The Complexity of Broadcasting in Planar and Decomposable Graphs

Andreas Jakoby*
Rüdiger Reischuk
Christian Schindelhauer
Medizinische Universität zu Lübeck†

January 1995‡

Abstract

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. The goal is to inform everybody using as few rounds as possible, that is minimize the broadcasting time.

Given a graph and a subset of nodes, the sources, the problem to determine its specific broadcast time, or more general to find a broadcast schedule of minimal length has shown to be $\mathcal{NP}$-complete. In contrast to other optimization problems for graphs, like vertex cover or traveling salesman, little was known about restricted graph classes for which polynomial time algorithms exist, for example for graphs of bounded treewidth. The broadcasting problem is harder in this respect because it does not have the finite state property. Here, we will investigate this problem in detail and prove that it remains hard even if one restricts to planar graphs of bounded degree or constant broadcasting time. A simple consequence is that the minimal broadcasting time cannot even be approximated with an error less than $\frac{1}{8}$, unless $\mathcal{P} = \mathcal{NP}$.

On the other hand, we will 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. Combining the negative and the positive results reveals the parameters that make broadcasting difficult. Depending on simple graph properties the complexity jumps from $\mathcal{NC}$ or $\mathcal{P}$ to $\mathcal{NP}$.

Classification: algorithms and data structures, computational complexity.

1 Introduction

Broadcasting in processor networks means disseminating a single piece of information, which is originally known only at 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

*supported by Deutsche Forschungsgemeinschaft, Grant DFG Re672/2, and German-Israeli Foundation for Scientific Research & Development, Grant GIF I-2007-199.06/91
†Institut für Theoretische Informatik, Wallstraße 40, 23560 Lübeck,Germany
email: jakoby/reischuk/schindel@informatik.mu-luebeck.de
‡revised June 1995

one round each processor can send a message to at most one of its neighbours. The goal is to inform everybody using as few rounds as possible. This number is called the *minimum broadcasting time* of the network.

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]. In several papers the broadcast capabilites of well known families of graphs like hypercubes, cube-connected-cycles, shuffle exchange graphs or de Bruijn graphs have been investigated and compared. In [HJM90] 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] Listman 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 paper 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 what seems to be known was that broadcasting is easy for trees 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 polylogarithmic parallel time for more general classes of graphs: graphs of bounded treewidth (see for example the paper by Arnborg, Lagergren and Seese [ALS91]) and graphs of small connectivity ([R91a]) – an overview can be found in [R91b].

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 [R91a] are not directly applicable. Still, modifying the framework developed in [R91a] 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. Furthermore, 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 favourably, 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.

2 Definitions and Previous Results

A formal definition of the broadcasting problem can be given as follows.

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* $E_i \subseteq \{ (u,v) \in E \mid u \in V_{i-1} \}$ and $\forall u \in V_{i-1} : \ | E_i \cap (\{u\} \times V)| \ \leq \ 1$.
Let us distinguish between the multiple source problem **MB** *and the restricted version with only a single source* **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 at 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 A: *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 B: *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 C: *SB can be solved in linear time for trees. This also holds for the constructive version of this problem finding an optimal broadcast schedule.*

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.

3.1 Lower Bounds

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

Theorem 1 *MB restricted to planar graphs with bounded degree at least* 4 *and a fixed deadline* T^* *at least* 3 *is* $\mathcal{NP}$ *-complete.*

The reduction to prove this result uses graphs of a specific kind that are guaranteed to have a broadcast schedule of length 4. Now consider an approximation algorithm that for a network G gives an estimate of its minimum broadcast time $T(G)$. The estimate $\tilde{T}(G)$ may be an arbitrary real number, but is required to be within a precision γ of the correct value: $(1+\gamma)^{-1} \cdot T(G) \leq \tilde{T}(G) \leq (1+\gamma) \cdot T(G)$. In this case, any estimate with a precision $\gamma \leq \frac{1}{8}$ could be used to solve the decision problem. Thus we also get

Theorem 2 *There exists no polynomial-time approximates algorithm for MB with a precision* $\frac{1}{8}$*, unless* $\mathcal{P} = \mathcal{NP}$*.*

If for this minimization problem one restricts to integer values and approximations from above then the statement of the theorem can be improved from precision $\frac{1}{8}$ to any value less than $\frac{1}{3}$.

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

Theorem 3 *SB restricted to graphs* $G = (V, E)$ *with bounded degree at least* 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 4 *SB restricted to planar graphs* $G = (V, E)$ *of degree* 3 *is* $\mathcal{NP}$*-complete (in this case the deadline grows like* $\sqrt{|V|}$*).*

For completeness, we should remark that after we have presented the lower bound in theorem 1 the first time Middendorf was able to improve it to degree 3 and deadline 2 [M92].

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 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 [H90] and [HR89]. There we have only considered node separators. 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 connected any pair of components:*

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

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

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

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 $G_i \in V_H$:*

$$|\, \mathrm{cut}(G_i) \,| \ \leq \ \kappa \ , \qquad |\, V_i \,| \ \leq \ \mu \qquad \text{and} \qquad \mathsf{cc}(G_i) \leq c \ ,$$

where $\mathsf{cc}(G_i)$ denotes the number of connected component of G_i. In this case $\mathrm{cut}(H)$ 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 $\sqrt{n} \times \sqrt{n}$–2-dimensional grid is $(O(\sqrt{n}), (O(\sqrt{n}), 1)$–edge decomposable into a tree. For a cycle of length n the parameters are $(4, 2, 2)$. 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 treewidth ([RS86]), see for example [ALS91],[BK91],[L90]. In the following we assume that an edge decomposition of the network is given and do not bother how to obtain such a decomposition. Although to construct an optimal decomposition in general is $\mathcal{NP}$-complete, efficient approximation schemes are known that achieve decompositions with at most a constant factor increase in the relevant parameters.

Theorem 5 *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*

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

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 at different rounds and each node of the network may have its individual deadline. Let us call this the *general broadcasting problem* **GB**.

The time bound becomes polynomial for classes of graphs that can be decomposed into smaller components using not too large separators. Let **llog** and **lllog** denote the logarithm function iterated twice, resp. three times.

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

- $(\mathrm{O}(\frac{\log n}{\mathrm{llog}\, n}), \mathrm{O}(\frac{\log n}{\mathrm{llog}\, n}), \mathrm{O}(1))$ *–edge decomposition trees or*

- *to graphs with bounded degree and* $(\mathrm{O}(\frac{\log n}{\mathrm{llog}\ n}), \mathrm{O}(\log n), \mathrm{O}(1))$ *–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 [R91a] one can also design an algorithm for constructing an optimal broadcast schedule by using the same techniques as for the decision problem.

Theorem 6 *Constructing an optimal broadcast schedule can be done in the same time bounds as stated for the decision problem in Theorem 5.*

Using the machinery developed in [R91a] 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 2 components can be replaced by a single component that externally behaves identically with respect to broadcasting.

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 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*

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

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

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

- *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 *form* 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. 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* $(\boldsymbol{\kappa}, \boldsymbol{\mu}, \boldsymbol{c})$ **–node decomposable** *if there exists a node decomposition graph* $H = (V_H, E_H)$ *such that for all* $G_i \in V_H$ *holds:*

$$|\ \mathrm{cut}(G_i)\ | \ \leq \ \kappa, \quad |\ V_i\ | \ \leq \ \mu, \quad \text{and} \quad \mathtt{cc}(G_i) \leq c \ .$$

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

Theorem 8 *Given a graph $G = (V, E)$ of maximal degree d with a (κ, μ, c) –node decomposition tree MB can be solved in time*

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

Similarly, we get in the parallel case

Theorem 9 *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*

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

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

All these bounds apply to GB as well. The constructive variant can be solved with the same effort, too.

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

Let us first observe how a nondeterministic TM can solve MB efficiently.

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

Proof: For a graph $G=(V,E)$ specified by adjacency lists, a set of sources V_0, and deadline $T^*\leq|V|$ do the following:

1. For each $x\in V\setminus V_0$ guess a triple (x,t_x,y_x) where $t_x\in[1,T^*]$ and y_x is a neighbour of x (linear time).
2. Add for each tripel (x,t_x,y_x) a tuple (x,t_x,x) to this list and sort the tripels according to the second component first and then to the third component. Let $L(x)$ be the sublist where the third component is x. For each node x verify that the first element of $L(x)$ is (x,t_x,x) (so for each tripel (x,t_x,y_x) $t_{y_x}<t_x$ holds) and for all $(y_1,t_{y_1},x),(y_2,t_{y_2},x)\in L(x)$ with $y_1\neq y_2$ holds $t_{y_1}\neq t_{y_2}$ (that means each node x does not inform more then one node at each round) (in time $\mathrm{O}(\,|V|\cdot\log^2|V|\,)$). ∎

The $\mathcal{NP}$-hardness of MB will be proved by a reduction of 3-Dimentsional 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:*
does M contain a matching, i. e., a subset $M'\subseteq M$ such that $|M'|=q$ and no two elements of M' agree in any coordinate. □

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 3DM** 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 1: 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 1). The other nodes and edges remain unchanged. $V_{A,1}$ is chosen as the set of sources, and the deadline is set to 3.

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

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

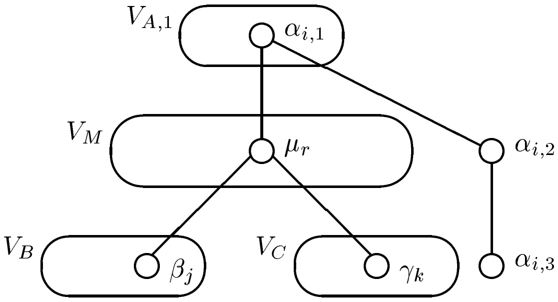

Figure 1: *The broadcasting graph corresponding to an instance of the 3DM problem.*

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

Round 2: The q sources informe the nodes in $V_{A,2}$, and the q of $V_{M'}$, informed at Round 1, the nodes of V_B: $V_2 := V_1 \cup V_{A,2} \cup V_B$.

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 send the information to the nodes of $V_M \setminus V_{M'}$: $V_3 := V_2 \cup V_{A,3} \cup V_C \cup (V_M \setminus V_{M'}) = V$.

All nodes of V_B can be informed at round 2, and of V_C at round 3 by the nodes in $V_{M'}$, since M' is a matching for A, B, and C.

It is also possible to inform the nodes in $V_M \setminus V_{M'}$ at 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 with 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 size at least $|V''|$. Therefore there exists a matching for $V_M \setminus V_{M'}$ 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 $\alpha_{i,2}$ at 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 tripels corresponding to V' establish a matching M' of A, B, C. ∎

Theorem 2 follows easily. Since any algorithm that approximates the minimum broadcast time with a deviation smaler than $\frac{1}{8}$ has to yield a value larger then 3.5 if the optimal schedule length is 4 and smaler then 3.5 if it is 3.

5 Modular Cunstruction 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 graphs with special brodacast 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 round in which a node v get the information first is called its* **starting round** $t_s(v)$ *. A node v is called* **active** *at a round t if sends the information to another node u at this round. Let $t_f(v)$ be the round all neighbors of v have received the information. A node v is called* **busy** *in $\mathcal{E}$ if it is active in round $t_s(v)+1, \ldots, t_f(v)$. $\mathcal{E}$ is* **busy** *if all nodes are busy.* □

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

The proofs of Theorems 3 and 4 contain rather complicated constructions. So this section precedes with an analysis of some special subgraphs. The basic ideas used here are

1. *granularity:*. The graph entirely consists of subgraphs described in these section.
2. *information flow:*. Information enters or leaves a subgraph always at the same specified nodes.

Edges between such graphs are input or output edges. The output edges of a subgraph G' are input edges of another subgraph G''. So we call G'' a successor of G' and G' a predecessor of G''. In the following we mark input and output edges by a corresponding arrow.

Definition 6 *Let $e_1, e_2, \ldots, e_k$ be the input edges of a graph G' and $t_1, t_2, \ldots, t_k$ be the rounds these edges have been used in a particular broadcast schedule. Then the vector $\tau = < t_1, t_2, \ldots, t_k >$ is called a* **time table** *of G' and the set of all possible time tables is called the* **time sheet** $\mathcal{T}(G')$ *of G'.* □

The construction of an optimal broadcast schedule of the graphs for the proofs of Theorem 3 and 4 guarantees that the difference between the rounds when the different input edges of a subgraph are used is at most 2.

Definition 7 *For a subgraph G' and a time table $\tau = < t_1, t_2, \ldots, t_k >$ of G' we call* **delay**(G', τ) *the minimal number of rounds needed to send the information across an output edge if the input edges are used as specified at τ.* □

We get a lower bound for the rounds an input edge of G' can be used if we add all delays on the shortest path to the sources.

Definition 8 *Let v_0 be the source and $\mathrm{path}(v_0, G')$ the set of all paths from v_0 to the subgraph G' then*

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

□

If in a subgraph G' a node is informed at round t, we call $t - \mathrm{dawn}(G')$ the relative round this node is informed.

To ensure that information enters a graph only at input edges we use the notion of a ghost message. A ghost message is an information transfer from a successor G'' of a subgraph G' to G' across an output edge of G'. To prevent such information flow we use the following techniques.

1. For any subgraph all successing subgraphs have the same dawn.
2. For a subgraph G an input edge can be used from the preceding subgraph at round $\text{dawn}(G)+2$ at the latest.
3. We measure the minimum rounds information needs to reach an input edge of a subgraph G' while starting at another input edge of G' and using only edges of G'. We call this quantity ghost time. We ensure that the ghost time of all subgraphs is at least 3.

Let us call the mapping from a time table τ of a subgraph G' to the set of rounds using output edges of G' in an optimal and busy broadcast schedule of G' the *broadcast relation* $\mathcal{B}$ of G'.

Definition 9 *Let $\mathcal{G}$ be the set of graphs described below then we define the broadcast relation*

$$\mathcal{B} : \mathcal{G} \times \mathbb{N} \times \mathbb{N}^n \to \mathsf{P}(\mathbb{N}^m) \cup \perp$$

as follows: $(a_1 \ldots a_n) \in \mathcal{B}(G', T^*, t_1 \ldots t_m)$ *iff*

- $< t_1 \ldots t_m > \in \mathcal{T}(G')$,
- *the n output edges of G' can be used in a broadcast schedule at round a_1 , a_2 , … a_n and it is still possible to inform all nodes of G' within the deadline T^* , where the m input edges of G' were used at round t_1 , t_2 , … t_m .*

$\mathcal{B}(G', T^*, t_1 \ldots t_m) = \perp$ *iff it is not possible to inform all nodes of G' within the deadline T^* , when the m input edges of G' were used at round t_1 , t_2 , … t_m .* □

Note that τ and $\mathcal{B}(G', T^*, \tau)$ describe the functionality of the graph G'.

5.1 Timer

The graph I_l shown in Figure 2 consists of a chain with $l+1$ nodes. It serves as an information sink.

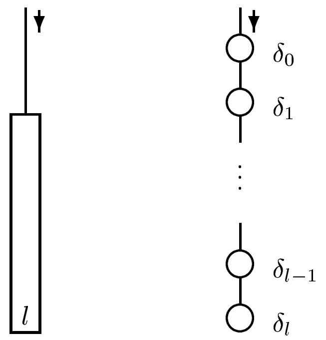

Figure 2: *The subgraph timer I_l*

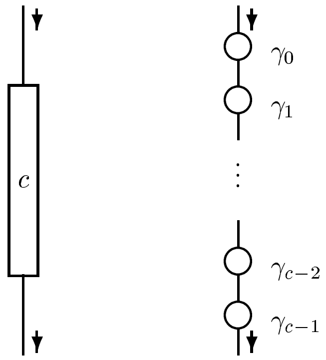

Figure 3: *A postponer J_c.*

Since there is no output node there does not exist a parmater delay for this subgraph. Its Broadcast relation is given by

$$\mathcal{B}(I_l, T^*, t) = \begin{cases} \emptyset & \text{if } l \le T^* - t \\ \perp & \text{if } l > T^* - t \end{cases}$$

5.2 Postponer

We call the chain J_c with one input node and one output node shown in figure 3 a postponer. It will be used to synchronize the dawn of subgraphs. Its characteristics are given by:

$$
\begin{aligned}
\mathrm{delay}(J_c) &= c \\
\mathcal{B}(J_c, T^*, t) &= \begin{cases} \{(t+c)\} & \text{if } c \le T^* - t \\ \perp & \text{if } c > T^* - t \end{cases}
\end{aligned}
$$

Since for any type of subgraph G and for $x > T^* - \mathrm{dawn}(G) - \mathrm{delay}(G)$ we get $\mathcal{B}^G(x) = \perp$ we will omit this case in the following.

5.3 Initializer

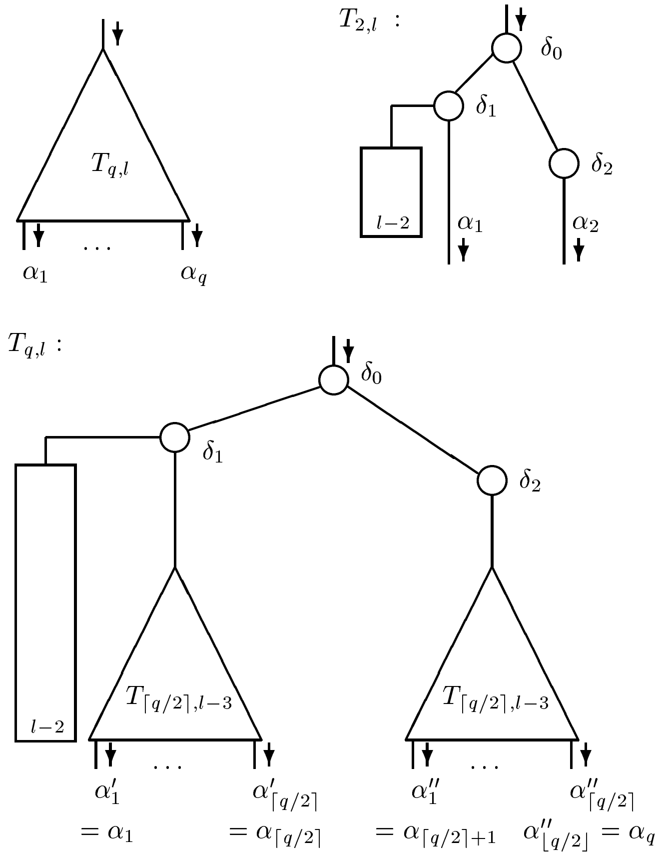

Figure 4: *The recursive construction of the trees $T_{q,l}$. The timer form a chain that forces the left son δ_1 of the root δ_0 to be informed first in order to obey the deadline l. For $q > 2$ the tree $T_{q,l}$ is obtained from 2 copies of $T_{\lceil q/2\rceil,l-3}$.*

This subgraph recursivly described in Figure 4 transmits the information simultaneously on all its q output edges. For the parameters we choose $l \ge 3\log q - 1$ $l = T^* - \mathrm{dawn}(T_{q,l})$ with $l \ge 3\log q - 1$.

Node δ_1 must be informed at relative round 1 to finish the timer graph on the left on time. Then δ_0 can inform δ_2 at relative round 2. Hence, nodes δ_1 and δ_2 can information simultanously at relative round 3 to their successors.

$$
\begin{aligned}
\mathrm{delay}(T_{q,T^*-t}) &= 3\log q \\
\mathcal{B}(T_{q,T^*-t}, T^*, t) &= \begin{cases} \{(\underbrace{3\log q, \ldots, 3\log q}_{q})\} & \text{if } t \le T^* - 3\log q \\ \perp & \text{if } t > T^* - 3\log q \end{cases}
\end{aligned}
$$

5.4 Chooser

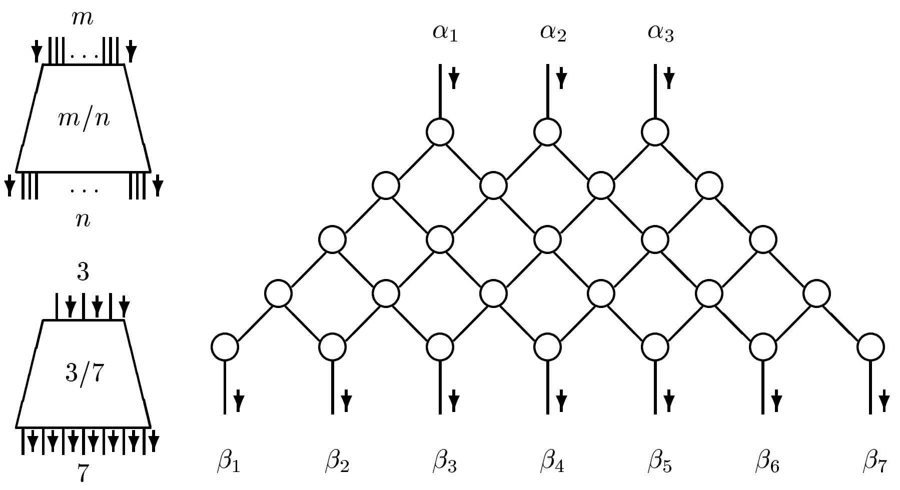

Figure 5: *The chooser subgraph* M_n^m.

This is the first subgraph that provides more than one minimal output round tupel and therefore requires a broadcast strategy to make a selection. It has m inputs and n outputs where $m < n$.

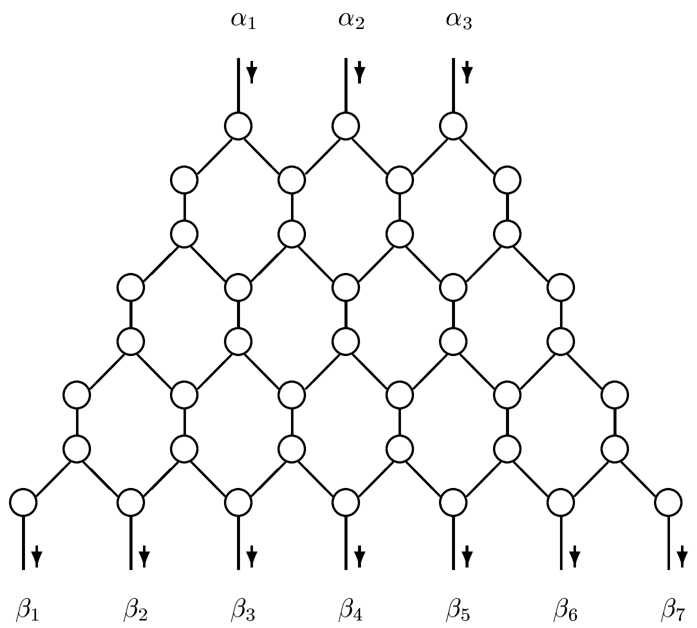

Figure 6: *A general 3–regular chooser subgraph*

Figure 5 shows a version of a chooser graph with degree 4. This general graph can be revised to a degree 3 version illustrated in Figure 6. Since we will only need the version M_3^1 chooser, the first design in version of Figure 5 already achieves degree 3.

The m input nodes can only inform m of the n of output edges in round $\text{delay}(M_n^m)$. The other output edges need at least one more round to be informed.

Note, that here a ghost message needs at least 3 rounds to make a short cut from one input node to an outer neighbour of another input.

Since of the regularity of the chooser the following characteristics can easily be checked.

$$
\begin{aligned}
\text{delay}(M_n^m) &= n-m+1 \\
\mathcal{B}(M_n^m, T^*, \underbrace{t,\ldots,t}_{m}) &= \begin{cases} \{(y_1,\ldots,y_m) \mid |\{i|y_i = t+n-m+1\}| = m & \text{if } n-m+2 \le T^*-t \\ \quad \text{and} \quad y_i \in \{t+n-m+1, t+n-m+2\} & \\ \perp & \text{if } n-m+2 > T^*-t \end{cases}
\end{aligned}
$$

5.5 Collector

This quasi inverse subgraph collects m inputs and delaies the information only if n out of m inputs were informed at dawn.

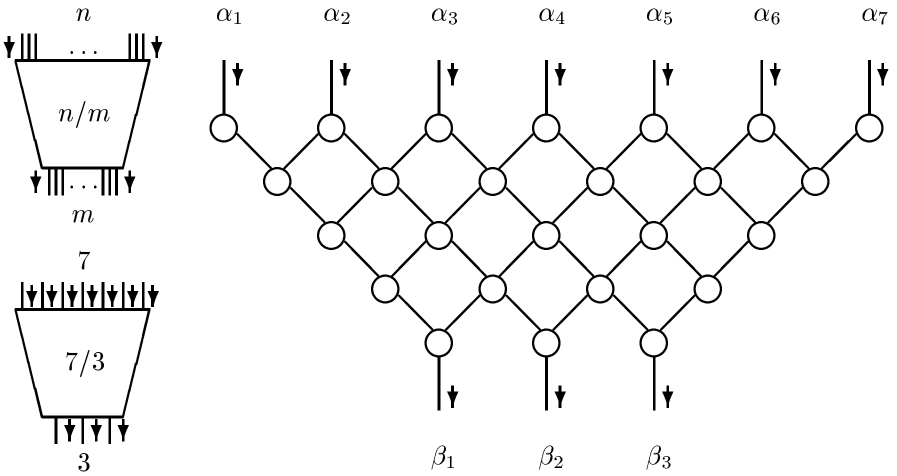

Figure 7: *The collector subgraph* W^c.

Again we introduce two versions of this subgraphs (Figure 7 and Figure 8). The details below refer to the version of Figure 7 since only the 3/1 variant will be used later.

$$
\begin{aligned}
\text{delay}(W_n^m) &= m-n+1 \\
\mathcal{B}(M_n^m, T^*, t_1, \ldots t_m) &= \begin{cases} \{\text{dawn}(M_n^m)+m-n+1\}^n & \text{if } |\{\, i \mid t_i = \text{dawn}(M_n^m)\}| \ge n \\ & \quad \text{and} \\ & m-n+2 \le T^* - \text{dawn}(M_n^m) \\ \{t \in \mathbb{N}^n | t > [\text{dawn}(M_n^m) & \text{if } |\{\, i \mid t_i = \text{dawn}(M_n^m)\}| < n \\ \qquad +m-n+1]^n\} & \quad \text{and} \\ & m-n+3 \le T^* - \text{dawn}(M_n^m) \\ \perp & \text{else} \end{cases}
\end{aligned}
$$

with $t_i \in \{\text{dawn}(M_n^m), \text{dawn}(M_n^m)+2\}$.

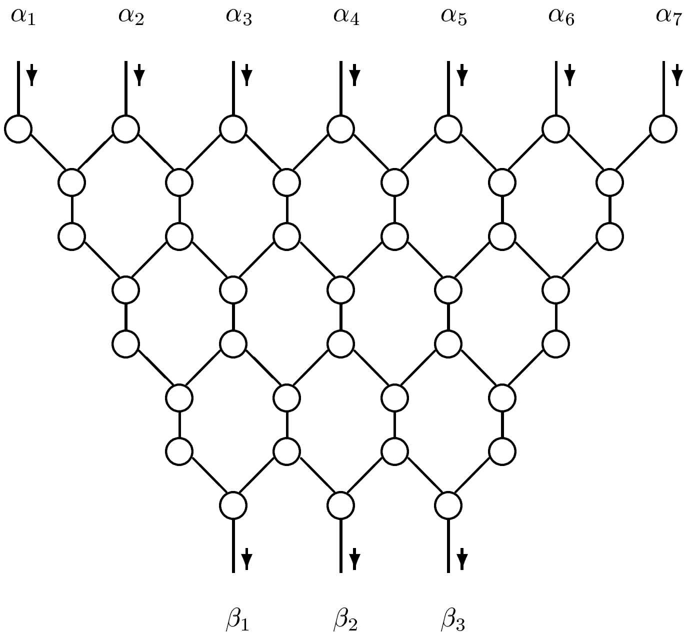

Figure 8: *A degree 3 variant of the collector subgraph.*

5.6 Padder

This subgraph E_d^l (Figure 9) enlarges the difference arrival rounds on an input edge. If the information enters λ_0 at round $\mathrm{dawn}(E_d^l)$ with $l = T^* - \mathrm{dawn}(E_d^l) + 1$ it has to inform the timer of length $l-1$ at round $\mathrm{dawn}(E_d^l) + 1$. One round later, it can inform λ_1. But again, it is necessary to inform the second timer of length $l-3$ first. Analogously all other nodes λ_i each cause one round delay.

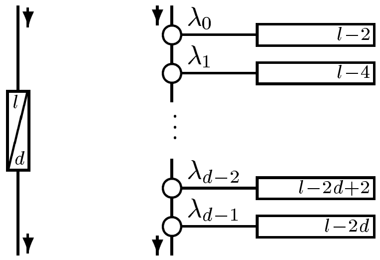

Figure 9: *The padder subgraph* E_d^l.

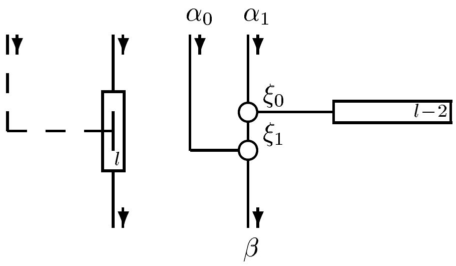

Figure 10: *The separator subgraph* Y_l.

If the information arrives only one round earlier it can pass through all λ_i without informing the timers first.

$$
\begin{aligned}
\mathrm{delay}(E_d^l) &= d \\
\mathcal{B}(E_d^l, T^*, t) &= \begin{cases} \{t+d\} & \text{if } l \leq T^* - t \\ \{t+2d\} & \text{if } l = T^* - t + 1 \\ \perp & \text{else} \end{cases}
\end{aligned}
$$

5.7 Separator

Figure 10 shows the separator subgraph Y_l with input edges α_0, α_1 and output edge β. We will choose for $l = T^* - \mathrm{dawn}(Y_l)$ with $l \geq 2$. We consider only the case that α_0 is used at $\mathrm{dawn}(Y_l)$.

This subgraph separates round constellations into two groups.

- α_1 is used at $\text{dawn}(Y_l)$ or $\text{dawn}(Y_l)+1$. Then, there are enough round to solve the timer directly and ζ_1 can immediately inform its outer sucessor.
- α_1 is informed later. Then node ζ_1 has to inform ζ_0, so ζ_1 can inform the timer at $\text{dawn}(Y_l)+2$. Therefore, ζ_1 can only inform the outer successor not before $\text{dawn}(Y_l)+2$.

Here, the mechanism of analyzing the ghost time fails. Remember, that ghost time was the round a ghost message needs before it can start. Formally the ghost time is 2. This is the number of rounds information needs from node ζ_0 to ζ_1, resp from ζ_1 to ζ_0. We will embed this subgraph, such that ζ_1 can always be informed at dawn. Now the relevant ghost time is the number of rounds information needs from ζ_1 to the other neighbour of ζ_0. Since ζ_0 has to inform its timer before it can continue, the relevant ghost time is 3.

$$
\begin{aligned}
\text{delay}(Y_l) &= 1 \\
\mathcal{B}(Y, T^*, \text{dawn}(Y_l), t) &= \begin{cases} \{t+1\} & \text{if } t \leq \text{dawn}(Y_l)+1 \\ \{t+2\} & \text{if } t \geq \text{dawn}(Y_l)+2 \end{cases}
\end{aligned}
$$

5.8 Crossing

This elementary subgraphs realizes the planar crossing of one late incoming information to the opposite output. There exist two versions of crossing graphs. The simplier version, shown in Figure 11, has degree 4. We will use the slightly more complex subgraph described in Figure 12 which has degree 3.

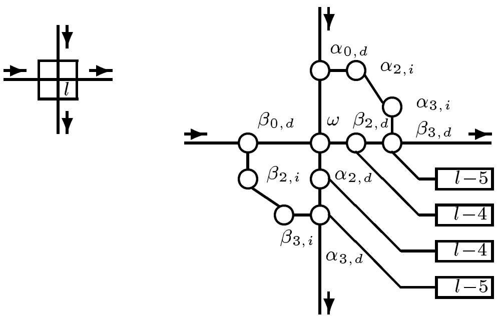

Figure 11: *A variant of a crossing not used in this paper.*

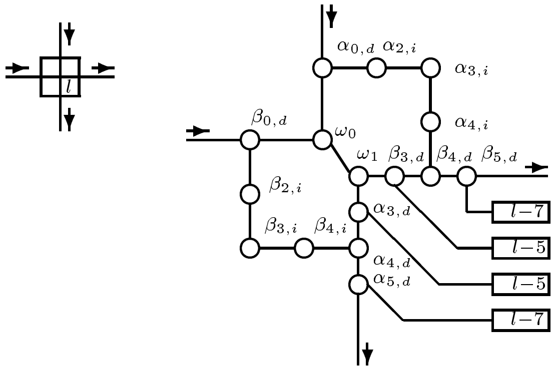

Figure 12: *The planar crossing X_l with degree 3.*

The input edges are connected to $\alpha_{0,d}$ and $\beta_{0,d}$. Choose $l = T^* - \text{dawn}(G)$ with $l \geq 7$.

- $\alpha_{0,d}$ and $\beta_{0,d}$ are informed at $\text{dawn}(X_l)$.

  Figure 13 shows that there exists a scheduling with relative number of rounds such that information can rush through this subgraph while all timers in this subgraph can be finished on time.

- $\alpha_{0,d}$ is informed at $\text{dawn}(X_l)$ and $\beta_{0,d}$ at $\text{dawn}(X_l)+2$.

  Then, $\alpha_{0,d}$ has to inform ω_0 at first. If not, one of the timers with length $t-5$ (at $\alpha_{3,d}$ or at $\beta_{3,d}$) can not be informed totally within the deadline (see Figure 14)

$\beta_{0,d}$ is informed at $\mathrm{dawn}(X_l)+2$. So, to prevent artificial delay node ω_0 informs ω_1 at $\mathrm{dawn}(X_l)+2$.

Now if ω_1 informs $\beta_{3,d}$ at $\mathrm{dawn}(X_l)+3$ then $\alpha_{3,d}$ is informed at $\mathrm{dawn}(X_l)+4$. So, $\alpha_{3,d}$ must inform its timer. Then, $\alpha_{4,d}$ is informed at $\mathrm{dawn}(X_l)+6$ and $\alpha_{5,d}$ is unable to inform its timer in time.

But if ω_1 informs $\alpha_{3,d}$ at $\mathrm{dawn}(X_l)+3$ information can rush to the outer successor of $\alpha_{5,d}$ (see fig 15) while the other output successor can still be informed at $\mathrm{dawn}(X_l)+8$.

- Both input nodes are informed at $\mathrm{dawn}(X_l)+2$.

  Figure 16 gives an impression that there does not exist a suited broadcast strategy. A proof or this property is not necessary, because we will focus our interest in preventing such a input constellation.

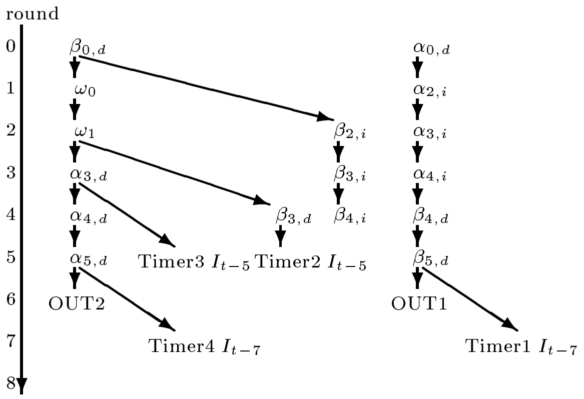

Figure 13: *The scheduling scheme for a crossing for inputs schedule at* $(\mathrm{dawn}(X_l), \mathrm{dawn}(X_l))$.

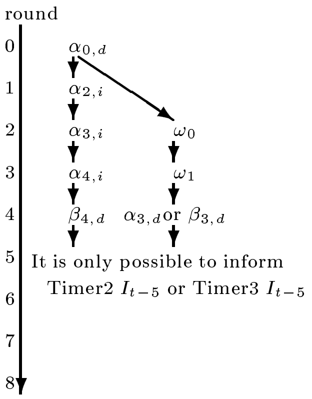

Figure 14: *A scheduling scheme for inputs schedule at* $(\mathrm{dawn}(X_l), \mathrm{dawn}(X_l)+2)$ *unable to hold the deadline.*

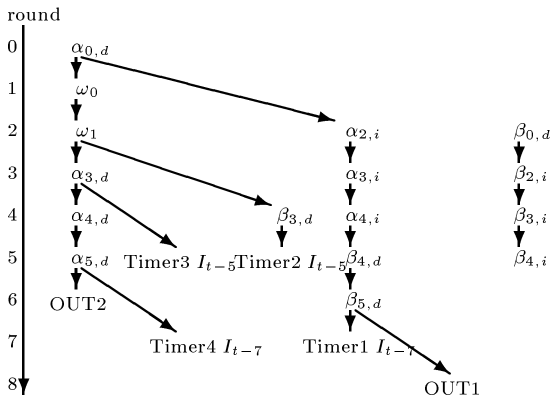

Figure 15: *The scheduling scheme for inputs schedule at* $(\mathrm{dawn}(X_l), \mathrm{dawn}(X_l)+2)$.

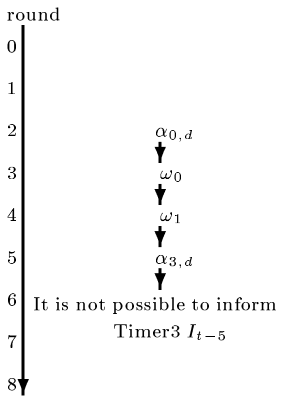

Figure 16: *No scheduling possible for inputs schedule at* $(\mathrm{dawn}(X_l)+2, \mathrm{dawn}(X_l)+2)$.

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

5.9 Guess–Graph

The following subgraphs will be used for the proof of Theorem 4. In this proof a reduction of a satisfiability problem to planar broadcasting is constructed. Thus an equivalent to the choice of the setting of a variable is needed.

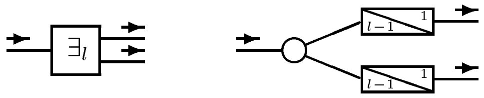

Figure 17: *The guess subgraph* $\exists_l$.

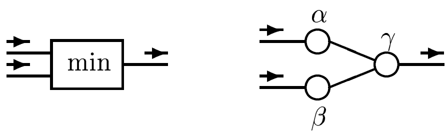

Figure 18: *The min subgraph* Min.

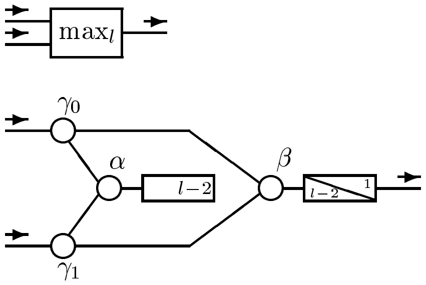

Figure 19: *The max subgraph* Max_l.

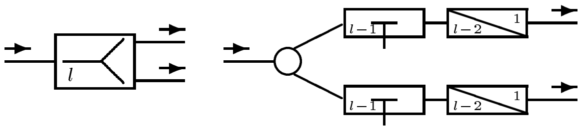

Figure 20: *A simplified version of the duplicator subgraph.*

This is done by the guess–graph shown in Figure 17 containing two padders of delay 1. If informations arrives at $\text{dawn}(\exists_l)$, the broadcast strategy can choose between two minimal output tupels $(\text{dawn}(\exists_l)+2, \text{dawn}(\exists_l)+4)$ or $(\text{dawn}(\exists_l)+4, \text{dawn}(\exists_l)+2)$, if we choose $l \geq 2$ and $l = T^* - \text{dawn}(\exists_l)$.

After informing the node connected to the input edge the broadcast strategy has to decide to take the upper or lower path first. The padders enlarge this round difference to $(\text{dawn}(\exists_l) + 2, \text{dawn}(\exists_l) + 4)$ or $(\text{dawn}(\exists_l) + 4, \text{dawn}(\exists_l) + 2)$.

$$
\begin{aligned}
\text{delay}(\exists) &= 2 \\
\mathcal{B}(\exists_l, T^*, t) &= \begin{cases} \{(\text{dawn}(\exists_l) + 2, \text{dawn}(\exists_l) + 4), & \text{if } t = \text{dawn}(\exists_l) \\ \quad (\text{dawn}(\exists_l) + 4, \text{dawn}(\exists_l) + 2)\} & \\ \perp & \text{if } t > \text{dawn}(\exists_l) \end{cases}
\end{aligned}
$$

5.10 Min-Graph

This graph shown in Figure 18 is self–explanary.

$$
\begin{aligned}
\text{delay}(\text{Min}) &= 2 \\
\mathcal{B}(\text{Min}_l, T^*, t_1, t_2) &= \begin{cases} \{\min(t_1, t_2) + 2\} & \text{if } \min(t_1, t_2) \leq T^* - 2 \\ \perp & \text{else} \end{cases}
\end{aligned}
$$

5.11 Max-Graph

This subgraph does not entirely simulate the maximum of input rounds. The case that both inputs are used later than $\text{dawn}(\text{Max}_l)$ causes a breakdown.

We choose $l = T^* - \text{dawn}(\text{Max}_l)$ with $l \geq 4$. If any node γ_0 or γ_1 is informed at $\text{dawn}(\text{Max}_l)$, one of it has to inform α immediately because of the timer in the middle.

So β is informed at $\text{dawn}(\text{Max}_l) + 1$ iff both input nodes were informed at $\text{dawn}(\text{Max}_l)$. If only one input arrives at $\text{dawn}(\text{Max}_l)$ and the other at $\text{dawn}(\text{Max}_l) + 2$ or later then β will be informed at $\text{dawn}(\text{Max}_l) + 2$. The padder will enlarge the possible output rounds to $\text{dawn}(\text{Max}_l) + 3$ or $\text{dawn}(\text{Max}_l) + 5$.

$$
\begin{aligned}
\text{delay}(\text{Max}_l) &= 3 \\
\mathcal{B}(\text{Max}_l, T^*, t_1, t_2) &= \begin{cases} \text{dawn}(\text{Max}_l) + 3 & \text{if } t_1 = \text{dawn}(\text{Max}_l) \quad \text{and} \quad t_2 = \text{dawn}(\text{Max}_l) \\ \text{dawn}(\text{Max}_l) + 5 & \text{if } |\{t_i | t_i = \text{dawn}(\text{Max}_l), i \in \{1,2\}\}| = 1 \\ \perp & \text{else} \end{cases}
\end{aligned}
$$

5.12 Duplicator

Later we establish a coding system where we will use only relative rounds 0 and 2. So, this subgraph that is built using padders and partitioners duplicates binary encodings.

A simplified version of a duplicator can be seen in Figure 20. The additional inputs of the separators are informed at relative round 1 each. Figure 21 shows a planar version of a duplicator.

We assume that the second input is informed at $\text{dawn}(\text{Dup}_\text{l})$ and the first one at round $\text{dawn}(\text{Dup}_\text{l})$ or $\text{dawn}(\text{Dup}_\text{l}) + 2$.

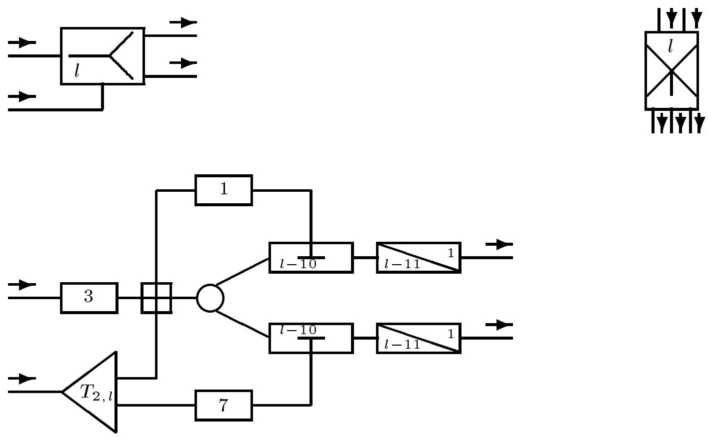

Figure 21: *The complete duplicator subgraph* Dup_l.

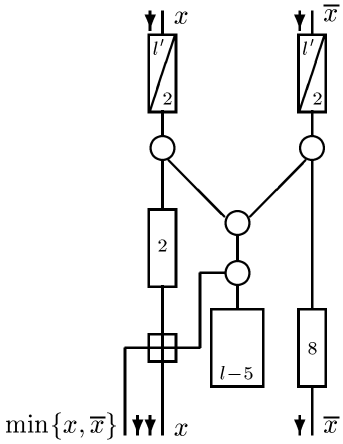

Figure 22: *The generator subgraph* Gen_l *where* $l' = l - 1$.

We choose $l = T^* - \text{dawn}(\text{Dup}_\text{l})$ with $l \geq 13$. Combining all delay paramters described before we get the new delay. By combining the broadcast relation of partitioners, padders, the crossing, initializers and postponers we get the desired broadcast relation of a duplicator.

$$
\begin{aligned}
\text{delay}(\text{Dup}_\text{l}) &= 12 \\
\mathcal{B}(\text{Dup}_\text{l}, \text{T}^*, \text{t}, \text{dawn}(\text{Dup}_\text{l})) &= \begin{cases} \{(\text{dawn}(\text{Dup}_\text{l}) + 12, \text{dawn}(\text{Dup}_\text{l}) + 12)\} & \text{if } t = \text{dawn}(\text{Dup}_\text{l}) \\ \{(\text{dawn}(\text{Dup}_\text{l}) + 14, \text{dawn}(\text{Dup}_\text{l}) + 14)\} & \text{if } t = \text{dawn}(\text{Dup}_\text{l}) + 2 \end{cases}
\end{aligned}
$$

5.13 Generator

For our construction of a planar graph we need a subgraph, that extracts from a binary coding of information streams an early information without changing the information contents of the stream. We call such a graph described in Figure 22 generator.

We choose $l = T^* - \mathrm{dawn}(\mathrm{Gen_l})$ with $l \geq 14$. We only consider the following time tables $(\mathrm{dawn}(\mathrm{Gen_l}), \mathrm{dawn}(\mathrm{Gen_l}) + 2)$ or $(\mathrm{dawn}(\mathrm{Gen_l}) + 2, \mathrm{dawn}(\mathrm{Gen_l}))$. Then, the padders inform their successors at $(\mathrm{dawn}(\mathrm{Gen_l}) + 2, \mathrm{dawn}(\mathrm{Gen_l}) + 6)$ or $(\mathrm{dawn}(\mathrm{Gen_l}) + 6, \mathrm{dawn}(\mathrm{Gen_l}) + 2)$. Now the earlier node has to inform the timer directly at $\mathrm{dawn}(\mathrm{Gen_l})) + 5$. So the left crossing has input $(\mathrm{dawn}(\mathrm{Gen_l}) + 6, \mathrm{dawn}(\mathrm{Gen_l}) + 6)$ if the left input node was informed at dawn or $(\mathrm{dawn}(\mathrm{Gen_l}) + 8, \mathrm{dawn}(\mathrm{Gen_l}) + 6)$ else.

Hence, the output edges can used at $(\mathrm{dawn}(\mathrm{Gen_l}) + 12, \mathrm{dawn}(\mathrm{Gen_l}) + 12, \mathrm{dawn}(\mathrm{Gen_l}) + 14)$ or $(\mathrm{dawn}(\mathrm{Gen_l}) + 12, \mathrm{dawn}(\mathrm{Gen_l}) + 14, \mathrm{dawn}(\mathrm{Gen_l}) + 12)$.

$$
\begin{aligned}
\mathrm{delay}(\mathrm{Gen_l}) &= 12 \\
\mathcal{B}(\mathrm{Gen_l}, \mathrm{T}^*, \mathrm{t_1}, \mathrm{t_2}) &= \begin{cases} \{(\mathrm{dawn}(\mathrm{Gen_l}) + 12, \mathrm{dawn}(\mathrm{Gen_l}) + 12, & \text{if } t_1 = \mathrm{dawn}(\mathrm{Gen_l}) \\ \qquad \mathrm{dawn}(\mathrm{Gen_l}) + 14)\} & \\ \{(\mathrm{dawn}(\mathrm{Gen_l}) + 12, \mathrm{dawn}(\mathrm{Gen_l}) + 14, & \text{if } t_2 = \mathrm{dawn}(\mathrm{Gen_l}) \\ \qquad \mathrm{dawn}(\mathrm{Gen_l}) + 12)\} & \end{cases}
\end{aligned}
$$

5.14 Binary–Unary–Converter

The crossing-graph does not enable a total crossing of information we use two methods to achieve it though.

- We code a binary information redundant by pairs of in-round ($\mathrm{dawn}(\mathrm{BU_t})$) or late ($\mathrm{dawn}(\mathrm{BU_t})+2$) (that means $(\mathrm{dawn}(\mathrm{BU_t}), \mathrm{dawn}(\mathrm{BU_t}) + 2)$ for true and $(\mathrm{dawn}(\mathrm{BU_t}) + 2, \mathrm{dawn}(\mathrm{BU_t}))$ for false).
- For the construction of a crossing we first convert two information pairs to a unary coding and then decode this unary notation to the binary exchanged counterpart.

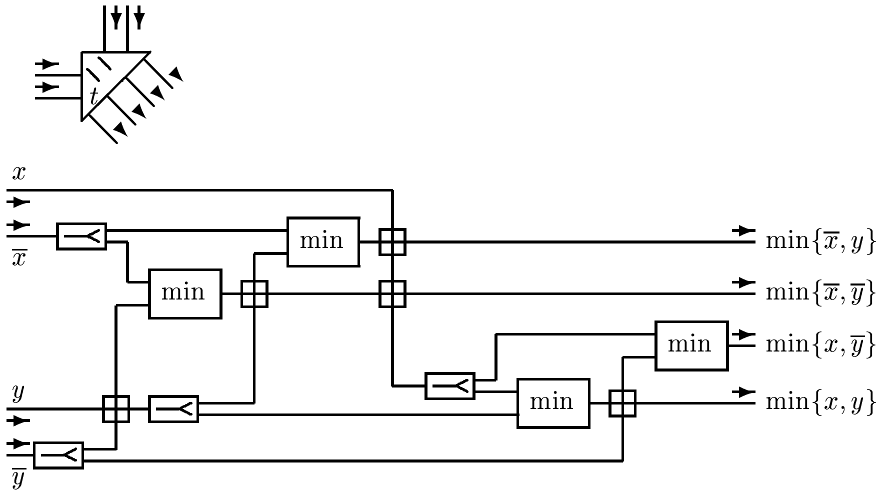

Figure 23: *A simplified version of a binary-unary-converter.*

Figure 23 shows the binary–unary–converter in simplified version. In this figure all transponers are missing that synchronize the dawns of equal leveled crossings, min-graphs and duplicators. Note, that there is no possibility that two late information arrive at any crossing.

The additional eight inputs are necessary for the duplicators.

We enhance this construction by planarizing the additional inputs and the missing postponers. Figure 25 shows the complete subgraph. The additional fifth input will be informed at dawn. The other four carry two binary codings. We choose $t = T^* - \mathrm{dawn}(\mathrm{BU_t})$ with $t \geq 86$.

$$
\begin{aligned}
\mathrm{delay}(\mathrm{BU_t}) &= 84 \\
\mathcal{B}(\mathrm{BU_t}, \mathrm{T}^*, \mathrm{i}, \mathrm{l}, \mathrm{i}, \mathrm{l}, \mathrm{dawn}(\mathrm{BU_t})) &= \{(i', l', i', i')\} \\
\mathcal{B}(\mathrm{BU_t}, \mathrm{T}^*, \mathrm{i}, \mathrm{l}, \mathrm{l}, \mathrm{i}, \mathrm{dawn}(\mathrm{BU_t})) &= \{(l', i', i', i')\} \\
\mathcal{B}(\mathrm{BU_t}, \mathrm{T}^*, \mathrm{l}, \mathrm{i}, \mathrm{i}, \mathrm{l}, \mathrm{dawn}(\mathrm{BU_t})) &= \{(i', i', l', i')\} \\
\mathcal{B}(\mathrm{BU_t}, \mathrm{T}^*, \mathrm{l}, \mathrm{i}, \mathrm{l}, \mathrm{i}, \mathrm{dawn}(\mathrm{BU_t})) &= \{(i', i', i', l')\}
\end{aligned}
$$

with $i = \mathrm{dawn}(\mathrm{BU_t})$, $l = \mathrm{dawn}(\mathrm{BU_t}) + 2$, $i' = \mathrm{dawn}(\mathrm{BU_t}) + 84$, and $l' = \mathrm{dawn}(\mathrm{BU_t}) + 86$.

5.15 Unary–Binary–Converter

Here, the corresponding unary–binary–converter. Again in Figure 24 all needed postponers are not shown and note, that no unary input is able to force a double occurence of late informations at input nodes of crossings or max-graphs.

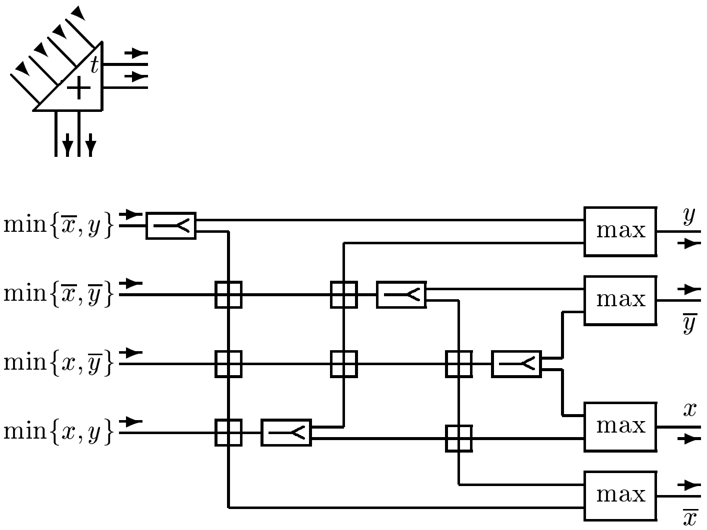

Figure 24: *The simplified version of an unary–binary–converter subgraph* $\mathrm{UB_t}$.

We planarize the additional inputs for the duplicators and get the desired parameters of this graph by combining those of the subgraphs duplicater, postponer, crossing and max-graph in the complete graph shown in Figure 26.

We choose $t = T^* - \mathrm{dawn}(\mathrm{UB_t})$ with $t \geq 131$.

$$
\begin{aligned}
\mathrm{delay}(\mathrm{UB_t}) &= 129 \\
\mathcal{B}(\mathrm{UB_t}, \mathrm{i}, \mathrm{l}, \mathrm{i}, \mathrm{i}, \mathrm{dawn}(\mathrm{UB_t})) &= \{(i', l', i', l')\} \\
\mathcal{B}(\mathrm{UB_t}, \mathrm{l}, \mathrm{i}, \mathrm{i}, \mathrm{i}, \mathrm{dawn}(\mathrm{UB_t})) &= \{(l', i', i', l')\} \\
\mathcal{B}(\mathrm{UB_t}, \mathrm{i}, \mathrm{i}, \mathrm{l}, \mathrm{i}, \mathrm{dawn}(\mathrm{UB_t})) &= \{(i', l', l', i')\} \\
\mathcal{B}(\mathrm{UB_t}, \mathrm{i}, \mathrm{i}, \mathrm{i}, \mathrm{l}, \mathrm{dawn}(\mathrm{UB_t})) &= \{(l', i', l', i')\}
\end{aligned}
$$

with $i = \mathrm{dawn}(\mathrm{UB_t})$, $l = \mathrm{dawn}(\mathrm{UB_t}) + 2$, $i' = \mathrm{dawn}(\mathrm{UB_t}) + 129$, and $l' = \mathrm{dawn}(\mathrm{UB_t}) + 131$.

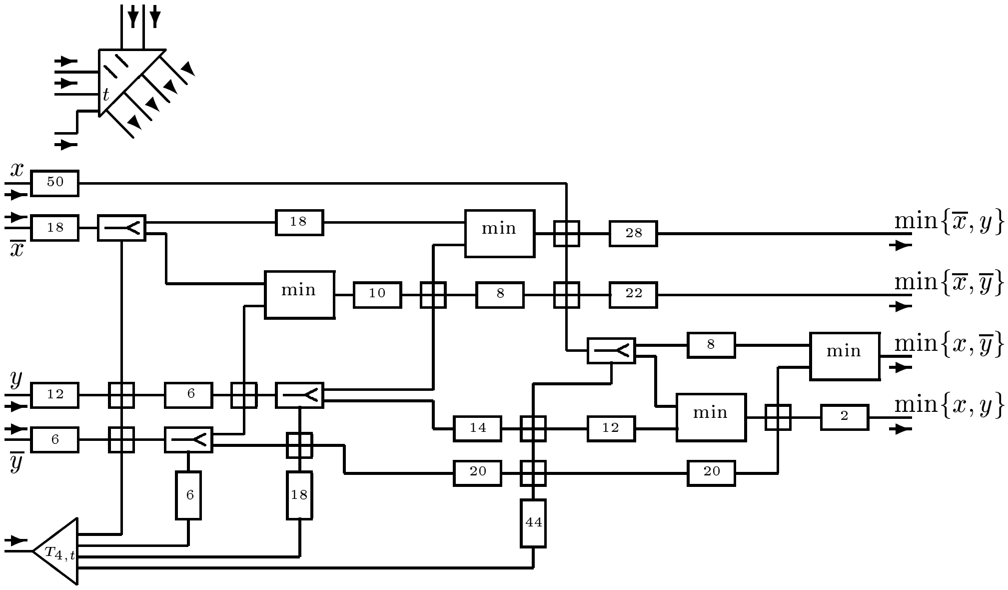

Figure 25: *The complete binary-unary-converter subgraph* $\mathrm{BU_t}$.

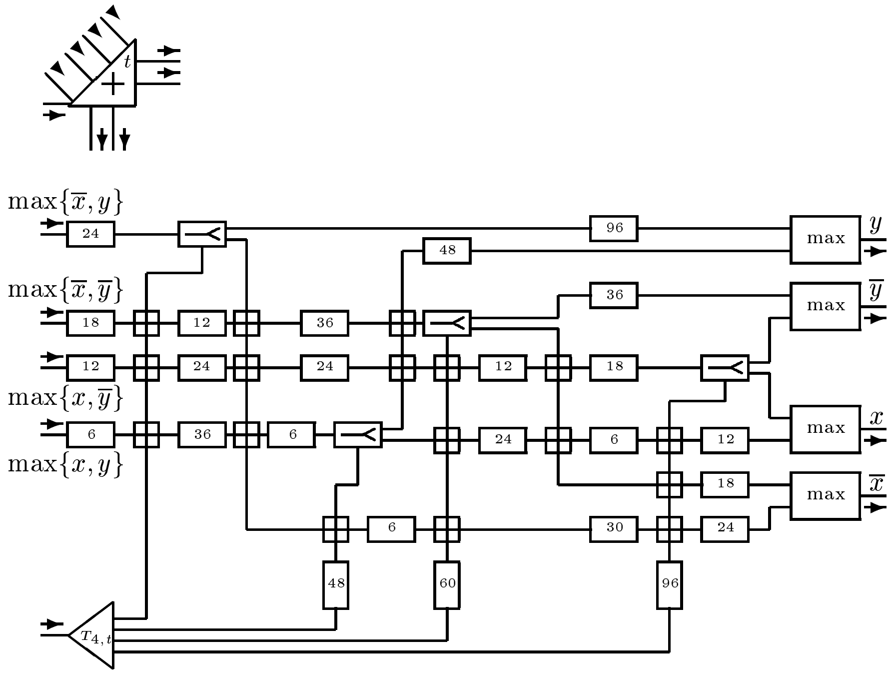

Figure 26: *The complete unary–binary–converter subgraph* $\mathrm{UB_t}$.

5.16 Double Crossing

This is the realisation of planar information crossing. Here we simply combine the binary–unary–converter with the binary–unary–converter (Figure 27).

We use the generator from above to generate early inputs for both converters. See Figure 29 for details. We choose $t = T^* - \mathrm{dawn}(\mathrm{XX_t})$ with $t \geq 230$.

$$
\begin{aligned}
\mathrm{delay}(\mathrm{XX_t}) &= 228 \\
\mathcal{B}(\mathrm{XX_t}, \mathrm{T}^*, \mathrm{d+x}, \mathrm{d+2-x}, \mathrm{d+y}, \mathrm{d+2-y}) &= \{(d+y+228, d+2-y+228, \\
&\qquad d+x+228, d+2-x+228)\}
\end{aligned}
$$

for $x, y \in \{0,2\}$ and $d = \mathrm{dawn}(\mathrm{XX_t})$.

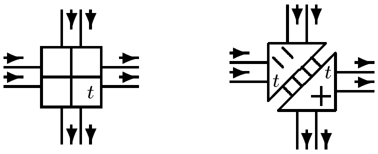

Figure 27: *The simplified version of a double crossing.*

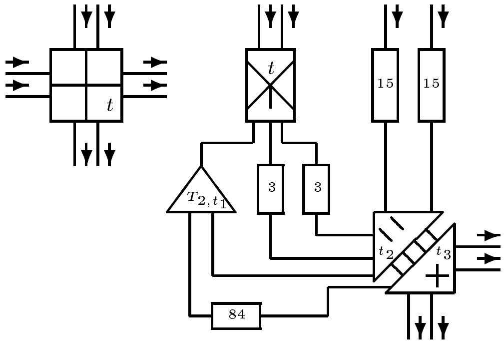

Figure 29: *The double crossing* $\mathrm{XX_t}$ *with* $t_1 := t-12$, $t_2 := t-15$ *and* $t_3 := t-99$.

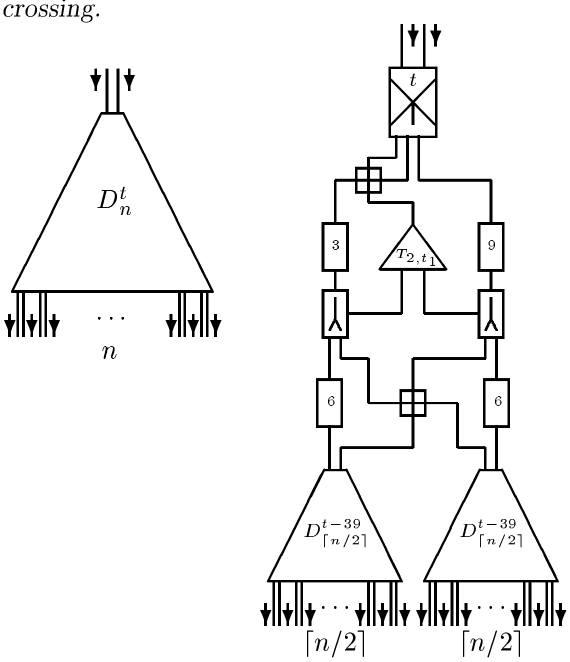

Figure 28: *The complete multiplicator graph* $\mathrm{D_n^t}$ *with* $t_1 := t-18$.

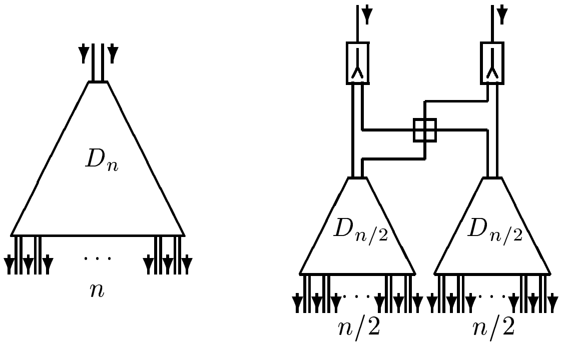

Figure 30: *The Multiplicator subgraph* $\mathrm{D_n^t}$.

5.17 Multiplicator

Combining the duplicator and the crossing it is possible to construct a planar multiplicator of information-pairs representing binar digits.

Figure 30 shows the construction of this subgraph. Again one has to add postponers parallel to the crossing.

We can achieve planarization and eliminate additional inputs and get a complete muliplicator subgraph in Figure 28. We choose $t = T^* - \mathrm{dawn}(\mathrm{D_n^t})$ with $t \geq 39 \log n + 2$.

$$
\begin{aligned}
\mathrm{delay}(\mathrm{D_n^t}) &= 39 \log n \\
B(\mathrm{D_n^t}, \mathrm{T}^*, \mathrm{dawn}(\mathrm{D_n^t}) + \mathrm{x}, \mathrm{dawn}(\mathrm{D_n^t}) + 2 - \mathrm{x}) &= \{[\mathrm{dawn}(\mathrm{D_n^t}) + 39 \log \mathrm{n} + \mathrm{x}, \\
&\qquad \mathrm{dawn}(\mathrm{D_n^t}) + 39 \log \mathrm{n} + 2 - \mathrm{x}]^n\}
\end{aligned}
$$

for $x \in \{0,2\}$.

5.18 Or–Graph

This graph will end our later construction. Here , we choose the deadline such that $\mathrm{dawn}(\mathrm{Or}^3_{b_x,b_y,b_z}) = T^* - 4$ resp. $\mathrm{dawn}(\mathrm{Or}^2_{b_x,b_y}) = T^* - 4$. It is necessary that at least one input has delaied through its path. If not, some nodes of this graph stay uninformed.

Figure 31 shows the realization of an $\mathrm{Or}^3_{0,1,0}$ –graph: $x \vee \overline{y} \vee z$ and an $\mathrm{Or}^2_{1,0}$ –graph: $\overline{x} \vee y$.

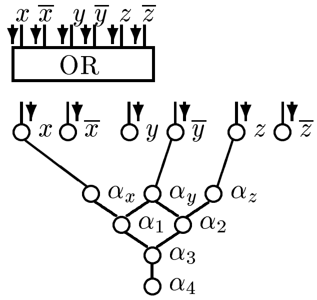

Figure 31: *The OR–subgraph.*

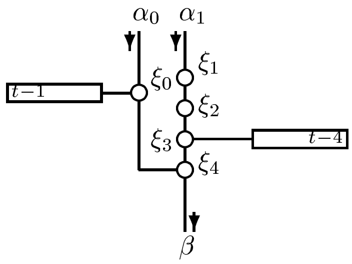

Figure 32: *The bipartite separator.*

We assume that all input nodes x , $\overline{x}$, y , $\overline{y}$, z , $\overline{z}$ are informed as early as possible and that this can only happen if one of each pair $(u, \overline{u})$ with $u \in \{x, y, z\}$ is informed at $\mathrm{dawn}(Or)$ and the other two rounds later.

So, it is only possible to inform node α_3 in round $\mathrm{dawn}(Or) + 3$ if there is at least one node wired with α_x or α_y or α_z is informed at $\mathrm{dawn}(Or)$. One can see that in this case all other nodes of the OR–graph can be informed before the deadline T^* .

If all nodes connected to α_x or α_y or α_z are informed at round $\mathrm{dawn}(Or)+2$ there is no broadcasting schedule for this given deadline.

For $t^x_0, t^x_1, t^y_0, t^y_1, t^z_0, t^z_1 \in \{\mathrm{dawn}(Or), \mathrm{dawn}(Or) + 2\}$

$$\mathcal{B}(\mathrm{Or}^3_{b_x,b_y,b_z}, T^*, t^x_0, t^x_1, t^y_0, t^y_1, t^z_0, t^z_1) = \begin{cases} \emptyset & \text{if } t^x_{b_x} = \mathrm{dawn}(\mathrm{Or}^3_{b_x,b_y,b_z}) \vee t^y_{b_y} = \mathrm{dawn}(\mathrm{Or}^3_{b_x,b_y,b_z}) \\ & \quad \vee t^z_{b_z} = \mathrm{dawn}(\mathrm{Or}^3_{b_x,b_y,b_z}) \\ \perp & \text{else} \end{cases}$$

$$B^{\mathrm{Or}^2_{b_x,b_y}}(x_0, x_1, y_0, y_1) = \begin{cases} \emptyset & \text{if } t^x_{b_x} = \mathrm{dawn}(\mathrm{Or}^2_{b_x,b_y}) \vee t^y_{b_y} = \mathrm{dawn}(\mathrm{Or}^2_{b_x,b_y}) \\ \perp & \text{else} \end{cases}$$

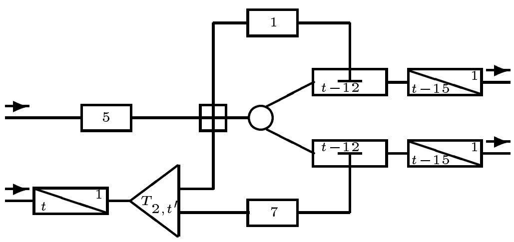

Figure 33: *The bipartite duplicator with* $t' := t - 1$.

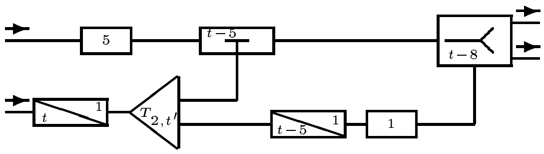

Figure 34: *The bipartite duplicator which is used at binary–unary–converters and unary–binary–converters with* $t' := t - 1$.

5.19 Bipartite Subgraph

Most of the graphs can easily be transfomed into bipartite graphs with the same functionality. One only has to add several postponers and padders. For separators, general duplicators and duplicators which are used at binary–unary–converters and unary–binary–converters little more complicated trasformations are used.

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 the 3DM problem. The main idea is similar to the reduction in the proof of Theorem 1.

Consider the tree $T_{q,t}$ in Figure 4 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 $T_{q,t}$ 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 and call this new graph G'. Let $t := 3\lceil \log q \rceil + 3$ and let w be the the root of $T_{q,t}$. Then, G' with source w has a broadcast schedule of length at most $3 \log q + 3$ 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 crossed by the edges leaving $T_{q,t}$. 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 with less then $\log |V|$ rounds. But the following theorem shows that the problem to find a minimal broadcast is $\mathcal{NP}$-comlete for a deadline $c \cdot \log \mathcal{N}$.

Proof of Theorem 3 Let A, B, C, M be an instance of 3DM with $|A| = q$. The corresponding graph G with a source s consist of 4 levels (see Figure 37): The first level consiting of one initializer that duplicates the number of sources. The second level consisting of some choosers connected with some postponers which simulate the nodes of $V_{A,1}$ in the proof of Theorem 1. The third level consisting of some padders and a choosers which simulate the nodes of V_C, $V_{A,2}$, and $V_{A,3}$. The fourth level consisting of some collector connected with some postponers and a timer which simulate the nodes of V_B and V_C.

More explicitly the graph is built as follows using the subgraphs defined above:

1. Start with an initializer $T_{q,10+3\log q}$ whose root serves also as the source of G.
2. Each element i of A is represented by a subgraph $G_{\alpha,i}$ (see Figure 35) that consists of a chooser and several postponers. The type of the chooser, the number of postponers, and the length of the postponers depends on the number $\alpha_i \leq 3$ of triples r with $i \in r$. The rush time of each subgraph is 3. Connect the outputs of the initializer with the inputs of these subgraphs.
3. Each element r of M is represented by a subgraph $G_{\mu,r}$ that consits of a padder and a chooser. The output of the padder is connected with the input of the chooser.
4. Connect the outputs of subgraph $G_{\alpha,i}$ with the inputs of a subgraph $G_{\mu,r}$ if i is an element of the triple r.
5. Each element j of B (resp. k of C) is represented by a subgraph $G_{\beta,j}$ consisting of some postponers, a collector (see Figure 36), and a timer of length 3 (resp. 2). The number of postponers, the length of the postponers, and the type of collector depend on the number β_i (resp. γ_i) of triples r with $j \in r$. Thus the rush time of each subgraph is 6 (resp. 5).

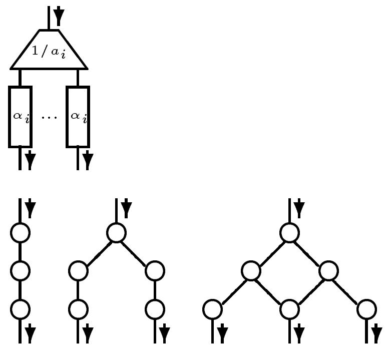

Figure 35: *The 3 different types for $G_{\alpha,i}$ for $\alpha_i = 1,\ 2$ or 3.*

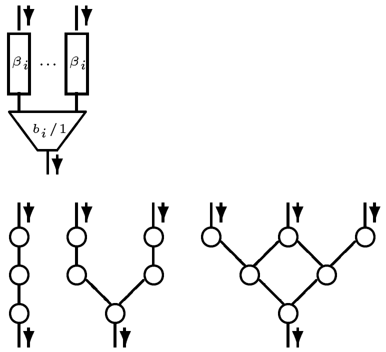

Figure 36: *The 3 different types of the first part of $G_{\beta,j}$ and $G_{\gamma,k}$ for $\beta_i = 1,\ 2$ or 3.*

6. Connect the outputs of a subgraph $G_{\mu,r}$ with an inputs of the subgraphs $G_{\beta,j}$ and $G_{\gamma,k}$ if j and k are elements of the triple r.

The set of sources is chosen as $V_0 := \{s\}$ and the deadline as $T^* := 11 + 3 \cdot \log q$.

Observe that G has maximal degree 3.

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

Proof: Let $M' \subseteq M$ be a matching for A, B, and C. Then it is possible to inform G by the following strategy:

Round $1 \ldots 3\log q - 1$**:** Inform all output nodes of the initializer.

Round $3\log q \ldots 2 + 3\log q$**:** Inform the output nodes of all $G_{\alpha,i}$ that are connected to graphs $G_{\mu,r}$ with $r \in M'$. The other outputs can be informed by round $4 + 3\log q$.

Round $3 + 3\log q \ldots 6 + 3\log q$**:** Inform the output nodes of the $G_{\mu,r}$ graphs that represent elements of M' which are connected to $G_{\beta,j}$. The other outputs of these graph can be informed by round $8 + 3\log q$, and the outputs of other $G_{\mu,r}$ with $r \notin M'$ can be informed by round $10 + 3\log q$.

Round $7 + 3\log q \ldots 11 + 3\log q$**:** inform the graphs $G_{\beta,j}$ and $G_{\gamma,k}$.

Observe that all timers of the initializer and all timers off the padder graphs can be informed within the deadline T^* by this strategy.

For the reverse direction observe that no input of a $G_{\alpha,i}$ graph can be reached by the information before the rush time of the initializer. Thus all subgraphs that represent elements of A, B, C, and M have to be informed within 12 rounds.

The $G_{\alpha,i}$ graphs have the property that only one output can send the information to a $G_{\mu,r}$ graph 3 rounds after the information has reached $G_{\alpha,i}$. The other outputs can send the information one round later. If an information reach a $G_{\mu,r}$ graph at round $3 + 3\log q$ it can send the information to one succsessor graph at round $6 + 3\log q$ and to the other succsessor at round $7 + 3\log q$. But if the information reach $G_{\mu,r}$ at round $4 + 3\log q$ it can send the information to one succsessor at round $8 + 3\log q$ and to the other succsessor at round $9 + 3\log q$ due to the functionality of a padder graph. The minimal number of rounds to inform a $G_{\beta,j}$ graph is 6 and the minimal number of rounds to inform a $G_{\gamma,k}$ graph is 5.

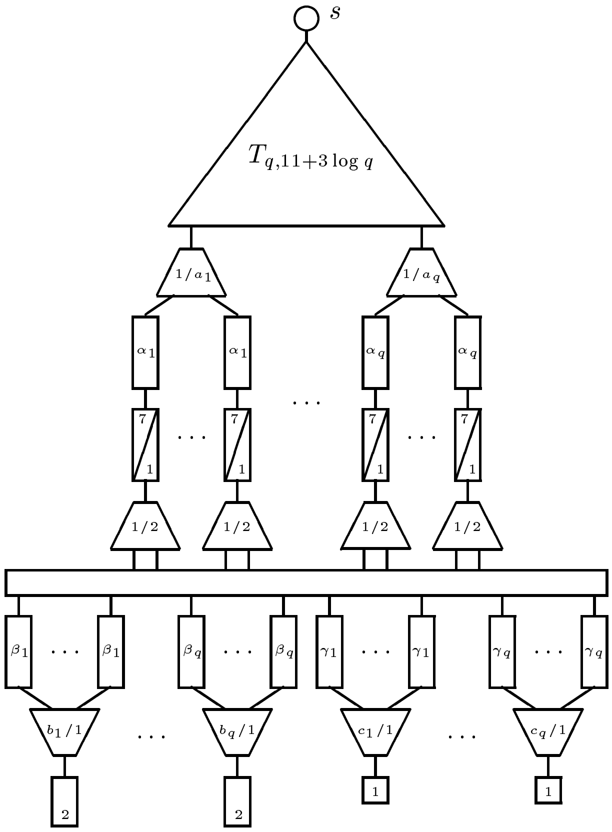

Figure 37: *A broadcastgraph corresponding to an instance of the 3DM problem.*

Thus if we can inform G within $11+3\log q$ rounds, we find q graphs $G_{\mu,r}$ that get the information at round $3+3\log q$ and that are connected to all graphs $G_{\alpha,i}$, $G_{\beta,j}$, and $G_{\gamma,k}$. Thus these graphs represent a matching M' of M. ∎

These proves Theorem 3. ∎

7 Single Source Broadcasting of Planar Graphs is NP–Complete

To achieve planarity in the single source case we construct a direct reduction of 3SAT. Although there are planar versions of 3SAT that remain $\mathcal{NP}$-complete they does not help much in this case because the connections to the source will destroy planarity. A simple exchange of an edge crossing by a planar subgraph with 2 inputs and 2 outputs as in figure 12 does not seem to work.

We have found a way to allow such a replacement under special circumstances, namely if the direction of the information flow over the edges is known in advance and if at most one of the two input edges is used. The first property can easily be achieved for 3SAT, while the second requires special codings. For this purpose we will use the converters discussed in figure 23 and 24.

We reduce a restricted version of the satisfiability problem to the planar SB–problem.

Definition 10 3–bounded 3SAT
A collection C of clauses of length at most 3 over a set U of variables, where each variable or its negation occur in no more than 3 clauses. □

The reduction will use a restricted satisfiability problem. Here, for each variable $x_i \in U$ there are at most 3 clauses in C that contain either x_i or $\overline{x_i}$ and each clause $c \in C$ satisfies $|c| \leq 3$.

This restricted version of 3SAT receives NP–complete.

Proof of Theorem 4:

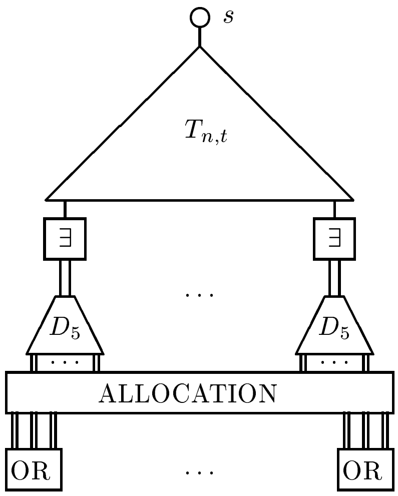

Figure 38: *A planar broadcasting graph corresponding to an instance of 3–bounded 3SAT.*

We reduce a given instance F of 3–bounded 3SAT with clauses $C_1 \dots C_m$ and variables $x_1 \dots x_n$ to a graph consisting of 5 levels (Figure 38).

1. Start with an initializer T_{n,T^*-1} whose root serves also as the source of G.
2. The information enters n parallel guess–graphs. If they are used without artificial delay each of them has two possibilities: It gives out information round tupels $(0,2)$ or $(2,0)$ relative to the dawn of the successing graphs. We interprete these pairs as codings for true and false setting of the corresponding variable.
3. Now we use multiplicators $D_{\alpha_1} \dots D_{\alpha_n}$ to increase the number of binary encodings where α_i is the number of variable x_i or its negations appears in F.

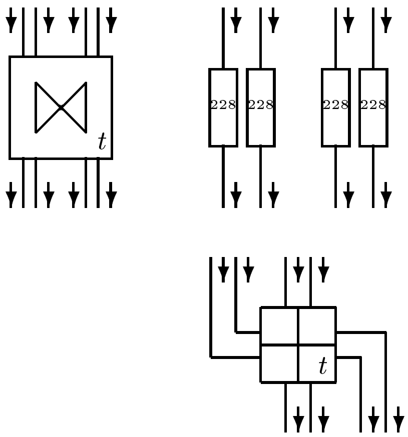

Figure 39: *The allocation subgraph consists of 4 postponers or a double crossing*

4. On this level we have to perform a coding distribution to the equivalents of the or–Clauses. In this network we will use double crossings and postponers to achieve this. Fig. 39 shows both alternatives.

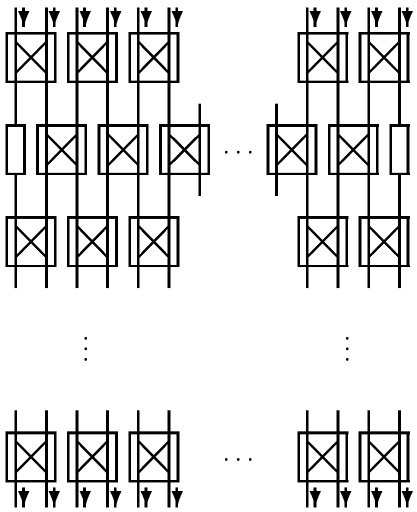

Figure 40: *The allocation network*

These components are combined in an allocation network (see Figure 40). Let d be the number of inputs of this network, then each permutation can be switched in depth $228d$ and size $O(d^2)$.

5. Finally, we connect the output nodes of the allocation network to the OR–graphs. These OR–graph play the part of the clauses. Below we will prove that each OR–graphs needs at least one rushing input if the whole graph is to be finished within T^* rounds.

For the resulting graph we choose a deadline $\mathrm{dawn}(\mathrm{Or}) + 4$. Explicitly this is

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

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

Proof: First the broadcast strategy informs all input nodes of the guess graphs in $3\log n$ rounds.

Then, for a satisfying assigenment $b_1, \ldots b_n \in \{0,1\}$ of variables in U for the clauses C we choose the following strategy for a guess graph $\exists_i$. If $b_i = 1$, then $\exists_i$ informs its outer left successor within $3\log n + 2$ rounds and its right one within $3\log n + 4$ rounds, otherwise the timing of the outputs is switched.

These pairs of binary codes will be multiplied by the following subgraphs D.

If these multiplicators and the following allocation network will be used as described above, each of the Or–graphs will get rushing inputs corresponding to the setting of the variables involved.

Since we have chosen for the variables a satisfiable setting there is at least one relevant input node informed at $3\log n + 3 \cdot 39 + 2 + 228d$ for every Or–graph.

Since we provided for every subgraph a behavior of input rounds as discussed above, the subgraphs informed earlier will be informed totally within these four last rounds like all Or–graphs.

For the reverse direction note:

1. All ghost drivers can be simulated without any delay

   A subgraph in a broadcast scheme starts a ghost driver if information travels from an input node to a precessor of an input node. For a broadcast scheme let S be a subgraph which starts a ghost driver and has no preceding subgraph with this property.

   Because of the construction every input node of a subgraph can be informed until $\mathrm{dawn}(S)+2$ at latest and at earliest at $\mathrm{dawn}(S)$. Among all preceding graphs we optimize the strategy of informing nodes described in the broadcast relations. Since we look at minimal tupels no subgraphs will be informed later. Earlier information of non preceding subgraphs can be simulated by artificial delay on output nodes to those subgraphs.

   For all subgraphs except the separator the round a ghost driver needs a number of rounds for traveling from an input node to a precessor is higher than 3. Here, we can simulate a strategy with ghost drivers by a local strategy without ghost drivers.

   We have combined the subgraphs such that the first separator input node can always be informed at dawn. So, the only relevant direction of a ghost driver is from the first to the second input node. This ghost driver needs at least 3 rounds from input node to the successor of the other one.

   Therefore even for this subgraph a ghost driver can be simulated.

   If these considerations are applied to all other subgraphs we can eliminate ghost drivers from the broadcasting schedule without leaving any nodes uninformed.

2. All subgraphs can be sped up.

   Now we perform a local optimization of the broadcast in each subgraph such that all subgraphs have input and output rounds given by the description of their broadcast relation.

3. Then the output of the guess–graphs define a satisfiable setting for the collection of clauses C.

   If we look now at the output tupels of the guess graph $\exists_i$ an output $(0,2)$ or $(2,0)$ will be multiplied and distributed to the corresponding Or–clauses. $(0,2)$ stands for $x_i = 1$ and $(2,0)$ stands for $x_i = 0$. Since for every Or–clause one input is rushing, the setting $x_1, \ldots, x_n$ satisfies all clauses in C. ∎

This finishes proof of theorem 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 at round 0. In the more general case, a source v may get the information at 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 theoretic decision and optimization problems.

Definition 11 GENERAL BROADCAST–*problem* **[GB]**:
Given a graph $G = (V, E)$ and two partial functions $\sigma, \rho : V \to \mathbb{N}$, decide whether there exists a broadcast schedule $E_1, E_2, \ldots,$ with

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

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

The set of sources V_s are given by the domain of σ.

The GB–problem can be solved similarly to the strategy of Lemma 1. Thus, this problem is also $\mathcal{NP}$–complete. If we restrict the GB–problem to graphs G with maximal degree d we can solve this problem by the following nondeterministic strategy:

Step 0: Generate a new graph $G' := (V, E \cup \{\{v,v\} \mid v \in V_s\}$ with maximal degree $d+1$. Note that every source v of G gets an additional edge at G'.

Step 1: For each node v choose one one edge $(v', v) \in E'$, meaning that v receives the message from its neighbour v' (if $v = v'$ this interpreted as v being a source, does not receive the message from somebody else).

Note that the number of different choices is bounded by

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

where $\delta'(v)$ denote the degree of v in G'.

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 the root.

Step 3: Solve the general broadcast time problem for the trees constructed in step 1. Analysing the time complexity of the strategy in [SCH81] for broadcasting in trees one can show that quadratic time suffices for this task.

The correctness follows from the fact that each broadcasting schedule can be described by a directed forest, in which the edges are labelled by the round the 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 $\rho(v)$. Thus we have shown

Lemma 5 *For a graph $G = (V, E)$ the GB–problem can be solved in time*

$$\mathrm{O}\Big(|V|^2 \cdot \Big(2 \cdot \tfrac{|E|+|V|}{|V|}\Big)^{|V|}\Big) .$$

■

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 above 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 the 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 choises.

Lemma 6 *Let V' denotes the set of nodes of G of degree 1. Then the GB–problem for G can be solved in time*

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

∎

The strategy above can be parallelised in a simple way

Lemma 7 *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 5: 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 41 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**.

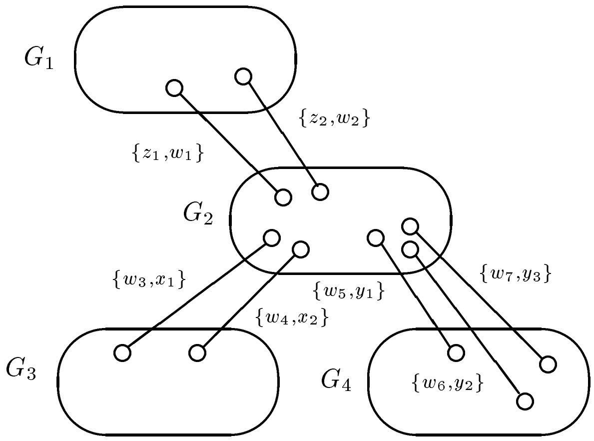

Figure 41: *A node G_2 of an edge decomposition tree and its neighbors.*

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 $\mathrm{cut}(G_i^a)$ as the set of edges of $\mathrm{cut}(G_i)$ with one endpoint in G_i^a. Define

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

To describe a broadcasting schedule $\mathcal{E}$ of G, each edge $\{u, v\}$ of G is labelled by a tupel $(\tau(u, v), r(u, v))$. 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 $\mathrm{cut}(G_i^a)$ it suffices to denote the first round $\boldsymbol{\tau(G_i^a)}$ a node of G_i^a gets the broadcast information and 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 $\mathrm{cut}(G_i^a, G_j^b)$ is used. If no edge in $\mathrm{cut}(G_i^a, G_j^b)$ 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 8 *If $\mathcal{E}$ is a busy broadcast schedule then for all $\{u,v\} \in \mathrm{cut}(G_i^a, G_j^b)$ holds: the numbers $\widehat{\tau}(u,v)$ and $\widehat{\tau}(G_i^a, G_j^b)$ are smaller than $|\mathrm{cut}(G_i^a)| + |V_i^a| - 1 \;\leq\; \kappa + \mu - 1$.*

Proof: Let $G_i' = (V_i^a \cup \mathrm{border}(G_i^a), E_i^a \cup \mathrm{cut}(G_i^a))$ 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 $\kappa + \mu$. Thus 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 $\{u,v\}$. ∎

Let G_i, G_j be neighbouring components, then define a **relative state** between G_i and G_j as a tupel

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

Figure 42 illustrates a complex information flow between a component and its neighbours.

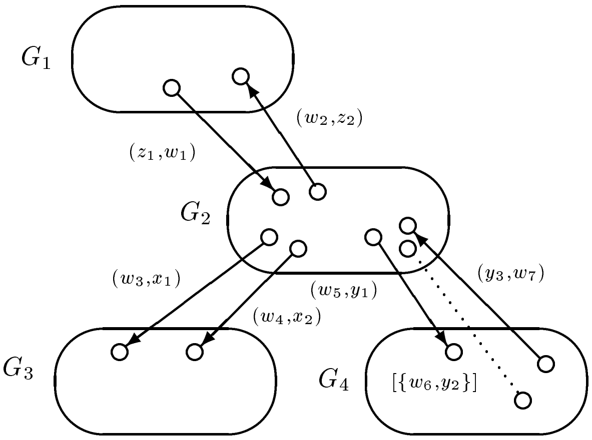

Figure 42: *A possible information flow within a broadcast schedule: from G_2 to other components in one or in both directions, some edges may not be used.*

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 neighbours 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 $\mathrm{cut}(G_i^a, G_j) \neq \emptyset$. 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 tupels $\gamma_{i,j}$ for all neighbouring 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 9 *For a component G_i with cutsize $|\mathrm{cut}(G_i)| \le \kappa_i$, size $|V_i| \le \mu_i$ and $c_i = \mathsf{cc}(G_i)$ 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:
$$\begin{aligned} |\Gamma_i| &\le \prod_{G_i^a \in \mathcal{G}_i} |V| \quad \cdot \prod_{\mathrm{cut}(G_i,G_j)\neq\emptyset} |\,\Gamma_{i,j}\,| \\ &\le |V|^{c_i} \;\cdot\; \prod_{\mathrm{cut}(G_i,G_j)\neq\emptyset} (2(\kappa_i+\mu_i))^{|\mathrm{cut}(G_i,G_j)|} \\ &= |V|^{c_i} \;\cdot\; (2(\kappa_i+\mu_i))^{\sum |\mathrm{cut}(G_i,G_j)|} \;=\; |V|^{c_i} \;\cdot\; (2(\kappa_i+\mu_i))^{|\mathrm{cut}(G_i)|} \\ &= |V|^{c_i} \cdot (2(\kappa_i+\mu_i))^{\kappa_i} \;. \end{aligned}$$
∎

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

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 $\{v, w\} \in \mathrm{cut}(G_i)$ with $w \notin G_i$ and $\widehat{\tau}(v, w) \ge 0$ 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 $E(S_i)$ the set of new edges $\{w^{[v]}, v\}$. We define a local GB-problem with respect to G_i and S_i as follows:

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

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 42 is illustrated in Figure 43.

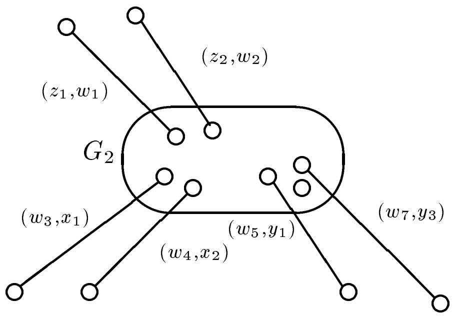

Figure 43: *The minimum deadline of the general broadcast problem for this graph is used to calculate the minimum broadcast time for the graph above.*

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 10 *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*

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

with

$$\begin{aligned}\Gamma_{j,i}(S_i) \;:=\; \{(\ &\left[\tau(G_{j,i}^b)=\tau_k+\hat\tau(G_i^k,G_{j,i}^b)-\hat\tau(G_{j,i}^b,G_i^k)\mid \mathrm{cut}(G_i^k,G_{j,i}^b)\neq\emptyset\right],\\ &\left[(\hat\tau(e)-\hat\tau(G_i^k,G_{j,i}^b)+\hat\tau(G_{j,i}^b,G_i^k),r(e))\mid e\in\mathrm{cut}(G_i^k,G_{j,i}^b)\ \textit{and}\ (\hat\tau(e),r(e))\in\gamma_{i,j}\right]\ )\}\ .\end{aligned}$$

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

If G_i is a leave 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 $< h_i$ 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

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

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 10.

According to Lemma 6 the computation of $\Delta(G_i,S_i)$ requires at most $\mathrm{O}\left((\mu_i+\kappa_i)^2\cdot(2\cdot(|E|+|V|)/|V|)^{\mu_i}\right)$ steps. for each G_i and S_i. From Lemma 9 follows that step 1 can be executed in time

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

The computation of $\Delta(G_i^*,S_{i,0})$ is independent of the remaining structure of G. Note that given S_i $\hat\tau(G_i^a,G_j^b)=\min_{e\in\mathrm{cut}(G_i^a,G_j^b)}\hat\tau(e)$, 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

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

This finishes the proof of Theorem 5 and 6. ∎

By using tree contraction methods the evaluation of the Δ-function can also be done in parallel requiring only a logarithmic number of iterations, to obtain Theorem 7. The details are described in [R91a].

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 8: 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 44 shows a component G_2 which is connected to three other components G_1, G_3, and G_4.

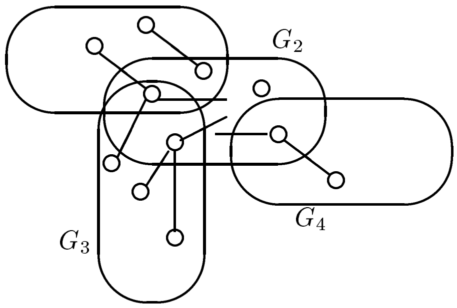

Figure 44: *A node G_2 of a edge decomposition tree and its neighbors.*

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 $\mathrm{cut}(G_i^a)$ as the set of nodes of $V_i^a \cap \mathrm{cut}(G_i)$.

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

For a node $u \in V$ let $\tau(u)$ be the round u gets the information, and for an edge $e \in E$ $\tau(e_l)$ the round when e_l is used. To describe a broadcasting schedule of a graph G, each node u of G is labelled by a vector of rounds $\boldsymbol{\gamma_u} := (\tau(u), \hat{\tau}(e_1), \ldots, \hat{\tau}(e_{\eta(u)}))$, where $e_1, \ldots, e_{\eta(u)}$ denote the edges which are incident to u and $\boldsymbol{\hat{\tau}(e_l)} := \tau(e_l) - \tau(u)$. Note that the values $\hat{\tau}(e_l)$ 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 a 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 $\boldsymbol{\hat{\tau}(u)} := \tau(u) - \tau(G_i^a)$ where $\tau(G_i^a)$ denotes the first round a node of V_i^a receives the information.

Lemma 11 *For a busy broadcast schedule $\hat{\tau}(u)$ 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 $G_i' = (V_i^a \cup \mathrm{border}(G_i^a), E_i^a \cup \{\{u, v\} \in E \mid u \in \mathrm{cut}(G_i^a)\}))$ be the extended component of G_i^a (see Figure 45). 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 getting 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 $\mathrm{cut}(G_i, G_j)$. The **surface** $\boldsymbol{\Gamma_{i,j}}$ of a $\mathrm{cut}(G_i, G_j)$ is the set of all possible states $\gamma_{i,j}$ that may appear in busy broadcast schedule. A **state** $\boldsymbol{S_i}$ of a component G_i is a vector of states γ_u, one for each node of $\mathrm{cut}(G_i)$. The **surface** $\boldsymbol{\Gamma_i}$ of a component G_i is the set of all possible states S_i that may appear in busy schedule.

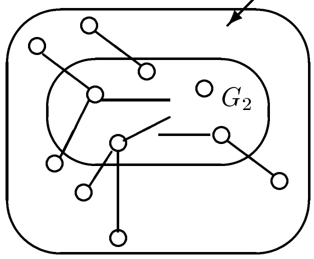

Figure 45: *The extended component of G_i.*

Lemma 12 *For a component G_i with cutsize $|\mathrm{cut}(G_i)| \le \kappa_i$, size $|V_i| \le \mu_i$ and $c_i = \mathtt{cc}(G_i)$ 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 neighbours G_i and G_j is a vector

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

The **surface** $\boldsymbol{\overline{\Gamma}_i}$ of $\mathrm{cut}(G_i)$ is the set of all possible states $\overline{\gamma}_i$ that may appear in busy broadcast schedule. Note that $|\overline{\Gamma}_i| = |\Gamma_i|$. So

$$\begin{aligned}
|\Gamma_i| &\le \prod_{G_i^a \in \mathcal{G}_i} |V| \cdot \prod_{u \in \mathrm{cut}(G_i^a)} |\Gamma_u^r| \\
&\le \prod_{G_i^a \in \mathcal{G}_i} |V| \cdot (((d-1)\cdot\kappa_i + \mu_i)\cdot(d+1)^d)^{|\mathrm{cut}(G_i^a)|} \\
&\le |V|^{c_i} \cdot (((d-1)\cdot\kappa_i + \mu_i)\cdot(d+1)^d)^{\sum_{G_i^a \in \mathcal{G}_i} |\mathrm{cut}(G_i^a)|} \\
&= |V|^{c_i} \cdot (((d-1)\cdot\kappa_i + \mu_i)\cdot(d+1)^d)^{|\mathrm{cut}(G_i)|} \\
&\le |V|^{c_i} \cdot (((d-1)\cdot\kappa_i + \mu_i)\cdot(d+1)^d)^{\kappa_i} \ .
\end{aligned}$$

∎

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

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

For each edge $\{v, w\} \in E_i$ with $u \in \mathrm{cut}(G_i)$ 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:

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

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 by Figure 46.

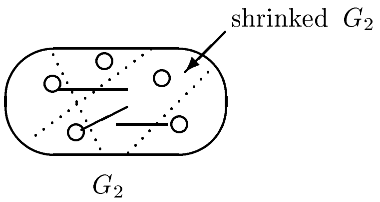

Figure 46: *The minimum deadline of the general broadcast problem of this graph is used to calculate the minimum broadcast time for the graph above.*

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 13 *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*

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

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

The proof is almost identical to the one of Lemma 10. ∎

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 13. According to Lemma 6 the computation of $\Delta(G_i, S_i)$ requires at most $\mathrm{O}\left(((d-2) \cdot \kappa_i + \mu_i)^2 \cdot (d+1)^{\mu_i} \}\right)$ steps. From Lemma 12 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 9.

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.

References

[ALS91] S. Arnborg, J. Lagergren, and D. Seese, *Easy Problems for Tree-Decomposable Graphs* J. Algorithms 12, 1991, 308-340.

[BHLP92] J.-C. Bermond, P. Hell, A. Liestman, and J. Peters, *Broadcasting in Bounded Degree Graphs*, SIAM J. Disc. Math. 5, 1992, 10-24.

[BK91] H. Bodlaender, T. Kloks, *Better Algorithms for the Pathwidth and Treewidth of Graphs,* Proc. 18 ICALP, 1991, 544-555.

[DF86] M. Dyer and A. Frieze, *Planar 3DM Is $\mathcal{NP}$ -Complete*, J. Algorithms 7, 1986, 174-184.

[GJ79] M. Garey and D. Johnson, *Computers and Intractability, A Guide To the Theory of $\mathcal{NP}$ -Completeness*, Freeman 1979.

[H90] W. Hohberg, *The Decomposition of Graphs into k –Connected Components for Arbitrary k ,* Technical Report, TH Darmstadt, 1990.

[HHL88] S. Hedetniemi, S. Hedetniemi, and A. Liestman, *A Survey of Gossiping and Broadcasting in Communication Networks*, Networks 18, 1988, 319-349.

[HJM90] J. Hromkovič, C.–D. Jeschke, and B. Moinien, *Optimal Algorithms for Dissemination of Information in Some Interconnection Networks*, Proc. 15. MFCS, 1990, 337-346.

[HR89] W. Hohberg, R. Reischuk, *Decomposition of Graphs – A Uniform Approach for the Design of Fast Sequential and Parallel Algorithms on Graphs,* Technical Report, TH Darmstadt, 1989.

[L90] J. Lagergren, *Efficient Parallel Algorithms for Tree-Decomposition and Related Problems,* Proc. 31. FoCS, 1990, 173-182.

[LP88] A. Liestman and J. Peters, *Broadcast Networks of Bounded Degree*, SIAM J. Disc. Math. 4, 1988, 531-540.

[M92] M. Middendorf, *Minimum Broadcast Time is $\mathcal{NP}$ –complete for 3–regular planar graphs and deadline 2*, Technical Report, Universität Karlsruhe, 1992.

[R91a] R. Reischuk, *An Algebraic Divide–and–Conquer Approach to Design Highly Parallel Solution Strategies for Optimization Problems on Graphs*, Technical Report, TH Darmstadt, 1991.

[R91b] R. Reischuk, *Graph Theoretical Methods for the Design of Parallel Algorithms*, Proc. 8. FCT, 1991, 61-67.

[RS86] N. Robertson, P. Seymour, *Graph Minors II. Algorithmic Aspects of Tree-Width*, J. Alg. 7, 1986, 309-322.

[SCH81] P. Slater, E. Cockayne, and S. Hedetniemi, *Information Dissemination in Trees*, SIAM J. Comput. 10, 1981, 692-701.