# Advanced Graph Theoretical Topics

Delpensum I238 - Institutt for informatikk

Pinar Heggernes

August 23, 2001

Many graph problems that are NP-complete on general graphs, have polynomial time solutions for special graph classes. In this document, a number of such graph classes are reviewed. In addition, some important graph theoretical definitions and notions are mentioned and explained.

## 1 Subsets, decompositions, and intersections

**Definition 1.1** *Given a graph $G = (V, E)$ and a subset $U \subseteq V$, the* subgraph *of $G$* induced *by $U$ is the graph $G[U] = (U, D)$, where $(u, v) \in D$ if and only if $u, v \in U$ and $(u, v) \in E$.*

**Definition 1.2** *A* tree-decomposition *of a graph $G = (V, E)$ is a pair*

$$(\{X_i \mid i \in I\}\,,\;\; T = (I, M))$$

*where $\{X_i \mid i \in I\}$ is a collection of subsets of $V$, and $T$ is a tree, such that:*

- $\bigcup_{i \in I} X_i = V$
- $(u, v) \in E \Rightarrow \exists i \in I$ *with* $u, v \in X_i$
- *For all vertices $v \in V$, $\{i \in I \mid v \in X_i\}$ induces a connected subtree of $T$.*

The last condition of Definition 1.2 can be replaced by the following equivalent condition:

- $i, k, j \in I$ and $j$ is on the path from $i$ to $k$ in $T \Rightarrow X_i \cap X_k \subseteq X_j$.

**Lemma 1.3** [6] *Let $(\{X_i \mid i \in I\}\,,\;\; T = (I, M))$ be a tree decomposition of $G = (V, E)$, and let $K \subseteq V$ be a clique in $G$. Then there exists an $i \in I$ with $K \subseteq X_i$.*

The *width* of a decomposition $(\{X_i \mid i \in I\}\,,\;\; T = (I, M))$ is $\max_{i \in I} |X_i| - 1$. The *treewidth* of a graph $G$ is the minimum width over all tree-decompositions of $G$.

**Corollary 1.4** *The treewidth of a graph $G$ is at least one less than the size of the largest clique in $G$.*

A *path-decomposition* is a tree-decomposition $(\{X_i \mid i \in I\},\ \ T = (I, M))$ such that $T$ is a path. The *pathwidth* of a graph $G$ is the minimum width over all path-decompositions of $G$.

**Example 1.5** *Let $G$ be the graph shown in Figure 1 a).*

*Let $I = \{1,2,3,4\}, X_1 = \{a,b,f\}, X_2 = \{b,d,f\}, X_3 = \{b,c,d\}, X_4 = \{d,e,f\}$. Let $T$ be the tree shown in Figure 1 b). $(\{X_i \mid i \in I\}, T)$ is a tree-decomposition of $G$.*

*Let $J = \{1,2,3\}, Y_1 = \{a,b,f\}, Y_2 = \{b,c,e,f\}, Y_3 = \{c,d,e\}$. Let $P$ be the path shown in Figure 1 c). $(\{Y_j \mid j \in J\}, P)$ is a path-decomposition of $G$.*

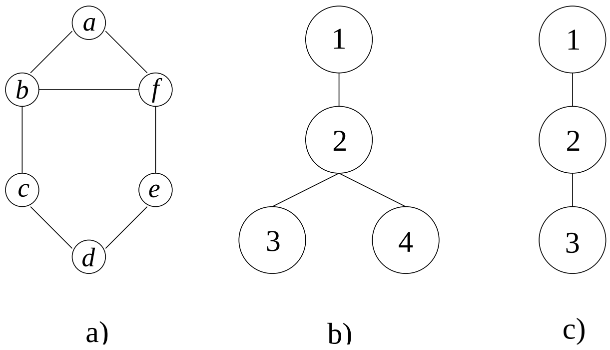


Figure 1: The graphs of Example 1.5.

The graph $G$ of Example 1.5 has treewidth 2 and pathwidth 2 (why?). Since every path decomposition is also a tree decomposition, pathwidth $\geq$ treewidth for all graphs.

**Theorem 1.6** [1] *The following problems are NP-complete:*

- *Given a graph $G = (V,E)$ and an integer $c < |V|$, is the treewidth of $G \leq c$?*
- *Given a graph $G = (V,E)$ and an integer $c < |V|$, is the pathwidth of $G \leq c$?*

The $c$-treewidth problem can be solved in polynomial time [1]: Given a graph $G$, is the treewidth of $G$ at most $c$? In this case, $c$ is a constant and not a part of the input.

We end this section with the definition of a minimal separator, which is central in coming sections. Given a graph $G = (V,E)$, a set of vertices $S \subset V$ is a *separator* if the subgraph of $G$ induced by $V - S$ is disconnected. The set $S$ is a $uv$-separator if $u$ and $v$ are in different connected components of $G[V - S]$. A $uv$-separator $S$ is minimal if no subset of $S$ separates $u$ and $v$.

**Definition 1.7** *S is a* minimal separator *of G if there exist two vertices u and v in G such that S is a minimal uv-separator.*

# 2 Partial $k$-trees

**Definition 2.1** *The class of* $k$-trees *is defined recursively as follows:*

- *The complete graph on $k$ vertices i a $k$-tree.*
- *A $k$-tree $G$ with $n+1$ vertices ($n \geq k$) can be constructed from a $k$-tree $H$ with $n$ vertices by adding a vertex adjacent to exactly $k$ vertices, namely all vertices of a $k$-clique of $H$.*

The following theorem gives several alternative characterizations of $k$-trees.

**Theorem 2.2** [15] *Let $G = (V, E)$ be a graph. The following statements are equivalent:*

- *$G$ is a $k$-tree.*
- *$G$ is connected, $G$ has a $k$-clique, but no $(k+2)$-cliques, and every minimal separator of $G$ is a $k$-clique.*
- *$G$ is connected, $|E| = k|V| - \frac{1}{2}k(k+1)$, and every minimal separator of $G$ is a $k$-clique.*
- *$G$ has a $k$-clique, but not a $(k+2)$-clique, and every minimal separator of $G$ is a clique, and for all distinct non-adjacent pairs of vertices $x, y \in V$, there exist exactly $k$ vertex disjoint paths from $x$ to $y$.*

**Definition 2.3** *A* partial $k$-tree *is a graph that contains all the vertices and a subset of the edges of a $k$-tree.*

**Theorem 2.4** [17] *$G$ is a partial $k$-tree if and only if $G$ has treewidth at most $k$.*

**Proof.** $\Rightarrow$: Let $G$ be a partial $k$-tree. We can assume that $G$ is a $k$-tree since the treewidth cannot increase for subgraphs. Let thus $G = (V, E)$ be a $k$-tree with $|V| > k+1$. There is a vertex $v \in V$ such that $G[V - \{v\}]$ is a $k$-tree, and the neighbors of $v$ induce a clique $K$ of size $k$ in $G$. Using induction, we can assume that $G[V - \{v\}]$ has treewidth at most $k$ with a corresponding tree decomposition $(\{X_i \mid i \in I\}, T = (I, M))$. (The base case of induction is when $G[V - \{v\}]$ is a complete graph on $k$ vertices; such a graph has clearly treewidth $k-1$.) By Lemma 1.3, there is an $i' \in I$ with $X_{i'}$ containing all neighbors of $v$ in $G$, with $K \subseteq X_{i'}$. Let $J = I \cup \{j\}$, where $j \notin I$, and let $X_j = K \cup \{v\}$. Now, $(\{X_i \mid i \in J\}, T = (J, M \cup \{i', j\}))$ is a tree decomposition of $G$ with width $k$.

$\Leftarrow$: Let $G = (V, E)$ be a graph with $|V| > k+1$, and let $(\{X_i \mid i \in I\}, T = (I, M))$ be a tree decomposition of $G$ of width at most $k$. We will prove, with induction on $|V|$, that there is a $k$-tree $H = (V, E')$ such that for every $i \in I$, $G[X_i]$ is a subgraph of a clique of $H$ with $k+1$ vertices. If $|V| = k+1$ then we are done. Otherwise, take a leaf node $l \in I$ and let $j \in I$ be the only neighbor of $l \in T$. If $X_l \subseteq X_j$, then we can remove $l$ from $T$ and continue with the remaining tree decomposition. Let $v$ be the only vertex in $X_l - X_j$ (otherwise $X_l$ can be divided into several tree nodes such that the resulting new leaf node satisfies this requirement). Note that $v$ does not appear in any $X_i$ for $i \neq l$. Thus all neighbors of $v$ must appear in $X_l$. Remember that $|X_l| \leq k+1$, thus $v$ has at most $k$ neighbors. Suppose that the induction hypothesis on $G[V - \{v\}]$ with tree decomposition $(\{X_i\} \mid i \in I, i \neq l\}, T[I - \{l\}])$ results in a $k$-tree $H'$. Since $v \notin X_j$, all neighbors of $v$ must belong to $X_j$. Therefore, the neighbors of $v$ induce a subgraph of a $k$-clique $C$ of $H'$ in $G$. Now, we can add $v$ with edges to all vertices of $C$ to $H'$ (which will make a $(k+1)$-clique), and get the desired $k$-tree $H$. □

Several problems that are NP-complete on general graphs have polynomial time algorithms for partial $k$-trees. We will look at one such example, INDEPENDENT SET [5]. In this problem we are looking for the maximum size of a set $X \subseteq V$ in a graph $G = (V, E)$ such that no two vertices belonging to $X$ are neighbors in $G$.

Given a tree decomposition of $G$, it is easy to make one that is binary with the same treewidth. Suppose that we have a binary tree decomposition $(\{X_i \mid i \in I\}, T = (I, M))$ of input graph $G$, with root $r$ and treewidth $k$. For each $i \in I$, let $Y_i = \{v \in X_j \mid j = i \text{ or } j \text{ is a descendant of } i\}$.

Note that if $v \in Y_i$, and $v \in X_j$ for some node $j \in I$ that is not a descendant of $i$ in $T$, then $v \in X_i$. Similarly, if $v \in Y_i$ and $v$ is adjacent to a vertex $w \in X_j$ in $G$ with $j$ not a descendant of $i$ in $T$, then $v \in X_i$ or $w \in X_i$ (or both). As a consequence, when we have an independent set $W$ of $G[Y_i]$, and we want to extend this to an independent set of $G$, then we only need to examine the vertices of $X_i$ rather than whole $Y_i$. The only important part is which vertices of $X_i$ belong to $W$, and we do not need to consider which vertices of $Y_i - X_i$ belong to $W$. Of the latter, only the *number* of the vertices in $W$ is important.

For $i \in I$ and $Z \subseteq X_i$, let $s_i(Z)$ be the maximum size of an independent set $W$ in $G[Y_i]$ with $W \cap X_i = Z$. Let $s_i(Z) = -\infty$ if no such independent set exists.

The algorithm to solve the independent set problem on $G$ basically consists of computing all tables $s_i$ for all nodes $i \in I$. This is done in a bottom-up manner in the tree $T$: each table $s_i$ is computed after the tables of the children of node $i$ are computed. For a leaf node $i$ in $T$, the following formula can be used to compute all $2^{|X_i|}$ values in the table $s_i$.

$$s_i(Z) = \begin{cases} |Z| & \text{if } \forall\, v, w \in Z : (v, w) \notin E \\ -\infty & \text{if } \exists\, v, w \in Z : (v, w) \in E \end{cases}$$

For an internal node $i$ with children $j$ and $k$, $s_i(Z)$ equals to:

$$\max_{Z \cap X_j = Z' \cap X_i \text{ and } Z \cap X_k = Z'' \cap X_i} \{s_j(Z') + s_k(Z'') + |Z \cap (X_i - X_j - X_k)| - |Z \cap X_j \cap X_k|\}$$

when $\forall\, v, w \in Z : (v, w) \notin E$, and $s_i(Z) = -\infty$ if $\exists\, v, w \in Z : (x, w) \in E$.

The idea behind the formula for internal nodes is to take the maximum over all sets $Z' \subseteq X_j$ that agree with $Z$ in which vertices of $X_i \cap X_j$ belong to the independent set, and similarly for $Z'' \subseteq X_k$. Vertices in $Z \cap (X_i - X_j - X_k)$ are not counted yet, so their number should be added, while vertices in $Z \cap X_j \cap X_k$ are counted twice, hence their number should be subtracted once.

For each node $i \in I$ the table $s_i$ is computed in a bottom-up order until the table $s_r$ is computed. Note that the maximum size of an independent set in $G$ is equal to $\max_{Z \subseteq X_r} s_r(Z)$, and can be found easily. This describes an algorithm that solves the independent set problem on $G$ in $O(2^{3k}n)$ time. It is also possible to construct the independent set itself using the standard dynamic programming techniques.

The idea behind this example is that each table entry gives information about an equivalence class of partial solutions. The number of such equivalence classes is bounded by some constant when the treewidth is bounded by a constant. Tables can be computed using only the tables of the children nodes. This idea and technique works for many examples.

# 3 Chordal graphs

In addition to the fact that several NP-complete problems have polynomial time solutions for chordal graphs, chordal graphs are a large and important class of graphs with applications in several areas.

**Definition 3.1** *A graph is* chordal *if every cycle of length* $> 3$ *has a chord.*

A *chord* is an edge joining two nonconsecutive vertices of a cycle. Equivalent to Definition 3.1, a chordal graph does not contain an induced subgraph isomorphic to $C_k$ for $k > 3$. Clearly, all induced subgraphs of a chordal graph are also chordal.

**Theorem 3.2** [7] *A graph $G$ is chordal if and only if every minimal separator of $G$ is a clique.*

**Proof.** $\Rightarrow$: Let $G = (V, E)$ be chordal and let $S$ be a minimal separator of $G$. Let $x$ and $y$ be any two vertices in $S$. We will show that $(x, y)$ must be an edge of $G$. Let $a$ and $b$ be the vertices for which $S$ is a minimal $ab$-separator, and let $A$ and $B$ be the connected components of $G[V - S]$ containing respectively $a$ and $b$. There must exist a path between $x$ and $y$ through vertices belonging to $A$. Let $p_1$ be a shortest such path. Let analogously $p_2$ be a shortest path between $x$ and $y$ through vertices of $B$. Paths $p_1$ and $p_2$ joined together make a cycle of length at least 4. Since $G$ is chordal, this cycle must have a chord. Since no edges exist between vertices of $A$ and vertices of $B$, the edge $(x, y)$ must be present and a chord of the mentioned cycle.

$\Leftarrow$: Let $G$ be a graph where each minimal separator is a clique. Assume that $G$ is not chordal, and let $w, x, y, z_1, ..., z_k, w$ be a chordless cycle of length at least 4 in $G$ ($k \geq 1$). Any minimal $wy$-separator of $G$ must contain $x$ and at least one $z_i$ for $1 \leq i \leq k$. Since all minimal separators are cliques, the edge $(x, z_i)$ must belong to $G$ contradicting that the mentioned cycle is chordless. □

**Definition 3.3** *A vertex is called* simplicial *if its adjacency set induces a clique.*

**Lemma 3.4** [7] *A chordal graph is either complete or has at least two nonadjacent simplicial vertices.*

**Proof.** Let $G$ be a chordal graph which is not complete. The proof is by induction on the number of vertices $n$. The base case is when $n = 2$ and $G$ has two isolated vertices that are both simplicial. Let $n > 2$ and assume that the lemma holds for all such graphs with fewer than $n$ vertices. Let $a$ and $b$ be two nonadjacent vertices of $G$ and let $S$ be a minimal $ab$-separator. The induced subgraph $G[V - S]$ has at least two connected components. Let $G[A]$ be the connected component containing $a$ and $G[B]$ the connected component containing $b$. Now $G[A \cup S]$ is a chordal graph with fewer than $n$ vertices and thus is either complete (every vertex of $A$ is simplicial) or has at least two nonadjacent simplicial vertices one of which must belong to $A$ since $S$ is a clique. Since $A$ has no neighbors outside of $A \cup S$, all simplicial vertices of $G[A \cup S]$ that belong to $A$ are also simplicial vertices of $G$. Thus $G$ has at least one simplicial vertex that belongs to $A$. With the same argument, $G$ has another simplicial vertex that belongs to $B$, and the proof is complete. □

This lemma gives a method for recognizing chordal graphs [8] with the following algorithm. Repeat the following step until no simplicial vertices are left: Find a simplicial vertex and remove it from the graph. If, at the end of this process, the remaining graph is empty, then we can conclude that the graph we started with is chordal. Otherwise it is not chordal. If the graph is chordal, the order in which the vertices are removed is called a *perfect elimination order.* Chordal graphs are exactly the class of graphs having perfect elimination orders, as will be proven in the following theorem.

**Theorem 3.5** [8] *A graph is chordal if and only if it has a perfect elimination ordering.*

**Proof.** $\Rightarrow$: We have already explained, in the discussion above, how a perfect elimination ordering of a chordal graph can be found. More formally, assume that $G$ is a chordal graph with $n$ vertices, and assume that the theorem is true for chordal graphs with fewer vertices. Let $v$ be a simplicial vertex of $G$. Now, the graph $G[V - \{v\}]$ has a perfect elimination ordering $\beta$. Let $\alpha$ be an ordering that orders $v$ first and the rest of the vertices in $G$ in the same order as $\beta$. Clearly, $\alpha$ is a perfect elimination ordering of $G$.

$\Leftarrow$: Let $G$ be a graph and let $v_1, v_2, ..., v_n$ be a perfect elimination ordering of the vertices of $G$. Assume that $G$ has a chordless cycle $c$ of length greater than 3. Let $v_i$ be the vertex on $c$ whose label $i$ is smaller than any other vertex on $c$. Since the given ordering is a perfect elimination ordering, the higher numbered neighbors of $v_i$ induce a clique in $G$. But then $c$ must have at least one chord, namely the edge joining the two neighbors of $v_i$ in $c$. $\Box$

The above idea can also be used to make any graph chordal by adding edges. Choose any vertex $x$ to start with, and add the necessary edges so that the neighbors of $x$ become a clique. Remove $x$ from the modified graph, and continue this process until all vertices are processed. In the end, all the added edges of each step are added to the original graph $G$ and this results in a *filled* graph. This algorithm is popularly referred to as *the elimination game.* Let $\alpha = v_1, v_2, ..., v_n$ be the order in which the elimination game processes the vertices of a given graph $G$ and produces the filled graph $G^+_\alpha$. To see that $G^+_\alpha$ is indeed chordal, observe that $\alpha$ is a perfect elimination order of $G^+_\alpha$. How many edges $G^+_\alpha$ has depends on the ordering $\alpha$. It is an interesting and widely studied problem to find an ordering $\alpha$ that results in the fewest number of edges in the filled graph $G^+_\alpha$. Unfortunately, this is an NP-hard problem [18].

## 3.1 Chordal graphs as intersection graphs

**Definition 3.6** *Let $F$ be a family of nonempty sets. The* intersection graph *of $F$ is obtained by representing each set in $F$ by a vertex, and connecting two vertices by an edge if and only if their corresponding sets intersect.*

**Definition 3.7** *A family $\{T_i \mid i \in I\}$ of subsets of a set $T$ is said to satisfy the* Helly property *if the following condition is satisfied:*

$$J \subseteq I \text{ and } T_i \cap T_j \neq \emptyset \ \forall i, j \in J \Rightarrow \bigcap_{j \in J} T_j \neq \emptyset$$

**Lemma 3.8** *A family of subtrees of a tree satisfies the Helly property.*

**Proof.** Let $T$ be a tree and let $\{T_i \mid i \in I\}$ be a set of subtrees of $T$. Suppose $T_i \cap T_j \neq \emptyset$ for all $i, j \in J$. Consider three vertices $a, b$, and $c$ on $T$. Let $S$ be a set of indices $s$ such that $T_s$ contains at least two of these three vertices, and let $P_1, P_2$, and $P_3$ be paths in $T$ connecting $a$ with $b$, $b$ with $c$, and $a$ with $c$, respectively. Since $T$ is a tree, it follows that $P_1 \cap P_2 \cap P_3 \neq \emptyset$. But each $T_s, s \in S$, contains one of these paths $P_i$. Therefore,

$$\bigcap_{s \in S} T_s \supseteq P_1 \cap P_2 \cap P_3 \neq \emptyset. \tag{1}$$

Assume now by induction that

$$T_i \cap T_j \neq \emptyset \ \forall i, j \in J \Rightarrow \bigcap_{j \in J} T_j \neq \emptyset$$

for all index sets $J$ of size $\leq k$. This is certainly true for $k = 2$. Consider then a family of subtrees $\{T_1, ..., T_{k+1}\}$. By the induction hypothesis there exist vertices $a, b, c$ on $T$ such that

$$a \in \bigcap_{j=1}^{k} T_j \ , \ b \in \bigcap_{j=2}^{k+1} T_j \ , \ c \in T_1 \cap T_{k+1}.$$

Moreover, every $T_j$ contains at least two of the vertices $a, b, c$. Hence, by (1), $\bigcap_{j=1}^{k+1} T_j \neq \emptyset$. $\square$

A clique $C$ in a graph $G$ is *maximal* if $C$ is not a subset of any other clique in $G$. We denote the neighbors of a vertex $v$ by $N(v)$.

**Theorem 3.9** [10] *Let $G = (V, E)$ be an undirected graph, and let $\mathcal{K}$ be the set of maximal cliques of $G$, with $\mathcal{K}_v$ the set of all maximal cliques that contain a vertex $v$ of $G$. The following statements are equivalent:*

*$(i)$ $G$ is chordal.*

*$(ii)$ $G$ is the intersection graph of a family of subtrees of a tree.*

*$(iii)$ There exists a tree $T = (\mathcal{K}, \mathcal{E})$ whose vertex set is the set of maximal cliques of $G$ such that each of the induced subgraphs $T[\mathcal{K}_v]$ is connected.*

**Proof.** $(iii) \Rightarrow (ii)$ : Assume that there exists a tree $T = (\mathcal{K}, \mathcal{E})$ that satisfies $(iii)$ and let $u, v \in V$. Now $(u, v) \in E \Leftrightarrow u, v \in A$ for some clique $A \in \mathcal{K} \Leftrightarrow \mathcal{K}_u \cap \mathcal{K}_v \neq \emptyset \Leftrightarrow T[\mathcal{K}_u] \cap T[\mathcal{K}_v] \neq \emptyset$. Thus $G$ is the intersection graph of the family of subtrees $\{T[\mathcal{K}_v] \mid v \in V\}$ of $T$.

$(ii) \Rightarrow (i)$ : Let $\{T_v \mid v \in V\}$ be a family of subtrees of a tree $T$ such that $(u, v) \in E \Leftrightarrow T_u \cap T_v \neq \emptyset$. Suppose that $G$ contains a chordless cycle $v_0, v_1, ..., v_{k-1}, v_0$ with $k > 3$ corresponding to the sequence of subtrees $T_0, T_1, ..., T_{k-1}, T_0$ of the tree $T$; that is $T_i \cap T_j \neq \emptyset \Leftrightarrow i$ and $j$ differ by at most 1 modulo $k$. All arithmetic will be done in modulo $k$.

Choose a vertex $a_i$ from $T_i \cap T_{i+1}$ for $i = 0, ..., k-1$. Let $b_i$ be the last common vertex on the unique simple paths from $a_i$ to $a_{i-1}$ and $a_i$ to $a_{i+1}$. These paths lie in $T_i$ and $T_{i+1}$ respectively, so that $b_i$ also lies in $T_i \cap T_{i+1}$. Let $P_{i+1}$ be the simple path connecting $b_i$ and $b_{i+1}$. Clearly $P_i \subseteq T_i$, so $P_i \cap P_j = \emptyset$ for $i$ and $j$ differing by more than 1 mod $k$. Moreover, $P_i \cap P_{i+1} = \{b_i\}$ for $i = 0, ..., k-1$. Thus, $\bigcup_i P_i$ is a simple cycle in $T$ contradicting the definition of a tree.

$(i) \Rightarrow (iii)$ : We use induction on the size of $G$. Assume that the implication is true for all graphs having fewer vertices than $G$. If $G$ is complete then $T$ is a single vertex and the result is trivial. If $G$ is disconnected then by induction there exists a corresponding tree satisfying $(iii)$ for each connected component of $G$. These trees can combined to a connected tree satisfying $(iii)$ by adding edges between the trees.

Let us assume that $G$ is connected and not complete. Choose a simplicial vertex $a$ of $G$, and let $A = \{a\} \cup N(a)$. Clearly $A$ is a maximal clique of $G$. Let $U = \{u \in A \mid N(u) \subset A\}$, and $Y = A - U$. The sets $U, Y$, and $V - A$ are nonempty since $G$ is connected and not complete. Consider the induced subgraph $G' = G[V - U]$, which is chordal and has fewer vertices than $G$. By induction, let $T'$ be a tree whose vertex set $K'$ is the set of maximal cliques of $G'$ such that for each vertex $v \in V - U$, the set $K'_v = \{X \in K' \mid v \in X\}$ induced a connected subtree of $T'$.

Now, either $K = K' + \{A\} - \{Y\}$ or $K = K' + \{A\}$ depending on whether or not $Y$ is a maximal clique of $G'$. Let $B$ be any maximal clique of $G'$ containing $Y$. 1. If $B = Y$, then we obtain $T$ from $T'$ by renaming $B \rightarrow A$. 2. If $B \neq Y$, then we obtain $T$ from $T'$ by connecting a new vertex $A$ to $B$. In either case (1 or 2), $K_u = \{A\}$ for all $u \in U$ and $K_v = K'_v$ for all $v \in V - A$, each of which induces a subtree of $T$. We need only worry about the sets $\{K_y \mid y \in Y\}$. In case 1, $K_y = K'_y + \{A\} - \{B\}$, which induces the same subtree as $K'_y$ since only names were changed. In case 2, $K_y = K'_y + \{A\}$, which clearly induces a subtree.

Thus we have constructed the required tree $T$ and the proof of the theorem is complete. □

**Example 3.10** *Let $G$ be the graph shown in Figure 2 a). Let $T_1$ be the tree given in b) and $T_2$ the tree shown in c) of the same figure. We will show that both trees have families of subtrees which $G$ is an intersection graph of.*

*Let $F_1$ be the family of subtrees of $T_1$ induced by the following subsets: $\{1\}, \{1,2,3\}, \{3\}, \{3,2,4\}, \{4\}, \{1,2,4\}$. It is easy to see that $G$ is the intersection graph of $F_1$, where vertices $a, b, c, ..., f$ of $G$ correspond to the subsets in the given order.*

*Let $F_2$ be the family of subtrees of $T_2$ induced by the following subsets: $\{t,u\}$, $\{u,v,w\}$, $\{w,x\}$, $\{w,v,y\}$,$\{y,z\}$, $\{y,v,u\}$. Again, $G$ is the intersection graph of $F_2$, where the vertices of $G$ correspond to the subsets in the presented order.*

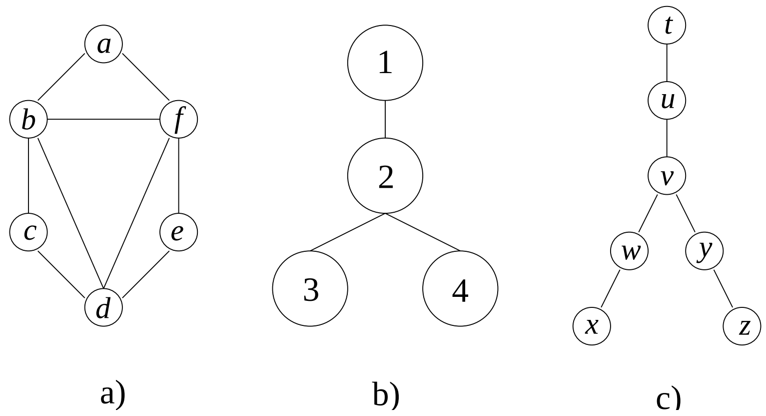


Figure 2: The graphs of Example 3.10.

## 3.2 Clique trees

A tree with the property described in Theorem 3.9 $(iii)$ is in fact called a *clique tree* of $G$. Recall the definition of a tree decomposition. A clique tree is definitely a tree decomposition of a chordal graph. However, the opposite is not necessarily true. The proof of Theorem 3.9 described how to construct a clique tree. In this section, we will concentrate on constructing the clique tree in practice. We will first see how to find all the maximal cliques of a chordal graph efficiently.

The problem of finding all the maximal cliques of a general graph is NP-hard. However, for chordal graphs, the number of maximal cliques is at most $|V|$, and these can be found in linear time by a modification of Maximum Cardinality Search (MCS) [16]. The original MCS algorithm proceeds as follows: Start with an arbitrary vertex $v$ and give $v$ the number $|V|$. Next, select a vertex $w$ with highest number of already numbered neighbors and give $w$ the number $|V|-1$. Continue this process until all the vertices are numbered. The resulting numbering is actually a perfect elimination ordering of $G$ when $G$ is a chordal graph [16]. The MCS algorithm can be implemented to run in $O(n+m)$ time for a graph with $n$ vertices and $m$ edges.

The modified MCS algorithm finds all the maximal cliques of a chordal graph [4]. It proceeds as follows: Start with an arbitrary vertex $v$, give $v$ the number $|V|$, and start a new maximal clique that contains $v$. Let $k=0$. At each next step, select a vertex $w$ with highest number of already numbered neighbors and let this number be $k'$; give $w$ the number $|V|-1$. If $k'>k$ then this means that we are still within the same maximal clique as in the previous step; thus place $w$ in the current clique. Otherwise, start a new clique which $w$ and all its already numbered neighbors belong to. In either case, set $k'=k$ at the end of each step. With a few additions to this algorithm, it can actually construct the clique tree, too.

A general graph can have many minimal separators. However, for connected chordal graphs the number of minimal separators is at most $|V|-1$. The clique tree is a useful structure to express the information on maximal cliques and minimal separators of a chordal graph.

**Definition 3.11** *The* clique graph *of a chordal graph $G$ is a graph $\mathcal{G}$ whose vertices are the cliques of $G$ and two vertices are connected by an edge if their corresponding cliques intersect in $G$.*

Let us define an edge $(C_i, C_j)$ in a clique graph to be equivalent to the set of vertices in $C_i \cap C_j$ for the cliques $C_i$ and $C_j$ in $G$. It is important to note that both the vertices and the edges represent cliques that belong to $G$. The vertices represent the maximal cliques, where the edges represent the intersection between maximal cliques. For the next theorem, let the *weight* of an edge of $\mathcal{G}$ be the number of vertices in the intersection it represents.

**Theorem 3.12** [3] *A* clique tree *of a chordal graph $G$ is a maximum weight spanning tree of the clique graph of $G$.*

**Example 3.13** *Let $G$ be the graph given in Figure 3 a). The clique graph $\mathcal{G}$ of $G$ is given in Figure 3 b), and a clique tree, which for this example is unique, is given in Figure 3 c). The numbers on the edges of the clique graph are the weights of the edges.*

Note that the clique graph of $G$ is unique, whereas $G$ may have several different clique trees.

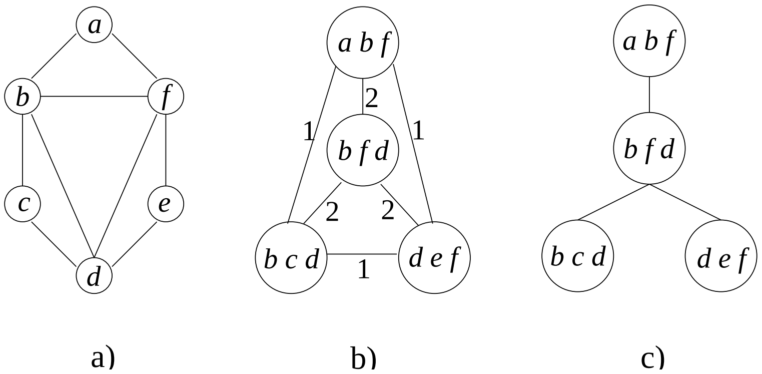


Figure 3: The graphs of Example 3.13.

**Theorem 3.14** [13] *Given a chordal graph $G$ and a clique tree $T$ of $G$, a set of vertices $S$ is a minimal separator of $G$ if and only if $S = C_i \cap C_j$ for an edge $(C_i, C_j)$ in $T$.*

Thus a clique tree is a convenient tool to represent all the maximal cliques and the minimal separators of a chordal graph. It follows that the number of minimal separators of a chordal graph $G = (V, E)$ is at most $|V| - 1$.

The following connections can be proven using the previous lemmas and theorems. A *chordal completion* of a non-chordal graph $G$ is a chordal graph obtained by adding edges to $G$ (e.g., $G^+_\alpha$ is a chordal completion of $G$).

**Exercise 3.15** *Prove the following lemmas:*

1. *Given any graph $G$ and a tree decomposition $(\{X_i \mid i \in I\},\ T = (I, M))$ of $G$, let $H$ be the graph obtained by adding edges to $G$ so that each $X_i$ becomes a clique. Then $H$ is chordal.*

2. *Let $G$ be any graph, and let $H$ be a chordal completion of $G$ with the smallest possible treewidth. Then the treewidth of $G$ is equal to the treewidth of $H$.*

From the lemmas of the exercise above, we can readily conclude that any clique tree of a chordal graph $G$ is a tree decomposition of $G$ of minimum width. Thus for a chordal graph $G$, the treewidth is one less than the size of the largest clique in $G$, and hence can be found in linear time. Note also that, as a consequence, finding a chordal completion of minimum treewidth is equivalent to finding a chordal completion of minimum largest clique size (i.e., the largest clique is as small as possible). Finding such a chordal completion is clearly an NP-hard problem.

**Corollary 3.16** (of Theorem 3.2) *$k$-trees are chordal.*

# 4 Interval graphs

In the previous sections we mentioned that chordal graphs are intersection graphs of subtrees of a tree. Interval graphs are an important subclass of chordal graphs, and they are the intersection graphs of subpaths of a path. This will be clear from the following definition:

**Definition 4.1** *A graph $G = (V, E)$ is an* interval graph *if there is a mapping $I$ of the vertices of $G$ into sets of consecutive integers such that for each pair of vertices $v, w \in V$ the following is true: $(v, w) \in E \Leftrightarrow I(v) \cap I(w) \neq \emptyset$.*

**Theorem 4.2** [11] *$G$ is an interval graph if and only if $G$ has a clique tree that is a simple path.*

**Example 4.3** *Let $G$ be the graph shown in Figure 4 a). A clique tree $T$ of $G$ is given in b). To see that $G$ is an interval graph, we can use the following mapping $I$: $I(a) = \{1\}$, $I(b) = \{1,2,3\}$, $I(c) = \{3,4\}$, $I(d) = \{4\}$, $I(e) = \{2,3,4\}$, $I(f) = \{1,2\}$.*

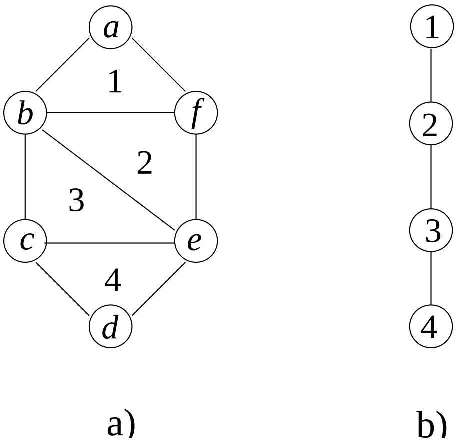


Figure 4: The graphs of Example 4.3.

Observe that the given clique tree is also a tree decomposition and a path decomposition of $G$. Since interval graphs have clique trees that are paths, these will also correspond to path decompositions. Because of the connection between tree decompositions and clique trees for chordal graphs, we can see that pathwidth equals to treewidth for interval graphs.

For a general graph $G$, the pathwidth is one less than the minimum size of the largest clique in an interval completion of $G$. Finding an interval completion where the size of the largest clique is as small as possible is an NP-hard problem. Even for chordal graphs it is an open question whether or not such an interval completion can be found in polynomial time.

# 5 Perfect graphs

Before we give the definition of perfect graphs, we repeat some properties for general graphs.

$\omega(G)$ is the size of the largest clique of $G$, and it is called the *clique number* of $G$.

A *clique cover* of size $c$ is a partition of the vertices of $G$ into $c$ cliques. (Note that an edge is a clique consisting of two vertices.) $k(G)$ is the size of a smallest possible clique cover of $G$.

An *independent set* is a set of vertices no two of which are adjacent. $\alpha(G)$ is the number of vertices in an independent set of maximum cardinality.

A *c-coloring* is a partition of the vertices into $c$ independent sets. The vertices belonging to the same independent set are "colored" with the same color, and adjacent vertices receive different colors. We say that $G$ is $c$-colorable. $\chi(G)$ is the smallest possible number $c$ for which there exists a $c$-coloring of $G$, and it is called the *chromatic number* of $G$.

It is easy to see that $\omega(G) \leq \chi(G)$ and $\alpha(G) \leq k(G)$, since every vertex of a maximum clique (maximum independent set) must be contained in a different partition segment in any minimum coloring (minimum clique cover). Observe also the following equalities: $\omega(G) = \alpha(\bar{G})$ and $\chi(G) = k(\bar{G})$.

**Definition 5.1** *A graph $G = (V, E)$ is* perfect *if it satisfies the following properties:*

1. $\omega(G[X]) = \chi(G[X])$ *for all* $X \subseteq V$
2. $\alpha(G[X]) = k(G[X])$ *for all* $X \subseteq V$.

Obviously, any graph $G$ satisfies 1. if and only if its complement graph $\bar{G}$ satisfies 2. In fact, it can be shown that these two requirements are equivalent.

**Theorem 5.2** [14] (The perfect graph theorem) *For a graph $G = (V, E)$, the following are equivalent:*

- $\omega(G[X]) = \chi(G[X])$ *for all* $X \subseteq V$
- $\alpha(G[X]) = k(G[X])$ *for all* $X \subseteq V$.
- $\omega(G[X])\alpha(G[X]) \geq |X|$.

**Theorem 5.3** [12] *Chordal graphs are perfect.*

As we have also previously seen, finding the maximum clique size and the clique cover number of a chordal graph can be done in polynomial time [9]. As a consequence, finding $\omega(G), \chi(G), \alpha(G)$, and $k(G)$, which are NP-hard for general graphs, can be done in polynomial time for chordal graphs (and thus interval graphs and $k$-trees).

Perfect graphs were defined by Berge in 1960. He also made an interesting attempt to characterize these graphs with the following well-known open conjecture.

**Conjecture 5.4** [2] (The strong perfect graph conjecture) *A graph is perfect if and only if it contains no induced copies of $C_{2k+1}$ or $\bar{C}_{2k+1}$ for $k \geq 2$.*

Here, $C_{2k+1}$ is a cycle on $2k+1$ vertices and $\bar{C}_{2k+1}$ is the complement of such a cycle. It is still an open problem to prove that the conjecture is true or false.

## References


[1] S. Arnborg, D. G. Corneil, and A. Proskurowski, *Complexity of finding embeddings in a k-tree*, SIAM J. Alg. Disc. Meth., 8 (1987), pp. 277–284.

[2] C. Berge, *Färbung von graphen, deren sämtliche bzw. deren ungeraden kreise starr sind (zusammenfassung)*, Wiss. Z. Martin-Luther-Univ. Halle-Wittenberg Math.-Natur., Reihe 10 (1961), p. 114.

[3] P. A. Bernstein and N. Goodman, *Power of natural semijoins*, SIAM J. Comput., 10 (1981), pp. 751–771.

[4] J. R. S. Blair and B. W. Peyton, *An introduction to chordal graphs and clique trees*, in Sparse Matrix Computations: Graph Theory Issues and Algorithms, J. A. George, J. R. Gilbert, and J. W. H. Liu, eds., Springer Verlag, 1993, pp. 1–30. IMA Volumes in Mathematics and its Applications, Vol. 56.

[5] H. L. Bodlaender, *A tourist guide through treewidth*, Acta Cybernetica, 11 (1993), pp. 1–21.

[6] H. L. Bodlaender and R. H. Möhring, *The pathwidth and treewidth of cographs*, SIAM J. Disc. Meth., 6 (1993), pp. 238–255.

[7] G. Dirac, *On rigid circuit graphs*, Anh. Math. Sem. Univ. Hamburg, 25 (1961), pp. 71–76.

[8] D. Fulkerson and O. Gross, *Incidence matrices and interval graphs*, Pacific Journal of Math., 15 (1965), pp. 835–855.

[9] F. Gavril, *Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph*, SIAM J. Comput., 1 (1972), pp. 180–187.

[10] ——, *The intersection graphs of subtrees in trees are exactly the chordal graphs*, J. Combin. Theory Ser. B, 16 (1974), pp. 47–56.

[11] P. C. Gilmore and A. J. Hoffman, *A characterization of comparability graphs and of interval graphs*, Canadian Journal of Mathematics, 16 (1964), pp. 539–548.

[12] M. C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, 1980.

[13] C. W. Ho and R. C. T. Lee, *Counting clique trees and computing perfect elimination schemes in p arallel*, Information Processing Letters, 31 (1989), pp. 61–68.

[14] L. Lovasz, *A characterization of perfect graphs*, J. Comb. Theory, 13 (1972), pp. 95–98.

[15] D. Rose, *On simple characterization of $k$-trees*, Disc. Math., 7 (1974), pp. 317–322.

[16] R. E. Tarjan and M. Yannakakis, *Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs*, SIAM J. Comput., 13 (1984), pp. 566–579.

[17] J. van Leeuwen, *Graph algorithms*, in Handbook of Theoretical Computer Science, A: Algorithms and Complexity Theory, North Holland, 1990, pp. 527–631.

[18] M. Yannakakis, *Computing the minimum fill-in is NP-complete*, SIAM J. Alg. Disc. Meth., 2 (1981), pp. 77–79.