virtual Player Protocol

Proof of Theorem 32: We now introduce a discrete version of the VPP that guarantees to reach a fair and efficient allocation within a small number of stages. Here "discrete" means that rates of active players are of the form $(1 + \epsilon)^s$, for integral s and positive, real ϵ. We use $\lceil\cdot\rceil$ to indicate upward rounding w.r.t. this representation. Let $\epsilon > 0$, $\alpha \geq 1$, and $\gamma \geq 1$ denote global parameters whose actual values will be determined during the analysis.

Discrete Virtual Player Protocol
Suppose player j performs update operation σ_t. For every edge e, let

$$\bar{r}(e) = B(e) - \sum_{i \in \mathcal{K}(e)} r_i(e)$$

denote the free bandwidth on edge e. Set $M = \min_{e \in path(j)}(\bar{r}(t))$.
If $r_j(t-1) \notin [\alpha M, (1+\epsilon)^\gamma \alpha M]$ then

$$r_j(t) = \left\lceil \frac{\alpha}{\alpha + 1}(r_j(t-1) + M) \right\rceil \;.$$

For analyzing the discrete VPP, we use a similar approach as for the fractional VPP. We define a virtual player for each edge e. The rate of this player is denoted by r_e and we define $r_e(t) = \lceil \alpha \bar{r}(t) \rceil$. Let us ignore the rounding for a moment. Then we can summarize the above protocol as follows. An update of player j brings the rate r_j in line with the minimum virtual rate r over all $e \in path(j)$, unless r_j is only slightly larger than r, that is, unless $r \in [r, (1+\epsilon)^\gamma r]$. In the following, the minimal bandwidth over all players including virtual players is denoted by $r_{\min}(t)$, for $t \geq 0$.

Observation 1 *The sequence of rates $r_{\min}$ is non-decreasing.*

Let us partition time into super-phases. Each of these super-phases consists of $2CD$ phases or CD double phases. We will use the discrete rates in order to show that $r_{\min}$ increases by a factor of $1+\epsilon$ in each super-phase until the system of rates runs into a bottleneck. More formally, for $T \geq 0$ let $r_{\min}^{(T)}$ denote the value of $r_{\min}$ at the end of super-phase T. We will show by induction that $r_{\min}^{(T)} \geq r_{\min}^{(0)}(1+\epsilon)^T$, where T^* denotes the first super-phase in which at least one edge "settles down". In super-phase T, an edge is called *settled* if the rates of all players on the edge are within the interval $[r_{\min}^{(T-1)}, r_{\min}^{(T-1)}(1+\epsilon)^\gamma]$ and the rate of the virtual player is exactly $r_{\min}^{(T-1)}$.

Observation 2 *Once an edge settles during any super-phase, the rates of the players crossing this edge are fixed forever.*

Now let us fix an arbitrary super-phase. We assume that there is no settled edge at the beginning of the super-phase. W.l.o.g., the super-phase starts with update σ_1 and the smallest initial rate in the super-phase is $r_{\min} = 1$. We need the following three lemmas in order to show $r_{\min} \geq (1+\epsilon)$ at the end of the super-phase.

Lemma 43 *For every non-virtual player i, if $r_i(t) \geq (1+\epsilon)$ then $r_i(t+1) \geq (1+\epsilon)$, for every $t \geq 0$.*

Proof: As all players including the virtual players have at least rate one, we can conclude that $\bar{r}_e(t) \geq \frac{1}{\alpha(1+\epsilon)}$, for every $e \in E$. Thus, $r_i(t) \geq 1+\epsilon$ implies

$$r_i(t+1) \;\geq\; \frac{\alpha\left((1+\epsilon)+\frac{1}{\alpha(1+\epsilon)}\right)}{\alpha+1} \;>\; \frac{\alpha(1+\epsilon)+(1-\epsilon)}{\alpha+1} \;,$$

so that $r_i(t+1)$ is rounded up to $1+\epsilon$. □

Lemma 44 *For every non-virtual player i, if i is called in step $t+1$ with $r_i(t) = 1$ and all virtual player on its paths have at least rate $1+\epsilon$ then $r_i(t+1) \geq 1+\epsilon$, for $t \geq 0$.*

Proof: As all virtual players on i's path have at least rate $1+\epsilon$, the unused bandwidth on each of these edges is at least $\frac{1+x}{\alpha}$, for some $x > 0$. Consequently,

$$r_i(t+1) \;\geq\; \left\lceil \frac{\alpha\left(1+\frac{1+x}{\alpha})\right)}{\alpha+1} \right\rceil \;\geq\; 1+\epsilon \;.$$

□

Lemma 45 *For every non-settled edge e, in every phase there is at least one update σ_t after or before which the virtual player of e has at least rate $1+\epsilon$.*

Proof: Either the virtual player has rate $1+\epsilon$ already at the beginning of the phase or at least one of the players must have rate larger than $(1+\epsilon)^\gamma$, otherwise the edge would be settled. Let us assume that the virtual player has rate one. Then the unused bandwidth is at most $\frac{1}{\alpha}$. Let j denote a player with at least rate $(1+\epsilon)^\gamma$. During the phase, j updates its rate at least once. Let t denote the corresponding time step. Then

$$r_i(t+1) \;\geq\; \left\lceil \frac{\alpha\left(1+\epsilon)^s+\frac{1}{\alpha}\right)}{\alpha+1} \right\rceil \;.$$

We set $\gamma = 1 - \log_{1+\epsilon}(1 - \alpha\epsilon)$. (If we assume $\epsilon \leq \frac{1}{2\alpha}$, then $\gamma \leq 3\alpha$.) This is the minimal assignment yielding $r_i(t+1) \leq (1+\epsilon)^{s-1}$. This way, the rate of player j is decreased by more than ϵ, which in turn implies that the unused bandwidth increases by more than ϵ, so that the rate of the virtual player is at least $\lceil \frac{1}{1-\epsilon} + \alpha\epsilon \rceil \geq 1 + \epsilon$. □

Lemma 43 implies that we only need to show that each player with rate one at the beginning of the super-phase is *lifted up* (i.e., its rate is set to $1+\epsilon$) once during the super-phase in order to show that all players have at least rate $1+\epsilon$. Now consider a double phase. Lemma 45 shows that every virtual player gets *loaded* (i.e., the virtual rate is set to minimum $1+\epsilon$) at least once during the first phase of the double phase. Furthermore, Lemma 44 shows that a player with rate one is lifted up if all virtual players on its paths are loaded. We conclude that every player with initial rate one is lifted up at his first update during the second phase of the double phase unless there is one virtual player on his path that it not loaded anymore, which means that this virtual player has lifted up another player before.

Let us call players sharing an edge *neighbors*. We conclude that, for every double phase κ and every player i with initial rate one, either player i or at least one of his neighbors is lifted up during κ. This implies that all players are lifted up during a super-phase consisting of DC double phases as each player has at most $DC - 1$ neighbors.

We summarize, the minimum rate $r_{\min}$ increases by a factor of $1+\epsilon$ in every super-phase until at least one edge settles down. Now let us set $\alpha = \frac{1}{\delta}$ and $\epsilon = \frac{\delta^2}{6}$.

Lemma 46 *The set of settled edges and players satisfy δ-max-min fairness.*

Proof: By definition, the players on settled edges have a rate in $[1, (1+\epsilon)^\gamma]$ and the unused bandwidth is at most $1/\alpha$. Hence, one cannot increase the rate r of one of the players by a factor of $1 + \frac{1}{\alpha} = 1 + \delta$ without exceeding the unused bandwidth or decreasing the bandwidth of a player with rate $(1+\epsilon)^\gamma r \leq (1+\epsilon)^{3\alpha} \leq (1 + 6\alpha\epsilon)r \leq (1+\delta)r$, which corresponds to the definition of δ-max-min fairness. □

Now suppose one or more of the edges settle down. Then we can exclude these edges and the rates of those players using one of them from our considerations as the corresponding rates are fixed forever. Hence, we can treat the system of remaining edges and players analogously to the original system. This way, we continue following the allocation process until we find that the rates of all players are fixed in a δ-max-min fair state.

It remains to analyze how many super-phases it takes until all players are settled. W.l.o.g., let us assume that the capacities of the edges are from the interval $[1, B]$. Then one can show that the minimum rate after the execution of only one double phase is $\Omega(2^{-C}/k)$. (This follows analogously to the lower bound on the increase of rates per double phase that we have done for the fractional VPP.) Furthermore, after the last super-phase the minimum rate among the players having survived until the end of our construction is $O(B/k)$. As the minimum rate among the surviving players increases by a factor of $1+\epsilon$ per super-phase, we conclude that the process settles down after $O(C + (\log B)/\epsilon)$ super-phases, which corresponds to $O(DC^2 + DC(\log B)/\epsilon) = O(DC^2 + DC(\log B)/\delta^2)$ phases. Thus, Theorem 32 is shown. □

8.6 A Lower Bound for VPP

The k players $\{1, \ldots, k\}$ start with bandwidths $0, 1, 2, \ldots, k-1$ with no wasted bandwidth, i.e. $r_i(0) = 0$ and $B = k(k-1)/2$. In round t we activate player $j = (t-1) \bmod k + 1$ and update his bandwidth by $r_j = \frac{1}{2}(r_j + r_0)$. This implements the Virtual Player Protocol for $\alpha = \frac{1}{2}$. So, player 1 is the first to begin in a phase which consists of k rounds. Note that for a closed system period the VPP protocol is equivalent to a *balancing circuit*. Such a circuit can be described by a directed acyclic graphs with n sources and n sinks where every node has in- and out-degree 2. At the inputs $r_0, \ldots, r_{n-1} \in \mathbb{R}$ are given. Every node balances the values on the inputs a, b to the outgoing values $\frac{1}{2}(a+b)$, $\frac{1}{2}(a+b)$. The behavior of such a circuit can be described as the product P of the matrices denoting the balancing of the inputs. If we iteratively apply this circuit it corresponds to repeat the activation strategy. Now we can apply standard theory of Markov chains and it follows that the rate of convergence depends on the eigenvalues of P, see [AHS94, Mih89, Fil91]. Using such Markov chain methods it is possible to analyze this example for fixed k. However, it is not clear how these results can be generalized. Therefore we follow a different approach.

Note that in [RSW98] it is shown that the convergence behavior of balancing circuits also holds if applied to a discrete domain. Therefore, this result transfers to the VPP in closed system periods with periodical adversarial behavior.

Lemma 47 *At the end of phase τ we have for the bandwidths r_i of players $i \geq 2\tau \log k$:*

$$i - \tau - \frac{\tau}{k} \leq r_i \leq i - \tau + \frac{\tau}{k} \, .$$

Proof: We prove this claim by induction. For the first round observe that for $t \in \{1, \ldots, k\}$: $r_i(t) = r_0(t) = t - 1 + t2^{-t}$.

For the inductive step we know that $r_0(\tau k + 2\tau \log k) \in [0, k-1]$ (From now on we use interval arithmetic and $x \pm y$ as convenient notation for $[x-y, x+y]$). We claim that for $i \geq 0$:

$$r_{\tau k + 2\tau \log k + i}((\tau+1)k) = r_0(\tau k + 2\tau \log k + i) \in i - \tau \pm \left(k + \frac{\tau}{k}\right) 2^{-i} \, .$$

This follows by

$$\begin{aligned}
r_{\tau k + 2\tau \log k + i}(\tau k) &= \frac{1}{2} r_{(\tau-1)k + 2\tau \log k + i}((\tau+1)k) + \frac{1}{2} r_0(\tau k + 2\tau \log k + i) \\
&\in \frac{1}{2}\left(i - (\tau - 1) \pm \left(k + \frac{\tau - 1}{k}\right) 2^{-i-2\log k}\right) \\
&\quad + \frac{1}{2}\left(i - 1 - \tau \pm \left(k + \frac{\tau}{k}\right) 2^{-i+1}\right) \\
&\subseteq i - \tau \pm \left(\frac{1}{k} 2^{-i} + \frac{\tau}{k^2} 2^{-i} + k 2^{-i} + \frac{\tau}{2k} 2^{-i}\right) 2^{-i} \pm k 2^{-i} \\
&\subseteq i - \tau \pm \left(k + \frac{\tau}{k}\right) 2^{-i}
\end{aligned}$$

□

This Lemma implies that VPP cannot reduce the maximum bandwidth difference by a factor of two within $\frac{k}{4 \log k}$ phases and $\frac{k^2}{4 \log k}$ rounds.

A similar but more lengthy proof improves this bound to $\frac{k - 2\log n}{4}$ phases. Then, we replace the activation schedule by double-phases of sequences $1, 2, \ldots, k, k \ldots, 2, 1$ for the same start configuration.

Chapter 9

Tree Network Design for the Cost-Distance-Model

9.1 Introduction

Given n terminal points in the Euclidean space we investigate the problem of constructing a network with small cost and short distances. This research is motivated by a number of practical problems arising in network design for traffic in communication networks as well as real traffic in street or railway networks. If one minimizes only the network size, i.e. the sum of all edge lengths, some distances between terminals had to be considerably increased. On the other hand if we minimize the distances between all terminals we face a complete network with large costs.

We want to investigate a measure considering the static network size and a more dynamic component that considers the point-to-point distances as well as the number of messages/vehicles using this route. In the case of a street network the static costs account for construction and maintenance, while the dynamic costs described by the sum of the mileage of all cars account for the fuel cost of all cars. In the case of a communication network we observe that there is a fixed cost for the physical network, while highly used connections need additional hardware, such as more parallel wires or additional hardware, describing this dynamic component.

In practice network designers model the demand in a network by a so-called *origin-destination matrix* $w(u, v)$. For sites u, v it describes the traffic starting at u with destination v. We model the cost of the network for each edge by a linear function $c_1||e||_2 + c_2 \sum_{(u,v)\in P(e)} w(u, v)||e||_2$ for $c_1, c_2 > 0$, where $||e||_2$ denotes the Euclidean length of the edge and $P(e)$ is the set of all pairs (u, v) such that the shortest path between u and v contains e. By summing over all edges we define the *Weighted Cost-Distance* (WCD) of a network N and a weighting w:

$$\mathrm{WCD}_w(N) \quad := \quad \sum_{e\in E(N)} \left(c_1||e||_2 + c_2 \sum_{(u,v)\in P(e)} w(u, v)||e||_2 \right) . \tag{9.1}$$

Thus, for a pair u, v with large weight $w(u, v)$ (frequent traffic) a detour between u and v implies higher costs than between pairs with smaller weight.

There is a trade-off between cost and weighted distance. If we choose $c_2 = 0$ we face the intensively studied *minimum network problem*. If we choose $c_1 = 0$, the optimal solution is a complete network for sites in general position and positive weights. As we scale the parameter c_1/c_2 from 0 to ∞, we see a gradual transformation from the Steiner tree to the complete network. We are interested in the structure of the intermediate states.

For simplicity we replace the above definition by the following. Since we only consider $c_2 > 0$, we can set $c_1 = c_2 = 1$ if we simultaneously modify the weighting by $w'(u, v) = \frac{c_1}{c_2} w(u, v)$. This results in the following equivalent version of the Weighted Cost-Distance:

$$\text{WCD}_w(N) \quad := \quad \sum_{e \in E(N)} c(e) + \sum_{u,v \in V(N)} w(u,v) L_N(u,v) \ , \tag{9.2}$$

where $c(e)$ denotes the cost of an edge and $L_N(u,v)$ the length of the shortest path from u to v in the network N. We use this notation throughout this chapter. The corresponding optimization problem is defined as follows.

Definition 12 *Let $L_G(u,v)$ denote the minimum length of a path of node u to v in graph G.*

- Weighted Cost-Distance Network problem (CDN): *Given a set of sites V in Euclidean space and a weighting $w : V \times V \mapsto \mathbb{R}^+$, find a network $N = (V,E)$ that optimizes the Cost-Distance WCD$_w(N)$ (according to equation (9.2)).*
- Weighted Cost-Distance Tree problem (CDT): *Given V and $w : V \times V \mapsto \mathbb{R}^+$, find a tree $T = (V,E)$ that optimizes the Cost-Distance WCD$_w(T)$.*

In addition to the sites we allow the use of a non-terminal node set, if not explicitly stated otherwise.

9.1.1 Previous Work

If the weights are set to zero, and no restrictions for the non-terminals are given the Weighted Cost-Distance problem reduces to the *Euclidean Steiner Tree problem*. It was shown to be NP-hard by Garey, Graham and Johnson [GGJ76]. However, in his groundbreaking paper Arora [Aro98] showed that this problem admits a polynomial time approximation scheme.

In [KRY95] the *Balanced Spanning Tree* problem was introduced. Here, the task is to find a tree which optimizes the term

$$\sum_{e \in E(T)} c(e) + \sum_{s \in V} L_T(s,r)$$

for a given root r under a metric c (not necessarily Euclidean). Non-terminal sites are not available.

The authors prove the existence of trees where the dilation of all nodes' distances from the root is bounded by any $\alpha > 1$ and the trees cost is at most β times the cost of the minimum spanning tree, where $\beta = 1 + \frac{2}{\alpha - 1}$. This leads to a constant polynomial time bounded approximation algorithm.

The Balanced Spanning Tree problem is a variant of the Weighted Cost-Distance Network problem, if we allow general metrics and exclude non-terminal nodes. The weighting is limited to $w(r,u) = 1$ and $w(u,v) = 0$ for $u,v \in V \setminus \{r\}$. For this problem in [KRY95] it is shown that a tree is always part of the optimal solution and approximating networks can be pruned to trees. Hence, here the Cost-Distance Network problem reduces to the Cost-Distance Tree problem.

Meyerson et al. [MMP00] generalize this problem by introducing a positive node weighting, and by allowing two different metrics for cost and distance: the length metric ℓ and the cost metric c. The Cost-Distance measure is given by

$$\sum_{e \in E(T)} c(e) + \sum_{s \in V} w(s) L_T(s,r)$$

for a root r. They present a polynomial time bounded randomized algorithm approximating the problem within a factor of $O(\log n)$. Furthermore, they show that the optimal solution is always a tree.

A t-spanner is a connected partial graph of a given graph G such that for all nodes $u,v \in V(G)$ the corresponding shortest path in the t-spanner is at most t times longer than in G. There exist t-spanners in Euclidean space, the sizes of which are bounded linearly by the size of the minimum spanning tree [ADM$^+$95]. It turns out that these spanning networks already allow us to state constant factor approximation algorithms for the Weighted Cost-Distance Network problem.

Theorem 33 ([ADM$^+$95]) *In k-dimensional Euclidean space, for any $t > 1$ there exists a t-spanner with size $O(c(MST)$ which can be computed in time $O(n \log n)$.*

This immediately implies that t-spanners allow constant factor approximation for the CDN-problem.

Corollary 7 *For Euclidean space the Weighted Cost-Distance Network problem can be approximated by a constant factor within time $O(n \log n)$.*

For the two-dimensional Euclidean space we can pin down the constant very accurately by using the result of [LL89].

Lemma 48 *[LL89] For $r > 0$, there exists a $(1 + \frac{1}{r}) \frac{2\pi}{3\cos(\pi/6)}$-spanner of the complete graph, the size of which is at most $2r + 1$ times the costs of the minimal spanning tree.*

Optimizing the choice of t leads to the following result:

Theorem 34 *For the two-dimensional Euclidean space there exists a polynomial time approximation of the Weighted Cost-Distance Network problem with non-terminal nodes by a factor of $\frac{2\pi+3+\sqrt{4\pi^2+36\pi+9}}{3\sqrt{3}} \approx 4.23\ldots$.*

For the complete proof we refer to [Web01].

Using the results in [Bar98] and [CCG+98] one can transfer the t-spanner result of [ADM+95] to arbitrary metrics. However the cost is increased by a logarithmic term. Such t-spanners give an approximative solution for CDN:

Corollary 8 *For metric costs and distances the Weighted Cost-Distance Network problem can be approximated in polynomial time within a factor of $O(\log n)$.*

9.1.2 The Optimal Network is not a Tree

For the minimum network problem it is known that introducing non-terminal nodes helps to reduce the network costs (i.e. size) by a constant factor. The optimal choice of such nodes are Steiner points.

Many properties are known for these Steiner networks. First of all minimum networks are trees. Further, in the plane Steiner points have degree three and the angle of neighbored edges is $120°$. The number of these non-terminal points is bounded by $n - 2$.

A complete analysis of even small graphs shows that non-terminal sites also allow an improvement of a constant factor for the CDN-problem. Nevertheless, the angles between the adjacent edges may differ from $120°$.

In contrast to the Cost-Distance Problems investigated so far, it turns out that the optimal solution is not a tree. We will prove in section 9.3 that a tree can differ by at least a factor of $\Omega(\log n)$ from the optimal network. Even more surprisingly, non-terminal (quasi-Steiner points) may be involved in cycles and there may be cycles connecting only quasi-Steiner points.

Another interesting observation is that the optimal network may include crossing edges where the placement of a quasi-Steiner point onto the crossing point does not improve the solution. This reminds of the open problem [Epp00] whether optimal dilation trees contain crossings.

Examples for crossings and quasi-Steiner points can be seen in Figures 9.1, 9.2 and 9.3. A detailed discussion of these examples can be found in [Web01]. In the following section we will prove that the optimal Cost-Distance network can be approximated by a tree within a factor of $O(\log n)$. Furthermore, there is a polynomial time bounded algorithm computing such a tree, given the weighting and the sites in Euclidean k-dimensional space. In section 9.3 we prove the optimality of this approximation factor. We finally conclude these results and present some open problems for further research.

9.2 A Tree-Approximation by a Factor $O(\log n)$

Note that for k-dimensional Euclidean space the quality of the minimum networks differs from the minimum spanning tree only by a constant factor. For the Cost-Distance problem the situation is similar. Therefore we will not use any non-terminals in the following construction.

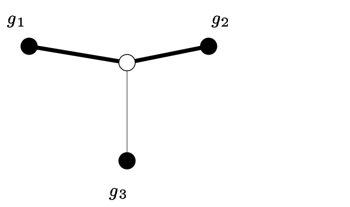

Figure 9.1: The optimal WCD-network contains a quasi-Steiner point.
$w(g_1, g_2) \gg w(g_1, g_3) \wedge w(g_1, g_2) \gg w(g_1, g_2)$

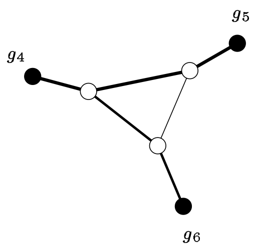

Figure 9.2: The optimal WCD-network contains a cycle.
$w(g_4, g_5) \approx w(g_4, g_6) \wedge w(g_4, g_5) > w(g_5, g_6)$

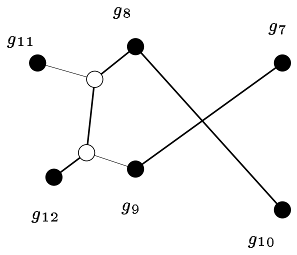

Figure 9.3: An instance where a crossing is part of the optimal solution.
$w(g_8, g_{10}) \approx w(g_7, g_9) \wedge w(g_8, g_{12}) > w(g_9, g_{11}) \wedge w(g_9, g_{10}) = w(g_8, g_7) = 0$

We use the notion of a *split tree* [ADM+95]. A *split tree* is a tree stemming from a hierarchical decomposition of a point set into k-dimensional rectangles of bounded aspect ratio, say in the range $[\frac{1}{3}, 3]$. We start with the smallest possible rectangle, $R_0 = R(V)$, including the point set V. Let r_0 be the root of the split tree. This rectangle R_0 is split into two smaller rectangles R_1 and R_2. Let $V(R)$ be the subset of nodes in rectangle R. The split tree of R_1 is the split tree for the nodes $V(R_1)$, and similarly for R_2 and $V(R_2)$. These sub-trees are connected to the root r_0, see Figure 9.4.

We will construct a *fair split tree* (FST) where each sub-tree with node set V' has a diameter of $O(kd(V'))$, where $d(V') := \max_{u,v \in V'} ||u,v||_2$. Let $\ell(R)$ be the length of the longest edge of a rectangle R. We will use the following recursive construction given a rectangle R, a root $r \in V(R)$ and a weighting w such that for some $c > 1$: $W := \sum_{u,v} w(u,v) = O(n^c)$.

1. If $\ell(R) \leq \frac{d(V)}{n^c}$, then we choose an arbitrary node $r \in R$ and connect all nodes $V(R)$ to r.
2. Otherwise, we partition the rectangle R by a hyper-plane orthogonal to an edge e with length $\ell(R)$. The distance between the hyper-plane and the ends of the longest edge is at least $\frac{1}{3}\ell(R)$. The exact position depends on the weighting and will be described in the proof of Theorem 35.

   The resulting two axis-parallel adjacent rectangles partitioning R are called R_1 and R_2.

   (a) If r is in $V(R_1)$ let $r_1 = r$ and take an arbitrary node $r_2 \in V(R_2)$ and vice versa if $r \in V(R_1)$. Insert the edge $\{r_1, r_2\}$.

   (b) Recursively, proceed with R_1, r_1 and R_2, r_2.

Note that $d(V) \leq \ell(R_0)$ and observe that after k rounds the length of the longest edge is reduced by at most a factor of $\frac{2}{3}$. So there are only $O(k \log n)$ rounds until the size of the rectangles is bounded by $\frac{\ell(R_0)}{n^c}$.

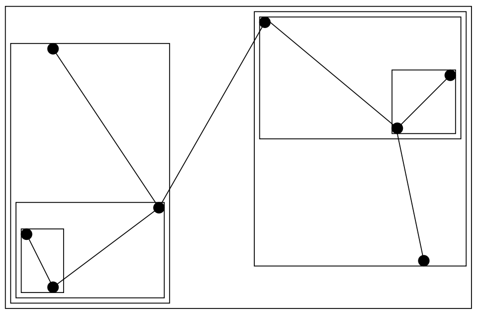

Figure 9.4: A split tree resulting form a hierarchical rectangle decomposition.

The length of every path in the resulting tree is bounded by $3k\ell(R_0)$: starting from the node of the path closest to the root, following the path downwards in both directions, the lengths of the edges $e_1, e_2, \dots$ and $e'_1, e'_2, \dots$ are upper bounded by $||e_i||_2, ||e'_i||_2 \leq (\frac{2}{3})^{\lfloor i/k \rfloor} d(V)$.

Lemma 49 *Fair split trees have diameter $3kd(V)$ and weight $O(s(MST(V))k \log n)$.*

Proof: We apply the Lemma of [Epp00, DHN93] using the isolation property. If we add non-intersecting cylinders to all edges with radius $r/3$ and distance $r/3$ to the end points, then the cost of the corresponding network is linearly bounded by the cost of the MST. (The isolation property also holds if the cylinder is replaced by other geometric objects). Note that for the edges of each recursion step, we can attach such a cylinder to an edge such that the cylinder is completely in the corresponding rectangle. Since there are at most $O(k \log n)$ recursion steps this implies the claim. □

We have not presented where we place the split. The following Lemma helps us to make a good selection.

Lemma 50 *Given rectangle R_0 and a weighting $w : V \times V \mapsto \mathbb{R}_0^+$. There exists partition of V into rectangles R_1 and R_2 with node sets V_1, V_2 such that*

$$\sum_{(u,v) \in \ V_1 \times V_2 \cup V_2 \times V_1} w(u,v) \ \leq \ \frac{3D}{\ell(R_0)}$$

where $D := \sum_{u,v \in V} w(u,v)||u,v||_2$.

Proof: Define $p := \left\lfloor \frac{\ell(R_0)}{3\Delta} \right\rfloor$ adjacent parallel rectangles R_i of thickness $\Delta := \frac{D}{W}$, where $W := \sum_{u,v \in V} w(u,v)$. These rectangles have distance of at least $\ell(R_0)/3$ to the left and right end of the longest edge of R_0. We will partition between a pair R_i and R_{i+1}.

Next consider a pair of nodes u, v with $u \in R_i$ and $v \in R_j$. Then, we have $||u,v||_2 \geq \Delta \cdot |i-j|$. Measure v_i which is the weight of all connections crossing the right border between R_i and R_{i+1}:

$$v_i = \sum_{j \leq i < k} \sum_{u \in R_j} \sum_{v \in R_k} w(u,v) + w(v,u) \ .$$

Let $i = I(u)$ denote the index of the rectangle R_i with $u \in R_i$. Note that

$$\sum_i v_i \ \leq \ \sum_{u,v \in \bigcup_i R_i} w(u,v)|I(u) - I(v)| \ \leq \ \sum_{u,v} \frac{w(u,v)||u,v||_2}{\Delta} \ = \ W \ .$$

Hence, for at least one of the rectangles R_i we have $v_i \leq \frac{W}{p} \leq \frac{3D}{\ell(R_0)}$. □

Of course, this split can be found in polynomial time if the number of partitions is not too high. If we use $2p$ rectangles, then a random partition fulfills this property with probability of at least $\frac{1}{2}$. However, the number of sites n is a lower bound of the number of different values v_i. Using this observation one can find an algorithm that always determines such a split in polynomial time, even if Δ is arbitrarily small.

Theorem 35 *Given a set of sites V in k-dimensional Euclidean space and a non-negative weighting w such that the sum of all weights is polynomial in $n = |V|$; there exists a tree with a weighted distance which differs from the optimal Weighted Cost-Distance by at most a factor of $O(k \log n)$. Such a tree has size $O(c(MST(V))k \log n)$ and can be computed in polynomial time.*

Proof: We construct a fair split tree using the partition introduced in Lemma 50. We consider the node pair sets $P_1 := V_1^2$, $P_2 := V_2^2$, and $Q := V \times V \setminus (P_1 \cup P_2)$.

It holds:

$$\sum_{(u,v) \in Q} w(u,v) L_T(u,v) \ \leq \ \max_{(u,v) \in Q} \frac{3D}{\ell(R)} L_T(u,v) \ \leq \ \frac{3D}{\ell(R)} 3k\ell(R) \leq 9kD \ ,$$

where $D := \sum_{u,v} w(u,v)||u,v||_2$ is a lower bound for the weighted distance of the optimal network. For the disjoint pair sets P_1 and P_2 we apply this technique recursively for at most $O(ck \log n)$ rounds. As we have already observed, the length of the longest edge of the sub-rectangles is at most $\ell' := \frac{\ell(R_0)}{n^c}$. Then we face partitions $P_1, \dots, P_m$ with partial weight sums $W_1, \dots, W_m$ ($W_i := \sum_{(u,v)\in P_i} w(u,v)$). The sum of all weights $W := \sum_{u,v} w(u,v)$ is bounded by a polynomial $O(n^c)$. Therefore, $\sum_i W_i \leq W = O(n^c)$. The corresponding normalized weighted distances $\Delta_i := \sum_{u,v \in P_i} \frac{w(u,v)}{W_i}||u,v||_2$ are bounded by ℓ_i, which is the length of the longest edge of the partition P_i's rectangle. Note that

$$\begin{aligned}\sum_{i\in P_i}\sum_{u,v\in P_i} L_T(u,v)w(u,v) &\leq 2\sum_i \ell_i w(u,v) \leq 2\sum_i \ell_i W_i \\ &\leq 2\ell' W \leq c'\ell(R_0) \leq c's(\text{MST}(V))\end{aligned}$$

for a suitable constant c'. This and the recurrency over $O(k \log n)$ rounds imply

$$\begin{aligned}\sum_{u,v} w(u,v)L_T(u,v) &\leq c's(\text{MST}(V)) + c''k(\log n)D \\ &\leq c'''k(\log n)\text{WCD}_w(N)\end{aligned}$$

for a suitable constant c'' and c''' and every network N. □

9.3 A Lower Bound for Tree-Approximations

Trees cannot approximate the optimal Weighted Cost-Distance graph better than stated in Theorem 9.2. To show this, we construct a counter-example where the sites are uniformly distributed and the weighting supports only neighbored sites.

In particular, we consider an $n \times n$ unit square grid G and the following weighting function:

$$w(u,v) = \begin{cases} 1 & : \quad ||u,v||_2 = 1 \\ 0 & : \quad ||u,v||_2 \neq 1 \,. \end{cases}$$

Clearly, the weighted Cost-Distance of the grid consisting of all positive weighted edges is $O(n^2)$ and since the minimum spanning tree has at least cost $n^2 - 1$, this network is optimal up to a constant factor. We will show that every spanning tree T has weighted distance $\Omega(n^2 \log n)$ even if we allow T to use non-terminal nodes.

Let G_i be the set of nodes with distance $i-1$ to the convex hull of the grid, i.e. G_1 is the convex hull and G_{i+1} is the convex hull of $G \setminus \bigcup_{j\leq i} G_j$.

Lemma 51 *For every spanning tree T of the grid and for all $i \leq n/2$ there exist two grid neighbors $u, v \in G_i$ such that the connecting path in T has at least length $\frac{n}{2}$.*

Proof: Assume the contrary and consider the upper row of G_i. Note that neighbored nodes (in the grid) are connected by a path which is too short to reach the other half of the grid. Therefore, in the upper row the leftmost and the rightmost node must be connected by a path, which lies completely in the upper half of the rectangle.

For symmetry reasons the analogous property is true for the the left column, the lower row, and the right column. Therefore, there exists a cycle enclosing the center of the grid, contradicting the tree property. □

Definition 13 (spanning cut) *A spanning cut splits a tree $T = (V, E)$ by a straight line s into trees $T_1 = (V_1 \cup S_1, E_1)$ and $T_2 = (V_2 \cup S_2, E_2)$. These sub-trees are entirely in the left or right half-space defined by s. All nodes in V_2 (resp. V_1) are orthogonally projected onto s and will be used as non-terminals S_1 in T_1 (resp. S_2 in T_2). All edges in trees T_1 and T_2 are copied from the original tree.*

So, we copy every tree into both half spaces without increasing any edge length, for an example see Figure 9.5.

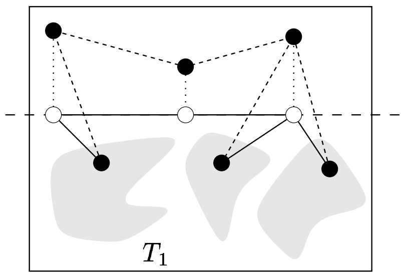

Figure 9.5: A spanning cut and the resulting sub-tree in the lower half-space

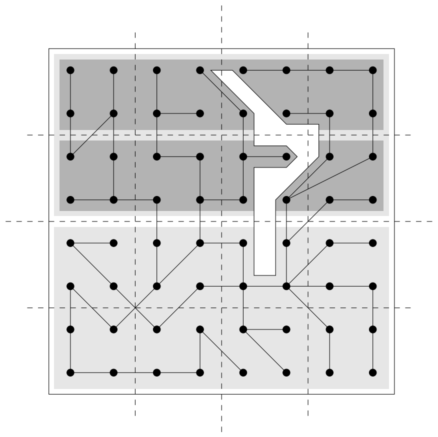

Figure 9.6: The white marked p̀-shaped area induces long paths for a number of neighbored pairs. For the lower bound the grid is tiled into 16 sub-grids

Lemma 52 *For a spanning cut of T into T_1 and T_2 we have for all $u_1, u_2 \in V(T_1)$ and $v_1, v_2 \in V(T_2)$:*

$$L_G(u_1, u_2) \geq L_{G_1}(u_1, u_2) \text{ and } L_G(v_1, v_2) \geq L_{G_2}(v_1, v_2)$$

Theorem 36 *For every spanning tree T of the $n \times n$-grid, where $w(u, v) = 1$ if u and v are neighbored nodes and $w(u, v) = 0$ elsewhere, the weighted Cost-Distance is at least $\Omega(n^2 \log n)$, while the optimal Cost-Distance network has cost and weighted distance $O(n^2)$.*

Proof: We will split this grid into 16 sub-grids of size $\frac{n}{4} \times \frac{n}{4}$ by 15 spanning cuts (Figure 9.6). By Lemma 52 the sum of the weighted distances of the sub-grids is a lower bound for the over-all grid (We also split the weighting into 16 local weighting functions).

Lemma 51 implies that in every subset G_i there are paths $p_1, \ldots, p_{n/2}$ between neighboring nodes with length of at least $n/2$. Furthermore, we can choose these paths such that the spanning cut reduces the lengths of all of them by at least $\frac{n}{4}$, since they reach the other side of the grid.

This way, we can account the length $\frac{n}{4}$ of these $\frac{n}{2}$ paths for this recursion level. This leads to the following recurrency for the weighted distance $W(n)$ of spanning trees of an $n \times n$-grid:

$$\mathrm{W}(n) \geq \frac{n^2}{8} + 16\,\mathrm{W}(n/4)\ .$$

Resolving this recurrency proves the claim. □

Applying the algorithm of Section 9.2 to this instance produces trees structured similar to the U-Layout shown in Figure 9.7. Such trees optimize the weighted Cost-Distance of an $n \times n$ grid by a factor of $\Theta(\log n)$.

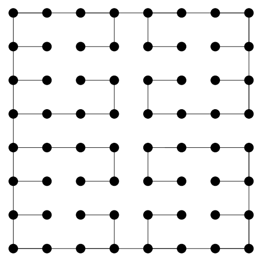

Figure 9.7: The U-Layout approximates the Cost-Distance of this instance by a factor of $\Theta(\log n)$.

9.4 Conclusions and Future Research

As an immediate implication of Theorem 35 we can state the following approximation result:

Corollary 9 *For polynomial weights the Weighted Cost-Distance-Tree problem can be polynomially approximated within a factor of $O(\log n)$.*

There is some hope that the approximation techniques introduced by Arora [Aro98] may lead to a polynomial time approximation scheme. Another follow-up result may be the extension to general metrics. We conjecture that the results of [Bar98] lead to an $O(\log^2 n)$ approximation.

An interesting open question is: If W, the sum of all weights, is super-polynomial, does the upper bound of Section 35 also apply? Or can the lower bound factor be increased for such weights? This mirrors the case in the original setting (Equation (9.1)) that the fixed costs are sub-polynomial compared to the linear costs.

Another extension of these results may be to consider different metrics for cost and distance as introduced in [MMP00]. They proved a $O(\log n)$-approximation for the two-metrics Cost-Distance problem with weights only on the root-node pairs. We have shown that for pairwise weight trees do not approximate better than $\Theta(\log n)$, while for node-root weights Meyerson et al. [MMP00] showed that a tree is always part of the optimal solution. It is an interesting open question whether trees approximate this Weighted Cost-Distance problem with different metrics within a factor of $O(\log n)$.

Chapter 10

Energy, Congestion and Dilation in Wireless Networks

10.1 Introduction

In this chapter we contribute to modeling wireless communication networks, to modeling congestion, energy consumption and delay for routing in such networks, and to designing routing paths in order to minimize these cost measures. One major insight is the fact that trade-offs are unavoidable: Minimizing one measure is only possible at the cost of enlarging another one.

Wireless ad hoc networks consist of nodes that can communicate via short-range wireless connections. Each node can be a source, a destination and a router for data packets, thus no explicit infrastructure is required to set up and maintain an ad hoc radio network. The area of application for radio networks is broad, especially in niches such as search and rescue missions or environmental monitoring. But ad hoc networks can also be used as a last-mile technology to provide access to the Internet in high-populated environments.

In wireless ad hoc networks, energy-intensive long-range connections should be avoided, and the overall distance between two communicating nodes respectively hop count should be minimized to achieve low latencies. To use the available network capacity efficiently and to achieve high bandwidths, congested connections should also be avoided by balancing the traffic over all reasonable connections.

These requirements can be expressed using three measurable quantities: congestion, energy and hop count. Traditional routing protocols such as AODV, DSDV and DSR [Per01] usually choose the path with the lowest hop count. There also exist power-aware routing protocols using different metrics (e.g., energy consumed per packet, variance in node power level) to choose the best route in order to extend the lifetime of individual nodes or the whole network [SR98, SW98, CT00]. The congestion of a route is usually not regarded directly, but some routing protocols choose routes with the shortest route discovery, assuming that the route with the quickest response is less congested (e.g., SSA [DRWT97]). However, to our knowledge, no practical work or theoretical studies exist that consider the interdependencies between these three quantities.

In radio networks it is not clear how to choose nodes as communication partners because links can interfere with each other. Our main goal in this chapter is to determine the optimal choice of this network given a set of nodes $V \subseteq \mathbb{R}^2$ (Random choices of node sets have been investigated in [AS98a, GK00]). Hence, we disregard the mobile and dynamic components of ad hoc-networking and determine the optimal static wireless network. We present a general model for congestion, energy and dilation for a given solution of the routing problem of the radio networks. (cf packet radio network model or more realistic wireless network models, for instance as in [AS98a, UY98, GK00, ABBS01, KKKP00, CNP01]). Besides the load the congestion also measures the interferences between edges.

In Section 10.2 we start our considerations with the paths of all packets solving a routing problem in a radio network. The union of all these paths, called path system, gives a natural definition of the communication network. These paths induce a load in the communication links which can interfere with each other. Combining the load and the interferences we achieve an intuitive model for the congestion of an edge of the communication network. Our definition is very similar to those in [AS98a], although they use a slightly different approach. Likewise in [AS98a] we relate the congestion and the dilation, also known as hop-distance, to the routing time of the routing problem. Then, we define measures for energy consumption, which is important for autonomous nodes that have to ”carry their energy”.

The main contributions concern path selection in wireless networks: Given a set of routing requests, find routing path so that the congestion, delay, and/or energy consumption is minimized. We introduce the notion of diversity to describe locations of node sets where high interferences are unavoidable. It turns out that if the diversity is small, i.e., all point to point distances differ only by a polynomial factor, then the interferences of communication networks can be kept small. This is key factor for the congestion avoidance analysis in this chapter.

In section 10.3 we present strategies for this path selection which provably optimize energy consumption and give a $O(g(V)^2)$-factor approximation of congestion. In section 10.4 as a main insight, we can conclude that not any two of these measures can be minimized simultaneously, but that trade-offs between measures are unavoidable. Finally, section 10.5 concludes this chapter.

10.2 Modeling Wireless Networks

We consider a set $V \subseteq \mathbb{R}^2$ of n radio stations, featuring both transmitter and receivers, called sites or nodes, in 2-dimensional Euclidean space. Let $d = \max_{u,v \in V} ||u,v||_2$ denote the geometric diameter of V.

As in the model of [MBmH01] each node $u \in V$ can adjust its transmission radius to some $r \geq 0$ for sending a packet to a neighbor $v \in V$ in range r. Then, the communication network $N = (V, E)$ has the edge $\{u, v\}$, where $||u, v||_2 = r$. Note that for adjusting the transmission power nodes exchanging packets must interact during the transmission. Just imagine that one of the nodes is moving and therefore the distance between these nodes continuously changes. In our model we simplify this interaction by assuming that the sending and acknowledging part of this interaction may interfere with any other such bi-directional connection if the distance is too small.

In particular, this means: To acknowledge this packet the receiving site adjusts its transmission radius to the same radius r as the sending radius. The transmission needs a unit time step and the area covered by sending and acknowledging a packet along $e = (u, v) \in E$ is $D(e) = D_r(u) \cup D_r(v)$, where $D_r(u)$ denotes a disk with center u and radius r. Of course edges only interfere when the routing protocol tries to send a packet at the same time and if $D(e')$ contains u or v (cp. Figure 10.1. We keep the timing aspect of interferences in mind and expand the notion of interferences to edges: Edge (u, v) interferes with edge e' if u or v is in the area $D(e')$.

We define the set of interfering edges by $\text{Int}(e) := \{e' \in E(N) \mid e' \text{ interferes with } e\}$. Note that sending a packet along e is successful only if no edge from Int(e) sends concurrently. These interferences of network N describe the directed *interference graph* $G_{Int}(N)$. Its node set are all edges of N and its edges describe all interferences, i.e., $(c, e) \in E(G_{\text{Int}}(N))$ iff $c \in \text{Int}(e)$. The interference graph can be interpreted as an additional constraint for routing. An edge of the radio network can only be used for sending a packet in a time unit if all interfering edges remain silent. The number of this interfering edges is given by the in-degree of an edge in the interference graph and is called the *interference number* of a communication link. The maximum interference number of a site u is the maximum interference number of all edges with receiving site u. The interference number of the network is the maximum interference number of all edges.

Now consider a routing problem $w : V \times V \to \mathbb{N}$, where $w(u, v)$ packets have to be sent from u to v. We subdivide the design of a routing strategy for w into the following steps:

- *Path selection*: Select a system $\mathcal{P}$ of paths R_p from source to destination for the packets p in the graph on V. The union of all edges $E_{\mathcal{P}}$ of the path system gives the links of communication network $N = (V, E_{\mathcal{P}})$.

- *Collision avoidance*: As noted above sending a packet along edge e is only successful if no $e' \in$ Int(e) sends at the same time.

Therefore, the following holds: Consider any routing strategy that routes w in T steps using the path system P. Let $\Gamma(e) \subseteq \{1, \ldots, T\}$ denote the time steps in which e sends successfully. Then clearly

$$|\Gamma(e)| + \sum_{e' \in \text{Int}(e)} |\Gamma(e')| \leq T \ .$$

As $|\Gamma(e)|$ is just the load $\ell(e)$ of e, i.e., the number of packets whose path goes through e, the above quantity is $\ell(e) + \sum_{e' \in \text{Int}(e)} \ell(e')$. We denote this quantity (which is uniquely defined by the path system $\mathcal{P}$) as congestion of the edge $C_{\mathcal{P}}(e)$. The **congestion of the path system** $\mathcal{P}$ is defined by

$$C_{\mathcal{P}}(V) := \max_{e \in E_{\mathcal{P}}} \{C_{\mathcal{P}}(e)\} \ .$$

We will denote by the dilation $D_{\mathcal{P}}(V)$ the length of a longest path in $\mathcal{P}$, also known as the hop-distance. By definition the optimal routing time T using $\mathcal{P}$ fulfills $T \ \geq \ D_{\mathcal{P}}(V)$, but also congestion gives a lower bound on the time T:

Theorem 37 *Consider a radio network M with path system $\mathcal{P}$, maximum interference number I, and a routing problem w with dilation D and congestion C. Let T be its optimal routing time, when the path system $\mathcal{P}$ is used. The following holds.*

1. $T \ \geq \ \max\{C/12, D\} = \Omega(C + D)$
2. *It exists an offline routing protocol with routing time $O(C + D \cdot I)$, with high probability.*
3. *There is an online routing protocol that needs routing time $O(C + D \cdot I \cdot \log n)$, w.h.p.*

Proof: 1. Let $e = (u, v)$ be an edge with maximum congestion C. We partition the plane into 6 regions $R_1, \ldots, R_6$ with center at u by six half-lines starting at u where the angle between neighbored half-lines is $\pi/3$. Similary we consider the analogous partitioning $R_7, \ldots, R_{12}$ with v as the starting point of the 6 half-lines.

Define

$$E_i \ := \ \{\{p, q\} \ | \ (p \in R_i \vee q \in R_i) \wedge \{p, q\} \in \text{Int}(e)\} \ .$$

Note that by a straight-forward geometric argument for two edge $e', e'' \in E_i$ it holds either $e' \in \text{Int}(e'')$ or $e'' \in \text{Int}(e')$. Therefore, all transmissions over edges in $E_i \cup \{e\}$ have to be done sequentially. Let $\ell_i := \ell(e) + \sum_{e' \in E_i} \ell(e')$. Then, $\sum_{i=1}^{12} \ell_i \geq C$. Hence,

$$T \ \geq \ \max_{i \in [12]} \{\ell_i\} \ \geq \ \frac{1}{12} \sum_{i=1}^{12} \ell_i \ \geq \ \frac{C}{12} \ .$$

The upper bounds of 2. and 3. can be proved using the same arguments as shown in Theorem 2.12 and Theorem 2.13 of [AS98a]. Note that in [AS98a] the notion dilation differs from our approach. $\square$

The variable choice of the transmitter power allows to reduce the energy consumption, saving on the tight resources of batteries in portable radio stations and reducing interferences. Theoretically, the energy needed to send over a distance of r is given by $O(r^2)$. It turns out that in practice one can model the energy by $O(r^4)$ or even $O(r^5)$. Throughout this chapter we model energy costs by $O(r^2)$. However, most results in this chapter can be easily transferred for higher exponents.

We distinguish two energy models. In the first model, called **unit energy model**, we assume that maintaining a communication link e is proportional to $O((||e||_2)^2)$, where $||e||_2$ denotes its Euclidean length. Therefore, the unit energy *U-Energy* used by radio network N is given by

$$\text{U-Energy}_{\mathcal{P}}(V) := \sum_{e \in E_{\mathcal{P}}(N)} (||e||_2)^2 \ .$$

The **flow energy model** reflects the energy actually consumed by transmitting all packets. Here, the power consumption of a communication link is weighted by the actual load $\ell(e)$ on an edge e:

$$\text{F-Energy}_{\mathcal{P}}(V) := \sum_{e \in E_{\mathcal{P}}(N)} \ell(e)(||e||_2)^2 \ .$$

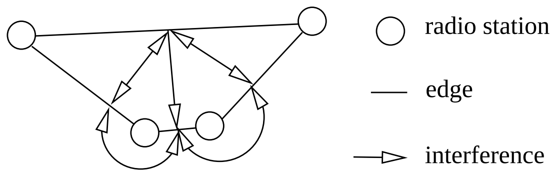

Figure 10.1: Radio stations, edges and induced interferences

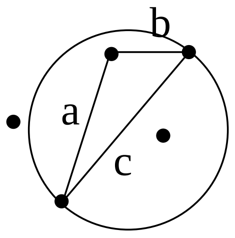

Figure 10.2: For an edge c of a Gabriel-graph no node may be in inside its disk

In this chapter we focus on the question: Given some sites, which path selection is best possible to obtain small congestion, low energy consumption and small dilation? Clearly, the optimal network for hop-distance is the complete graph. Hence, we investigate only energy and congestion.

10.3 Minimizing Energy and Congestion

10.3.1 Energy

The unit energy of a path system for a radio network is defined as the energy consumption necessary to deliver one packet on each communication link. It turns out that the minimal spanning tree optimizes unit energy. Note that the hardness results shown in [KKKP00, CPS00] do not apply because in our model the transmission radii are adjusted for each packet.

Theorem 38 *The minimal spanning tree is an optimal path system for a radio network with respect to the unit energy.*

Proof: Consider the graph defined by all edges $E \subseteq V \times V$ with edge weight $(||e||_2)^2$. The minimum energy network can be constructed using Prim's or Kruskal's algorithm for minimum spanning tree. Note that the decisions in this algorithm are based on comparison of the length of some edges e and e', i.e., $||e||_2 \leq ||e'||_2$. Thus, the minimal network for energy is also the minimum spanning tree for Euclidean distances. $\square$

For the flow energy model, the minimal network is not necessarily a tree. However, one can compute the minimal flow energy network in polynomial time.

Theorem 39 *For a given node set V a sub-graph of the Gabriel Graph [GS69, JT92] is an optimal path system for a radio network with respect to the flow energy.*

Proof: If in the interior of the circle defined by the diameter (u, v) there exists a node w, then the edges $(u, w), (w, v)$ need less energy than the original edge. This follows by the Theorem of Thales (otherwise energy is not optimal because $(||u, w||_2)^2 + (||w, v||_2)^2 < (||u, v||_2)^2$). Therefore, one can add an edge into the communication network iff there are no sites in the interior of this circle, see Figure 10.2. This matches the definition of a Gabriel graph of V.

For two nodes u and v the sub-graph providing the lowest energy for routing information from u to v is given by the shortest path in the Gabriel graph if the length of an edge is redefined by $(||e||_2)^2$. The flow

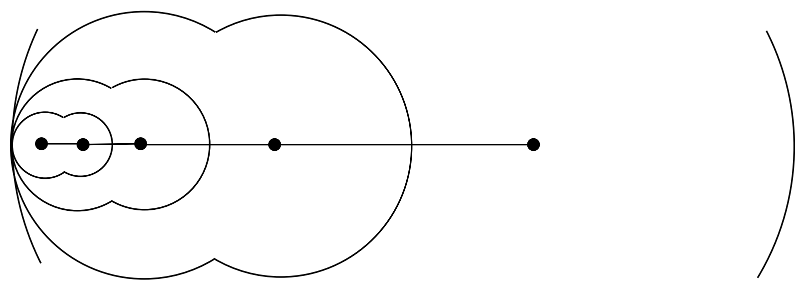

Figure 10.3: The high diversity of the node set increases the interference number, resulting in high congestion

energy of the optimal network consists of a linear combination of these lowest-energy-paths between pairs. Using an all-pair-shortest-path algorithm gives the optimal network. □

Note, that there are situations where edges of the Gabriel-graph can be replaced by less energy-consuming paths, even if no site lies inside the disk described by the edge. Then, the edge of the Gabriel graph is not part of any energy optimal route.

10.3.2 Diversity of a Node Set

Sometimes the location of the radio stations does not allow small congested routes for the radio networks at all. Consider a node set $V = \{v_1, \ldots, v_n\}$ on a line, with distances $|v_i, v_{i+1}| = 2^i$. The edge (v_i, v_{i+1}) interferes with all edges (v_j, v_{j+1}) for $j \leq i$, see Figure 10.3. Therefore, the interference number of the network is $n - 1$. Suppose only v_1 and v_n want to communicate, then the better solution for congestion is to disconnect all interior points and to realize only the edge (v_1, v_n). Of course this is not an option when interior nodes need to communicate.

It turns out that a determining parameter for the realization of optimal communication networks for radio networks, is the number of magnitudes of distances. Distances have different magnitude if they differ more than a factor of 2.

Definition 14 *The diversity $g(V)$ of a point set V in Euclidean space is defined by*

$$g(V) := |\{m \mid \exists u, v \in V \ : \ \lfloor \log ||u, v||_2 \rfloor = m\}| \, .$$

Note that in the above scenario we observe the maximum diversity of n (and a high interference number). For point sets V on the line with small diversity the interference number is small, too. It is easy to see that the interference number for a node set V on the line is at most $O(g(V))$.

Lemma 53 *The diversity of n points in $\mathbb{R}^2$ is at least $\Omega(\log n)$ and at most $O(n)$. For a point set randomly distributed in a square of $\mathbb{R}^2$ the diversity is $O(\log n)$ with high probability (i.e., $1 - n^{-c}$ for any fixed constant $c > 0$). Furthermore,*

$$g(V) \leq 1 + \log \frac{\max_{u,v \in V} ||u, v||_2}{\min_{u,v \in V \wedge u \neq v} ||u, v||_2} \, .$$

Proof:

- For a point set $V \subset \mathbb{R}^2$: $g(V) \geq \log_{19} |V|$.

  We consider the following sequential process starting with $r := \min_{u,v \in V, u \neq v} \{2^{\lfloor \log_2 ||u,v||_2 \rfloor}\}$. We start with a non-marked node set $V = \{v_1, \ldots, v_n\}$ and we sequentially visit every non-marked node v_i. Now we mark every node v_j with $j > i$ and $||v_i, v_j||_2 \in [r, 2r)$.

  After having visited all non-marked nodes we end up with a non-marked node set V', where for all $u, v \in V'$ we have $||u, v||_2 \geq 2r$. Since $r \geq \min u, v \in V, u \neq v ||u, v||_2$, a straight-forward geometric argument shows that each node may cause only 18 other nodes to be marked. Therfore, we have $|V'| \geq \frac{1}{19}|V|$. This leads directly to the claim.

- $g(V) \leq 6n - 10$ for $V \in \mathbb{R}$ with $|V| = n \geq 2$.

  Let $V := \{v_1, \ldots, v_n\}$ denote the nodes sorted from left to right. Define

  $$Q_i := \{\{x \in \mathbb{Z} \mid \exists j > i \ : \ x = \lfloor r + \log ||v_i, v_j||_2 \rfloor\} ,$$

  where r is a uniform random variable (the same for all i) of the interval $[0, 1]$. Note that $Q_n = \emptyset$ and $|Q_{n-1}| = 1$. We will prove that

  $$\mathbf{E}[|Q_i \setminus Q_{i+1}|] \ \leq \ 4 \, . \tag{10.1}$$

  Let $q_r(i, j) := \lfloor r + \log |v_i, v_j| \rfloor$ and $p := q_r(i, i+1)$. Now for all $j > i+1$ with $q_r(i+1, j) \leq q+1$ we have $q_r(i, j) \in \{q, q+1, q+2\}$ inducing three elements into the difference set $Q_i \setminus Q_{i+1}$.

  For all $j > i + 1$ with $q_r(i + 1, j) = q' \geq q + 2$ we have analogously $q_r(i, j) \in \{q', q' + 1\}$. But since $|v_i, v_{i+1}| \leq 2^{q'-q+1}|v_{i+1}, v_j|$, the probability that $q' + 1$ actually occurs in Q_i but not in Q_{i+1} can be bounded by $2^{q'-q+1}$. Note that $q' \in Q_{i+1}$ and that $q' + 1$ is added to Q_i with probability $2^{q'-q+1}$. Therefore, the expected number of elements larger than $q + 2$ added into Q_i is at most 1.

  Note that $|\bigcup_i Q_i| = g(\{2^r v_1, \ldots, 2^r v_n\})$ and therefore inequality 10.1 implies that

  $$\mathbf{E}[g(\{2^r v_1, \ldots, 2^r v_n\})] \ \leq \ 4n - 7 \, .$$

  Hence, there exists a choice $r \in [0, 1]$ with

  $$g(\{2^r v_1, \ldots, 2^r v_n\}) \ \leq \ 4n - 7 \, .$$

  Now the different rounding points by introducing a factor 2^r may at most double the diversity, which implies $g(V) \leq 8n - 14$.

- $g(V) \leq 32n - 56$ for $V \in \mathbb{R}^2$ and $n = |V|$.

  The 2-dimensional case can be reduced to the 1-dimensional case. Now let $p_1, p_2 : \mathbb{R}^2 \mapsto \mathbb{R}$ be the orthogonal projection onto the axes. Then we have

  $$\forall u \neq v \ \exists i \in [2] \ : \quad \log ||p_i(u), p_i(v)||_1 - \frac{1}{2} \ \leq \ \log ||u, v||_2 \ \leq \ \log ||p_i(u), p_i(v)||_1 \, .$$

  Now $g(p_i(V)) \leq 8n - 14$ implies that $g(V) \leq 2g(p_1(V)) + 2g(p_2(V)) \ \leq \ 32n - 56$. Of course this argument can be generalized to $\mathbb{R}^k$ for a constant k.

- The last inequality follows directly from the definition. It implies logarithmic diversity for random point sets since the probability to choose a node within a n^{-c-2}-neighborhood of another can be bound by n^{-c-1}. Hence, for all nodes the probability that $g(V) \geq (c + 2) \log n$ can be bounded by at most n^{-c}.

□

There are many reasons why in the real world the diversity can always be estimated by $O(\log n)$, e.g. the accuracy of determining locations; and the ratio between the physical size of a radio station and its transmitting range.

10.3.3 Congestion

To approximate congestion-optimal communication networks for radio networks we will use the t-spanner with bounded degree introduced in [AS94]. A t-spanner is a graph such that for each pair (u, v) there exists a path of at most length $t||u, v||_2$. Note that t-spanners are a common choice for the communication links in radio networks, e.g. see [GGH$^+$01].

The algorithm *AS-spanner*(V, η, w) shown in Figure 10.4 constructs such a spanner for a point set V in $\mathbb{R}^2$. The algorithm considers all ordered pairs (p, q) of sites in increasing order of their distances. The edge (p, q) is added to the graph iff there is no edge (r, s) in the current graph such that (p, q) and (r, s) have roughly the same direction and the points p and r are close to each other, or the points q and s are close to each other.

```
Algorithm AS-spanner($V, \eta, w$)
            (* $0 < \eta < \frac{\pi}{4}, 0 \leq w < \frac{1}{2}(\cos\eta - \sin\eta)$ *)
begin
    $L :=$ list of all pairs of $V$ sorted according to their distances;
    $E := \emptyset$
    for all ordered pairs $(p, q) \in L$ do
        add := true
        for each edge $(r, s) \in E$ do
            if angle$(q - p, s - r) \leq \eta$ then
                add := add $\wedge$ $(||p, r||_2 > w||r, s||_2)$
            fi
            if angle$(p - q, r - s) \leq \eta$ then
                add := add $\wedge$ $(||q, s||_2 > w||r, s||_2)$
            fi
        od
    if add = true then $E := E \cup \{(p, q)\}$ fi
    od
    return($E$)
end
```

Figure 10.4: The algorithm of Arya and Smid [AS94] for the construction of a t-spanner

Theorem 40 *[AS94] The graph constructed by algorithm* AS-spanner *is a t-spanner with bounded degree $\frac{2\pi}{\eta}$ for $t = 1/(\cos\eta - \sin\eta - 2w)$.*

Such a t-spanner causes only a small number of interferences if the node set V has a small diversity.

Lemma 54 *For a node set V with diversity $g(V)$ the interference number of the t-spanner constructed by algorithm* AS-spanner *is bounded by $O(g(V))$.*

Proof: We choose $w = \frac{1}{4}(\cos\eta - \sin\eta)$ for $\eta = \pi/8$. Hence the stretch-factor is given by $t = 2/(\cos\eta - \sin\eta)$.

We define $Q := \{q_1, q_2, \dots, q_g\} \subset \mathbb{Z}$ such that $Q := \{m \mid \exists u, v \in V \; : \; \lfloor \log ||u, v||_2 \rfloor = m\}$. We consider an edge $e = (u, v)$ of length $l \in [2^{q_i}, 2 \cdot 2^{q_i}]$. We try to insert as many edges interfering e as possible. Their number is bounded by $O(\frac{2\pi}{\eta w})$ since the constructed spanner has bounded degree 16 and for every edge to a neighbored node we can construct at most $O(\frac{1}{w}) = O(1)$ parallel edges interfering with e because of the restriction that the distance between the edges must be at least $w \cdot l$ (see figure 10.5). We have $g(V) = |Q|$ and therefore the number of interferences is bounded by $O(\frac{g(V)}{\eta \cdot w})$. Using that η and w are constant, this completes the proof. □

A typical feature of radio communication is that transmitting information blocks a region for other transmission. We formalize this observation and define the capacity of a region following a similar approach presented in [GK00]. Let $A(R)$ denote the area of a geometric region R.

Definition 15 *The capacity $\kappa(R)$ of a geometric region R is defined as follows:*

1. *If in every point of R the same set of edges E interfere, then $\kappa(R) := \sum_{e \in E} \ell(e) \cdot A(R)$ where $A(R)$ denotes the area of R. Such a region is called elementary.*
2. *Otherwise partition R into elementary regions $R_1, \dots, R_m$ and define $\kappa(R) := \sum_{i=1}^{m} \kappa(R_i)$.*

This definition implies the following relationship between capacity, area and congestion.

Lemma 55 *Let R be a region and C the congestions of a path system $\mathcal{P}$. Then, the capacity of R is bounded by $\kappa(R) \; \leq \; A(R) \cdot C$.*

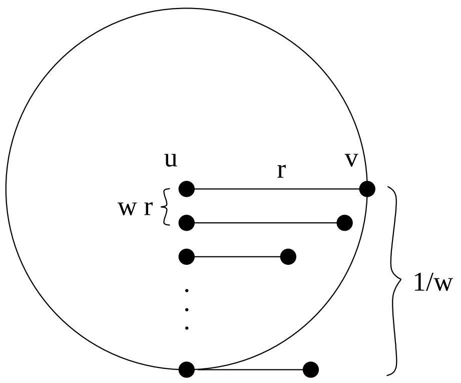

Figure 10.5: For an edge e to a neighbored node there exist at most $c \cdot \frac{1}{w}$ parallel edges interfering with e

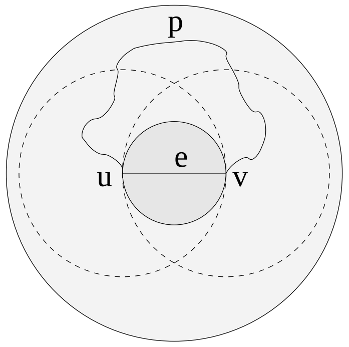

Figure 10.6: The edge e interferes with other edges (at least) within the central disk. Its information is rerouted on p, lying completely within the outer-disk with radius $\frac{1}{2}t||e||_2$

Every edge e with load $\ell(e)$ has a certain impact on the capacity of the area covered by the radio-network.

Lemma 56 *An edge e with load $\ell(e)$ occupies the capacity $c\ell(e)(||e||_2)^2$ for a constant $c > 0$.*

The proof follows from the definition of the interference area.

Lemma 57 *Let C^* be the congestion of the congestion-optimal path system $\mathcal{P}^*$ for a node set V. Then, every t-spanner N can host a path system $\mathcal{P}'$ such that the induced load $\ell(e)$ in N is bounded by $\ell(e) \leq c' g(V)\ C^*$ for a positive constant c'.*

Proof: Given a path P of the path system $\mathcal{P}^*$, we replace every edge $e = (u,v)$ that does not exist in the t-spanner N with the shortest path p from u to v in N (which by definition has length of at most $|p| \leq t||u,v||_2$). Therefore, the new route lies completely inside a disk $C_t(e)$ of radius $\frac{1}{2}t||u,v||_2$ and center $\frac{1}{2}(u+v)$.

For the path system $\mathcal{P}^*$ there may have been interferences between e and other edges. For simplicity we underestimate the area where e can interfere other communication by the disk $C_1(e)$ with center $\frac{1}{2}(u+v)$ and radius $\frac{1}{2}||u,v||_2$ (see Figure 10.6).

We want to describe the impact of rerouting all edges in $E(N^*)$ to a specific edge $e_0 \in E(N)$ in the t-spanner N. If this edge $e_0 = (u_0, v_0) \in E(N)$ transmits the traffic of a detour of an edge $e = (u,v) \in E(N^*)$, then the distance between the central points $z_0 := \frac{1}{2}(u_0 + v_0)$ of e_0 and $C = \frac{1}{2}(u+v)$ is bounded by $||z_0, z||_2 \leq \frac{1}{2}t||e||_2$.

Now consider the edge set $E_{i,e_0} \subseteq E(N^*)$ of edges e with length $||e||_2 \in [2^i, 2^{i+1}]$ for $i \in \mathbb{Z}$ which reroute their traffic to e_0. Their center points are located inside a disk with radius $2^i t$ and center z_0. The interference area of every edge e is described by $C_1(e)$. It occupies an area of at least $\pi 2^{2i}$, which lies completely inside a disk D with radius $2^i(t+1)$ and center z_0. The area of D is $\pi 2^{2i}(t+1)^2$.

Lemma 56 shows that every edge e reduces the capacity in D by at least $c\ell(e)2^{2i}$. Because of Lemma 55, the over-all capacity of C is at most $A(D) = \pi 2^{2i}(t+1)^2 C^*$. Therefore, we have for the sum of the loads $\ell(e)$ for $e \in E_{i,e_0}$ that $\sum_{e \in E_{i,e_0}} \ell(e) \leq \pi(t+1)^2 C^*/c$. By definition there are at most $g(V)$ non-empty sets E_{i,e_0}. This implies for the sum of loads $\ell(e)$ of the set $E_{e_0} \subseteq E(N^*)$: $\sum_{e \in E_{e_0}} \ell(e) \leq g(V)(t+1)^2 \pi C^*/c = c' C^* g(V)$, where $c' := (t+1)^2\pi/c$. $\square$

Combining the last two lemmas we can show that the t-spanner approximates the optimal network by a factor of $O(g(V)^2)$. Since in practice the diversity can be seen as a logarithmic term, such a t-spanner provides a $O((\log n)^2)$-approximation for the congestion.

Theorem 41 *The t-spanner of [AS94] contains a path system $\mathcal{P}$ with congestion $O(g(V)^2 C_{\mathcal{P}^*}(V))$, where $\mathcal{P}^*$ denotes the congestion optimal path system for the node set V.*

Proof: From Lemma 57 it follows that there exists a routing on a t-spanner such that the load of an edge e is bounded by $\ell(e) \leq c' g(V) C_{\mathcal{P}^*}(V)$. Lemma 54 shows that the interference number of the network is bounded by $O(g(V))$. This implies $C_{\mathcal{P}}(V) = O(g(V)^2 C_{\mathcal{P}^*}(V))$. □

10.4 Trade-Offs

As we have seen there are efficient ways for selecting paths to optimize energy and to approximate congestion. One might wonder whether an algorithm can compute a path system for a radio network such that energy, congestion and dilation can be optimized at the same time. It turns out that this is not the case.

10.4.1 Congestion versus Dilation

For a node set G_n placed on the crossings of a $\sqrt{n} \times \sqrt{n}$-grid the best choice to minimize congestion is to connect grid points only to their neighbors given the demand $w(u,v) = W/n^2$ for all nodes (Figure 10.7). Then the congestion is $O(W/\sqrt{n})$ and the dilation is given by $O(\sqrt{n})$. In [GK00] it is shown that such a congestion is best possible in a radio network. A fast realization is given by a tree featuring a hop-distance of $O(\log n)$ and congestion $O(W \log n)$ (Such a tree-construction for the Cost-distance problem is presented in [SW01]). In both cases we observe $C_{\mathcal{P}}(G_n) D_{\mathcal{P}}(G_n) \geq \Omega(W)$. This is also true for any other path selection:

Theorem 42 *Given the grid node set G_n, then for every path system $\mathcal{P}$ the following trade-off between delay $D_{\mathcal{P}}(G_n)$ and congestion $C_{\mathcal{P}}(G_n)$ exists: $C_{\mathcal{P}}(G_n) \cdot D_{\mathcal{P}}(G_n) \geq \Omega(W)$.*

Proof: For $n = 9p^2$ partition the grid into three $p \times 3p$ rectangle shaped node sets V_1, V_2, V_3, such that V_1 contains all left nodes, V_3 all right nodes and V_2 the nodes in the middle.

We consider only an $\frac{1}{9}$th of the demand starting at V_1 heading for nodes in V_3. Let $D \leq 3p$ be the delay of the network and $p_{i,j}$ denote the route from node v_i to node v_j. Let $r(p_{i,j}) = w(u_i, u_j)$ denote the information flow on path $p_{i,j}$.

Consider two nodes $v_i \in V_1$ and $v_j \in V_3$. Then the path $p_{i,j}$ has at most $D_{\mathcal{P}}(G_n)$ edges. The induced capacity $\kappa(p_{i,j})$ of the path $p_{i,j}$ is at least $\kappa(p_{i,j}) \geq c_1 \ell(p_{i,j}) \sum_{e \in p_{i,j}} (||e||_2)^2$. This term is minimized if the path uses the maximum possible number $D_{\mathcal{P}}(G_n)$ of edges with equal length of at least $\frac{d}{3 D_{\mathcal{P}}(G_n)}$. Then we have $\kappa(p_{i,j}) \geq \frac{c_1 d^2 W}{9n^2 D_{\mathcal{P}}(G_n)}$.

The sum of the capacity over all paths cannot extend the capacity of the $(3d) \times (3d)$-square containing all possible interference areas. This gives $\sum_{v_i \in V_1} \sum_{v_j \in V_3} \kappa(p_{i,j}) \leq 9d^2$. Combining the inequalities states the claim: $C_{\mathcal{P}}(G_n) D_{\mathcal{P}}(G_n) \geq \frac{c_1}{81} W$. □

10.4.2 Dilation versus Energy

The simplest location of sites is the line node set L_n as investigated in [KKKP00], see Figure 10.8. Here all nodes $L_n = \{v_1, \dots, v_n\}$ are placed on a line with equal distances $||v_i, v_{i+1}||_2 = \frac{d}{n}$. Only the first and the last node want to exchange messages, i.e., $w(v_1, v_n) = W$ and $w(v, w) = 0$ for all other pairs (v, w). The energy-optimal network for unit and flow energy is the path $(v_1, v_2, \dots, v_n)$, given the unit energy U-Energy$_{\mathcal{P}}(L_n) = \frac{d^2}{n}$, the flow energy F-Energy$_{\mathcal{P}}(L_n) = \frac{d^2 W}{n}$ and the delay n.

The fasted network realizes only the edge (v_1, v_n) with hop-distance 1 and unit energy d^2 (and flow energy $W d^2$). There are path systems that can give a compromise between these extremes. However, it turns out that the product of delay and energy cannot be decreased:

Theorem 43 *Given the node set L_n with diameter d, then for every path system $\mathcal{P}$ the following trade-offs between delay D and unit energy* U-Energy *(resp. flow energy* F-Energy*) exist:*

$$\begin{aligned} D_{\mathcal{P}}(L_n) \cdot \textit{U-Energy}_{\mathcal{P}}(L_n) &\geq \Omega(d^2) , \\ D_{\mathcal{P}}(L_n) \cdot \textit{F-Energy}_{\mathcal{P}}(L_n) &\geq \Omega(d^2 W) . \end{aligned}$$

Proof: Let m be the number of edges of the longest path of the radio network (wlog. we assume that there are only edges with non-zero information flow $\ell(e) > 0$). For the unit energy model we can assume that there is only a path p from v_1 to v_n (because introducing more edges needs additional energy without decreasing the delay). We have to minimize U-Energy$_{\mathcal{P}}(p) := \sum_{i=1}^{m} (\ell_i)^2$ defined by the edge lengths $\ell_1, \dots, \ell_m$, where $\sum_{i=1}^{m} \ell_i = d$. Clearly, the sum is minimal for $\ell_1 = \ell_2 = \cdots = \ell_m = \frac{d}{m}$ giving U-Energy$_{\mathcal{P}}(p) \cdot D_{\mathcal{P}}(p) \geq d^2$.

The bound for the flow energy follows analogously. □

10.4.3 The Incompatibility of Congestion and Energy

We will show that for some node sets congestion and energy are incompatible. This is worse than a trade-off-situation since there is no compromise possible for energy and congestion.

The node set $U_{\alpha,n}$ for $\alpha \in [0, \frac{1}{2}]$ consists of two vertical parallel line graphs L_{n^α}. Neighbored (and opposing) nodes have distance $\frac{d}{n^\alpha}$. There is only demand W/n^α between the horizontal pairs of opposing nodes of the line graphs. The rest of the $n - n^{-\alpha}$ nodes are equidistantly placed between the nodes of each line graph and the lowest horizontal pair of nodes, see Figure 10.9.

The minimum spanning tree consists of n nodes where all edges have length $\Theta(dn^{-1})$. This results in a total unit energy of U-Energy$_{\text{MST}}(U_{\alpha,n}) = O(d^2 n^{-1})$ and congestion $C_{\text{MST}}(U_{\alpha,n}) = O(W)$. The flow energy of the (same) minimal network is given by F-Energy$_{\text{MST}}(U_{\alpha,n}) = O(Wd^2 n^{-1})$.

The congestion optimal path system $\mathcal{P}'$ connects only nodes with non-zero demand. Its congestion is $C_{\mathcal{P}'}(U_{\alpha,n}) = O(Wn^{-\alpha})$ and its unit energy is U-Energy$_{\mathcal{P}'}(U_{\alpha,n}) = O(d^2 n^{-\alpha})$. The flow energy is given by F-Energy$_{\mathcal{P}'}(U_{\alpha,n}) = O(Wn^{-2\alpha} d^2)$.

Lemma 58 *For $\alpha \in [0, \frac{1}{2})$ and the node set $U_{\alpha,n}$ with diameter d let $x \in \{0, \dots, n^\alpha\}$ be the number of edges of length $dn^{-\alpha}$ of a path system for the radio network and let $f \in [0, W]$ be the information flow on these edges. Then, we have for the congestion C, unit energy and flow energy:*

$$\textit{U-Energy}_{\mathcal{P}}(U_{\alpha,n}) \geq \frac{d^2}{n} + x\left(\frac{d^2}{n^{2\alpha}} - \frac{d^2}{n^{1+\alpha}}\right) , \tag{10.2}$$

$$C_{\mathcal{P}}(U_{\alpha,n}) \geq \frac{W}{x+1} , \tag{10.3}$$

$$\textit{F-Energy}_{\mathcal{P}}(U_{\alpha,n}) \geq f\frac{d^2}{n^{2\alpha}} + (W-f)\left(\frac{d^2}{n} - \left\lfloor \frac{f}{n^\alpha} \right\rfloor \frac{d^2}{n}\right) , \tag{10.4}$$

$$C_{\mathcal{P}}(U_{\alpha,n}) \geq \frac{f}{n^\alpha} + (W-f) . \tag{10.5}$$

Proof: The minimum unit energy network is given by the MST which is a U-shaped path. Note that no shortcut within the left and right vertical bars of this path can reduce energy or congestion. Therefore, the only reasonable choice for an edge is to connect some (x) of the horizontal nodes (and possibly to disconnect a route to a vertical neighbors). Adding the horizontal channel implies additional energy consumption of $\frac{d^2}{n^{2\alpha}}$. For $x + 1$ horizontal routes (including the original low energy route) the best choice is to fairly distribute the traffic.

For the flow energy the argument is analogous. □

Theorem 44 *There exists a node set V with a path system minimizing congestion to C^*, and another path system optimizing unit energy by U-Energy* and minimal flow energy by F-Energy* such for any path system $\mathcal{P}$ on this node set V we have*

$$C_{\mathcal{P}}(V) \geq \Omega(n^{1/3} C^*) \quad \text{or} \quad \textit{U-Energy}_{\mathcal{P}}(V) \geq \Omega(n^{1/3} \textit{U-Energy}^*) ,$$
$$C_{\mathcal{P}}(V) \geq \Omega(n^{1/3} C^*) \quad \text{or} \quad \textit{F-Energy}_{\mathcal{P}}(V) \geq \Omega(n^{1/3} \textit{F-Energy}^*) .$$

Proof: follows by Lemma 58 using the graph $V = U_{1/3,n}$. □

Hence, there is no hope that communication networks can optimize more than one parameter at a time. The network designer has to decide whether to go for small congestion or low energy consumption.

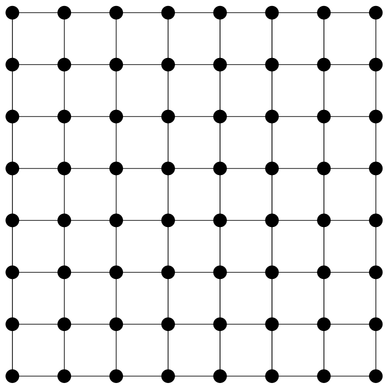
Figure 10.7: The grid G_n

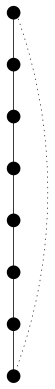
Figure 10.8: The line L_n

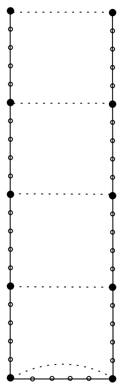
Figure 10.9: Node set U_α

10.5 Conclusions and Further Work

The main difference between wired networks and radio networks is that the choice of communication links in wireless networks influences the quality of the edges. We model the type of influences by the interference graph, which gives a very general description how links can interfere. If the sending and receiving characteristics of the radio stations are known, this interference graph can be described by the geometric properties like the location of sites and transmitter power.

However, the main difference is still that choosing a certain communication link for some time decreases the ability of transmitting information in some other parts of the radio network. Since the analysis of point-to-point communication (or permutation networks) in wireless networks is relatively young (see [AS98a]), we start our investigation with a static simplified model: The point-to-point communication and the location of the sites is fixed (unlike in mobile ad hoc networks). You can also see this model as a snapshot of a more dynamic model (where research has just begun [ABBS01]).

We investigate the question of what is the optimal choice of communication links to achieve the best possible network. We measure the quality by congestion, energy and delay. Given a path system for the packets we present a sound definition of congestion, which takes into account the actual information flow, i.e., load, over a link and the interferences of other links.

There already exists a probabilistic solution for solving interferences if the network parameters are known [AS98a]. We show how this algorithm can be applied to our setting. Further, we relate the routing to our notion of congestion and dilation, which is the maximum length of a path.

We prove that for our notion of energy (depending on the packet flow) the optimal path system can be computed in polynomial time. Furthermore, we prove that a t-spanner construction for the communication networks allows path systems with small congestion. Concretely, we show an approximation of a factor of $O(g(V)^2)$ of the minimal congestion, where $g(V)$ denotes the diversity of the node set. We introduce this measure to characterize malformed node locations. For practical applications we have $g(V) = \Theta(\log)$, e.g. if the node set is random, or if the ratio of maximum and minimum distance of nodes is at most polynomial. An overview of these results is shown in Table 10.1.

However there are situations where it is not possible to optimize two of these measures at the same time (see Table 10.2). We prove trade-off results for congestion versus dilation and energy versus dilation. For congestion and energy we show that every path system trying to approximate the congestion within a smaller factor than $o(n^{1/3})$ of the optimal congestion, suffers under an increased energy consumption of at least a factor of $\Omega(n^{1/3})$, and vice versa. Hence, energy and congestion minimization in radio networks are incompatible tasks.

Besides the standard model of omni-directional communication we are currently investigating a sector model where sender and receiver can focus signals (e.g. infrared). Such sector communication is a special case of so-called space multiplexing techniques to increase the network capacity (e.g. by using directional antennas [KSV00]). All results shown in this chapter can be easily transferred to such a model.

Another possibility to decrease interferences is to use multiple frequencies (as done in Bluetooth

	Congestion	Dilation	Unit Energy	Flow Energy
Structure	AS-spanner	Complete Network	MST	Gabriel Sub-Graph
Approx.-factor	$O(\log^2 n)$	optimal	optimal	optimal

Table 10.1: Approximation results for logarithmic diversity

	Delay	Congestion
Congestion	$C_{\mathcal{P}}(V) \cdot D_{\mathcal{P}}(V) \geq \Omega(W)$	—
Unit Energy	$D_{\mathcal{P}}(V) \cdot \mathrm{UE}_{\mathcal{P}}(V) \geq \Omega(d^2)$	$C_{\mathcal{P}}(V) \geq \Omega(n^{1/3} C^*_{\mathcal{P}}(V))$ or $\mathrm{UE}_{\mathcal{P}}(V) \geq \Omega(n^{1/3} \mathrm{UE}^*_{\mathcal{P}}(V))$
Flow Energy	$D_{\mathcal{P}}(V) \cdot \mathrm{FE}_{\mathcal{P}}(V) \geq \Omega(d^2 W)$	$C_{\mathcal{P}}(V) \geq \Omega(n^{1/3} C^*_{\mathcal{P}}(V))$ or $\mathrm{FE}_{\mathcal{P}}(V) \geq \Omega(n^{1/3} \mathrm{FE}^*_{\mathcal{P}}(V))$

Table 10.2: Trade-Offs and Incompatibilities on network parameters

[Miy00] or IEEE 802.11 [IEE97]). As long as number f of frequencies is small (which is the case in practice, because of governments' regulation of the entire frequency spectrae) this may improve the congestion by f. However, using frequency hopping cannot completely resolve the shown the trade-off and incompatibility problems shown here.

This work is part of a project where a prototype communication system is being developed based on infrared directed communication. The prototype will be able to communicate in eight sectors independently with adjustable transmission powers. Furthermore, it can be used as an extension module for the mobile mini robot Khepera ([MFG99, KTe00]). Thus, realistic scenarios for ad hoc networks can be reproduced by performing experiments with these mini robots. Thus, beside computer simulations, our communication strategies will also be validated under practical conditions. Such a network is technically more complicated, but our goal is to show that it is possible to set up a geometric spanner graph as a communication network. Notably, this chapter shows that such geometric spanners always provide good solutions for congestion minimization in radio networks.

Bibliography

[AAS97] Divyakant Agrawal, Amr El Abbadi, and Robert C. Steinke. Epidemic algorithms in replicated databases (extended abstract). In ACM, editor, *PODS '97. Proceedings of the Sixteenth ACM SIG-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 161–172, 1997.

[ABBS01] Baruch Awerbuch, Petra Berenbrink, Andreas Brinkmann, and Christian Scheideler. Simple Routing Strategies for Adversarial Systems. In *IEEE Symposium on Foundations of Computer Science (FOCS'01)*, pages 158–167, 2001.

[ACBFS95] Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *36th Annual Symposium on Foundations of Computer Science (FOCS'95)*, pages 322–331, 1995.

[ADM$^+$95] S. Arya, G. Das, D. Mount, J. Salowe, and M. Smid. Euclidean Spanners – Short, Thin and Lanky. In *Procedings of the 27th Annual ACM Symposium on the Theory of Computing (STOC'95)*, pages 489–498, 1995.

[AHS94] James Aspnes, Maurice Herlihy, and Nir Shavit. Counting networks. *Journal of the ACM*, 41:1020–1048, 1994.

[ALS91] Stefan Arnborg, Jens Lagergren, and Detlef Seese. Easy problems for tree-decomposable graphs. *Journal of Algorithms*, 12(2):308–340, June 1991.

[AMO96] Yehuda Afek, Yishay Mansour, and Zvi Ostfeld. Convergence complexity of optimistic rate based flow control algorithms (extended abstract). In *Proceedings of the 28th ACM Symposium on the Theory of Computing (STOC'96)*, pages 89–98, 1996.

[AMO00] Yehuda Afek, Yishay Mansour, and Zvi Ostfeld. Phantom: a simple and effective flow control scheme. *Computer Networks (Amsterdam, Netherlands: 1999)*, 32(3):277–305, March 2000.

[Aro98] Sanjeev Arora. Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems. *Journal of the ACM*, 45(5):753–782, 1998.

[AS94] S. Arya and M. Smid. Efficient Construction of a Bounded Degree Spanner with Low Weight. In *Proceedings 2nd European Symposium on Algorithms (ESA'94)*, pages 48–59, 1994.

[AS98a] M. Adler and Ch. Scheideler. Efficient Communication Strategies for Ad-Hoc Wireless Networks (extended Abstract). In *ACM Symposium on Parallel Algorithms and Architectures (SPAA'98)*, pages 259–268, 1998.

[AS98b] Baruch Awerbuch and Yuval Shavitt. Converging to approximated max-min fairness in logarithmic time. In *IEEE INFOCOM*, pages 1350–1357, April 1998.

[Aue00] Peter Auer. Using upper confidence bounds for online learning. In *41st Annual Symposium on Foundations of Computer Science (FOCS'00)*, pages 270–279, 2000.

[Bai75] Norman T. J. Bailey. *The Mathematical Theory of Infectious Diseases and its Applications*. Griffin, 2nd edition, 1975.

[Bar98] Yair Bartal. On approximating arbitrary metrics by tree metrics. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC-98)*, pages 161–168. ACM Press, May 23–26 1998.

[BEY98] Allan Borodin and Ran El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, Cambridge, 1998.

[BHLP92] Jean-Claude Bermond, Pavol Hell, Arthur L. Liestman, and Joseph G. Peters. Broadcasting in bounded degree graphs. *SIAM Journal on Discrete Mathematics*, 5(1):10–24, February 1992.

[BK91] Hans L. Bodlaender and Ton Kloks. Better algorithms for the pathwidth and treewidth of graphs. In *Automata, Languages and Programming, 18th International Colloquium (ICALP'91)*, pages 544–555. Springer-Verlag, 1991.

[BNGNS98] Amotz Bar-Noy, Sudipto Guha, Joseph (Seffi) Naor, and Baruch Schieber. Multicasting in heterogeneous networks. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC'98)*, pages 448–453, 1998.

[BNK94] Amotz Bar-Noy and Shlomo Kipnis. Designing broadcasting algorithms in the postal model for message-passing systems. *Mathematical Systems Theory*, 27(5):431–452, September/October 1994.

[Bod93] Hans L. Bodlaender. A tourist guide through treewidth. *Acta Cybernetica*, 11:1–21, 1993.

[Bun84] Gerhard Buntrock. A nice special case of 3dm is $\mathcal{NP}$-complete. Technical report, Technische Universit¨at Berlin, 1984.

[CBFH$^+$97] Nicolò Cesa-Bianchi, Yoav Freund, David Haussler, David P. Helmbold, Robert E. Schapire, and Manfred K. Warmuth. How to use expert advice. *Journal of the ACM*, 44(3):427–485, May 1997.

[CBFHW94] Nicolò Cesa-Bianchi, Yoav Freund, David P. Helmbold, and M. Warmuth. On-line prediction and conversion strategies. In *European Conference on Computational Learning Theory, (EuroCOLT'93)*, pages 205–216, 1994.

[CBL99] Nicolò Cesa-Bianchi and Gábor Lugosi. Minimax regret under log loss for general classes of experts. In *Proceedings of the 12th Annual Conference on Computational Learning Theory (COLT'99)*, pages 12–18, 1999.

[CCG$^+$98] Moses Charikar, Chandra Chekuri, Ashish Goel, Sudipto Guha, and Serge A. Plotkin. Approximating a finite metric by a small number of tree metrics. In *Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science (FOCS'98)*, pages 379–388, 1998.

[CJ89] Dah-Ming Chiu and Raj Jain. Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. *Computer Networks and ISDN Systems*, 17:1–14, 1989.

[CNP01] Ioannis Chatzigiannakis, Sotiris E. Nikoletseas, and P.Spirakis. An Efficient Communication Strategy for Ad-hoc Mobile Networks. In *In Proc. of the 15th Symposium on Distributed Computing (DISC'2001)*, 2001.

[CPS00] Andrea E. F. Clementi, Paolo Penna, and Riccardo Silvestri. On the power assignment problem in radio networks. *Electronic Colloquium on Computational Complexity (ECCC'00)*, 2000.

[CRL96] Anna Charny, K. K. Ramakrishnan, and Anthony Lauck. Time scale analysis scalability issues for explicit rate allocation in ATM networks. *IEEE/ACM Transactions on Networking*, 4(4):569–581, August 1996.

[CT00] Jae-Hwan Chang and Leandros Tassiulas. Energy conserving routing in wireless ad hoc networks. In *IEEE INFOCOM*, pages 22–31, March 2000.

[DF86] Martin E. Dyer and Alan M. Frieze. Planar 3DM is $\mathcal{NP}$-complete. *Journal of Algorithms*, 7(2):174–184, June 1986.

[DGH$^+$87] Alan Demers, Dan Greene, Carl Hauser, Wes Irish, John Larson, Scott Shenker, Howard Sturgis, Dan Swinehart, and Doug Terry. Epidemic algorithms for replicated database maintenance. In Fred B. Schneider, editor, *Proceedings of the 6th Annual ACM Symposium on Principles of Distributed Computing*, pages 1–12. ACM Press, 1987.

[DHN93] Gautam Das, Paul J. Heffernan, and Giri Narasimhan. Optimally sparse spanners in 3-dimensional euclidean space. In *Symposium on Computational Geometry*, pages 53–62, 1993.

[DR98] Devdatt P. Dubhashi and Desh Ranjan. Balls and bins: A study in negative dependence. *RSA: Random Structures & Algorithms*, 13, 1998.

[DRWT97] Rohit Dube, Cynthia D. Rais, Kuang-Yeh Wang, and Satish K. Tripathi. Signal Stability-Based Adaptive Routing (SSA) for Ad Hoc Mobile Networks. *IEEE Personal Communications*, pages 36–45, February 1997.

[EK01] Michael Elkin and Guy Kortsarz. Combinatorial logarithmic approximation for the directed telephone broadcast problem. Technical report, Unpublished Manuscript, 2001.

[Epp00] David Eppstein. Spanning trees and spanners. In J ̈org-Rudiger Sack and Jorge Urrutia, editors, *Handbook of Computational Geometry*, chapter 9, pages 425–461. Elsevier Science Publishing, 2000.

[Fei98] Uriel Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM*, 45(4):634–652, July 1998.

[Fil91] James Allen Fill. Eigenvalue bounds on convergence to stationarity for nonreversible markov chain with an application to the exclusion process. *Annals of Applied Probability*, 1:62–87, 1991.

[FJ92] Sally Floyd and Van Jacobson. Traffic phase effects in packet-switched gateways. *Journal of Internetworking: Practice and Experience*, 3(3):115–156, September 1992.

[FS99] Yoav Freund and Robert E. Schapire. Adaptive game playing using multiplicative weights. *Games and Economic Behavior*, pages 97–103, 1999.

[GGH$^+$01] Jie Gao, Leonidas J. Guibas, John Hershberger, Li Zhang, and An Zhu. Geometric spanner for routing in mobile networks. In *Proceedings of the 2nd ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '01)*, pages 45–55, 2001.

[GGJ76] Michael R. Garey, Ronald L. Graham, and David S. Johnson. Some NP-complete geometric problems. In *ACM Symposium on Theory of Computing*, pages 10–22, 1976.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the theory of of NP-Completeness*. Freeman and Company, San Francisco, 1979.

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

[GL91] Richard Golding and Darrell D. E. Long. Accessing replicated data in a large-scale distributed system. ucsc-crl-91-01, University of California, Santa Cruz, January 1991.

[GP96] Leszek Gasieniec and Andrzej Pelc. Adaptive broadcasting with faulty nodes. *Parallel Comput.*, 22:903–912, 1996.

[GPP93] Richard G. Guy, Gerald J. Popek, and Thomas W. Page Jr. Consistency algorithms for optimisic replication. In *Proceedings of the First International Conference on Network Protocols*, October 1993.

[GS69] K. R. Gabriel and R. R. Sokal. A new statistical approach to geographic variation analysis. *Systematic Zoology*, 18:259–278, 1969.

[Han57] James. Hannan. Approximation to bayes risk in repeated plays, 1957.

[Hås97] Johan Håstad. Some optimal inapproximability results. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing (STOC'97)*, pages 1–10, 1997.

[HHL88] Stephan Hedetniemi, T. Hedetniemi, and Arthur L. Liestman. A Survey of Gossiping and Broadcasting in Communication Networks. *NETWORKS*, 18:319–349, 1988.

[HKMP96] Juraj Hromkovic, Ralf Klasing, Burkhard Monien, and Regine Peine. Dissemination of information in interconnection networks. *Combinatorial Network Theory*, 53:125–212, 1996.

[HKW95] David Haussler, Jyrki Kivinen, and Manfred K. Warmuth. Tight worst-case loss bounds for predicting with expert advice. In *Proceedings of the 2nd European Conference on Computational Learning Theory (EuroCOLT'95)*, volume 904 of *LNAI*, pages 69–83, March 1995.

[HKW98] David Haussler, Jyrki Kivinen, and Manfred K. Warmuth. Sequential prediction of individual sequences under general loss functions. *IEEE Transactions on Information Theory*, 44:1906–1925, September 1998.

[HLL92] David P. Helmbold, Nicholas Littlestone, and Philip M. Long. Apple tasting and nearly one-sided learning. In *33rd Annual Symposium on Foundations of Computer Science (FOCS'92)*, pages 493–502, 1992.

[HLL00] David P. Helmbold, Nicholas Littlestone, and Philip M. Long. Apple tasting. *Inform. Comput.*, 161(2):85–139, September 2000.

[Hoc97] Dorit Hochbaum, editor. *Approximation Algorithms for NP-hard Problems*. PWS Publishing Company, 1997.

[Hoh90] Walter Hohberg. The decomposition of graphs into k-connected components for arbitrary k. Technical report, Technische Hochschule Darmstadt, 1990.

[HR89] Walter Hohberg and R¨udiger Reischuk. Decomposition of graphs – a uniform approach for the design of fast sequential and parallel algorithms on graphs. Technical report, Technische Hochschule Darmstadt, 1989.

[IEE97] IEEE. IEEE Standard 802.11 - Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Spezifikation. In *IEEE*, 1997.

[Jac88] Van Jacobson. Congestion Avoidance and Control. In *ACM SIGCOMM '88*, volume 18, 4, pages 314–329. ACM SIGCOMM, ACM Press, August 1988. Stanford, CA.

[Jaf81] Jeffrey M. Jaffe. Bottleneck flow control. *IEEE trans. on commun.*, COM-29:954–962, 1981.

[JRS94] Andreas Jakoby, R¨udiger Reischuk, and Christian Schindelhauer. The complexity of broadcasting in planar and decomposable graphs. In *WG: Graph-Theoretic Concepts in Computer Science, International Workshop WG*, volume 903, pages 219–231, 1994.

[JRS98] Andreas Jakoby, R¨udiger Reischuk, and Christian Schindelhauer. The complexity of broadcasting in planar and decomposable graphs. *Discrete Applied Mathematics*, 83:197–206, 1998.

[JT92] Jerzy W. Jaromczyk and Godfried T. Toussaint. Relative neighborhood graphs and their relatives. In *Proc. IEEE, 80(9):1502-1517*, 1992.

[KKKP00] Lefteris M. Kirousis, Evangelos Kranakis, Danny Krizanc, and Andrzej Pelc. Power consumption in packet radio networks. *Theoretical Computer Science*, 243(1-2):289–305, 2000.

[KKPS00] Richard Karp, Elias Koutsoupias, Christos H. Papadimitriou, and Scott Shenker. Optimization problems in congestion control. In *41st Annual Symposium on Foundations of Computer Science (FOCS'00)*, pages 66–74, 2000.

[KL93] Michael Kearns and Ming Li. Learning in the presence of malicious errors. *SIAM Journal on Computing*, 22(4):807–837, August 1993.

[KP95] Guy Kortsarz and David Peleg. Approximation algorithms for minimum-time broadcast. *SIAM Journal on Discrete Mathematics*, 8(3):401–427, August 1995.

[KR85] D. G. Kirkpatrick and J. D. Radke. A framework for computational morphology. *Computational Geometry*, pages 217–248, 1985.

[KRT99] Jon Kleinberg, Yuval Rabani, and Eva Tardos. Fairness in routing and load balancing. In IEEE, editor, *40th Annual Symposium on Foundations of Computer Science: October 17–19, 1999, New York City, New York,*, pages 568–578, 1109 Spring Street, Suite 300, Silver Spring, MD 20910, USA, 1999. IEEE Computer Society Press.

[KRY95] Samir Khuller, Balaji Raghavachari, and Neal Young. Balancing minimum spanning trees and shortest path trees. In *Algorithmica, 14*, pages 305–321, 1995.

[KSSV00] Richard Karp, Christian Schindelhauer, Scott Shenker, and Berthold V¨ocking. Randomized rumor spreading. In *41st Annual Symposium on Foundations of Computer Science (FOCS'00)*, pages 565–574, 2000.

[KSV00] Young-Bae Ko, Vinaychandra Shankarkumar, and Nitin H. Vaidya. Medium Access Control Protocols Using Directional Antennas in Ad Hoc Networks. In *IEEE INFOCOM*, pages 13–21, March 2000.

[KTe00] K-Team SA. *Khepera miniuature mobile robot*, 2000. http://www.k-team.com/robots /khepera.

[Lag90] Jens Lagergren. Efficient parallel algorithms for tree-decomposition and related problems. In IEEE, editor, *Proceedings: 31st Annual Symposium on Foundations of Computer Science (FOCS'90)*, pages 173–182, 1990.

[LL89] Christos Levcopoulos and Andrzej Lingas. There are planar graphs almost as good as the complete graphs and as short as minimum spanning trees. In *Proc. Internat. Symp. on Optimal Algorithms, volume 401 of LNCS*, pages 9–13. Springer-Verlag, 1989.

[LLSG92] Rivka Ladin, Barbara Liskov, Liuba Shrira, and Sanjay Ghemawat. Providing high availability using lazy replication. *ACM Transactions on Computer Systems*, 10(4):360–391, November 1992.

[LM97] T. V. Lakshman and Upamanyu Madhow. The performance of TCP/IP for networks with high bandwidth-delay products and random loss. *IEEE/ACM Transactions on Networking*, 5(3):336–350, June 1997.

[LMS92] F. Thomson Leighton, B. Maggs, and R. Sitaraman. On the fault tolerance of some popular bounded-degree networks. In IEEE, editor, *33rd Annual Symposium on Foundations of Computer Science (FOCS'92)*, pages 542–552, 1992.

[LP88] Arthur L. Liestman and Joseph G. Peters. Broadcast networks of bounded degree. *SIAM Journal on Discrete Mathematics*, 1(4):531–540, November 1988.

[LW94] Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108(2):212–261, February 1994.

[MBmH01] Jeffrey P. Monks, Vaduvur Bharghavan, and Wen mei Hwu. A Power Controlled Multiple Access Protocol for Wireless Packet Networks. In *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, volume 1, pages 1–11, April 2001.

[MFG99] Francesco Mondada, Edoardo Franzi, and André Guignard. The development of khepera. In *Proceedings of the 1st International Khepera Workshop*, pages 7–13, Paderborn, Germany, December 10.-11. 1999.

[Mid93] Martin Middendorf. Minimum broadcast time is $\mathcal{NP}$-complete for 3-regular planar graphs and deadline 2. *Information Processing Letters*, 46(6):281–287, July 1993.

[Mih89] Milena Mihail. Conductance and convergence of markov chains: a combinatorial treatment of expanders. In *IEEE Symposium on Foundations of Computer Science (FOCS'89)*, pages 526–531, 1989.

[Miy00] Kazuhiro Miyatsu. Bluetooth Design Background and Its Technologial Features. In *IEICE Trans. Fundamentals, vol. E83-A, no. 11*, 2000.

[MJ90] Burkhard Monien and Juraj Hromkovič and and Claus-Dieter Jeschke. Optimum algorithms for dissemination of information in some interconnection networks. In *Mathematical Foundations of Computer Science 1990*, pages 337–346, Berlin, August 1990. Springer.

[MMP00] Adam Meyerson, Kamesh Munagala, and Serge A. Plotkin. COST-DISTANCE: Two metric network design. In *Proc. 41st Symp. Foundations of Computer Science*. IEEE, 2000.

[MMR99] D. Malkhi, Y. Mansour, and M. Reiter. On diffusing updates in a byzantine environment. In *Proceedings of the 18th IEEE Symposium on Reliable Distributed Systems (SRDS '99)*, pages 134–143, Washington - Brussels - Tokyo, October 1999. IEEE.

[MRS$^+$98] Madhav V. Marathe, Ramamoorthy Ravi, Ravi Sundaram, S. S. Ravi, Daniel J. Rosenkrantz, and Harry B. Hunt III. Bicriteria network design problems. *Journal of Algorithms*, 28(1):142–171, July 1998.

[MSM97] Matt Mathis, Jeffrey Semke, and Jamshid Mahdavi. The macroscopic behavior of the TCP congestion avoidance algorithm. *Computer Communications Review*, 27(3), July 1997.

[MSVG01] Friedhelm Meyer auf der Heide, Christian Schindelhauer, Klaus Volbert, and Matthias Gr¨unewald. Congestion, energy and delay in radio networks. Technical Report tr-ri-01-229, Universit¨at Paderborn, Heinz Nixdorf Institut, 2001.

[Nag84] John Nagle. Congestion control in IP/TCP internetworks. *Computer Communication Review, Vol. 14, No. 4, pp. 11-17*, October 1984.

[Nev75] Jacque Neveu. *Discrete-Parameter Martingale*. North Holland, 1975.

[Pap94] Christos H. Papadimitriou. *Computational Complexity*. Addison-Wesley, New York, 1994.

[Per01] C. E. Perkins, editor. *Ad Hoc Networking*. Addision-Wesley, 2001.

[Pit87] Boris Pittel. On spreading a rumor. *SIAM Journal on Applied Mathematics*, 47(1):213–223, February 1987.

[PS01] Antonio Piccolboni and Christian Schindelhauer. Discrete prediction games with arbitrary feedback and loss. In *14th Annual Conference on Computational Learning Theory, (COLT'01) and 5th European Conference on Computational Learning Theory, (EuroCOLT'01)*, pages 208–223, 2001.

[Rav94] Ramamoorthi Ravi. Rapid rumor ramification: approximating the minimum broadcast time. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science (FOCS'94)*, pages 202–213, 1994.

[Rei90] R¨udiger Reischuk. *Einfihrung in die Komplexitätstheorie*. B.G. Teubner, Stuttgart, 1990.

[Rei91a] R¨udiger Reischuk. An algebraic divide–and–conquer approach to design highly parallel solution strategies for optimization problems on graphs. Technical report, Technische Hochschule Darmstadt, 1991.

[Rei91b] R¨udiger Reischuk. Graph theoretical methods for the design of parallel algorithms. Technical report, Technische Hochschule Darmstadt, 1991.

[RGK96] Michael Rabinovich, Narain H. Gehani, and Alex Kononov. Scalable update propagation in epidemic replicated databases. *Lecture Notes in Computer Science*, pages 207–222, 1996.

[Rob96] Lawrence G. Roberts. Performance of explicit rate flow control in atm. In *CompCom'96*, pages 22–25, 1996.

[RS83] Neil Robertson and Paul D. Seymour. Graph minors i. excluding a forest. *Comb. Theory Series B*, 35:39–61, 1983.

[RS86] Neil Robertson and Paul D. Seymour. Graph minors ii. algorithmic aspects of tree-width. *Journal of Algorithms*, 7:309–322, 1986.

[RSW98] Yuval Rabani, Alistair Sinclair, and Rolf Wanka. Local divergence of markov chains and the analysis of iterative load balancing schemes. In *IEEE Symposium on Foundations of Computer Science*, pages 694–705, 1998.

[SCH81] Peter J. Slater, Ernest J. Cockayne, and Stephan T. Hedetniemi. Information dissemination in trees. *SIAM Journal on Computing*, 10(4):692–701, 1981.

[Sch99] Robert E. Schapire. Drifting games. In *Proceedings 12th Annual Conference on Computational Learning Theory (COLT'99)*, pages 114–124, 1999.

[Sch00a] Christian Schindelhauer. Broadcasting time cannot be approximated within a factor of $57/56-\epsilon$. Technical Report TR-00-002, International Computer Science Institute, Berkeley, CA, March 2000.

[Sch00b] Christian Schindelhauer. On the inapproximability of broadcasting time. In *3rd International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'00)*, pages 226–237, 2000.

[Sch01] Robert E. Schapire. Drifting games. *Machine Learning*, 43(3):265–291, 2001.

[Sin92] Alistair Sinclair. *Algorithms for Random Generation and Counting, a Markov Chain Approach*. Birkha¨user, 1992.

[SR98] Suresh Singh and Cauligi S. Raghavendra. PAMAS - power aware multi-access protocol with signalling for ad hoc networks. *ACM Computer Communications Review*, 28:5–26, 1998.

[SW98] Suresh Singh and Mike Woo. Power-Aware Routing in Mobile Ad Hoc Networks. In *Proceedings of the fourth annual ACM/IEEE international conference on Mobile computing and networking*, pages 181–190, 1998.

[SW01] Christian Schindelhauer and Birgitta Weber. Tree-approximations for the weighted cost-distance problem. In *International Symposium on Algorithms and Computation (ISAAC'01)*, pages 185–195, 2001.

[UY98] Sennur Ulukus and Roy D. Yates. Stochastic power control for cellular radio systems. In *IEEE Trans. Comm., 46(6):784–798*, 1998.

[Vel92] Remco C. Veltkamp. The gamma-neighborhood graph. *Computational Geometry, Theory and Applications*, 1:227–246, 1992.

[Vov98] Vladimir Vovk. A game of prediction with expert advice. *Journal of Computer and System Sciences*, 56(2):153–173, April 1998.

[Vov99] Vladimir Vovk. Competitive on-line statistics. *In The 52nd Session of the International Statistical Institute, 1999. To be published in Bulletin of the International Statistical Institute.*, 1999.

[Web01] Birgitta Weber. *Netzwerke optimiert für lokale Kosten-Distanzen*. Universit¨at L¨ubeck, Diploma Thesis, 2001.

[WM01] Tsachy Weissman and Neri Merhav. Universal prediction of individual binary sequences in the presence of noise. *To appear in IEEE Trans. Inform. Theory*, 2001.