Graph Algorithms - Lecture 13

January 17, 2025

Tree decomposition - Definition

Definition

A tree decomposition of a graph $G = (V, E)$ is a pair $\mathcal{T} = (T, \{V_t : t \in T\})$, where T is a tree and $\{V_t : t \in V(T)\}$ is a family of subsets of vertices of G, $V_t \subseteq V$ for every node $t \in T$ such that:

- *(Node coverage)* $V = \bigcup_{t \in V(T)} V_t$;
- *(Edge coverage) For every* $e \in E$, *both endpoints of* e *are contained in* V_t *for some* $t \in V(T)$.
- *(Coherence) Let* t_1, t_2, t_3 *be three nodes in* T *such that* t_2 *lies on the path between* t_1 *and* t_3 *in* T. *Then, if* $v \in V$ *belongs to both* V_{t_1} *and* V_{t_3}, v *must also belong to* V_{t_2}.

Remarks

The coherence can be rephrased like follows

- (Coherence') Let $t_1, t_2, t_3 \in V(T)$ s. t. t_2 belongs to the path from t_1 to t_3 in T. Then $V_{t_1} \cap V_{t_3} \subseteq V_{t_2}$.
- (Coherence") For every $x \in V$, the subgraph of T induced by $\{t \in V(T) : x \in V_t\}$ is (a subtree of T) connected.

The sets V_t are called the bags of the corresponding tree decomposition.

Definition

Let $\mathcal{T} = (T, \{V_t : t \in T\})$ be a tree decomposition of G, the width of $\mathcal{T}$ is

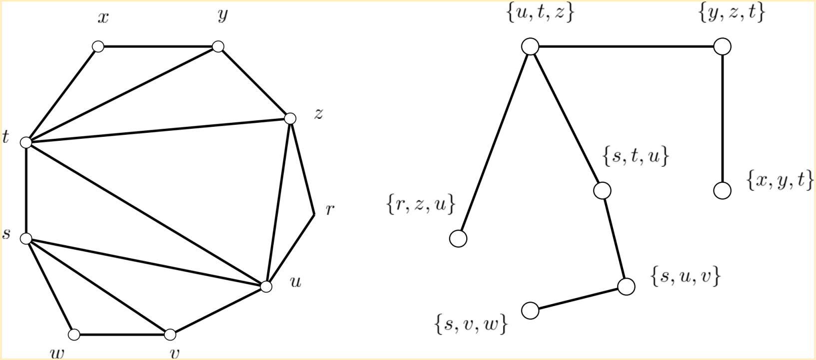
$$width(\mathcal{T}) = \max_{t \in V(T)} (|V_t| - 1).$$

Tree-width - Definition

Definition

The tree-width *of a graph G, is the minimum width of a tree decomposition of G:*

$$tw(G) = \min\{width(\mathcal{T}) \ : \ \mathcal{T} \text{ tree decomposition of } G\}.$$



Remark

$tw(G) = 0$ if and only if $E(G) = \varnothing$.

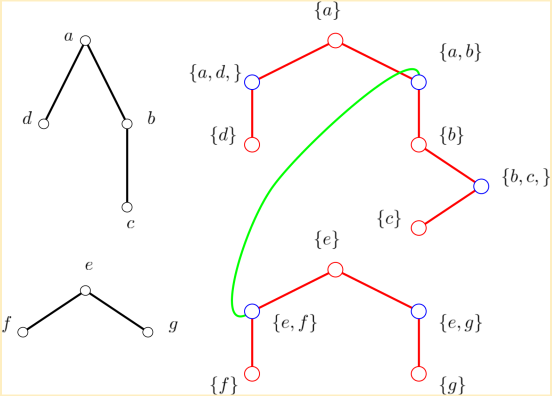
Proposition

If G is a forest with $E(G) \neq \varnothing$, then $tw(G) = 1$.

Proof. $tw(G) \geqslant 1$ by the above remark. If G is a tree, then

- let T be obtained from G by renaming t_v each vertex $v \in V(G)$,
- insert on each edge $t_u t_v$ ($uv \in E(G)$) a new vertex t_{uv},
- set $V_{t_u} = \{u\}$ for all t_u associated to $u \in V(G)$, and $V_{t_{uv}} = \{u, v\}$ for all $t_{uv} \in V(T)$ associated to $uv \in E(G)$.
- $(T, \{V_t : t \in V(T)\})$ is a tree decomposition of G with width 1.

Proof (cont'd). A tree decomposition of a forest with k components can be obtained by adding $k-1$ arbitrary edges to tree decompositions for the components (without creating cycles). $\square$



Definition

A tree decomposition, $\mathcal{T} = (T, \{V_t : t \in V(T)\})$, is small *if there are no distinct vertices $t_1, t_2 \in V(T)$ such that $V_{t_1} \subseteq V_{t_2}$.*

Proposition

Given a tree decomposition of G, a small tree decomposition of G with the same width can be constructed in polynomial time.

Proof. Let $\mathcal{T} = (T, \{V_t : t \in V(T)\})$ be a tree decomposition of G with $V_{t_1} \subseteq V_{t_2}$ for $t_1, t_2 \in V(T)$, $t_1 \neq t_2$. We can suppose that $t_1 t_2 \in E(T)$ (otherwise, we find adjacent nodes with this property, by considering a path from t_1 to t_2).
Contracting $t_1 t_2$ into a new node t_{12} with $V_{t_{12}} = V_{t_2}$, gives a smaller tree decomposition of G (it contains less pairs of vertices (t_1', t_2') with $V_{t_1'} \subseteq V_{t_2'}$).

Proof (cont'd). Repeat this reduction until a small tree decomposition is obtained

□

Proposition

If $\mathcal{T} = (T, \{V_t : t \in V(T)\})$ is a small tree decomposition of G, then $|T| \leqslant |G|$.

Proof. By induction on $n = |G|$. If $n = 1$, then $|T| = 1$.
In the inductive step, for $n \geqslant 2$, consider a leaf t_1 of T with neighbor t_2. $(T - t_1, \{V_t : t \in V(T - t_1)\})$ is a small tree decomposition of $G' = G \setminus (V_{t_1} \setminus V_{t_2})$. By induction hypothesis $|T - t_1| \leqslant |G'|$, therefore

$$|T| = |T - t_1| + 1 \leqslant |G'| + 1 \leqslant |G|.$$

□

Minors

Remarks

- If the graph H is obtained from G by contracting an edge uv into z, then $tw(H) \leqslant tw(G)$: in a tree decomposition of G, insert z in every bag containing u or v, and then remove u and v from every bag to obtain a tree decomposition of H.
- If H is a subgraph of G, then $tw(H) \leqslant tw(G)$.

Definition

H is a minor *of a graph G if it can be obtained from G by iteratively deleting and contracting edges.*

Corollary

If H is a minor of a graph G, then $tw(H) \leqslant tw(G)$.

Proof. Using the above remarks. $\square$

Let $TW_k = \{G : tw(G) \leqslant k\}$.

TW (Tree-Width - decision version)

Instance: G a graph and $k \in \mathbb{N}$.

Question: $G \in TW_k$?

Theorem

Tree-Width (decision version) problem is NP*-complete.*

Proof. Omitted.

Lemma

For every positive integer k, TW_k is minor closed.

Theorem

(Bodlaender) For every fixed k, the problem of determining whether or not $G \in TW_k$ can be solved in $\mathcal{O}(f(k) \cdot n)$ time.

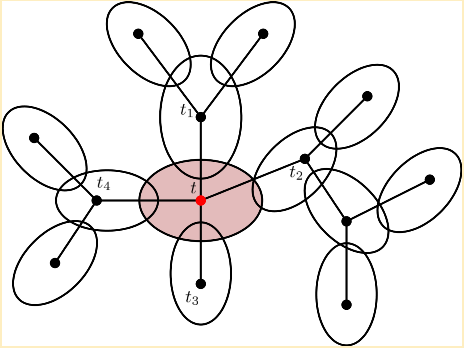
Proofs. Omitted. ($f(k)$ is exponential in k.)

Notation: Let $\mathcal{T} = (T, \{V_t : t \in V(T)\})$ be a tree decomposition of G. Then, if T' is a subgraph of T, $G_{T'}$ denotes the subgraph of G induced by the set of vertices $\bigcup\limits_{t \in V(T')} V_t$.

Tree decomposition properties

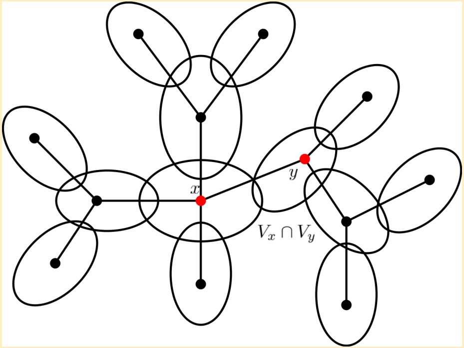
Theorem

(Node separation property.) *Suppose $T - t$ has connected components $T_1, T_2, \ldots, T_p$. Then the subgraphs $G_{T_1} - V_t, G_{T_2} - V_t, \ldots, G_{T_p} - V_t$ have no vertices in common and there are no edges between them.*



Theorem

(Edge separation property.) Let X and Y be the two connected components of T after the deletion of edge $xy \in E(T)$. Then, deleting $V_x \cap V_y$ disconnects G into two subgraphs $H_X = G_X - V_x \cap V_y$ and $H_Y = G_Y - V_x \cap V_y$. That is H_X and H_Y share no vertices and there is no edge in G with one endpoint in H_X and the other in H_Y.



Other properties:

- Let G be a connected graph with $tw(G) = k$, then $|G| = k + 1$ or G has a k-vertex cutset.
- If $tw(G) = 1$, then G is a forest.
- $tw(P_r \times P_s) = \min\{r, s\}$.
- $tw(K_n) = n - 1$.

Definition

A rooted tree decomposition *of G is a tree decomposition $\mathcal{T} = (T, \{V_t : t \in V(T)\})$ of G, where some vertex r of T is declared to be the root.*

Notations: let t be a vertex in a rooted tree decomposition $\mathcal{T} = (T, \{V_t : t \in V(T)\})$.

- T_t is the subtree of T rooted at t.
- $G[t]$ is the subgraph of G induced by the vertices in $\bigcup_{x \in V(T_t)} V_x$ (i. e., $G[t] = G_{T_t}$).

- Recall: a p-vertex coloring of a graph $G = (V, E)$ is a function $c : V \to \{1, 2, \dots, p\}$ such that for all $uv \in E$, $c(u) \neq c(v)$.
- Let H' and H'' be two subgraphs of G, with p-colorings c' and c'', respectively. c'' is c'-compatible if for all $v \in V(H') \cap V(H'')$, $c'(v) = c''(v)$.
- Let $\mathcal{T} = (T, \{V_t : t \in V(T)\})$ a rooted tree decompositions of G. For every $t \in T$ and every p-coloring c of G_t, define

$$Prev_t(c) = \begin{cases} 1, & \text{if } G[t] \text{ has an } c\text{-compatible } p\text{-coloring } \overline{c} \\ 0, & \text{otherwise.} \end{cases}$$

Proposition

$Prev_u(c) = 1$ if and only if for all children v of u, there exists a c-compatible coloring $\overline{c}$ of G_v with $Prev_v(\overline{c}) = 1$

Proof. "$\Longrightarrow$" If γ is a c-compatible coloring of $G[u]$, since G_v is a subgraph of $G[u]$, then the restriction of γ to G_v gives the required coloring $\overline{c}$.

"$\Longleftarrow$" Suppose that u has exactly two children v and w, and we have two c-compatible colorings $\overline{c}'$ and $\overline{c}''$, respectively (the proof is similar for more children).

Since $(T, \{V_t : t \in V(T)\})$ is a tree decomposition, $V(G[v]) \cap V(G[w]) \subseteq V_u$, so $\overline{c}'$ is $\overline{c}''$-compatible.

Combining $\overline{c}'$ and $\overline{c}''$ gives $\overline{c} : V(G[u]) \to \{1, 2, \ldots, p\}$. Since $(T, \{V_t : t \in V(T)\})$ is a tree decomposition, there are no edges $xy \in E(G)$ with $x \in V(G[v]) - V_u$ and $y \in V(G[w]) - V_u$, so $\overline{c}$ is a p-coloring of $G[u]$.

$\square$

Theorem

If G, a graph of order n, has a small tree decomposition $(T, \{V_t : t \in V(T)\})$ of width w, then we can decide if G is p-colorable in $\mathcal{O}(p^{w+1} \cdot n^{\mathcal{O}(1)})$ time complexity.

Proof. Transform $(T, \{V_t : t \in V(T)\})$ in a rooted tree decomposition (r is the root). For every $v \in V(T)$ and every p-coloring c of G_v, we compute $Prev_v(c)$: start at the leaves of T, and use the above proposition for the other nodes, in the right order.

$G = G[r]$ is p-colorable if and only if $Prev_r(c) = 1$ for some c. Testing whether c is a G_v coloring and computing $Prev_v(c)$ can be done in polynomial time $\mathcal{O}(n^{\mathcal{O}(1)})$, so the total time complexity is mainly determined by the number of candidates for c, which is $p^{|V_v|}$.

Complexity: $|V(T)| \cdot p^{w+1} \cdot n^{\mathcal{O}(1)} = \mathcal{O}(p^{w+1} \cdot n^{\mathcal{O}(1)})$.

□

Other applications - Similar approaches (more advanced dynamic programming)

Theorem

If G, a graph of order n, has a small tree decomposition $(T, \{V_t : t \in V(T)\})$ of width w, the size of a minimum vertex cover of G ca be computed in $\mathcal{O}(2^{w+1} \cdot n^{\mathcal{O}(1)})$ time complexity.

Theorem

If G, a vertex-weighted graph of order n, has a small tree decomposition $(T, \{V_t : t \in V(T)\})$ of width w, a maximum stable set of G ca be computed in $\mathcal{O}(4^{w+1} \cdot w \cdot n)$ time complexity.

The end!