Recognition of Simple Triangle Graphs

by
Udo Hoffmann

Master's thesis in Mathematics

submitted to
Faculty of Mathematics, Computer Science and Natural Sciences
at RWTH Aachen University
in March 2012

written at
Lehrstuhl für Informatik 1
Supervised by
Priv.-Doz. Dr. Walter Unger
Univ.-Prof. Dr. Eberhard Triesch

Acknowledgement

I wish to thank Walter Unger for his excellent care and supervision of this work.

In addition, I would like to thank Eberhard Triesch and Alfred Wagner for the possibility to write this thesis (officially) outside of my field.

Contents

§ 1 Introduction

A lot of problems are difficult to solve algorithmically on graphs. To get efficient algorithms in the field, the problems are studied on special graph classes. Some of these classes are restricted *intersection graphs*. Each vertex of an intersection graph corresponds to a set. Two vertices are adjacent if and only if their sets intersect. Every graph can be represented as an intersection graph by choosing one distinct element for every edge, and placing it in its vertices' set. Therefore, the sets are often given by a geometrical forms, for example disks in the plane (*disk graph*), disks of the same radius in the plane (*unit disk graphs*), chords within a circle (*circle graph*), intervals on a line (*interval graph*), intervals of the same length (*unit interval graphs*), or polygons with corners attached to two parallel lines. In the last case trapezoids (*trapezoid graphs*), triangles (*triangle* and *simple triangle graphs*), as well as lines (*permutation graphs*), are known.

The intersection models, described above, have different advantages and applications. One advantage they have in common is the memory required to represent the graph. Most of the graphs can be stored in a memory that grows linear in the number of vertices of the graph. The memory usage of a normal graph representation is larger, because of the edges, which may grow quadratic in the number of vertices. For example for trapezoid graphs can be represented by storing only the order of the corners on the two lines or for disk graphs only the coordinates of the centre and the radius has to be memorised.

Some of these structures are not primarily calculated from a graph, but occur in real world applications. For example an application of interval graphs is resource allocation, see [BNBYF$^+$01]. In a computer program variables have a certain lifespan – from the assignment of their first value to their last access. Knowing these times, it is possible to determine the memory usage of the program. Considering a line as time sequence, the lifespan of a variable is an interval. The maximal memory that is used at one time is given by the maximal set of intervals that intersect at one point in time. This corresponds to the largest set of pairwise adjacent vertices of the graph – its largest *clique*. The problem of calculating the largest clique of a graph is in general NP hard. Any way, with the help of the interval representation, in form of an ordered list of the starting and endpoints, it is possible to calculate the largest clique in time linear to the number of vertices, see [Gol04]. This is done by scanning the line, i.e. look upon the next starting or endpoint of an interval and keeping track of the number of opened, but not yet closed, intervals. The maximal number is the clique size of the graph. Moreover, the same idea leads to a colouring of the graph,

i.e. the assignment of a colour to each vertex, such that two adjacent vertices do not have the same colour. This is archived by assigning the smallest number $c \in \mathbb{N}$, that is not currently used by an unclosed interval, to an opening interval. The number of the used colours will be identical to clique size, which is a lower bound in general graphs. Hence this colouring is optimal.

Unit disk graphs model wireless broadcasting in a flat area, see [HS95]. Each centre of the circle represents one antenna, the radius corresponds to its operating range. For several NP-hard problems, such as the colouring and independent set, there are approximation algorithms, that guarantee a better approximation on unit disk graphs than on general graphs, see [MBHI$^+$95]. Unlike in the case of interval and simple triangle graphs, the representation as disks does not yield to efficient, exact algorithms for all mentioned problems. While the clique problem is still solvable in polynomial time, the colouring and independent set problem are NP-hard, see [CCJ90] for clique and independent set and [GSW98] for colouring. The linear arrangement, mentioned in the algorithms for calculating the maximum clique and the colouring of interval graphs, is missing to circle and disk graphs. The graph class, formalising this property, are the *cocomparability graphs*. Cocomparability graphs can be characterised as intersection graph of curves between two parallel lines, see [GRU83]. However, the algorithmically more useful definition uses partial orders. The intersection graph representations ,on one or between two lines, give the idea of considering the vertices of the graph in an order, e.g. "lie on the left" can be seen as smaller and "lie on the left" as larger while an intersection leads to an edge that marks the incomparability of the vertices. The partial order on the vertices can be calculated efficiently, see [Gol04]. Except disk, unit disk and circle graphs all mentioned graph classes are cocomparability graphs. The last facts are given in [BS$^+$99], which is a comprehensive reference book that gives a good overview on several graph classes. An even more comprehensive, interactive version can be found in [dR$^+$].

The graph class with the closest relation to cocomparability graphs are permutation graphs, since their complement is also a cocomparability graph, i.e. a permutation graph is a *comparability graph*. This can be seen by reversing the order of the points on one of the two lines. This leads to permutation representation of the complement. Permutation graphs have their name from the following construction. Given a permutation $\pi := [\pi(1), \pi(2), \ldots, \pi(n)]$, a described intersection model can be obtained by drawing points on two parallel lines and connecting the i-th point on the first line with the $\pi(i)$-th point on the second one by a straight line. The intersection graph

of these lines is given by

$$v_i v_j \text{ is an edge } \Leftrightarrow (i-j)(\pi(i)-\pi(j)) < 0,$$

i.e. two vertices are adjacent if and only if the order of the connected points is different on both lines.
An application of permutation graphs is the following, see [Gol04]. Box wagons are appended on a locomotive. To save time, the train driver does not want to drop a wagon from the middle of the train at every designation. So the wagons are sorted using parallel rails. The wagons come from one side on a ramification of the one rail into several ones. The workers have to distribute the wagons onto the rails, such that the train driver can take the first wagon of each rail on the other side until he has a train with sorted wagons, see Figure 1. So every rail is a queue. Now the question appears, how many rails the workers have to reserve for the sorting and where to put which wagon. This problem can be modelled using a permutation graph. Let $1, \ldots, n$ be the wagons in correct order and let $\pi = [\pi(1), \ldots, \pi(n)]$ be the permutation giving order the wagons are before sorting. In every partition of the wagons onto the rails, that leads to the correct order, the order of the wagons on one of the parallel rails does not change during the process. So the wagons on each rail are an independent set in the permutation graph of π, and the partition is a colouring. On the other hand distributing the wagons according to a colouring gives a partition correctly ordered chains. So the minimal wagon, that is not connected to a train yet, is on the first position of a rail and can be attached to the train. Hence, every colouring leads to a sorted train. The wanted optimal colouring of a permutation graph can be computed in $O(n \log n)$ by using the order of the vertices, again, see [Gol04].

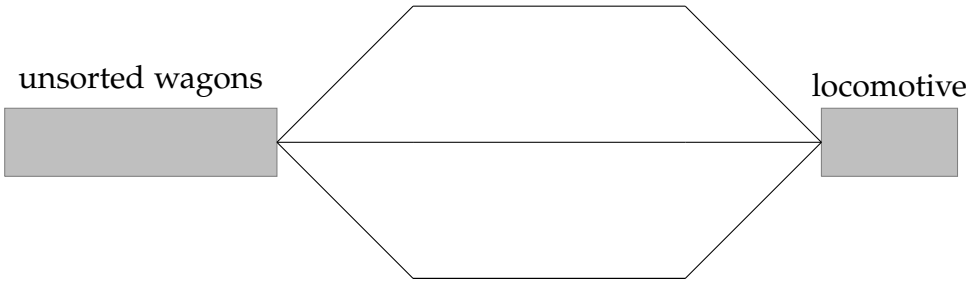


Figure 1: Sorting wagons on three parallel rails.

In the following the graphs of interest are simple triangle graphs, intersection graphs of triangles between two lines, where one corner is attached to the first line and two

corners are attached to the second one, see Figure 2. In contrast, in triangle graphs the line two corners of a triangle is connected to may differ.

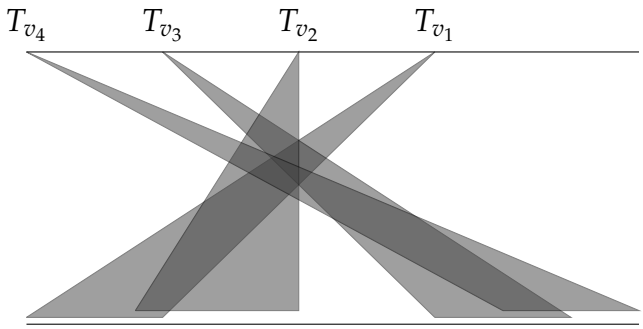


Figure 2: A simple triangle representation.

Algorithms for the in general NP-hard problems, that can be solved efficiently on this graph class, are usually based on cocomparability properties. One of these problems is the independent set problem, i.e. finding a set of triangles of maximal cardinality, such that each two triangles do not intersect. An algorithm for this problem is suggested in the following. First tag every triangle without a triangle that lies entirely on its left with a 0. The now tagged triangles intersect pairwise (they form a clique in the graph). Otherwise, one would lie entirely on the left of the other only would be tagged with 0. So only one of these triangles can be in a maximal independent set. Repeating this procedure for the untagged triangles with $1, 2, \ldots$ give a partition into sets, that are given by the triangles label. In an independent set only one triangle from each set can be contained, so if the largest label is k then there can be at most $k + 1$ pairwise not intersecting triangles. By construction every triangle labelled with i has triangle on its left, that is labelled with $i - 1$. Choosing these triangles iteratively leads to an independent set of cardinality $k + 1$ – the maximal possible size.

Beside that, most of the graph classes mentioned are cocomparability graphs, there are different inclusions of the graph classes. Every simple triangle graph is a triangle graph, too. Every triangle graph is a trapezoid graph, which can be shown by replacing the one corner on one line with a small interval, such that none of the corners of other triangles are in this new interval. With the same argument permutation graphs are simple triangle graphs, hence also triangle and trapezoid graphs. Arranging disks over an interval representation, such that the centres of the disks lie

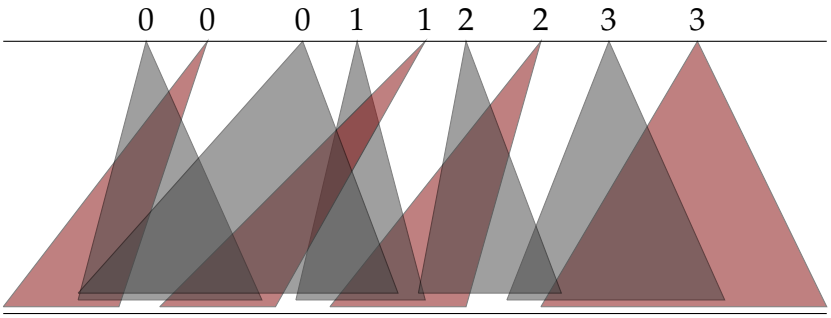


Figure 3: Independent set of a simple triangle graph.

on the line, and choosing the radius equal to the length of the interval, a disk representation of an interval graph is obtained. These inclusions give the possibility to use properties of larger graph classes on problems of the smaller class. An inclusion graph of the mentioned classes is shown in Figure 4.

Owing to the application of unit disk graphs, as antenna positions, and interval graphs, as a program sequence, a representation as intersection graph is already given. If this is not the case, the problem of testing the membership to a class and subsequently determining a representation occurs. For the class of simple triangle graphs this recognition problem will be the issue of this work. For every graph class Figure 4 except triangle and simple triangle, disk and unit disk graphs an efficient algorithm that calculates a representation is known, see [BS$^+$99] for further references. Disk and unit disk graphs are NP-complete, see [BK96], while triangle graph recognition is NP-complete, see [Mer11]. The recognition of simple triangle graphs is the sole open recognition problem among the graph classes mentioned in Figure 4.

For interval and permutation graphs characterising properties, that lead to a recognition algorithm, have already been mentioned.
A graph is an interval graph if and only if its maximal cliques can be ordered, such that every vertex appears only in consecutive cliques. Another characterisation will be discussed in Section 1.2.
As mentioned before the complement of a permutation graph is, as well, a permutation graph and hence a comparability graph. On the other hand, if a graph is a comparability and cocomparability graph, then it is a permutation graph. The two partial orders can be combined to one total order of the vertices. Reversing the partial order that compares adjacent vertices, i.e. the one obtained because G is a comparability graph, and uniting it with the other partial order again leads to an-

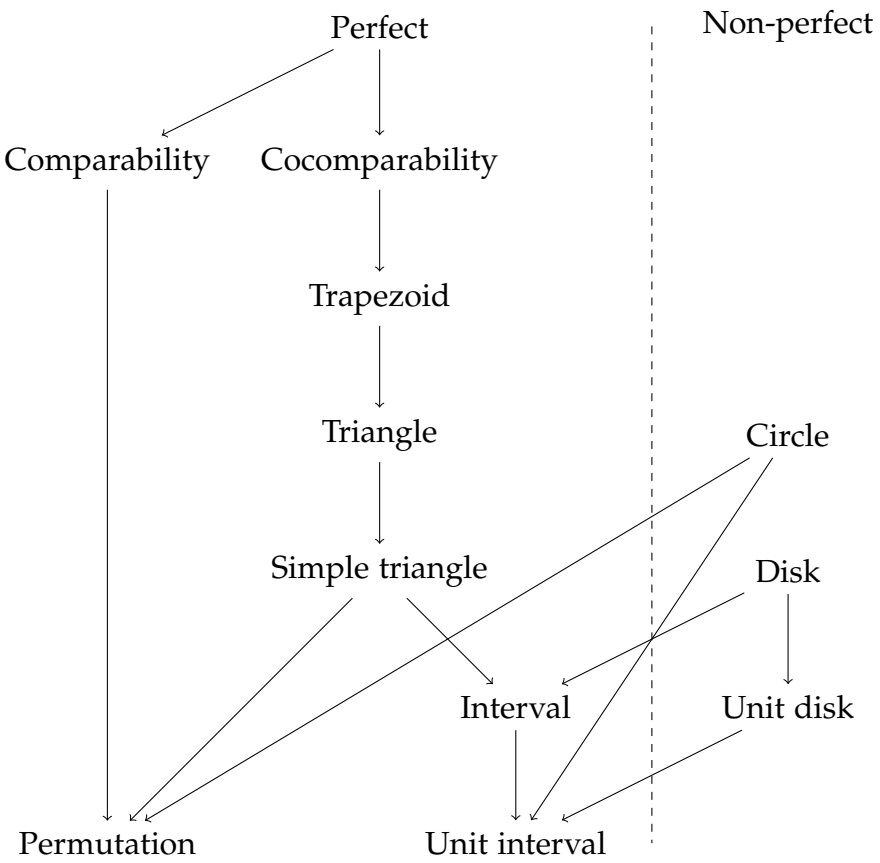


Figure 4: Inclusions of graph classes.

other total order. The two vertices are reversed in the two total orders iff they are adjacent – the property of the intersection model – see [Gol04].
The proof of the correctness of this algorithm proves the fact, that any partial order of the vertices can be used to build a permutation representation. This holds for trapezoid and interval graphs, too.

For triangle graphs such a simple criterion is not on-hand. The orientation of one triangle, normal or upside down, can lead, together with the orientation of another triangle, to restricted options in the construction of the whole triangle representation. In a simple triangle graph this choice of the orientation is not available. A method to detect other restrictions to the construction, that exist on simple triangle graphs, is given in Chapter 4.

In the following chapter some simple geometric constructions of simple triangle graphs, based on interval and trapezoid representations, are studied. These give some examples of simple triangle graphs and two algorithms to calculate simple triangle representations of two subclasses of trapezoid graphs.

In the third chapter an algorithmic method called modular decomposition is introduced to give the basis for an algorithm that determines partial orders on (co)comparablity graphs. With this algorithm in mind, the difference between two partial orders on the same graph occurs very structured, due to the possibility to transform one partial order into another one step by step – without changing the intersection graph. This led to the idea to find a possibility to reconstruct these steps on a simple triangle representation. In this way it is shown, that every partial order on a simple triangle graph is induced by a simple triangle representation.

Finally, in the fourth chapter an algorithm that determines a simple triangle representation from a total order on the top corners of the triangles is introduced. During the process a sufficient condition on this order developed. This leads to an algorithm, that calculates such an order, based on these conditions.

In the following section the notation, that is used in this work, is introduced.

1.1 Notations and basic graph theory

An *(undirected) graph* $G = (V(G), E(G))$ consists of a set of *vertices* $V(G)$ and a set of *edges* $E(G)$ where every element of $E(G)$ is a subset of V of size two. If $\{v, w\} \in E(G)$ for two vertices v and w then v is *adjacent* to w. The set of all adjacent vertices of

$v \in V$ is called the *neighbourhood* of v and is denoted by $N(v)$. If a vertex v is contained in an edge e then v and e are incident.

The *complement* of G, which is denoted by $\overline{G}$, is the graph $\overline{G} = (V(G), \overline{E(G)})$ with

$$\overline{E(G)} := \{e \subseteq V(G) | \ |e| = 2, \ e \notin E(G)\},$$

i.e. v and w are adjacent in $\overline{G}$ if and only if they are not adjacent in G.

A graph H is an *induced subgraph* of G if $V(H) \subseteq V(G)$ and

$$E(H) = \{\{v, w\} \in E(G) | v, w \in V(H)\}.$$

The induced graph is denoted by $G[V(H)]$. The notation $G - S$ is used for $G[V(G) \setminus S]$. If a graph G graph does not contain an induced subgraph H, the graph G is called H-free.

Analogously, a set of edges S *spans* a graph

$$G[S] = (\{v | v \in e, e \in S\}, S).$$

In a *directed graph* $\vec{G} = (V(\vec{G}), E(\vec{G}))$ the set of edges is a subset of $(V \times V)$. An edge (v, v) is called *loop*. The directed graphs in this thesis are *(self-)loop-free*. A directed graph with either $(v, w) \in E(\vec{G})$ or $(w, v) \in E(\vec{G})$ is called a *tournament*.

A *path* of *length* l is a sequence of distinct vertices $p_1 \dots p_l$, such that $\{p_i, p_{i+1}\} \in E(G)$ for $i \in \{1, \dots, l-1\}$. For a *(directed)* path (in a directed graph) $(p_i, p_{i+1}) \in E(\vec{G})$ is requested.

A sequence of distinct vertices $c_1 \dots c_l$ is called a *cycle* of length l if $\{c_i, c_{i+1}\} \in E(G)$ for $i \in \{1, \dots, n-1\}$ and $\{c_n, c_1\} \in E(G)$. Analogously, a sequence with $(c_i, c_{i+1}) \in E(\vec{G})$ for $i \in \{1, \dots, n-1\}$ and $(c_n, c_1) \in E(\vec{G})$ in a directed graph is a *directed cycle*.

A directed graph that contains no cycle is called *acyclic*. An acyclic tournament is called *transitive*.

A graph is called *connected* if there exists a path from every vertex to ever other vertex. A directed graph is called *strongly connected* if there exists a directed path from every vertex to every other vertex. A *(strongly) connected component* of a (directed) graph is a maximal (strongly) connected subgraph.

If $\{v, w\} \in E(G)$ for every $v \neq w$ in a set $C \subseteq V(G)$, then C is called a *clique* of size $|C|$. A graph where $V(G)$ is a clique is referred to as *complete* graph. The size of the largest clique of a graph is denoted with $\omega(G)$.

Let $V_1, \ldots, V_n$ be sets. The *intersection graph* of $V_1, \ldots, V_n$ is the graph

$$G = \left(\{V_1, \ldots, V_n\}, \left\{\{V_i, V_j\} \,|i \neq j,\ V_i \cap V_j \neq \emptyset\right\}\right),$$

i.e. V_i and V_j are adjacent if and only if their intersection is not empty.

The notation $v \sim w$ means the vertices v and w are adjacent. In terms of triangle graphs the intersection of two triangles is denoted by $T_v \sim T_w$.

Let L_1 and L_2 be two parallel lines and $T_1, \ldots, T_n$ triangles, such that each triangle has one corner on one line and two on the other one. Such a figure is called a *triangle representation* and its intersection graph a *triangle graph*. If every triangle has one corner on L_1 and two corners on L_2, the representation is called *simple triangle representation*, see Figure 2. A graph is called *simple triangle graph* if it admits a simple triangle representation.

To keep the difference between a vertex and its triangle the triangle of the vertex v is denoted with T_v. Since every triangle in a simple triangle graph is determined by its coordinates on L_1 and L_2, a triangle is written as $T_v = (t, l, r)$, where t is the coordinate of the corner on L_1, l the coordinate of the left corner on L_2 and r of the right corner on L_2.

1.2 Related graph classes

In this section some graph classes that are related to simple triangle graphs are introduced.

A *binary relation* R on a set S is a subset of $S \times S$. For $(a, b) \in R$ the notation aRb is used.

Definition 1.1
Let S be a set. A *non-strict partial order* on S is a binary relation $\preccurlyeq$, such that ...

a.) $\ldots a \preccurlyeq a$ (reflexivity),

b.) $\ldots a \preccurlyeq b$ and $b \preccurlyeq a$ implies $a = b$ (antisymmetry),

c.) $\ldots a \preccurlyeq b$ and $b \preccurlyeq c$ implies $a \preccurlyeq c$ (transitivity),

for all $a, b, c \in S$. If either $a \preccurlyeq b$ or $b \preccurlyeq a$ for every $a \neq b$, then $\preccurlyeq$ is called a *total* or *linear* order.

If 1.1.a is changed to

$$a \not\preccurlyeq a \quad \forall a \in S$$

this is the definition of a *strict* partial order. The last property is called *irreflexivity*. For a non-strict partial order $\preccurlyeq$ the symbol $\prec$ denotes its strict partial order, respectively for similar symbols accordingly. Two elements $a, b \in S$ are *comparable* if $a \prec b$ or $b \prec a$.

A totally ordered subset of S is called a $\prec$-*chain*, whereas a set of pairwise incomparable elements is referred to as *antichain*.

Binary relations have a close relationship to directed graphs. Let R be an irreflexive relation, then $\vec{G} = (S, R)$ is a directed graph. Properties of the relation can be translated to properties of the graph, see for example the following theorem.

Theorem 1.2 (Topological sorting,[CLR00])
Let R be an irreflexive relation on S.
There is a total order $<$, such that $R \subseteq <$ if and only if $G := (S, R)$ is acyclic.

This leads, with the following observation, to the fact that every strict and non-strict partial order can be extended to a total order:

Observation 1.3
Let $\prec$ be a strict partial order on V, then $G = (V, \prec)$ is acyclic.

Partial orders are the main structure used in this work.

Definition 1.4
A graph G is called a *comparability graph* or a *transitive orientable graph* if there exists a strict partial order on the vertices of G, such that $\{v_i, v_j\} \in E(G)$ iff v and w are comparable.

The complement $\overline{G}$ of a comparability graph G is called a *cocomparability graph* or an *incomparability graph*.

Using a non-strict partial order in the definition above allows loops in the resulting graph.

Triangle and simple triangle graphs are cocomparability graphs, see [BS$^+$99]. One partial order for an triangle graph can be defined using a triangle representation and using $v \prec w$ iff the triangle T_v lies entirely on the left of the triangle T_w.

Since loops are precluded in the definition of intersection graphs, it is useful to use strict partial orders. Since strict and non-strict partial orders are both used as partial orders in different literature, in the following a partial order is a strict partial order.

Since $\prec$ can be replaced by $\succ$ another order is given by the reversed partial order. Therefore, a partial order is not uniquely defined by the graph. Nevertheless, a graph that admits only two partial orders – one the inverse of the other – is called *uniquely orientable*.

An example of a cocomparability graph with $n!$ different partial orders is the complement of the complete graph on n vertices $\overline{K_n}$.

A good intuitive impression of cocomparability graphs is given by the following lemma.

Lemma 1.5 ([Gol04])
Let G be a graph and $\prec$ a partial order $V(G)$, such that G is a cocomparability graph. Let $v_1, v_2, c \in V(G)$ and $v_1 \prec c \prec v_2$. Let $v_1 = p_1 \dots p_k = v_2$ a path from v_1 to v_2. Then there exists $i \in \{1, \dots, k\}$, such that $\{c, p_i\} \in E(G)$.

Proof. Let $v_1 = p_1 \dots p_k = v_2$ be a path from v_1 to v_2. Let p_i be the vertex on the path with the largest i, such that $p_i \prec c$. Hence $p_{i+1} \not\prec c$. Assume $p_{i+1} \succ c$. Then $p_i \prec c \prec p_{i+1}$ implies that $\{p_i, p_{i+1}\} \notin E(G)$, a contradiction. So p_{i+1} and c are adjacent. □

For the intersection graphs on one or between two lines this can be visualised easily, see Figure 5.

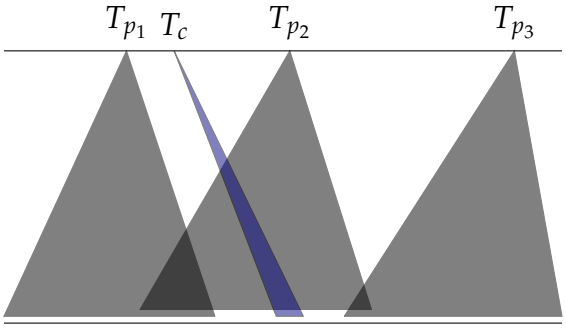


Figure 5: Lemma 1.5

The triangle T_c blocks every path from T_3 to T_1, hence the path from p_1 to p_3 has a vertex adjacent to c.

Lemma 1.6
Let G be a cocomparability graph with partial order $\prec$ and $a, b \in V(G)$ and $a \sim b$. Then the graph obtained by replacing a and b by one vertex c, such that $N(c) = (N(a) \cup N(b)) \setminus \{a, b\}$, is a cocomparability graph.

Proof. Let $\prec_1$ be the partial order of the cocomparability graph $G - a$ and $\prec_2$ the partial order of $G - b$. Replacing b resp. a in $\prec_1$ and $\prec_2$ gives two partial orders $\prec_1'$ and $\prec_2'$. Their intersection is a partial order for the cocomparability graph described above, see Figure 5. □

More theory of partial orders will be introduced and used in section 3.

The symbol $\prec$ is used for partial orders whereas $\ll_1$ and $\ll_2$ denote the orders on the lines L_1 and L_2, e.g. for triangles T_a and T_b we have $T_a \prec T_b$ if and only if $T_a \ll_1 T_b$ and $T_a \ll_2 T_b$. The symbol $\ll_2$ is also used for the order of the corners, for example $r_a \ll l_b$ is equivalent to $T_a \ll_2 T_b$ and implies $r_a \ll_2 r_b$ since $l_b \ll_2 r_b$ is always assumed.

The exact coordinates of the corners on the lines in a simple triangle representation is not important if the order of corners does not change. In consequence, the coordinates of new positions of corners during the construction of a representation are often not given, but the new order of the corners is. With $prec(a)$ for a corner a of a triangle, the position on the left of a and on the right of every corner, that lies on the left of a (i.e. the *predecessor*) is denoted, see Figure 6. Analogously, the *successor* is predicated with $succ(a)$.

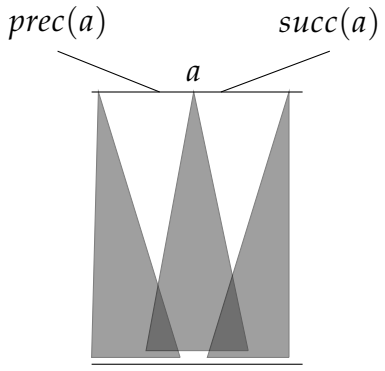


Figure 6: Definition of *prec* and *succ*.

A graph G is called k-colourable if there exists a map $c : V(G) \to \{1, \ldots, k\}$, such that $c(w) = c(v)$ implies $v \not\sim w$. The chromatic number $\chi(G)$ of a graph G is the smallest number k, such that G is k-colourable.

Obviously $\omega(G)$ is a lower bound for $\chi(G)$. A graph G is called *perfect* if $\chi(H) = \omega(H)$ for every induced subgraph H of G.

Theorem 1.7 ([CRSTm06])
A graph G is perfect if and only if G and $\overline{G}$ do not contain an induced cycle C of odd length $l \geq 5$.

The last theorem also shows that G is a perfect graph if and only if $\overline{G}$ is perfect.

In [Dil50] Dilworth has shown that the minimal size of a partitions into $\prec$-chains is equal to the length of the longest antichain of a partial order. Since $\prec$-chains are independent sets in a cocomparability graphs a partition into $\prec$-chains gives a vertex colouring of the graph. A antichain of a cocomparability graph is a clique. So we have $\chi(G) = \omega(G)$ for every cocomparability graph. For the reason that every induced subgraph of a cocomparablity graph is a cocomparability graph we get the following theorem.

Theorem 1.8 ([Gol04],[Dil50])
Comparability and cocomparability graphs are perfect graphs.

Perfect graphs are an important graph class. Several in general difficult problems can be solved more efficiently on perfect graphs, for example graph colouring, which is NP-complete in general, can be solved in polynomial time. Even more efficient are algorithms that uses geometric representations such as a simple triangle representation. Some other related graph classes, with efficient recognition algorithms, are introduced in the following.

Definition 1.9
The intersection graph of intervals on the real line is called an *interval graph*.

By drawing the third point of a triangle above the interval as in Figure 7 it can be shown that an interval graph is a simple triangle graph.

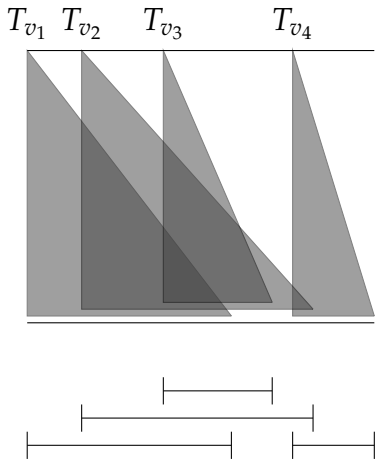


Figure 7: A simple triangle representation of an interval graph.

A more general characterisation for simple triangle representations of interval graphs is given in the end of this chapter.

Definition 1.10
A graph that contains no induced cycle C_l of length $l > 3$ is called *chordal*

Lemma 1.11 ([Gol04, BS$^+$99])
A graph is an interval graph iff it is a chordal cocomparability graph.

This is a very easy and intuitive characterisation of interval graphs which will be used several times.

Cocomparability graphs in general also limit the length of the longest induced cycle.

Lemma 1.12
Let G be a cocomparability graph and let l be the length of the longest induced cycle. Then $l \leq 4$.

Proof. Since C_5 is not perfect it is no cocomparability graph. If C_l for $l \geq 6$ is a cocomparability graph, then C_{l-1} is a cocomparability graph because of Lemma 1.6. □

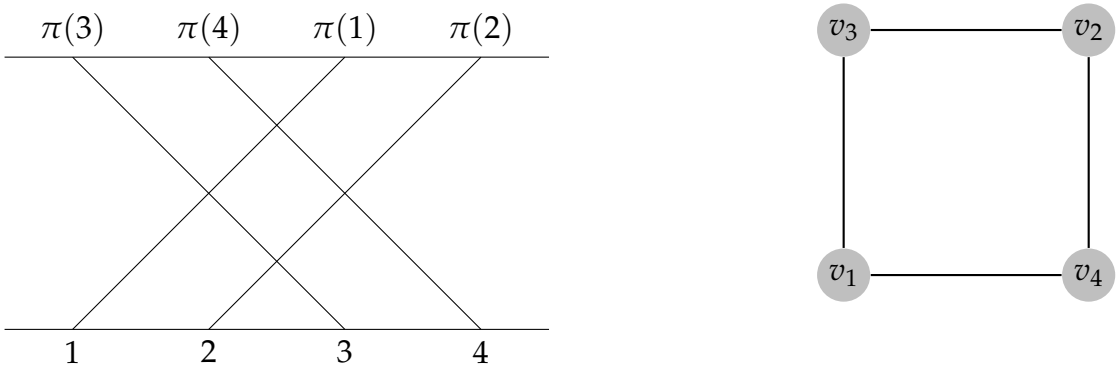
A subclass of simple triangle graphs, where circles of length 4 are allowed are, permutation graphs.

Definition 1.13
Let $\pi \in S_n$ be a permutation. Define

$$G[\pi] := (\{v_1, \ldots, v_n\}, \{(v_i, v_j) | (i-j) \cdot (\pi(i) - \pi(j)) < 0\}).$$

A graph G is a *permutation graph* if it is isomorphic to $G[\pi]$, for a $\pi \in S_{|V(G)|}$.

A permutation graph is the intersection graph of lines in the following manner:



(a) A permutation representation...

(b) ...and its intersection graph.

Figure 8: A permutation graph.

Lemma 1.14 ([Gol04])
A graph G is a permutation graph iff G and $\overline{G}$ are comparability graphs.

Let $G = (V, E)$ be a simple triangle graph. Then there exists $E' \subseteq E$ and $E'' \subseteq E$ with $E = E' \cup E''$, such that (V, E') is a permutation graph and (V, E'') is an interval graph. This is easy to see by removing one side of the triangle which does not lie on the line L_1. The resulting intersection graph of L's is still the same graph as the simple triangle graph. So the simple triangle graph representation is the same as the intersection graph of the following diagram.

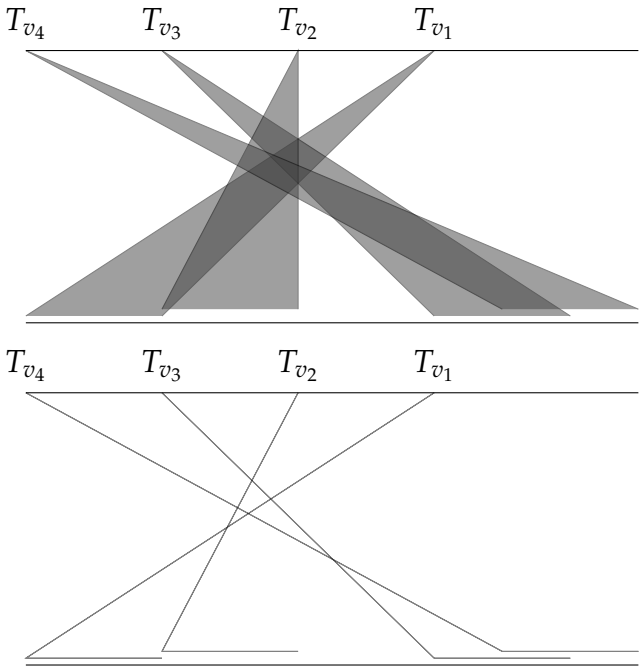


Figure 9: L representation.

By choosing the intervals in simple triangle graphs small, it can be seen that a permutation graph is a simple triangle graph, see Figure 10.

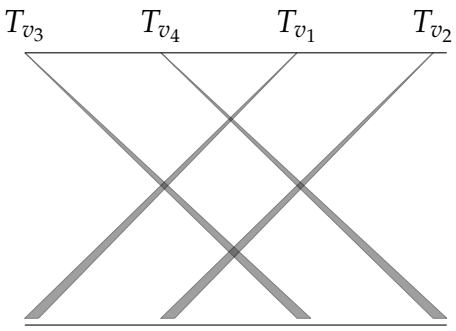


Figure 10: Simple triangle rep. of 8(a).

There is another characterisation of permutation graphs. Since a binary relation on a set M is a subset of $M \times M$, its intersection is defined.

Definition 1.15
Let $\prec$ be a partial order. Define $dim(\prec)$ as the smallest integer, such that there are total orders $<_i$, $i \in \{1, \ldots, dim(\prec)\}$, satisfying

$$M = \bigcap_{i=1}^{dim(\prec)} <_i .$$

The intersection of two total orders $<_1$ and $<_2$ can be visualised by the intersection model of a permutation graph. Drawing the elements according to the total orders on one line each and connecting the same elements by a straight line yields to an permutation representation. E.g. in Figure 8(a) the two total orders are given by $1 <_1 2 <_1 3 <_1 4$ and $3 <_2 4 <_2 1 <_2 2$. The intersection of two lines, one between the element a on L_1 and L_2 and the other between b on L_1 and L_2, implicates $a <_1 b$ and $b <_2 a$ or vice versa. This leads to the following characterisation of permutation graphs.

Theorem 1.16 ([BS$^+$99])
A cocomparability graph is a permutation graph iff the partial order on the complement is the intersection of two linear orders, i.e. $dim(\prec) \leq 2$.

An algorithm, that calculated these two total orders, has been given in the introduction.

Linear orders with $\prec \subset <$ are called *realisers* of $\prec$. Hence the dimension is the cardinality of the smallest set S of realisers with $\bigcap_{s \in S} s = \prec$.

A graph class with a quite similar characterisation is the class of *trapezoid graphs*. This class is defined analogously to triangle graphs, as the intersection of trapezoids between two parallel lines.

Definition 1.17
An *interval order* $\prec$ is a partial order that corresponds to an interval graph, i.e. a partial order on M is an interval order iff there is a map $f : M \to \{I \subset \mathbb{R} |\ I \text{ is interval}\}$, such that $a \prec b$ iff $x_a < x_b$ for all $x_a \in f(a)$ and $x_b \in f(b)$.

The graphical interpretation of an intersection of two interval orders leads exactly to trapezoid graphs: Drawing two interval graphs with the same amount of vertices on two parallel lines and connecting each starting and end point of an interval to the starting and endpoint of a related interval on the other line leads to a set of trapezoids between the two lines. Two trapezoids do not intersect of both intervals of one trapezoid lie on the same side of the intervals of the other trapezoid.

Definition 1.18
Let $\prec$ be a partial order. Define $idim(\prec)$ as the smallest integer, such that there are interval orders $\lhd_i$, $i \in \{1, \ldots, idim(\prec)\}$, satisfying

$$M = \bigcap_{i=1}^{idim(\prec)} \lhd_i \ .$$

Theorem 1.19 ([MS94])
A cocomparability graph is a trapezoid graph if and only if its partial order has interval dimension 2 at most.

Obviously every triangle and simple triangle graph is a trapezoid graph. A simple triangle representation gives a characterisation of simple triangle graphs in terms of intersection of orders.

Observation 1.20
Let G be a simple triangle graph. Then there exists a partial order $\prec$, such that G is its incomparability graph and $\prec$ is the intersection of an interval order and a total order.

Permutation, interval, (co)comparability, and trapezoid graphs can be recognised efficiently ([Gol04, BS$^+$99, MS94]). For (co)comparability graphs the recognition and properties are further discussed in Chapter 3. Interval and trapezoid graphs are used to construct some simple triangle graphs in the next chapter.

§2 Some simple triangle graphs

In this chapter some trapezoid graphs with strong additional structure are studied. Therefore, related intersection models, such as trapezoid and interval representations, are used.

2.1 An interval graph based approach

In this section an interval graph based approach for building a simple triangle graph will be introduced. This means the idea is to fix the induced cycles and use the interval representation of the resulting graph to build an triangle representation. To introduce the idea the graphs in the beginning of this chapter will be restricted to graphs with only one non-chordal cycle.

Theorem 2.1
A cocomparability graph with exactly one non-chordal cycle is an simple triangle graph.

Proof. The idea is to add an edge to the cycle, calculate an interval representation, and use the interval representation to build a simple triangle graph.

Let c_1, c_2, c_3, c_4 be the vertices of the non-chordal cycle. Without loss of generality assume the partial order on the vertices is given $c_1 \prec c_3$ and $c_2 \prec c_4$. At first it is shown that $G' := (V, E \cup \{\{c_1, c_3\}\})$ is a cocomparability graph, i.e. we show that the relation $c_1 \prec c_3$ is not induced by the rest of the partial order. It is sufficient to show that there is no vertex v which complies $c_1 \prec v$ and $v \prec c_3$.

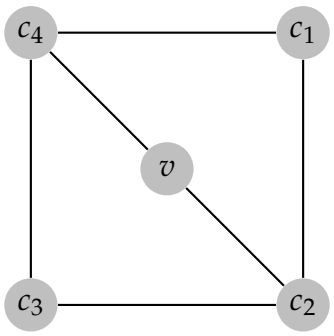


Figure 11: Proof of 2.1.

Since $c_1c_4c_2$ is a path from c_1 to c_3, Lemma 1.5 shows that v is connected to c_4. Analogously, c_2 and v are connected. Hence c_1, c_2, v, c_4 form a non-chordal cycle. This contradicts the uniqueness of the cycle c_1, c_2, c_3, c_4, see Figure 11, and G' is a cocomparability graph.

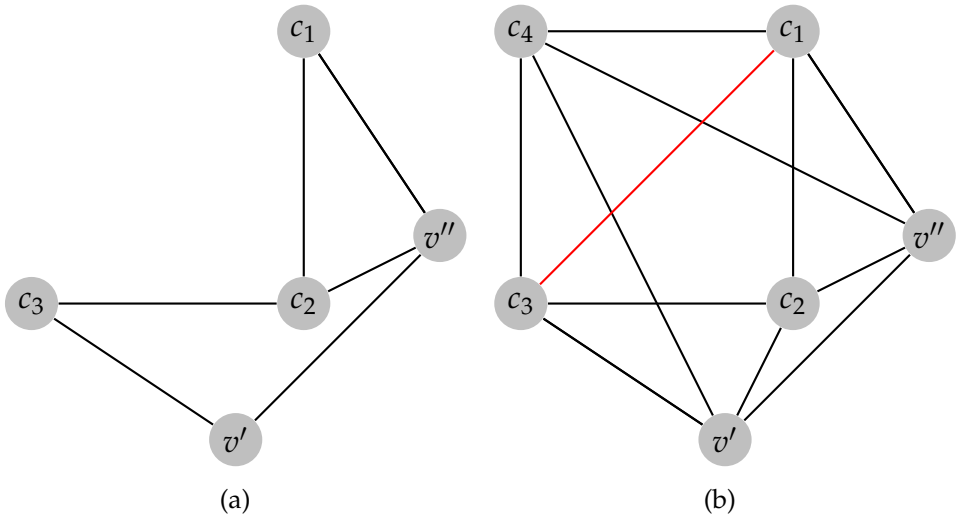


Figure 12: Proof of 2.1.

To obtain an interval graph we have to show that G' is chordal. Suppose it is not. Then there exists a non-chordal cycle which uses the edge $\{c_1, c_3\}$. This follows from the assumption that G had only one non-chordal cycle, which does not exist in G'. So the new cycle consists of the vertices $c_1c_3v'v''$. Assume c_2 and v' are not connected. Then c_2 and v'' are connected, because otherwise $v'v''c_1c_2c_3v'$ would form a non-chordal cycle of length 5 in G. With $\{c_2, v''\} \in E(G)$ we had a non-chordal cycle of length 4 in G, given by $v''c_2c_3v'$, which contradicts the assumption, see Figure 12. Analogously, we get $\{\{c_2, v'\}, \{c_2, v''\}, \{c_4, v'\}, \{c_4, v''\}\} \subseteq E(G)$. Hence $c_2v'c_4c_1c_2$ and $c_2v''c_4c_3c_2$ are two more vertex induced cycles in G, a contradiction. This proves that G' is chordal.

The construction of the simple triangle representation with the help of the interval representation is the following:

1. Calculate an interval representation of G' according to the given partial order, i.e. the interval $I_v = [s_v, e_v]$ represents the vertex v.
2. If $I_{c_1} \subset I_{c_3}$, i.e. $s_{c_1} > s_{c_3}$ and $e_{c_1} < e_{c_3}$, set $I_{c_1} := [s_{c_3}, e_{c_1}]$ and $I_{c_3} := [s_{c_1}, e_{c_3}]$.
3. If $I_{c_3} \subset I_{c_1}$, i.e. $s_{c_3} > s_{c_1}$ and $e_{c_3} < e_{c_1}$, set $I_{c_1} := [s_{c_1}, e_{c_3}]$ and $I_{c_3} := [s_{c_3}, e_{c_1}]$.

4. For $v \in V \setminus \{c_1, c_3\}$ set $T_v := (s_v, e_v, s_v)$.
5. For $v \in \{c_1, c_3\}$ set $T_v := (s_v, s_v, e_v)$.

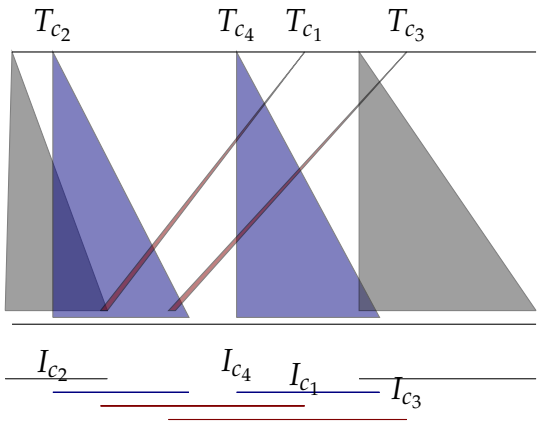


Figure 13: Simple triangle rep. with calculated interval rep. below.

An example for the steps from an interval representation to the simple triangle representation is shown in Figure 13. Step 2 (and step 3 likewise) are possible without changing any edge in the represented graph. Otherwise there would be an interval I_v intersecting $[s_{c_2}, s_{c_1}]$ with $I_v \cap [s_{c_1}, e_{c_1}] = \emptyset$. Since the interval graph has been build according to the given partial order this implies $v \prec c_1$. Since $c_1 \prec c_3$ holds we have $v \prec c_1$ and $\{v, c_3\} \notin E(G')$. Hence, we can change the interval representation according to step 2.
The resulting triangles are a triangle representation for G':
For $v, v' \in V \setminus \{c_1, c_3\}$ it is easy to see that T_v and $T_{v'}$ intersect if and only if their intervals I_v and $I_{v'}$ intersect.
The constructed triangles (lines) of c_1 and c_3 do not intersect by construction. For $v \in V \setminus \{c_1, c_3\}$ and $v' \in \{c_1, c_3\}$ we get the triangles $T_v = (s_v, e_v, s_v)$ and $T_{v'} = (s_{v'}, s_{v'}, e_{v'})$. These triangles intersect iff

$$\begin{aligned} & \neg \left((s_v < e_{v'} \wedge e_v < s_{v'}) \vee (s_v > e_{v'} \wedge e_v > s_{v'}) \right) \\ \overset{s_v \leq e_v}{\Leftrightarrow} \quad & e_v > s_{v'} \wedge s_v < e'_v. \end{aligned}$$

The last one is equivalent to the intersection of I_v and $I_{v'}$. Hence we have constructed an triangle representation for G' without the edge $\{c_1, c_3\}$ which is G. □

This method can be generalised to graphs, where the non-chordal cycles are structured.

Definition 2.2
Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be graphs. The graph $G_1 \times G_2 := (V_1 \times V_2, E)$ where

$$E = \{\{(v_1, v_2), (w_1, w_2)\} \,|(v_1 = w_1 \wedge \{v_2, w_2\} \in E_2) \vee (v_2 = w_2 \wedge \{v_1, w_1\} \in E_1)\}$$

is called the *Cartesian product* of G_1 and G_2.

Definition 2.3
The Cartesian product of paths L_i, $i \in \{1, \ldots, d\}$ is called a grid of dimension d. It is denoted with

$$G(n_1, \ldots, n_d) := L_1 \times \cdots \times L_d.$$

In the next step we will determine which grids can appear in cocomparability graphs.

Proposition 2.4
Let G be a comparability graph and $a, b, c \in V(G)$ with $\{a, b\}, \{b, c\} \in E(G)$ and $\{a, c\} \notin E(G)$. Then $a \prec b$ implies $c \prec b$ and $a \succ b$ implies $c \succ b$.

Proof. The relations $a \prec b$ and $b \prec c$ would imply $a \prec c$, hence $\{a, c\} \in E(G)$, which contradicts the assumption. □

The above proposition is also the main idea in the algorithm that calculates a partial order of a comparability graph.

Lemma 2.5
The complement of the grid $G(2, 3)$ has exactly two orientations.

Proof. Assume that $v_{1,1} \prec v_{1,3}$. Repeated application of Lemma 2.4 gives a orientation on the edges, see Figure 14. The implications are the following:

$$\begin{aligned}
v_{1,1} \prec v_{1,3} &\Rightarrow v_{1,1} \prec v_{2,3},\\
v_{1,1} \prec v_{1,3} &\Rightarrow v_{2,1} \prec v_{1,3},\\
v_{1,1} \prec v_{2,3} &\Rightarrow v_{1,1} \prec v_{2,2},\\
v_{1,1} \prec v_{2,3} &\Rightarrow v_{1,2} \prec v_{2,3},\\
v_{2,1} \prec v_{1,3} &\Rightarrow v_{2,2} \prec v_{1,3},\\
v_{2,1} \prec v_{1,3} &\Rightarrow v_{2,1} \prec v_{2,3},\\
v_{2,1} \prec v_{1,3} &\Rightarrow v_{2,1} \prec v_{1,2}.
\end{aligned}$$

Analogously $v_{1,1} \succ v_{1,3}$ gives the reverse orientation. Those orientation are given by the choices of the orientation of one edge. Hence we have found the only possible orientations. □

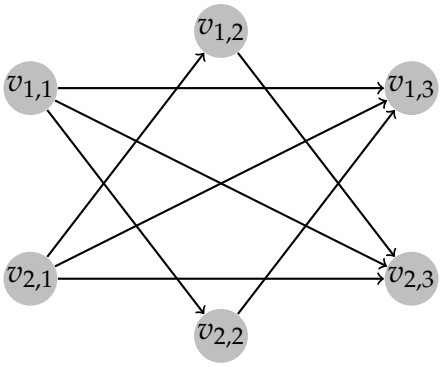


Figure 14: An orientation of $\overline{G(2,3)}$.

The construction of an orientation of the edges according to (2.4) is the main idea in the algorithm, that determines an partial order of a comparability graph, see [Gol04].

The application of Lemma 2.5 on the 2×3 sub-grids of $G(2,k)$ gives the following corollary.

Corollary 2.6
The complement of the grid $G(2,l)$ has exactly two orientations.

Theorem 2.7
Let G be a cocomparability graph where all non-chordal cycles form a grid $G(2,l)$, i.e. the vertices on non-chordal cycles induce the grid. Then G is a simple triangle graph.

Proof. As in Theorem 2.1, the idea will be to add an edge to the non-chordal cycles and build a simple triangle representation in the same way. Denote the grid vertices with $v_{1,j}$ and $v_{2,j}$, $j \in \{1,\ldots,l\}$, see Figure 15.

By 2.6 we can assume that $v_{1,1}$ and $v_{2,1}$ are the first vertices in the grid, i.e. there exist no vertex w in the grid with $w \prec v_{1,1}$ or $w \prec v_{2,1}$.

If there exists a vertex u with $u \prec v_{2,1}$ and $u \prec v_{1,1}$ in G, let us assume without loss of generality that $u \prec v_{2,1}$. The graph $G' = (V, E \cup \{\{v_{1,1}, v_{2,2}\}\})$ is a cocomparability graph. It is sufficient to show that there exists no vertex $w \in V$, such that $v_{1,1} \prec w \prec v_{2,2}$. Since $G - \{v_{1,2},\ldots,v_{1,l},v_{2,3},\ldots,v_{2,l}\}$ and $G - \{v_{1,3},\ldots,v_{1,l},v_{2,1},v_{2,3},\ldots,v_{2,l}\}$ are interval graphs and $\{w, v_{1,2}\}$ and $\{w, v_{2,1}\}$ are edges of G by Lemma 1.5 since $v_{1,1}v_{2,1}v_{2,2}$ is a path in $G - \{v_{1,2},\ldots,v_{1,l},v_{2,3},\ldots,v_{2,l}\}$ and $v_{1,1}v_{1,2}v_{2,2}$ is one in

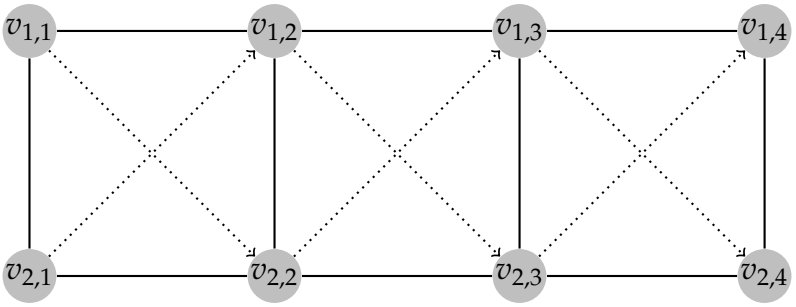


Figure 15: $G(2,4)$ with dotted orientation on the complement (except transitivity).

$G - \{v_{1,3}, \ldots, v_{1,l}, v_{2,1}, v_{2,3}, \ldots, v_{2,l}\}$. This leads to a cycle $v_{1,1}v_{1,2}wv_{2,1}v_{1,1}$, which is not part of the grid since $w \prec v_{2,2}$.

Assume the edge $\{v_{1,1}, v_{2,2}\}$ closes a new non-chordal cycle of length 4. Analogous to the proof of 2.1 we get another non-chordal cycle including the vertices $v_{1,1}$ and $v_{2,1}$, which is a contradiction to the assumed cycle structure. Hence, G' has less non-chordal cycles than G. Moreover these cycles form a $2 \times (l-1)$ grid. Repeating the foregoing procedure yields to chordal cocomparability graph, an interval graph.

The additional edges that have been added to the graph are the edges

$$\{v_{k \pmod 2),k}, v_{k+1 \pmod 2),k+1}\}, \quad k \in \{1, \ldots, l-1\}.$$

This is the case since $v_{k \pmod 2),k}$ has the neighbour $v_{k(\mathrm{mod}2),k-1}$, but $v_{k \pmod 2),k}$ has not. These vertices form an independent set in the original graph G. Hence, we can construct a simple triangle representation in Theorem 2.1 if we can ensure that we find an interval representation of our modified graph, such that the none of the intervals representing the vertices on the added path is a subinterval of another one. For the inner vertices of the path $v_{k-1(\mathrm{mod}2),k-1}$ $v_{k(\mathrm{mod}2),k}$ $v_{k+1(\mathrm{mod}2),k+1}$ this is true since $I_{v_{k \pmod 2),k}} \subseteq I_{v_{k-1 \pmod 2),k-1}}$ implies $N(v_{k \pmod 2),k}) \subseteq N(v_{k-1 \pmod 2),k-1})$, which is violated by $v_{k+1 \pmod 2),k+1}$. For $v_{1,1}$ we had an neighbour w of $v_{2,2}$ with $w \prec v_{1,1}$, as in Theorem 2.1, which is a contradiction to the original partial order where $v_{1,1} \prec v_{2,2}$. □

Larger grids of dimension 2 are not simple triangle graphs. This is due to the fact that they are not even cocomparability graphs.

Lemma 2.8
Let $G(n_1, n_2)$ be a grid with $n_1 > 2$ and $n_2 > 2$. Then $G(n_1, n_2)$ is not a cocomparability graph.

Proof. Since n_1 and n_2 are larger than 2 the grid contains the grid $G(3,3)$ as induced subgraph. Hence it is sufficient to show that the lemma holds for $H = G(3,3)$. By Theorem 2.5 we can assume that the orientation of one $G(2,3)$ grid is given as in Figure 16.

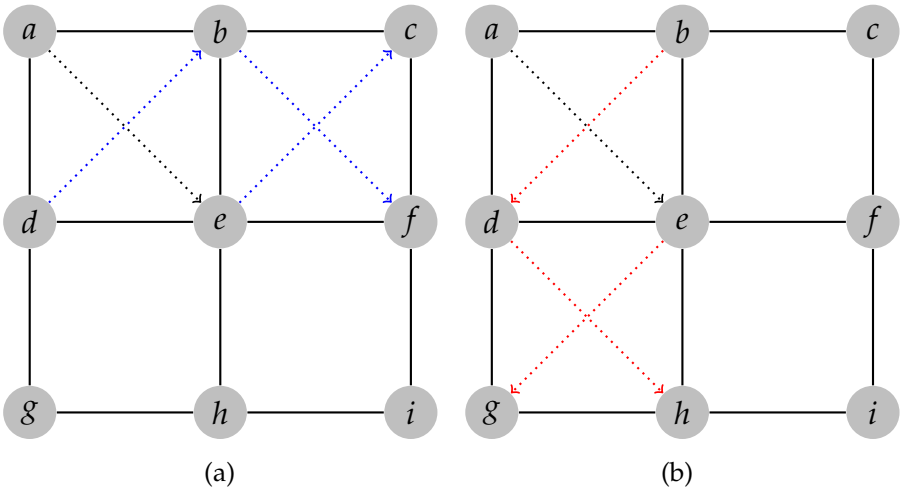


Figure 16: By $a \prec e$ forced orientations.

Now assume, without loss of generality, that $a \prec e$, see Figure 16. This implies that $d \prec b$ by using the implications of 2.5 applied on the (2,3)-grid induced by $\{a, b, c, d, e, f\}$ as shown in Figure 16(a). On the other hand, applying Lemma 2.5 on the grid induced by $\{a, b, d, e, g, h\}$ gives $b \prec d$, see Figure 16(b). Hence $a \prec e$ implies $b \prec d$ and $d \prec b$, a contradiction. □

Grids of higher dimension are not cocomparability graphs, neither.

Lemma 2.9
A grid $G(n_1, n_2, n_3)$ with $n_i > 1$, $i \in \{1, 2, 3\}$, is not a cocomparability graph.

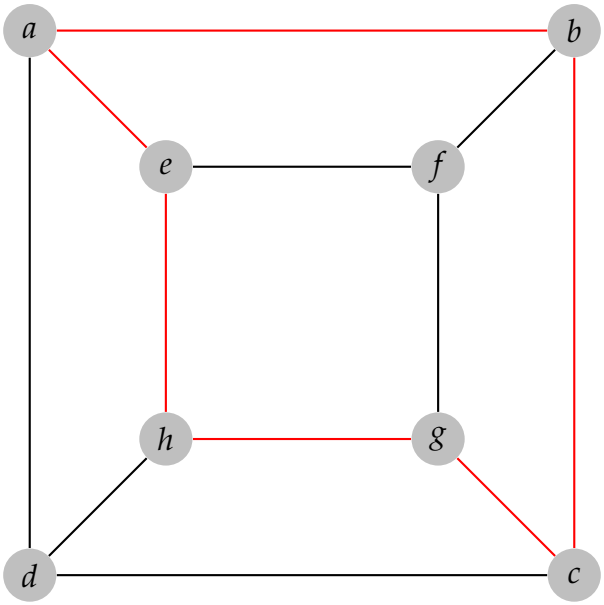


Figure 17: Induced cycle of length 6 (red) in $G(2,2,2)$.

Proof. The grid contains $G(2,2,2)$ as an induced subgraph. The graph $G(2,2,2)$ contains an induced cycle of length 6, see Figure 17. Therefore, it is no cocomparability graph by 1.12. □

2.2 Trapezoid graphs with bipartite complement

After using interval representations of slightly modified graphs in the last section trapezoid representations of graphs with bipartite complement are used to obtain a simple triangle representation.

The transitive orientations on the complement of these graphs are directly given by a partition of the vertices.

Lemma 2.10
Let G cocomparability graph whereby $\overline{G}$ is bipartite. Then, every connected component $\overline{G}$ is uniquely orientable. Additionally a partition is given by

$$X = \{x \in V(G) |\ \nexists v \in V(G) : v \prec x\}.$$

The sets X and Y are cliques in G.

Proof. Let v_1v_2 and $v_{k-1}v_k$ be (undirected) edges of $\overline{G}$. Since $\overline{G}$ is connected, there exists a path from v_1 to v_k. Without loss of generality assume that v_2 and $k-1$ lie

on such a path, i.e. one path is given by $v_1 v_2 \dots v_{k-1} v_k$. Otherwise just swap names of v_1 and v_2 or v_{k-1} and v_k.

Assume we have $v_l \prec v_{l+1}$. Since v_l and v_{l+2} are not connected in $\overline{G}$, since $\overline{G}$ is bipartite, we have $v_{l+2} \prec v_{l+1}$ by Proposition 2.4. Inductively applied on the given path we have that $v_1 \prec v_2$ implies $v_{k-1} \prec v_k$ if k is even and $v_k \prec v_{k-1}$ otherwise.

So every edge of $\overline{G}$ is in the same implication class and $\overline{G}$ is uniquely orientable.

The set X is a clique: Assume $x_1, x_2 \in X$ are not adjacent, then either $x_1 \prec x_2$ or $x_2 \prec x_1$ holds, hence x_2 or x_1 are not in X by its definition.

The set Y is a clique: Assume $y_1, y_2 \in Y$ are not adjacent, then w.l.o.g. $y_1 \prec y_2$. Since $y_1 \notin X$ there exists $x \in X$ such that $x \prec y_1$, hence $x \prec y_1 \prec y_2$ gives a 3-clique in $\overline{G}$. □

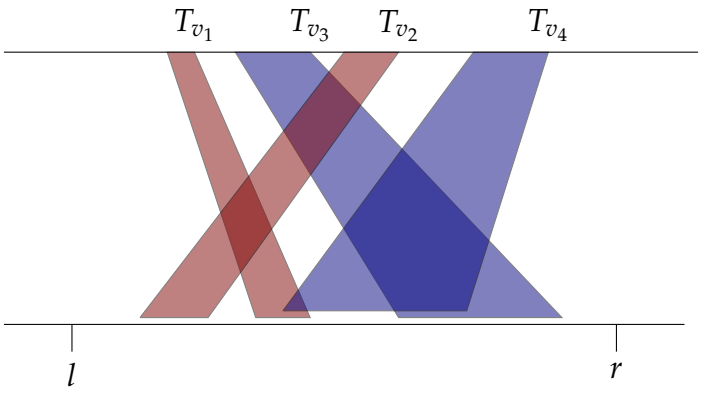
This leads to a short trapezoid representation. One side of the intervals on L_1 is not used since the vertices have no predecessor or they do not have a successor. The unused side can be shrinked onto the necessary other side of the interval. This leads to the following lemma.

Lemma 2.11
Let G be a trapezoid graph where $\overline{G}$ is bipartite. Then G is a simple triangle graph.

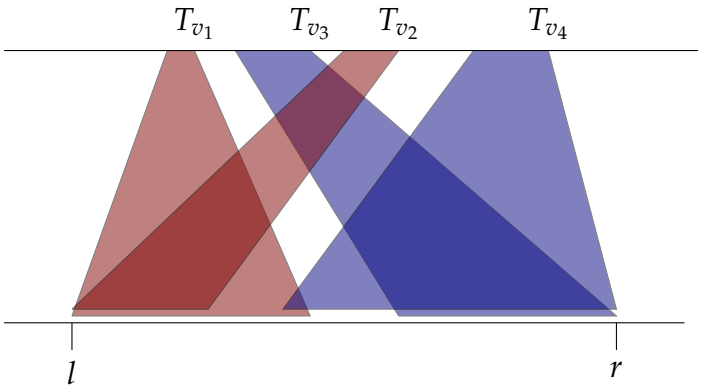
Proof. Since G is a trapezoid graph there exists a trapezoid representation of G. Because $\overline{G}$ is bipartite there exists a partition $V(G) = X \cup Y$, such that X and Y are cliques in G according to Lemma 2.10. Now let l be a point on L_2, such that $l \ll_2 l_v$ for all $v \in V(G)$, where l_v is the left corner of the trapezoid of x. Accordingly, r is a point on the right trapezoids right point, i.e. $r_v \ll r$ for all $v \in V(G)$.

Then, replacing the trapezoid $T_x = (l_1^x, r_1^x, l_2^x, r_2^x)$ by (l_1^x, r_1^x, l, r_2^x) for every $x \in X$ and the trapezoid $T_y = (l_1^y, r_1^y, l_2^y, r_2^y)$ by (l_1^y, r_1^y, l_2^y, r) for every $y \in Y$, leads to a trapezoid representation with the property, that its intersection graph is still G, see Figure 18.a/b: Since r and l lie entrirely on the right resp. left of the trapezoids, they are all expanded, hence it is sufficient to show that there arises no new intersection of trapezoids. Now assume x and y are not adjacent, but their trapezoids intersect in the newly constructed trapezoid representation. By construction of X and Y we may expect $x \in X$ and $y \in Y$. Since the right corner of T_x and the left corner of T_y have not changed we have still $T_x \prec T_y$, see Figure 18.c.

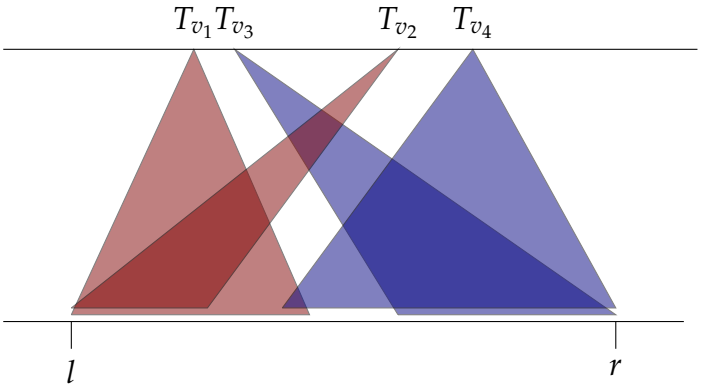
Now replacing every trapezoid $T_x = (l_1^x, r_1^x, l, r_2^x)$ by the triangle (r_1^x, l, r_2^x) for $x \in X$ and $T_y = (l_1^y, r_1^y, l_2^y, r)$ by (l^y, l_2^y, r) leads to a simple triangle representation of G: Since



(a) Original trapezoid representation



(b) Extended intervals on L_2



(c) Shrinked intervals on L_1

Figure 18: Constructing a simple triangle rep. from a trapezoid rep. of a graph with bipartite complement.

the trapezoids are narrowed the it is to show that formerly intersecting trapezoids still intersect. Two intersecting trapezoids of vertices in X have still the point l on L_2 in common, two trapezoids of vertices in Y intersect at least in r. For adjacent $x \in X$ and $y \in Y$ we have $r_1^x \gg_1 l_1^y$ or $r_2^x \gg_2 l_2^y$. With $l_2^x = l \ll_2 r = r_2^y$ we have an intersection of the segments between l and r_1^x of x and r and l_1^y of y or of the intervals on L_2, see Figure 18.c. □

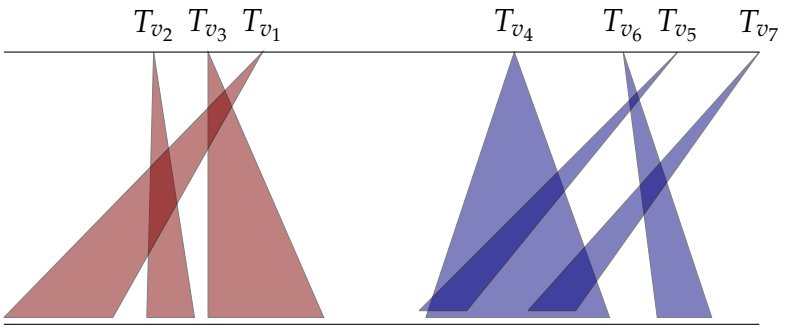
Due to the fact that every simple triangle graph is a trapezoid graph, Theorem 2.11 shows that a cocomparablity graph with bipartite complement is a simple triangle graph, if and only if it is a trapezoid graph.

A possibility to combine some of the known simple triangle graphs and, more importantly, decompose them, is discussed in the next chapter.

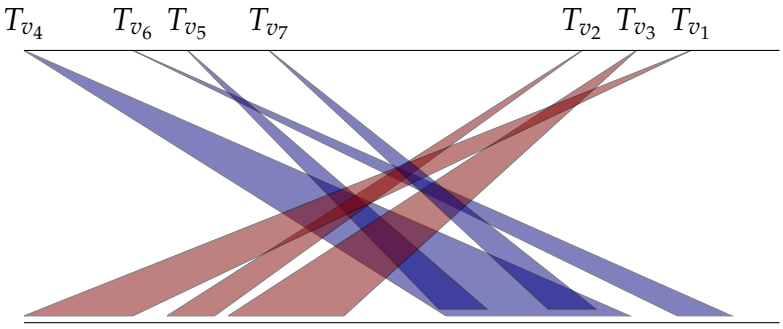
§ 3 Partial orders and simple triangle representations

In this chapter, some known theory about different partial orders on the same (co)-comparablity graph is introduced. Since it is well known how these orders behave it is possible to reconstuct the steps performed the partial order on a simple triangle representation, such that we can conclude that there exists a simple triangle representation for every partial order on the simple triangle graph's complement.

The simple triangle graphs in the last chapter have given a brief overview of some simple triangle graphs. Obviously, these constructions do not cover all simple triangle graphs. One way to obtain more simple triangle graphs is to combine two simple triangle graphs by connection every vertex of one of the graphs with vertex in the other graphs or by simply unify their vertices and edges. The according representations are given in Figure 19.



(a) Union of simple triangle graphs...



(b) ... and connection of them.

Figure 19: Combination of two simple triangle graphs.

The combination of these two simple triangle graphs is the *substitution* of the vertices in a independent set of size 2 and a 2-clique by one simple triangle graph each.

Remark 3.1
In the same way, the lines in any permutation graph can be substituted by a simple triangle representation to obtain a new simple triangle graph.

The inverse process, the decomposition into graphs that cannot be constructed by any (non-trivial) substitution, is known as *modular decomposition* or *substitution decomposition* and is studied to gain a further understanding of different partial orders on the same simple triangle graph.

3.1 More on partial orders

Let $G = (V, E)$ be a graph. A set of nodes $M \subseteq V$ is called a *module* if for every $x \in V \setminus M$ the node x is adjacent to every node of M or it is not adjacent to any node of M, i.e. $\Gamma(x) \cap M = \emptyset$ or $\Gamma(x) \cap M = M$. A module is also called *homogeneous set* or *paritive set*. If we have $1 < |M| < |V(G)|$ the module M is called *proper*.

Modules have a close connection to the structure of different partial orders on a graph. The theory of modules has been established in [Gal67] and is used to determine the number of partial orders of a comparability graph in [Gol04]. The notation of the last-mentioned book is used here.

Example 3.2
Connected components and connected components of the complement are modules.

Since partial orders have a natural interpretation on a comparability graph as a direction on the edges, i.e. edges are directed from a smaller to a larger vertex, comparability instead of cocomparability graphs are used in this first part.

As mentioned in Proposition 2.4, the main tool to calculate a partial order of a comparability graph is fixing the direction on one edge and using the transitivity of a partial order to determine which other directions on the different edges are implicated. Formally this is done by using the following notation:

$$(v, w)\Gamma(v', w') \Leftrightarrow \begin{cases} \text{either } v = v' \wedge \{w, w'\} \notin E(G) \\ \text{or} \quad w = w' \wedge \{v, v'\} \notin E(G) \end{cases} .$$

The notation (v, w) indicates that the edge between v and w is directed from v to w or in other words $v \prec w$. This *forces* $v' \prec w'$ by Proposition 2.4.

We define the relation Γ^* as the transitive, reflexive closure of Γ, i.e. we have $(v,w)\Gamma^*(v',w')$ if and only if there exists a chain

$$(v,w)\Gamma(a_1,b_1)\Gamma(a_2,b_2)\dots(a_k,b_k)\Gamma(v',w').$$

By Γ^* an equivalence relation on directed edges is given that indicates which orientation on one edge influences an orientation on other edges. The equivalence classes give a partition of the set

$$\{(v,w)|\{v,w\} \in E(G)\}.$$

The equivalence classes are called *implication classes* in [Gol04]. Implication classes are equivalence classes of directions of edges that are obtained by transitivity, i.e. if one directed edge is reversed in one implication class the others in this class are reversed in any partial order on the graph. For an implication class A we define

$$A^{-1} := \{(b,a)|(a,b) \in A\}.$$

The implication class of a directed edge (a,b) is denoted with $[(a,b)]$. Because $(v,w)\Gamma(v',w')$ implies $(w,v)\Gamma(w',v')$ we have

$$[(v,w)]^{-1} = [(w,v)].$$

If an implication class contains (v,w) and (w,v) for $w,v \in V(G)$ a graph is not an comparability graph since $(v,w)\Gamma^*(w,v)$ means: $v \prec w$ implicates $w \prec v$.

Theorem 3.3 ([Gol04])
A graph G is cocomparability graph if and only if $A \cap A^{-1} = \emptyset$ for every implication class A. All possible orientations are obtained by the different choices of the orientations of the implication classes, i.e. the choice of A or A^{-1}.

Note that not every choice of A or A^{-1} gives a transitive orientation, e.g. the complete graph on three vertices K_3 has three implication classes consisting of one edge each. Let $V(K_3) = \{v_1, v_2, v_3\}$ then $v_1 \prec v_2$, $v_2 \prec v_3$ and $v_3 \prec v_1$ is not a partial order, but $v_1 \prec v_3$ instead of the last relation gives a partial order.

Example 3.4
The complete graph on three vertex is a comparability graph. Its implication classes are

$$\begin{array}{ccc} A_1 = \{v_1v_2\} & A_2 = \{v_2v_3\} & A_3 = \{v_3v_1\} \\ A_1^{-1} = \{v_2v_1\} & A_2^{-1} = \{v_3v_2\} & A_3^{-1} = \{v_1v_3\}. \end{array}$$

Since there are three implication classes there are $2^3 = 8$ possibilities for the choice of A_i or A_i^{-1}. Because there are only $3! = 6$ total orders on three nodes, not all choices lead to an order. The problem is solved by removing the edges of the implication class from the graph before determining the next one. This forces other implication classes to union in remaining graph. For the correctness of this idea and the complete algorithm see [Gol04].

The important part of this section is the fact that the implication classes span structured set of vertices.

Theorem 3.5 ([Gol04])
Let Y be the set of vertices that is spanned by the implication class A. Then Y is a module.

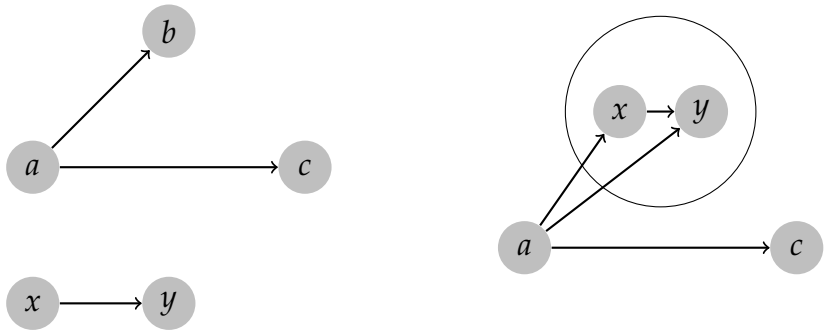
Proof. If $Y = V$ the theorem holds. Assume there exists a node $v \in V \setminus Y$, such that $\Gamma(v) \cap Y \notin \{Y, \emptyset\}$. Let $a \in Y \cap \Gamma(v)$ and $c \in Y \setminus \Gamma(v)$. Since a and c are spanned by Y there exists a Γ-chain

$$ab_0\Gamma a_1b_1\Gamma a_2b_2 \dots \Gamma a_kb_k$$

with $a_k = c$ or $b_k = c$. By definition $a_ib_i\Gamma a_{i+1}b_{i+1}$ implies either $a_i = a_{i+1}$ or $b_i = b_{i+1}$. Whence $\{a, v\} \in E(G)$ and $\{c, v\} \notin E(G)$ implies that there exists $i \in \{0, \dots, k\}$, such that v is adjacent to exactly one of a_i or b_i, which implies $a_iv\Gamma a_ib_i$ or $vb_i\Gamma a_ib_i$. This contradicts $v \notin Y$. □

Now consider the following construction:

Let $\vec{G}$ and $\vec{H}$ two directed graphs, that both represent a partial order each, see Figure 3.1. The substitution of one vertex b of $\vec{G}$ with $\vec{H}$ leads to a new transitive directed graph – another partial order.



(a) Substitution of b...

(b) ... with $x \to y$

Figure 20: Substitution in partial orders.

By construction $V(\vec{H}) = \{x, y\}$ is a module in the underlying undirected graph spanned by the implication class $[(x, y)] = \{(x, y)\}$. By the choice of another orientation on $V(\vec{H})$, another orientation of the whole graph is obtained.

Based on the underlying undirected graphs G and H of $\vec{G}$ and $\vec{H}$, we denote the resulting graph drawn $G[H]$. In general for a graph G and graphs H_i, $i \in V(G)$, the graph $I := G[H_1, \ldots, H_{|V(G)|}]$ is defined as

$$V(I) := \bigcup H_i$$

and

$$E(I) := \bigcup V(H_i) \cup \{xy | x \in V_i, y \in V_j \text{ and } ji \in E(G)\}.$$

The graph G is called *inner factor* and the graphs H_i are *outer factors*. The representation of I in terms of G and H_i is called *modular decomposition*.

Modular decomposition is the tool that the calculation of partial orders on a graph is based on: Given a graph with its decomposition $I = G[H_1 \ldots, H_n]$ and transitive orientations $\vec{G}$ and $\vec{H_i}$, a transitive orientation $\vec{I} = \vec{G}[\vec{H}_1, \ldots, \vec{H}_n]$ is obtained. So a graph is a (co)comparability graph if and only if all its inner and outer factors are (co)comparability graphs, see [Gol04].

Finding a proper decomposition (i.e. neither the outer nor an inner factor is the original graph) of a not uniquely orientable (co)comparability graph is ensured by the following lemma.

Lemma 3.6 ([Gol04])
There exists at most one implication class that spans all vertices of G.

Since graphs with only one implication class are uniquely orientable, and are not of our interest here, since, we can find a proper decomposition of a graph which will implicitly used in the next section.

The following lemma finally guarantees that we can work with the same modular decomposition on G as well as on $\overline{G}$.

Lemma 3.7
A let $M \subseteq V(G)$ be a module of G. Then M is a module of $\overline{G}$.

Proof. Consider $v \in V \setminus M$. If $\Gamma(v) \cap M \neq \emptyset$ then we have $M \subseteq \Gamma(v)$ by the definition of a module. Hence v is not adjacent to any node of M in $\overline{G}$. Otherwise $\Gamma(v) \cap M = \emptyset$. This means $\{v, w\} \in E(\overline{G})$ for every $w \in M$. Hence M is a module of $\overline{G}$. □

3.2 Partial orders and simple triangle representations

In this section the modular decomposition is used to change one partial order on the same graph into another one step by step on the simple triangle representation, which induces the first order. In this way, it is shown that every partial order can be chosen to calculate a simple triangle representation.

To be allowed to reverse the order of some triangles, we have to make sure not to be in the following situation:

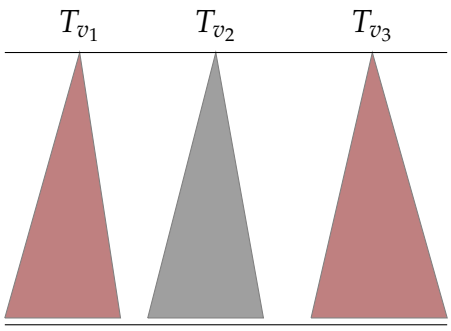


Figure 21: Module $\{v_1, v_3\}$

The module $M = \{v_1, v_3\}$ has been picked. Then it is not possible to reverse the order between v_1 and v_3, i.e. set $v_3 \prec v_1$ without changing the orientation between v_2 and the mentioned. This is the reason for the following 'definition'.

Lemma 3.8
Let $G = (V, E)$ be a cocomparability graph with partial order $\prec$ and M' a module of G. Define

$$M'' = \{v \in V | \exists m_1, m_2 \in M' : m_1 \prec v \prec m_2\}.$$

Then $C(M', \prec) := M := M' \cup M''$ is a module.

The module $C(M', \prec)$ is kind of the convex closure of the module M', i.e. every triangle between the rightmost and leftmost triangle of M' is included in $C(M', \prec)$.

Proof. Behold $v \in V \setminus M$. If v is adjacent to all $m' \in M'$, then it is adjacent to all $m'' \in M''$ since $v \prec m''$ (resp. $v \succ m''$) would imply the existence of $m_2 \in M'$ with $v \prec m'' \prec m_2$ (resp. $m_1 \in M'$ with $v \succ m' \succ m_1$). This contradicts $v, m_1 \in E(G)$. If v is not adjacent to any $m' \in M'$ then $v \prec m'$ (resp. $v \succ m'$) for all $m' \in M'$. This holds due to the fact that $v \notin M''$. Hence $v \prec m' \prec m''$ (resp. $v \succ m' \succ m''$) for all $m'' \in M''$. This proofs that M is a module. □

Consider a triangle representation between the upper line L_1 and the lower line L_2 with the points on L_1 and the intervals on L_2.

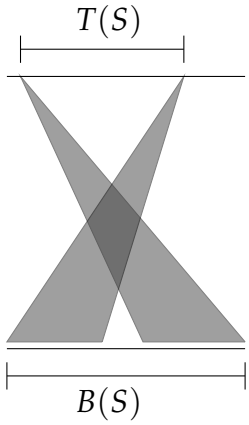


Figure 22: Definition of $T(S)$ and $B(S)$

Let S be a set of triangles of the triangle representation. Then define $T(S) := [t_l, t_r]$ as the interval on L_1 where t_l is the leftmost point and t_r the rightmost point of all the triangles in S on L_1, see Figure 22. Analogously, the interval $B(S) := [b_l, b_r]$ denotes the interval between the leftmost point b_l and the rightmost point b_r on L_2.

Lemma 3.9
Let T_w, T_x, T_y and T_z be triangles, such that $T_x \ll_1 T_y \ll_1 T_z$, $T_x \prec T_z$ and T_y intersects T_x and T_z. If T_w intersects T_x and T_z then T_w intersects T_y.

Proof. Consider the triangle $T_v = (t_y, succ(r_x), prec(l_z))$, see Figure 23.

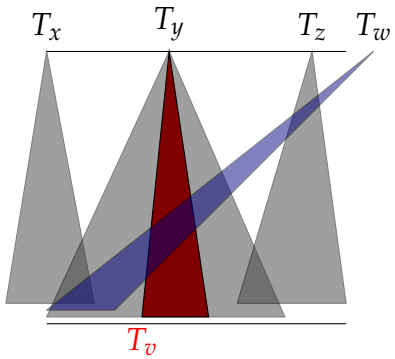


Figure 23: Proof of 3.9

This triangle is a subset of T_y, hence $N(T_v) \subset N(T_y)$. Because of $T_x \prec T_v \prec T_z$ and since x, w, z is a path from x to z Lemma 1.5 gives $v \sim x$ and so $y \sim x$. □

Now we have enough tools to show that the construction in Remark 3.1 – the substitution of a line in a permutation graph – is possible in general for modules $C(M)$, where M is a module spanned by an implication class. Therefore, the simple triangle representation is modified – without changing the resulting intersection graph nor its partial order – such that none of the triangles in $V(G) \setminus C(M)$ has a corner in the interval $B(C(M))$ or $T(C(M))$. In Figure 24 is a part of a simple triangle representation shown. In the red area are the triangles of $C(M)$. On L_1 the condition is satisfied – none of the other triangles has a corner in the red area there – whereas the blue triangle violates it on L_2. This will be the situation after the following lemma.

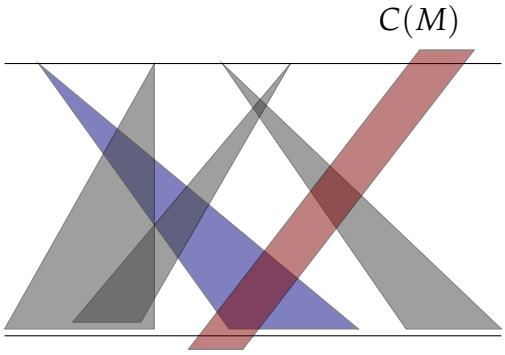


Figure 24: Simplified triangle representation after applying Lemma 3.10.

Lemma 3.10
Let M be a module of a triangle graph G that is spanned by an implication class in $\overline{G}$. There exists a triangle representation of G, such that none triangle T_v corresponding to $v \in V(G) \setminus C(M)$ has a corner in $T(C(M)) = [t_l, t_r]$.

Proof. Let $S(R)$ be the set of triangles violating the property on L_1 in a given simple triangle representation R. First note that $T_v = (t_v, l_v, r_v) \in S(R)$ implies $B(C(M)) \cap [l_v, r_v] \neq \emptyset$ because there exists a triangle $T_a = (t_a, l_a, r_a) \in S(R)$, such that $t_a < t$. Since $T_a \sim T_v$ we have $[l_a, r_a] \cap [l_v, r_v] \neq \emptyset$ or $r_v < l_a$. Analogously, there exists a $T_b = (t_b, l_b, r_b) \in S(R)$, such that $t_v < t_b$ and $[l_b, r_b] \cap [l_v, r_v] \neq \emptyset$ or $r_b < r_v$. Because of $[r_b, l_a] \subset B(C(M))$ we have $B(C(M)) \cap [l_v, r_v] \neq \emptyset$, see Figure 25. There is a triangle representation R' inducing the same partial order, such that $S(R') = \emptyset$. Proof by induction on $|S_T(R)|$. For $|S_T| = 0$ the assumption is true. Let $|S_T| = n > 0$.

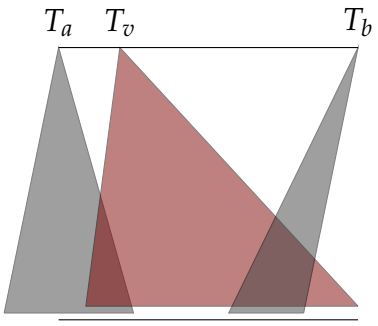


Figure 25: Proof of 3.10

Consider $T_v \in S(R)$. Then the sets of triangles

$$L := \{T_x | x \in C(M) : t_x < t_v\}$$

and

$$R := \{T_x | x \in C(M) : t_x > t_v\}$$

are not empty. Since $\overline{G[C(M)]}$ is connected, there are triangles $T_m \in L$ and $T_n \in R$, such that $T_m \not\sim T_n$. Lemma 3.9 gives $T_v \sim T_b$ for every triangle $T_b \sim M$. Define $l := \min\{prec(t_l), l_v\}$ and $r := \max\{succ(t_r), r_v\}$. Replacing T_v with the triangle $(prec(t_l), l, r)$ leads to a new triangle representation R', as shown in Figure 26.

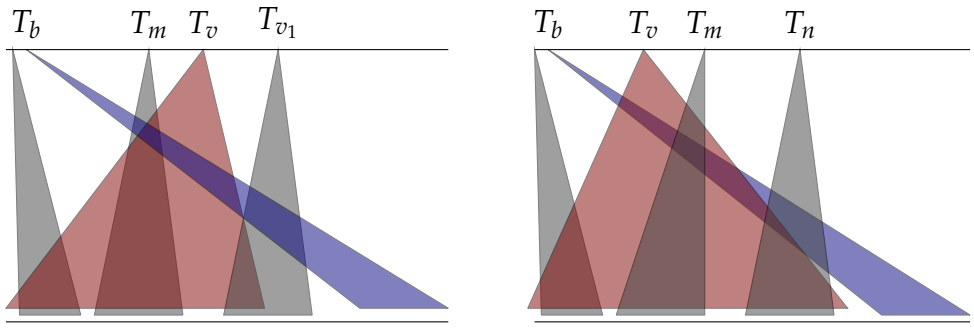


Figure 26: Replacing T_v in proof of 3.10.

The representation R' is another representation for G: Assume there is a triangle $T_c = (t_c, l_c, r_c)$, such that $T_c \sim T_v$ but $T_c \not\sim (prec(t_l), l, r)$. Since $[l, r] \supseteq [l_v, r_v]$ and $prec(t_l) < t_v$ this implies $prec(t_l) < t_c < t_v$ and hence $T_c \in S(R)$. This leads to $[l_c, r_c] \cap B(C(M)) \neq \emptyset$ and $B(C(M)) \subset [l, r]$ comes to $T_c \sim (prec(t_l), l, r)$. Now assume there exists a triangle $T_d = (t_d, l_d, r_d)$, such that $T_d \sim (prec(t_l), l, r)$ but $T_d \not\sim$

T_c. Then $t_d \in [t_l, t_v]$, $l_d \in [l_v, b_r]$, or $r_d \in [b_l, r, v]$. Each case implies $t_d \in T(C(M))$ or $[l_d, r_d] \cap B(C(M)) \neq \emptyset$, hence $T_d \sim T_v$.
Because of $prec(t_l) < t_l$ we have $|S_T(R')| < |S_T(R)|$ and the lemma is proven by induction. □

Lemma 3.11
Let T_a and T_b be triangles, such that $a \sim C(M)$ and $b \sim C(M)$. If $t_a < T(C(M))$ and $T(C(M)) < t_b$ then $T_a \sim T_b$.

Proof. The module $C(M)$ contains two not adjacent vertices x and y. Assume without loss of generality that $T_x \prec T_y$. Since T_a and T_y intersect we have $l_y < r_a$ and $T_b \sim T_x$ implicates $l_b < r_x$. Hence $l_b < r_x < l_y < r_a$ shows $l_b < r_a$. Because of $t_a < t_b$ the triangles T_a and T_b intersect. □

The result of the following lemma is that there exists a simple triangle representation, such that the module $C(M)$ is "independent" of the rest of the triangle representation, i.e. its representation can be replaced by another representation.

Lemma 3.12
Let M be a module of a triangle graph G that is spanned by an implication class in $\overline{G}$. There exists a triangle representation of G, such that none triangle T_v corresponding to $v \in V(G) \setminus C(M)$ has a corner in $T(C(M)) = [t_l, t_r]$ nor in $B(C(M)) = [b_l, b_r]$.

Proof. The property that no corner lies in $T(C(M))$ is fulfilled by Lemma 3.10. Given such a simple triangle representation we define the sets

$$L := \{v | t_v < t_l \wedge (r_v \in B(C(M)) \vee l_v \in B(C(M))\}$$

and

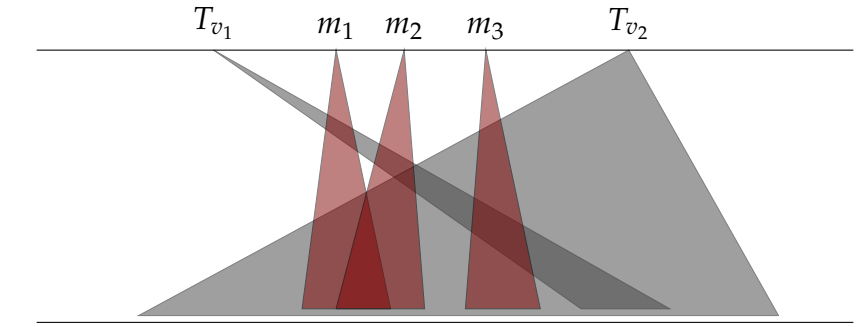
$$R := \{v | t_v > t_r \wedge (r_v \in B(C(M)) \vee l_v \in B(C(M))\}.$$

Let T_v be the triangle with the rightmost corner in of all $v \in L$ in $B(C(M))$. If the rightmost corner is l_v then replacing T_v with $(t_v, succ(b_r), r_v)$ leads to a new triangle representation of G. Otherwise the replacement with $(t_v, l_v, succ(r_v))$ gives a new representation. In both cases only the order of corners of triangles in L and R or L and $C(M)$ are changed. Since the triangles of L and R intersect by Lemma 3.11 and the triangles of L and $C(M)$ intersect as well, we have to show that all triangles of L intersect the triangles of R and of $C(M)$. Because we have $t_x \geq t_l$ and $l_x \leq b_r$ for every $x \in L \cup C(M)$ the result is a different triangle representation of G. For R the mirror-symmetric procedure, i.e. replacing T_v with $(t_v, l_v, prec(b_l))$ or $(t_v, prec(b_l), r_v)$ leads to a triangle new representation of G. Since the number of corners in $B(C(M))$ decreased with every replacement the lemma is proven inductively. □

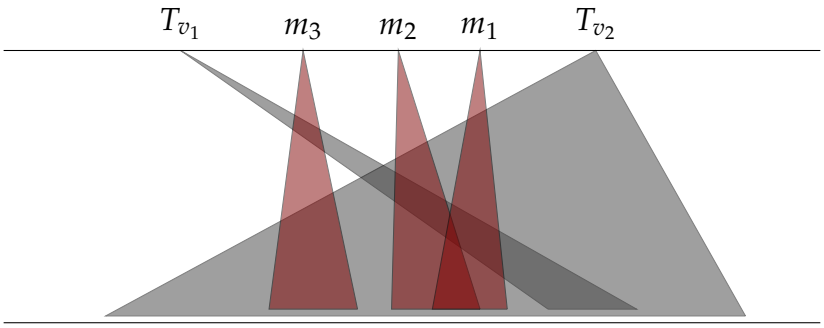
Now we have a modified simple triangle representation without any corner of a triangle of $V \setminus C(M)$ between any two corners of triangles in $C(M)$.

Lemma 3.13
Let R be a simple triangle representation of G and R' another triangle representation of $G[C(M)]$. Then, there exists a triangle representation R'' on G, such that R'' agrees with R on the order of the corners restricted to $C(M)$, i.e. the order of $\{t_v | v \in C(M)\}$ on L_1 and of $\{l_v | v \in C(M)\} \cup \{r_v | v \in C(M)\}$ on L_2 are the same in R and R''.

Proof. Consider the triangle representation constructed in Lemma 3.12. Replacing the i-th corner in the interval $T(C(M))$ on L_1 in this representation with the i-th corner on L_1 in R' and the j-th corner on in the interval $B(C(M))$ on L_2 with the j-th corner on L_2 in R', see Figure 27.



(a) A triangle rep. where the $C(M)$...



(b) ...is replaced by another rep. of $C(M)$

Figure 27: Lemma 3.13: Replacement of a triangle representation of a module.

Hence, the simple triangle representation of a module $C(M)$ for a module M that is spanned by an implication class can simple be replaced by another simple triangle representation.

Lemma 3.14
Let G be a cocomparability graph and $C_1 \neq C_2$ two connected components of G. If $v_1 \prec v_2$ for $v_1 \in C_1$ and $v_2 \in C_2$ then $v_1' \prec v_2'$ for all $v1' \in C_1$ and all $v_2' \in C_2$.

Proof. Assume there is $v_1 \prec v_2$ for $v_1 \in C_1$ and $v_2 \in C_2$, as well as $v_1' \succ v_2'$ for $v_1' \in C_1$ and $v_2' \in C_2$. If $v_1 \prec v_2 \prec v_1'$ we have an path from v_1 to v_1', hence Lemma 1.5 gives a vertex of C_1 that is adjacent to C_2. This is a contradiction, because they are in different connected components. This implies $v_2 \prec v_1 \prec v_2'$ which is a contradiction by the same lemma. □

The last lemma has shown that the different connected components can simply be reordered.

Theorem 3.15
Let G be a simple triangle graph. Then every partial order on G admits a simple triangle representation.

Proof. We show the theorem by induction over the number of vertices: For a graph with one vertex the theorem holds. So let $G = (V, E)$ be a simple triangle graph with $|V| = n > 1$. Since G is a simple triangle graph it has a simple triangle representation R, inducing a partial order O_1 denoted with $\prec_1$. Let O_2 be another arbitrary partial order, such that G is a cocomparability graph. We can assume without loss of generality that $O_1 \neq O_2$ and $O_1 \neq O_2^{-1}$. Otherwise, R respectively R^{-1} is a triangle representation inducing O_2. If $\overline{G}$ has an implication class that spans every vertex of G assume without loss of generality that this class is oriented in the same direction. Otherwise, consider the reversed orientation and the horizontally flipped triangle representation accordingly. If $\overline{G}$ has only this implication class then it is uniquely partial orderable, hence O_1 and O_2 are identical and it is nothing to prove.

Now we can choose an implication class A of $\overline{G}$ that is differently directed in O_1 and O_2. This implication class A does not span all vertices by Lemma 3.6. Denote the module spanned by A with M' and set $M := C(M')$.

If $M = V$, then G is not connected. To this end, we have $V = M' \cup \{v \in V \setminus M' | \exists m_1, m_2 \in M' : m_1 \prec v \prec m_2\}$ and the last set is not empty. Hence, every vertex in this set is not connected to at least one vertex in M'. Since M' is a module it is not connected to any and the graph is not connected.
Each connected component has simple triangle representation inducing O_2 by induction. Arranging these triangle representations according to O_2 gives a triangles representation inducing O_2 by Lemma 3.14.

If $M \neq V$ then $|M| < |V|$ and the subgraph induced by M admits a simple triangle representation that is identical to $\prec$ restricted on M by induction. Replacing the triangle representation of M according to O_1 with the new one according to O_2 on the intervals on L_1 and L_2 using Lemma 3.13 yields to a triangle representation with less different oriented edges of the induced partial order. Inductively a representation that induces O_2 is revealed and the lemma is proven. □

All in all we have shown, that any partial order on $\overline{G}$ can be used for the further calculation of a simple triangle representation. An approach for an algorithm is discussed in the next chapter.

§ 4 The order on the tops

The goal of this chapter is to understand the logic of the 'passing on of information' through the graph, i.e. the understanding of how the different choice of the order of corners on L_1 influence the further restrictions to possible constructions of other triangles. Therefore, Figure 28 gives an example of a triangle graph that is not a simple triangle graph. The red triangles are connected by the two blue triangles. Since the two vertices corresponding to the blue triangles are a module Lemma 3.12 gives a representation such that none of the red triangles has a corner between the blue top corners. Since the red triangles do not intersect, the proof of Lemma 3.10 shows that the top corners of the red triangles lie on the same side of the blue top corners. Hence, one of the red triangles (in the figure the one in headstand) cannot be intersected by the grey corners on L_1, so they have to intersect on L_2. This is not possible by this construction. A method to recognise this algorithmically is given in the following.

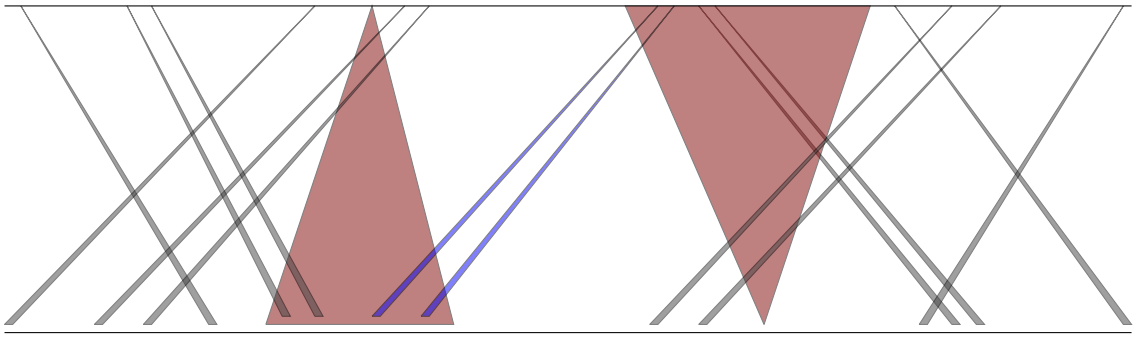


Figure 28: A non-simple triangle graph.

4.1 From an order on the tops to a triangle representation

In this section an equivalent condition for a graph being a simple triangle graph is introduced. This leads to a backtracking algorithm that is formulated in in second section.

The idea is based on an observation already mentioned in the introduction of this chapter.

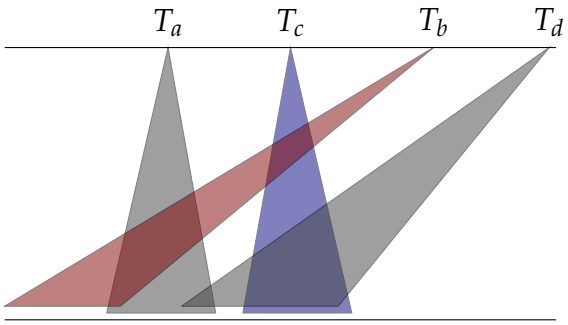


Figure 29: Order of the tops of a cycle.

Lemma 4.1
Let G be a simple triangle graph, $\ll_1$ the order on L_1 and $a, b, c, d \in G$ be an induced cycle with $a \prec c$ and $b \prec d$, see Figure 29. Then the following is satisfied in every triangle representation of G that induces $\prec$:

a.) $(t_c \ll_1 t_b) \oplus (t_d \ll_1 t_a)$

b.) $(t_c \ll_1 t_b) \Rightarrow (t_c \ll_1 t_d \wedge t_a \ll_1 t_b)$

c.) $(t_d \ll_1 t_a) \Rightarrow (t_d \ll_1 t_c \wedge t_b \ll_1 t_a)$

The symbol $\oplus$ denotes *XOR*, i.e.

$$a \oplus b \Leftrightarrow (a \vee b) \wedge (\overline{a} \vee \overline{b}).$$

Property a.) is similar to Lemma 3.10 for cycles of length 4 instead of modules: a and c as well as b and d are a module. Hence we have a representation such that no triangle has a top corner between a and c or b and d in this induced subgraph.

Proof. a.) Assume there is a simple triangle representation with $t_c \ll_1 t_b$ and $(t_d \ll_1 t_a)$. Without loss of generality $t_a \ll_1 t_b$. Then we have $t_d \ll_1 t_a \ll_1 t_b$ which is a contradiction to $b \prec d$. On the other hand, assume we have $t_b \ll_1 t_c$ and $t_a \ll_1 t_d$. Without loss of generality assume $r_a \ll_2 r_b$. Because of $b \prec d$ we have $r_a \ll_2 r_b \ll_2 l_d$ and hence $a \prec d$, a contradiction as well.

b.) With $t_c \ll_1 t_b$ and $b \prec d$ we have $t_c \ll_1 t_d$. The relation $t_b \ll_1 t_a$ would imply $t_a \gg_1 t_c$, a contradiction to $a \prec c$.

c.) Is proven analogously to b. □

The main statement is a.), since the other two are simple consequences of the properties of an order. It attaches some conditions to a realiser of $\prec$ that is the order on L_1.

In the following theorem we will see, that Lemma 4.1 is already a sufficient condition on a realiser of $\prec$ to calculate a simple triangle representation.

Theorem 4.2
Consider all non-chordal cycles $C_1, \ldots, C_k$ in the trapezoid graph G. The graph G is a simple triangle graph iff there is a total order $<$ satisfying all conditions of Lemma 4.1 for every cycle C_i, $i \in \{1, \ldots, k\}$, and agrees with $\prec$, i.e. $a \prec b$ implies $t_a < t_b$.

Proof. If G is a simple triangle graph, then the total order $\ll_1$ on L_1 satisfies the given conditions by Lemma 4.1.a for every non-chordal cycle.

Now let $\ll$ be a total order, such that 4.1 is satisfied. Let $V(G) = \{v_1, \ldots, v_n\}$ be the vertices of G enumerated according to $\ll$. Then the following procedure leads to a simple triangle representation of G:
The first triangle is defined as $T_{v_1} = (1, l_1, r_1)$ with $l_1 < r_1$.

1. Set $T_{k+1} := (k, l_{k+1}, r_{k+1}) := (k, x, x)$ where x is the successor of the last corner on L_2.
2. While there exists a triangle T_i ,$i \leq k$, such that
$$N(v_i) \cap \{v_{k+2}, \ldots, v_n\} \supset N(v_{k+1}) \cap \{v_{k+2}, \ldots, v_n\}$$
and $r_i < r_{k+1}$ update $T_{k+1} := (k, y, y)$ where $y := prec(z)$ and $z = prec(r_i)$.
3. While there exists a triangle T_i ,$i \leq k$, such that $v_i \sim v_{k+1}$ but $r_i < l_{k+1}$ update $T_{k+1} := (k, y, r_{k+1})$ where $y := prec(z)$ and $z = prec(l_i)$.

After the k-th iteration the algorithm leads to a simple triangle representation of $G[\{v_1, \ldots, v_k\}]$ with the property that the right corners r_i, $1 \leq i \leq k$ are ordered in the way that $r_i < r_j$ implies
$$N(v_i) \cap \{v_{\max(i,j)+1}, \ldots, v_n\} \subseteq N(v_j) \cap \{v_{\max(i,j)+1}, \ldots, v_n\}.$$

For $k = 1$ this property is true. Step 2 can be executed as long as z is not the right corner of a triangle $T_j = (t, l, z)$, that is not adjacent to T_{k+1} – in the other cases the operation performed does not change the intersection graph (if z is a left corner) or adds an edge that is part of G (if z is a right corner of a triangle adjacent to v_{k+1}). On the other hand, the loop ensures that there is another triangle T_i with $r_i < r_{k+1}$ (hence $r_i < z$), such that $v_i \sim v_{k+1}$. This is a contradiction to the assumption above.

It remains to show that there is no vertex v_l with $l > k+1$, such that $v_l \sim v_k + 1$ and $v_i \not\sim v_l$, $i < k+1$, but $r_{k+1} < r_i$. Since $r_{k+1} < r_i$ implies the existence of a vertex $v_m \in N(v_i)$, $m > k+1$ there are two possibilities: If $v_l \not\sim v_m$, then $v_l \prec v_m$ (resp. $v_m \prec v_l$) implies $v_{k+1} \prec v_l \prec v_m$ (resp. $v_i \prec v_m \prec v_l$), a contradiction to $v_{k+1} \sim v_l$ (resp. $v_m \sim v_i$). If $v_l \sim v_m$, then there is an induced cycle v_i, v_{k+1}, v_m, v_l with $v_i \prec v_m$ and $v_{k+1} \prec v_l$. This is a contradiction to $v_i \gg v_l \oplus v_{k+1} \gg v_m$, hence the property persists. The correctness of the algorithm is shown if the result is a simple triangle representation of $G[\{v_1, \ldots, v_{k+1}\}]$. Step 3 is executed until v_{k+1} is connected to all its neighbours v_i with $i < k+1$. Again every update is allowed until z is the right corner of a triangle T_j with $v_j \not\sim v_{k+1}$. If there exists a vertex v_i with $r_i < r_j$, such that $v_i \sim v_{k+1}$, then $v_{k+1} \in N(v_i)$ and

$$N(v_i) \cap \{v_{\max(i,j)+1}, \ldots, v_n\} \subseteq N(v_j) \cap \{v_{\max(i,j)+1}, \ldots, v_n\}$$

is violated by v_{k+1}. Hence the theorem is proven inductively. □

Instead of using the algorithm described in the proof above it is possible to calculate the interval representation on L_2 directly using the following theorem.

Theorem 4.3
Let $C_1, \ldots, C_k$ be the non-chordal cycles of length 4 of G. The cycle C_i consists of the vertices a_i, b_i, c_i and d_i where $a_i \prec c_i$ and $b_i \prec d_i$. Additionally assume that $c_i \ll_1 b_i$ and thus $a_i \ll_1 c_i \ll_1 b_i \ll_1 d_i$. Then $\vec{G} := (V(G), E(G) \setminus \{b_i c_i | i \in \{1, \ldots, k\}\})$ is an interval graph with induced partial order $\ll_2 = \prec \cup \{(b_i, c_i) | i \in \{1, \ldots, k\}\}$.

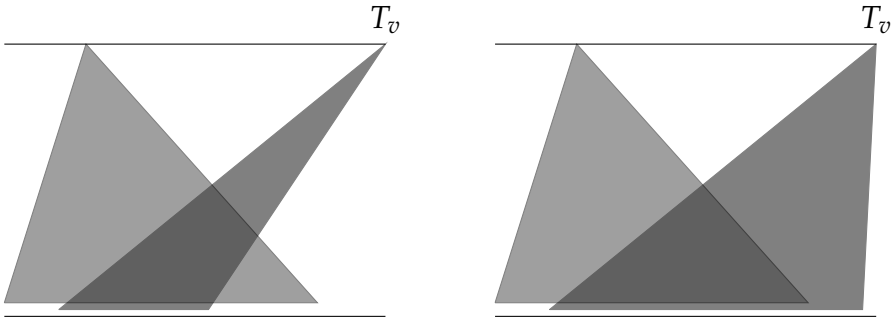


Figure 30: Extension of the interval constructed in proof of 4.3.

Proof. By Theorem 4.2, there exists a simple triangle representation of G, such that the order of the corners on L_1 is given by $\ll_1$. In the incomparability graph G' of the interval order $\ll_2$ on L_2 the vertices b_i and c_i are not adjacent. To this end, since

$a_i \sim d_i$ and $a_i \ll_1 d_i$ holds by assumption, we have $l_{d_i} \ll_2 r_{a_i}$ and because of $b_i \ll_2 d_i$ we have $l_{b_i} \ll_2 r_{a_i}$. With $a_i \ll_2 c_i$ we have $r_{b_i} \ll_2 l_{c_i}$, thus $b_i \ll_2 c_i$. Now assume there are two vertices v, w that are adjacent in G' but $v \ll_2 w$. If there is no triangle T_x with $T_x \not\sim T_v$ and $r_v \ll_2 l_x \ll_2 l_w$ then the interval of v can be extended by setting $r_v := succ(l_w)$. Otherwise if there is no triangle T_y with $T_y \not\sim T_w$ and $r_v \ll_2 r_y \ll_2 l_w$ then the interval of w can be extended by setting $l_w := prec(r_v)$.

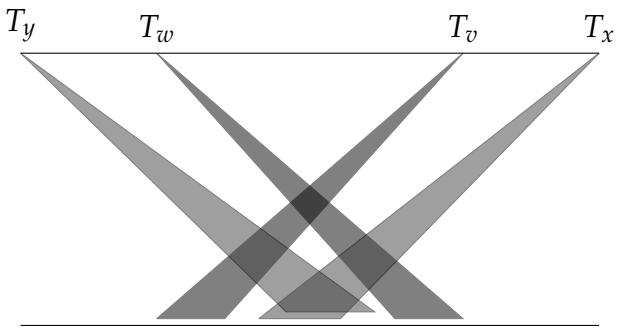


Figure 31: Simple triangle rep. constructed in proof of Theorem 4.3.

If both cases do not apply, then extend both intervals in the direction of the other one in the triangle graph, i.e. while this does not change the triangle representation, set $r_v := succ(s)$ and $l_v := prec(t)$, where s is the next corner on the right of r_v and the next corner on the left of l_v on L_2. If the result is not a triangle representation with $l_w \ll_2 r_v$, then $succ(r_v)$ is the left corner of a triangle T_x with $x \not\sim v$ and $prec(l_w)$ the right corner of a triangle T_y with $y \not\sim w$. Because of $y \prec w$ and $v \prec x$ we have $y \ll_1 v$ and $w \ll_1 x$, see Figure 31. Hence there exists an induced cycle y, v, w, x with $y \ll_1 w \ll_1 v \ll_1 x$, a contradiction to $vw \in E(G')$. □

Theorem 4.2 gives a tool to calculate a simple triangle representation of a simple triangle graph and to show that some graphs are not simple triangle graphs.

Example 4.4
Figure 32 shows a triangle representation of a graph, that is not a simple triangle graph.

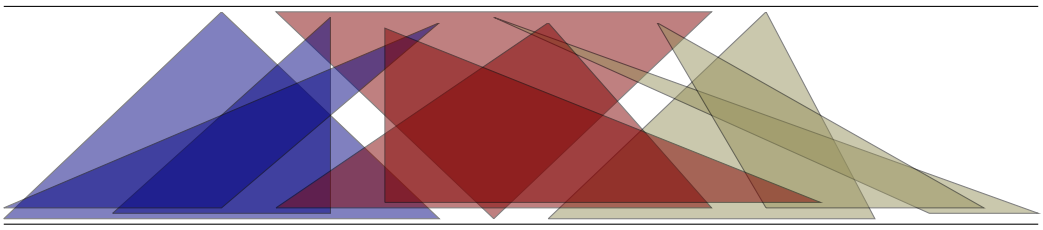


Figure 32: A triangle representation of a non-simple triangle graph.

The first two cliques (blue and red) induce a graph, that is isomorphic to the graph in Figure 33. The order of the corners on L_1 is (up to symmetry) unique: a top corner of a blue triangle is followed by the top of the red triangle, that is not adjacent to the blue one. Assume x_1, y_1 and z_1 are the blue triangles and x_2, y_2, z_2 the red ones, such that $x_1 \prec x_2$, $y_1 \prec y_2$ and $z_1 \prec z_2$. without loss of generality assume that $x_1 \ll_1 y_1 \ll_1 z_1$. Then we have $x_2 \ll_1 y_2 \ll_1 z_2$ because of 4.1 and moreover $x_1 \ll_1 x_2 \ll_1 y_1 \ll_1 y_2 \ll_1 z_1 \ll_1 z_2$. The red and yellow vertices (x_3, y_3, z_3 with $x_2 \prec x_3$,$y_2 \prec y_3$ and $z_2 \prec z_3$) also induce a to Figure 33 isomorphic graph, whence $x_2 \ll_1 x_3 \ll_1 y_2 \ll_1 y_3 \ll_1 z_2 \ll_1 z_3$. Now $x_3 \ll_1 y_2$ and $y_2 \ll_1 z_1$ give $x_3 \ll_1 z_1$, which is a contradiction to $z_1 \prec x_3$. So there exists no simple triangle representation for this graph.

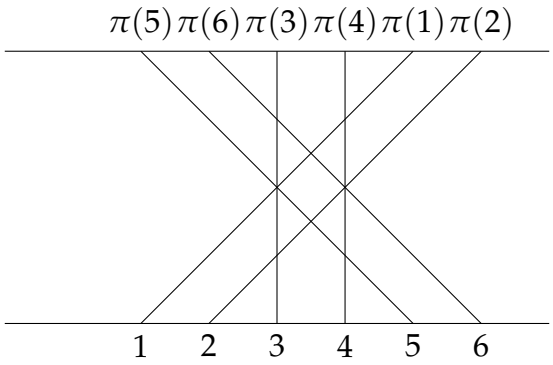


Figure 33: Permutation rep. of induced subgraph described in the example.

Hence the problem of recognising a simple triangle graph can be solved by finding a solution for a 2-satisfiability problem (2SAT), where the variables are given in the form $t_a < t_b$ with the additional condition that a solution has to lead to a total order,

i.e. the directed graph

$$\vec{G} := (\{t_v | v \in G\}, \prec \cup S),$$

where

$$S := \{(t_a, t_b) | t_a < t_b \text{ is given by the solution of the 2SAT}\}$$

has to be acyclic.

This leads to a generalised formal description of the problem.

Definition 4.5
Problem: An undirected graph $G = (V, E)$ and conditions of the form

$$(a, b) \in \vec{E} \vee (c, d) \in \vec{E},$$

along with

$$(a, b) \in \vec{E} \oplus (b, a) \in \vec{E} \text{ if } \{a, b\} \in E,$$

is given.
Question: Is there a possible choice for $\vec{E}$, such that $\vec{G} = (V, \vec{E})$ is acyclic?

This problem will be referred to as *acyclic 2SAT*.

Unfortunately determining a solution for a acyclic 2SAT is difficult in general.

Theorem 4.6
The acyclic 2SAT problem is NP-complete.

Proof. The theorem is shown by a reduction of not-all-equal-3SAT (NAE-3SAT). NAE-3SAT is a 3SAT with the additional condition that not all literals in a clause are allowed to be true. It is NP-complete, see [GJ79].

Let C_i be the clauses consisting of the literals $l_{i,1}, l_{i,2}$ and $l_{i,3}$. Then build the graph $G_i = (V_i, E_i)$, which is the complete graph on the vertices

$$V_i = \{x_i, y_i, z_i\}.$$

The literals are identified with on edge of this graph, e.g. $l_{i,1}$ with $\{x_i, y_i\}$, $l_{i,2}$ with $\{y_i, z_i\}$, and $l_{i,3}$ with $\{z_i, x_i\}$. The graph $G = (\bigcup G_i, \bigcup E_i)$ is embedded in the plane, such that x_i, y_i, z_i forms a clockwise cycle, see Figure 34.

If the literal l and l' are the same and l is associated with the edge $\{x, y\}$ and l' with $\{x', y'\}$ (where the edges are given in clockwise order) the conditions

$$(x, y) \in \vec{E} \vee (y', x') \in \vec{E},$$
$$(y, x) \in \vec{E} \vee (x', y') \in \vec{E}$$

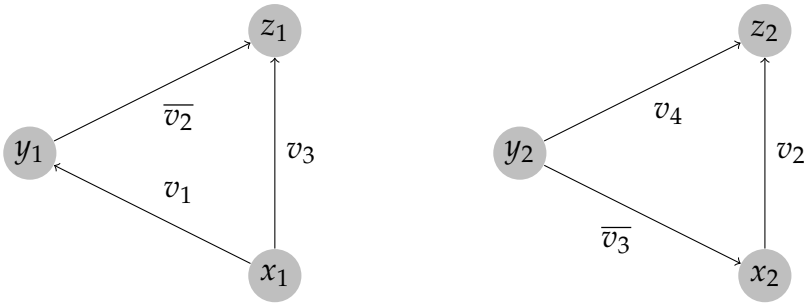


Figure 34: Reduction of the NAE-3SAT instance $(v_1 \vee \overline{v_2} \vee v_3) \wedge (\overline{v_3} \vee v_4 \vee v_2)$ with solution v_1, v_4 true and v_2, v_3 false.

are added. If the literal l is the negation of l' the conditions

$$(x, y) \in \vec{E} \vee (x', y') \in \vec{E},$$
$$(y, x) \in \vec{E} \vee (y', x') \in \vec{E}$$

are included. The first conditions cause that the edge $\{x, y\}$ is directed clockwise if and only if $\{x', y'\}$ is directed clockwise if l and l' are the same. If l is the negation of l' then the one edge is directed clockwise if and only if the other is directed anticlockwise. A clockwise orientation is considered as true, an anticlockwise orientation as false.

The constructed acyclic 2SAT problem is solvable if and only if the underlying not-all-equal-3SAT is solvable: Given a not-all-equal-3SAT solution a literal of the variable x is directed clockwise if the literal is true and anticlockwise otherwise. This satisfies the conditions above by construction. The directed graph is acyclic, since a clockwise (resp. anticlockwise) cycle in a connected component, that would imply that every literal of the not-all-equal-3SAT solution is true (resp. false), a contradiction. Analogously a solution of the acyclic 2SAT gives a solution for the not-all-equal-3SAT problem by seting a variable x true if associated edges of x are directed clockwise and of $\overline{x}$ anticlockwise and false otherwise. Again, all literals true (resp. false) leads to a clockwise (resp. anticlockwise) cycle in G.

If a given orientation of G satisfies the acyclic 2SAT can be checked in polynomial time, because 2SAT solving and topological sorting can be done in polynomial time. Hence, the acyclic 2SAT problem is NP-complete. □

The proof also shows, that the problem stays NP-complete, if the 2SAT is assumed to be an XOR-2SAT – which is a linear equation system in $\mathbb{Z}/2\mathbb{Z} = \mathbb{F}_2$ – i.e. every $\vee$ is replaced with $\oplus$.

Nevertheless, the instances of the problem given by trapezoid graphs may be solvable efficiently in virtue of the strong underlying structure of trapezoid graphs. In the following some neighbourhood relations, that guarantee acyclicity are discusse.

The transitivity of the order can not be formulated in terms of a 2-satisfiability problem. Some parts with one fixed edge can added to the 2SAT.

Lemma 4.7
Let $G = (V, E)$ be a cocomparability graph with partial order $\prec$ and the total order $<$ of Lemma 4.2 , $x, y, z \in V$ with $x \prec y$, $z \sim x$ and $z \sim y$. Then

a) $(t_z < t_x) \Rightarrow (t_z < t_y)$,

b) $(t_y < t_z) \Rightarrow (t_x < t_z)$.

Proof. Because of $x \prec y$ we have $t_x < t_y$, hence the lemma follows since $<$ is transitive. □

The lemma above gives the transitivity for some parts a solution of the 2SAT, i.e. a better chance of an acyclic graph, if only the 2SAT is solved. Also the examples from Figure 28 and Example 4.4 can be proven not to be simple triangle graphs, since the resulting extended 2SAT's are not satisfiable.

The (acyclic) 2SAT given by Lemma 4.1 and Lemma 4.7 is referred to as extended 2SAT.

The directed graph given by the solution of the extended 2SAT is still not acyclic in general.

Example 4.8
Consider the graph shown in Figure 35.

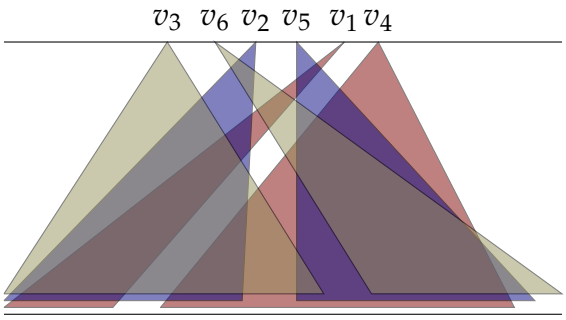


Figure 35: Example: Cyclic and acyclic solutions of the 2SAT.

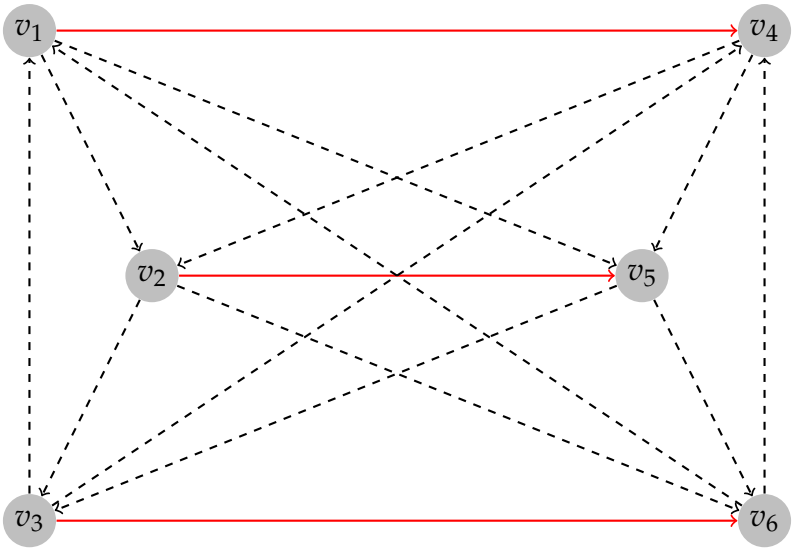


Figure 36: Red denotes the partial order, black the solution of the 2SAT.

Obviously, the order of the corners on L_1 satisfies all the conditions of Lemma 4.2, but there is a possible solution of the extended 2SAT that is not an order. This solution is shown in Figure 36.

Remark 4.9
Lemma 4.7 can also be applied if one edge between the vertices can be fixed because of other reasons than $\prec$, e.g. one variable takes the same value in all possible solutions of the extended 2SAT.

Some structured subgraphs give some more information on a possible solution or even avoid cycles in any solution. A few of them are introduced in the following.

For a cocomparability graph G with partial order $\prec$ and a clique $C \subseteq V(G)$ define

$$N^- := \{v \in N(x) | x \in C \wedge \exists y \in C : v \prec y\}.$$

The vertices in N^- are the *left neighbours* of C. Analogously the *right neighbours*

$$N^+ := \{v \in N(x) | x \in C \wedge \exists y \in C : y \prec v\}$$

are defined. For all neighbours of a clique the notation

$$N(C) := \bigcup_{v \in C} N(v) \setminus C$$

is used.

Proposition 4.10
a.) $N^+(C) \cap N^-(C) = \emptyset$

b.) $N^+(C) \cup N^-(C) = N(C) \Leftrightarrow C$ is a maximal clique.

Proof. a.) Assume there exists $v \in N^+(C) \cap N^-(C)$. Then there exists $x, y \in C$, such that $x \prec v$ and $v \prec y$. Hence $x \prec y$. A contradiction since C is a clique.

b.) Assume C is not maximal. Then there is $v \in V(G) \setminus C$, such that $v \sim x$ holds for all $x \in C$. Hence we have $v \notin N^-(C)$ and $v \notin N^+(C)$ but $v \in N(C)$.
On the other hand, a vertex $v \in N(C) \setminus (N^+(C) \cup N^-(C))$ is adjacent to every $x \in C$, hence C is not maximal. □

Trapezoid graphs can be characterised by the right (or analogously left) neighbourhood, see [MS94]. The characterisation of trapezoid graphs is a generalisation of a property of interval graphs.

Lemma 4.11
Let G be a trapezoid graph, C a clique in G and R a trapezoid representation according to the partial order $\prec$.
There are vertices $v, w, x, y \in C$, such that

$$N^-(C) \subseteq N(v) \cup N(w)$$

and

$$N^+(C) \subseteq N(x) \cup N(y).$$

In an interval graph there is one vertex v, that covers N^- and a vertex x that covers N^+. This has been implicitly used as sufficient property for interval graphs in the proof of Theorem 4.2.

Proof. Let $T_v = (t_v, l_v, r_v)$ be the triangle in the trapezoid representation R, such that t_v is the leftmost corner of all vertices of C on L_1 and $T_w = (t_w, l_w, r_w)$ the triangle, such that l_w is the leftmost corner on L_2 of all triangles in C, see Figure 37. Now assume there exists $z \in N^-(c)$, such that $z \not\sim v$ and $z \not\sim w$. By 4.10.a we have $z \prec v$ and $z \prec w$. With $T_z = (t_z, l_z, r_z)$ we have $t_z < t_v$ and $r_z < l_w$. Because of $t_v \prec t_u$ and $l_w < l_u$ for every $T_u = (t_u, l_u, r_u)$, we have $z \prec u$ for every $u \in C$, a contradiction to $z \in N^-(C)$. □

Lemma 4.11 gives a possibility to characterise a few structures, where no cycle occurs.

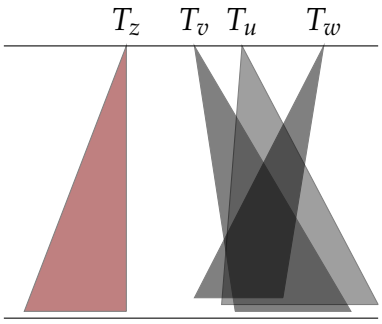


Figure 37: Trapezoid rep. in proof of 4.11.

Lemma 4.12
Let $C = \{x, y, z\}$ be a clique, such that $N^+(C) \subseteq N(x) \cup N(y)$ and neither $N^+(C) \subseteq N(x) \cup N(z)$ nor $N^+(C) \subseteq N(y) \cup N(z)$. Then x, y and z do not form a cycle in a solution of the 2SAT.

Proof. Because of $N^+(C) \supset N(x) \cup N(z)$ there is $a \in N(y)$, such that $x \prec a$ and $z \prec a$. Analogously we have $b \in N(x)$, such that $y \prec b$ and $z \prec b$. So we have a cycle x, b, a, y and either $b \ll_1 x$ or $a \ll_1 y$ by Lemma 4.1. Hence we have either $z \ll_1 x$ and $y \ll_1 x$ or $z \ll_1 y$ and $x \ll_1 y$. In both cases no cycle is possible. □

Another case where acyclicity is obtained is the following.

Lemma 4.13
Let $C = \{x, y, z\}$ be a clique, such that $N^+(C) \subseteq N(x) \cup N(y)$ and $N^+(C) \subseteq N(x) \cup N(z)$ but neither $N^+(C) \subseteq N(y) \cup N(z)$ nor $N^+(C) \subseteq N(x)$. Then x, y and z do not form a cycle in a solution of the 2SAT.

Proof. Since there is $a \in N^+(C)$ with $a \notin N(x)$ we have $a \in N(y)$ and $a \in N(z)$. On the other hand, there is $b \in N(x)$, such that $b \notin N(y) \cup N(z)$. Hence x, b, a, y and x, b, a, z form a non-chordal cycle. As before we have $x \ll_1 y$ if and only if $x \ll_1 z$ and so no cycle occurs on x, y, z. □

In the two preceding lemmata $N^+(C)$ can be replaced by $N^-(C)$. This refers to the same statements on the reversed partial order of G. A consequence of these properties is the following observation.

Observation 4.14
Let G be a trapezoid graph, where every 3-clique satisfies the assumptions of Lemma 4.12 or Lemma 4.13. Then every solution of the extended 2SAT leads to a solution of the acyclic 2SAT.

In the following, the possibility of repairing cycles in a solution of the 2SAT is analysed.

Definition 4.15
Let $\prec$ be a partial order on V, then define the height function $h_\prec : V \to \mathbb{N}_0$ by

$$\{v \in V | h_\prec(v) = 0\} = \{v \in V | \nexists w \in V : w \prec v\}$$

and

$$h_\prec(v) = k \Leftrightarrow h_{\prec_{k-1}}(v) = 0,$$

where $\prec_{k-1}$ is $\prec$ restricted on the set $V_{k-1} := V \setminus \{v \in V | h_\prec(v) \leq k-1\}$.

Hence, we can denote the vertices of height 0 as the *first clique*.

Lemma 4.16
Let $\ll_1$ be a non acyclic solution the 2SAT. Let C be the first clique in $G[S]$ according to $\prec$ where S is a strongly connected component with $|S| \geq 2$. Then C has no minimal vertex (according to $\ll_1$).

Proof. Assume there is a vertex $v \in C$, such that $v \ll_1 c$ holds, for all $c \in C \setminus \{v\}$. Since $v \in S$ there exists $x \in S$ with $x \ll_1 v$. By assumption $x \notin C$, hence there is $c \in C$ with $c \prec x$. This gives the implication in the 2SAT $x \ll_1 v \Rightarrow c \ll_1 v$, a contradiction since v is assumed to be minimal. □

Thus, it is an idea to repair a cycle in the first clique of a strongly connected component.

Lemma 4.17
Let S be a strongly connected component in $\vec{G}$, then every new order $\ll_S$ on S that satisfies the 2SAT restricted to S, fulfils the 2SAT on G together with $\ll_1$ unless there exists a non-chordal cycle a, b, c, d in G with $b, c \in S$ $a \prec c$ and $b \prec d$, such that $a \ll_1 s$, for all $s \in S$, or $s \ll_1 d$, for all $s \in S$.

Or, in other words, in the case described above the graph is a simple triangle graph if $G[S]$ is a simple triangle graph for every strongly connected component S in a solution of the 2SAT.

Proof. Consider a solution of the 2SAT with a strongly connected component S in the resulting directed graph $\vec{G}$ with $|S| \geq 2$. Now consider an order $\ll_S \subset \prec$ on S, such that $G[S]$ has a simple triangle representation with $\ll_S$ as order on the tops. Replacing the variables corresponding to two vertices in S by the value given in $\ll_S$

leads to another solution of the 2SAT:
There are two kinds of conditions in the 2SAT, those from Lemma 4.1 and those from Lemma 4.7. Since the resulting relation is an order, hence acyclic, Lemma 4.7 is satisfied. For Lemma 4.1 consider the vertices of a cycle a, b, c, d with $a \prec c$ and $b \prec d$. Without loss of generality assume that $a \ll_1 c \ll_1 b \ll_1 d$ holds. Now consider the combination of vertices that can be contained in S.
If all four vertices are contained in S, then the 2SAT is satisfied for this cycle, since $\ll_S$ is a solution of the 2SAT. With $a, c \in S$ and $b, d \notin S$, resp. $a, c \notin S$ and $b, d \in S$, the relation $a \ll_S c$, resp. $b \ll_S d$ holds because of $a \prec c$ resp. $b \prec d$. If only one vertex is contained in S the order in the cycle does not change. Since $b, c \in S$ and $a \notin S$ or $d \notin S$ is forbidden by assumption, the lemma is proven. □

Now we consider trapezoid graphs with maximal clique size 3. If, for these graphs, the 2SAT is satisfiable and the influence on one strongly connected component by Lemma 4.17 can be limited, then it is a simple triangle graph.

Theorem 4.18
Let G be a trapezoid graph with $\omega(G) = 3$. If for a solution of the 2SAT $\ll_1$ every strongly connected component is not influenced by Lemma 4.17 and does not contain a vertex $v \in S$, such that $v \ll_1 s$ for every $s \in S$ satisfies the 2SAT, then G is a simple triangle graph.

Proof. Let S be the first strongly connected component with $|S| \geq 2$ in the directed graph $\vec{G}$ given by the solution of the 2SAT. The first clique C of $G[S]$ according to $\prec$ has no minimal element by Lemma 4.16, hence $|C| = 3$, e.g. $C = \{x, y, z\}$ and $x \ll_1 y$, $y \ll_1 z$ and $z \ll_1 x$. Without loss of generality assume that there is no vertex $v \in C$, such that there exists no $w \in S \setminus C$, such that $w \ll_1 v$. Otherwise set $v \ll_1 i$, for all i in $C \setminus \{v\}$. Hence for every $v \in C$ there exists a vertex w_v, such that $w_v \ll_1 v$. Because of $\omega(G) = 3$ there is a $v \in C$, such that $v \not\sim w_x$. Since $x \ll_1 y$ and $y \prec w_x$ would imply $x \ll w_x$ we have $y \sim w_x$, hence $y \not\sim w_x$. This also shows $y \ll_1 w_x$ since the contrary implies $z \sim w_x$ as before. So we have $w_x \ll_1 x$, $y \ll w_x$, $y \sim w_x$ and $z \prec w_x$.

The analogous properties are also true for w_y and w_z, i.e. we have $w_y \ll_1 y$, $z \ll w_y$, $z \sim w_y$ and $x \prec w_y$ as well as $w_z \ll_1 z$, $x \ll w_z$, $x \sim w_z$ and $y \prec w_z$.

Moreover, these properties hold for every vertex $w \in S \setminus C$ with $w \ll_1 c$ for one $c \in C$. Obviously the sets W_x, W_y and W_z with

$$W_c := \{w \in S \setminus C | w \ll_1 c\}$$

are disjoint.

Now we show that for any three vertices $w_x \in W_x$, $w_y \in W_y$ and $w_z \in W_z$ the set $\{w_x, w_y, w_y\}$ is a clique: Assume the contrary. Then, without loss of generality $w_x \not\sim w_y$. If $w_x \prec w_y$ then $w_y \ll_1 y$ implies $w_x \ll_1 y$ or if $w_y \prec w_x$ then $w_x \ll_1 x$ implies $w_y \ll_1 x$, a contradiction. Now assume $a, b \in W_x$ are adjacent. Then $w_y \in W_y$ and $w_z \in W_z$ form a clique $\{a, b, w_y, w_z\}$, a clique of size 4, which is ruled out.

For any $w_x \in W_x$, $w_y \in W_y$ and $w_z \in W_z$ we have $w_y \ll_1 w_x$, $w_y \ll_1 w_z$ and $w_z \ll_1 w_x$: Because of $x \prec w_y$ and $y \prec w_z$ we have a cycle x, y, w_y, w_z, hence $x \ll_1 y$ gives $w_y \ll_1 w_z$. Analogously $x \prec w_y$ and $z \prec w_x$ leads to $w_x \ll_1 w_y$, and $y \prec w_z$ together with $z \prec w_x$ to $w_z \ll_1 w_x$.

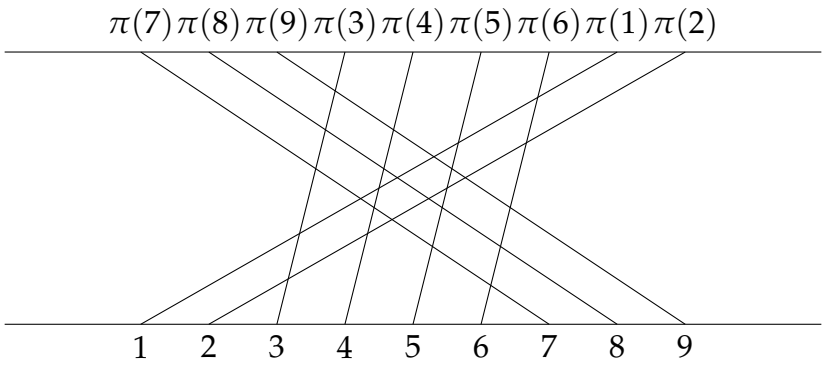


Figure 38: Permutation rep. of graph constructed in proof of 4.18.

Assume there is a $v \notin W_x \cup W_y \cup W_z \cup C$ with $v \prec w_x$ for one $w_x \in W_x$. Then $v \sim w_y$, $w_y \in W_y$ since $w_x \ll_1 w_y$. Then $w_y \ll_1 v$ (otherwise $v \sim w_z$, a contradiction to the clique size) and $w_z \prec v$ and $v \in W_z$. For symmetry reasons, there is also no v with $v \ll_1 w_y$ or $v \ll_1 w_z$. Hence $S = C \cup W_x \cup W_y \cup W_z$. So $G[S]$ is a permutation graph and any order on $W_x \cup \{z\}, W_y \cup \{x\}, W_z \cup \{y\}$ satisfies the 2SAT, see Figure 38. □

4.2 A backtracking algorithm

Based on Theorem 4.2 it is easy to formulate an algorithm that determines whether a given graph is a simple triangle graph and gives a representation if possible.

The basic idea will be to solve the 2SAT step by step and add some information about new transitivity information afterwards.

First some knowledge about the ideas to solve a 2SAT is required:

Given a 2SAT its *implication graph* I is defined as

$$V(I) := \{x, \overline{x} | x \text{ is variable of the 2SAT}\}$$

and

$$E(I) := \{(\overline{x}, y), (\overline{y}, x) | x \vee y \text{ is clause of the 2SAT}\}.$$

The idea behind the construction of the implication graph is that $x \vee y$ is logically equivalent to $\overline{x} \Rightarrow y$ resp. $\overline{y} \Rightarrow x$. Hence a directed edge corresponds to an logical implication.

The following theorem gives the basic idea of an algorithm for solving 2SAT problems.

Theorem 4.19 ([CLR00])
A 2SAT is solvable if and only if none of its implication graph's strongly connected components contains x and $\overline{x}$ for any x.

Proof. A strongly connected component containing x and $\overline{x}$ contains a path from x to $\overline{x}$ and one from $\overline{x}$ to x, i.e. $x \Rightarrow \overline{x}$ and $\overline{x} \Rightarrow x$, a contradiction. For the converse consider the *condensation* of I, i.e. a directed graph $C(I)$, where the set I is replaced by one vertex or more precisely

$$V(C(I)) := \{S | S \text{ is strongly connected component of } I\}$$

and

$$E(C(I)) := \{(S_1, S_2) \in V(C(I)) \times V(C(I)) \mid \exists (x, y) \in E(I) : x \in S_1 \neq S_2 \ni y\}.$$

The condensation is acyclic, hence topological sorting leads to an order of strongly connected components $S_1, \ldots, S_k$, such that there is no path from any vertex in S_l to any vertex in S_m if $l > m$. Now choose the variables iteratively in the determined order, such that S_l contains no true value if there is not already a true one. If a variable is already chosen, such that S_l contains a true value then choose the variables, such that S_l contains no false value.

The correctness of the algorithm above relies basically on the symmetry of the implication graph: For a path from x to y exists a path from $\overline{y}$ to $\overline{x}$. Assume there is a variable x set true but there exists a path from x to $\overline{x}$. Then w.l.o.g. there is a variable y set false, such that $\overline{y} \in S_x$ where S_x is the strongly connected component of x. Since there is a path from $\overline{y}$ to $\overline{x}$ and there is also a path from x to y and hence one from $\overline{y}$ to y. Since y and $\overline{y}$ are not in the same strongly connected component y is not set before $\overline{y}$, a contradiction to the assumption. □

The described algorithm, that uses a complete topological sorting in the beginning, is not ideal for a backtracking algorithm that gives a simple triangle representation if existing, since it cannot adapt on new transitivity conditions emerging by the choice of variables. The topological sorting assures that x is chosen to be false if $\overline{x}$ is one of its successors in the implication graph for any x. This can be done manually by checking whether a depth-first-search tree starting in x contains y and $\overline{y}$. If this is the case there is no possible solution for the 2SAT, where x is true, otherwise x and its successors can be set true, which is equivalent to setting all predecessors of $\overline{x}$ false. After the fixing of a variable the graph $\vec{G}$ can be updated with a new transitivity according to 4.9 and the transitive closure can be calculated, which corresponds to the choice of new variables. Iterative calculating of the transitive closure and logical consequences leads either to a consistent state, where a new variable can be chosen or an contradiction occurs, such that x cannot be chosen to be true.

For the runtime of the algorithm let us denote $|V(G)|$ with n. It is possible make the preparations for this algorithm in $O(n^4)$: Calculating a partial order of a trapezoid graph is possible in $O(n^2)$ by [MS94]. Building the 2SAT requires $O(n^4)$ steps. A simple algorithm, that checks every 4-tuple of vertices for cycles in G, runs in $O(n^4)$.

For the runtime of the whole algorithm consider the runtime of one decision step. After one variable is chosen, the algorithm performs a depth-first-search on I to calculate logical implications by the 2SAT. A depth-first-search on I, a graph with $n(n-1)/2$ vertices, requires up to $O(V(I)+E(I))$, hence $O(n^4)$, steps. Afterwards calculating the transitive closure of $\vec{G}$ is possible in $O(n^c)$ steps, where c is a constant, such that matrix multiplication of a $n \times n$-matrix runs in $O(n^c)$, see [CLR00]. Since the smallest c is known to be smaller than 3, see [VZGG03] for instance, one round of calculating the transitive closure and logical implications requires $O(n^c) + O(n^4) \in O(n^4)$ steps. These rounds have to be repeated until no new variable is set. This is limited by the number of variables, hence the number of rounds can be limited by $n(n-1)/2$. Thus, the runtime in one decision step can be limited by $O(n^6)$.

The runtime of the whole algorithm massively depends on the number of backtracking steps. If every choice of variables must be tried, the decision tree grows exponentially in the depth, i.e. $2^{O(n^2)}$ nodes are searched. If the algorithm finds a solution directly, the decision tree is a path, i.e. $O(n^2)$ nodes are searched. This leads to a worst case runtime of $O(n^6 2^{n^2})$. An implementation of the algorithm described has not needed a backtracking step on a set of example graphs. If this would be true in general the algorithm would run in $O(n^8)$ steps, hence.

§ 5 Conclusion

In this work the structure of different partial orders on one cocomparability graph was reconsidered. With this knowledge an algorithm to transform one partial order into another one was constructed. The steps of this algorithm were replicable on a simple triangle representation. Hence, it was proven that every order on a simple triangle graph is induced by a simple triangle representation. This led to the fact that every partial order on a simple triangle graph leads to a simple triangle representation.

An approach for an algorithm, that can determine such a representation was given in the last chapter. Therefore, a criterion on a realiser of the partial order of the cocomparability graph was given to ensure that a simple triangle representation with this realiser as order of the top corners exists. This was done by observing the consequences of reordering two vertices through the graph.
The attempt to calculate such a realiser led to the acyclic 2SAT problem – a combination of a 2SAT and topological sorting. The acyclic 2SAT problem was shown to be NP-complete.

If the recognition of simple triangle graphs is NP-complete, the reduction of NAE-3SAT on acyclic 2SAT gives an idea of the graphs that has to be rebuild. Possibilities to copy a variable were given in Lemma 4.1. Attempts to construct a trapezoid graphs, such that there is a clique that builds a cycle in every solution of the 2SAT failed. For an example see Figure 39.

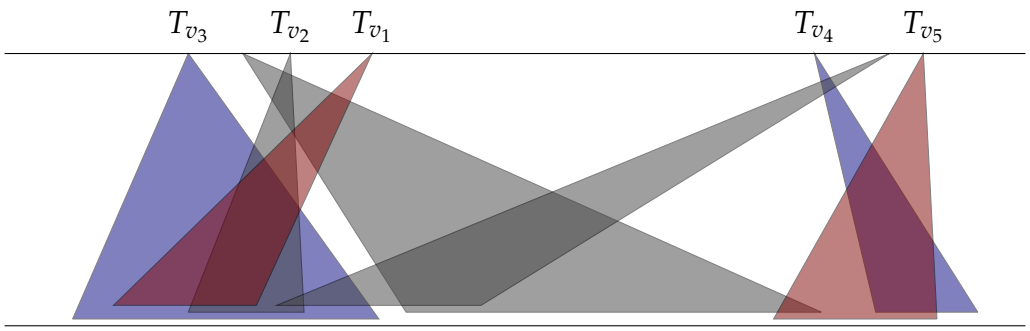


Figure 39: Copying the order between red and blue.

The grey triangles intersecting T_{v_4} resp. T_{v_5} form circles, such that the order of the red and blue triangles is the same. Attempts to use this construction to copy the

relation of the tops of v_1, v_2 and v_3 failed. Trying a copy of the construction with the grey triangles, for all three edges in the clique, lead to an influence between the grey Triangles.

An idea for an efficient algorithm is to have a look at the structure of the acyclic 2SAT, that is constructed from a trapezoid graph. Maybe it is possible to construct a comparability graph with an partial order, that leads to a partial order on the tops. This idea is based on the fact, that in [Gol04] it is shown, that calculating a partial order on a graph is basically solving an acyclic 2SAT, that has always a solution if the 2SAT is solvable.

References

[BK96] H. Breu and D. Kirkpatrick. On the complexity of recognizing intersection and touching graphs of disks. In *Graph Drawing*, pages 88–98. Springer, 1996.

[BL01] Andreas Brandstädt and Van Bang Le. *Graph-Theoretic Concepts in Computer Science*. Springer-Verlag Berlin Heidelberg, 2001.

[BNBYF+01] A. Bar-Noy, R. Bar-Yehuda, A. Freund, J. Naor, and B. Schieber. A unified approach to approximating resource allocation and scheduling. *Journal of the ACM (JACM)*, 48(5):1069–1090, 2001.

[BS+99] A. Brandstädt, J.P. Spinrad, et al. *Graph classes: a survey*. Number 3. Society for Industrial Mathematics, 1999.

[CCJ90] B.N. Clark, C.J. Colbourn, and D.S. Johnson. Unit disk graphs. *Discrete mathematics*, 86(1):165–177, 1990.

[CLR00] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to algorithms*. The MIT electrical engineering and computer science series. MIT Press [u.a.], Cambridge, Mass. [u.a.], 24. print. edition, 2000.

[CRSTm06] Maria Chudnovsky, Neil Robertson, Paul Seymour, and Robin Thomas. The strong perfect graph theorem. *Annals of Mathematics*, 164:51–229, 2006.

[dAdMG] S.M. de Almeida, C.P. de Mello, and A. Gomide. On the representation of a pi-graph.

[Dil50] R.P. Dilworth. A decomposition theorem for partially ordered sets. *The Annals of Mathematics*, 51(1):161–166, 1950.

[dR+] H. N. de Ridder et al. Information System on GraphClasses and their Inclusions (ISGCI). `http://www.graphclasses.org`.

[Gal67] T. Gallai. Transitiv orientierbare Graphen. *Acta Math. Acad. Sci. Hung.*, 18:25–66, 1967.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness (Series of Books in the Mathematical Sciences)*. W. H. Freeman, first edition edition, 1979.

[Gol04] Martin Charles Golumbic. *Algorithmic graph theory and perfect graphs*. Annals of discrete mathematics ; 57. Elsevier, Amsterdam, 2004.

[GRU83] M.C. Golumbic, D. Rotem, and J. Urrutia. Comparability graphs and intersection graphs. *Discrete Mathematics*, 43(1):37–46, 1983.

[GSW98] A. Gräf, M. Stumpf, and G. Weißenfels. On coloring unit disk graphs. *Algorithmica*, 20(3):277–293, 1998.

[HS95] M.L. Huson and A. Sen. Broadcast scheduling algorithms for radio networks. In *Military Communications Conference, 1995. MILCOM'95, Conference Record, IEEE*, volume 2, pages 647–651. IEEE, 1995.

[MBHI+95] M.V. Marathe, H. Breu, H.B. Hunt III, S.S. Ravi, and D.J. Rosenkrantz. Simple heuristics for unit disk graphs. *Networks*, 25(2):59–68, 1995.

[Mer11] George B. Mertzios. The Recognition of Triangle Graphs. In Thomas Schwentick and Christoph Dürr, editors, *28th International Symposium on Theoretical Aspects of Computer Science (STACS 2011)*, volume 9 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 591–602, Dagstuhl, Germany, 2011. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.

[MS94] T. H. Ma and J. P. Spinrad. On the 2-chain subgraph cover and related problems. *Journal of Algorithms*, 17(2):251–268, 1994.

[She95] N.A. Sherwani. *Algorithms for VLSI physical design automation*. Kluwer Academic Publishers, 1995.

[VZGG03] J. Von Zur Gathen and J. Gerhard. *Modern computer algebra*. Cambridge Univ Pr, 2003.

Erklärung

Ich versichere, die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt zu haben.

Aachen, den 30. März 2012