Parameterized algorithms: Tree width and dynamic programming

Maria Serna

Spring 2023

Graph parameters

- For a given graph G we can consider graph measures as candidates for parameters.
- Diameter, degree, vertex cover, . . . , or a combination of them.
- Many hard graph problems can be solved in polynomial time in trees.
- We are going to explore a parameter that measures the closeness of a graph to a tree: treewidth.
- A similar parameter measures closeness of a graph to a path: pathwidth.

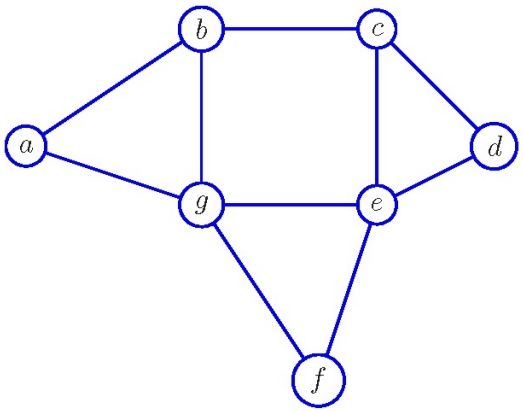
Recall some graph notation

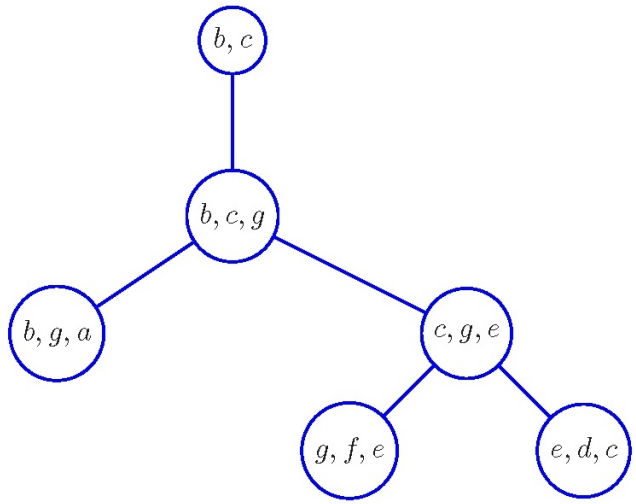
- For a graph G and $v \in V(G)$, $G - v$ denotes the graph obtained by deleting v (and all incident edges).
- For a set S, $S + v$ denotes $S \cup \{v\}$, and $S - v$ denotes $S \setminus \{v\}$.
- For a vertex $v \in V(G)$, $N(v)$ denotes the set of neighbors of v. $N[v] = N(v) + v$. $d(v) = |N(v)|$.
- For a graph $G = (V, E)$, $\delta(G) = \min_{v \in V} d(v)$, and $\Delta(G) = \max_{v \in V} d(v)$.

- A tree is a connected graph without cycles.
- A forest is a graph without cycles.
- A unicyclic graph has only one cycle.
- A graph is outerplanar if it can be drawn as cycle with non-crossing chords.

Tree decomposition

- A tree decomposition of a graph G is a tuple (T, X) where T is a tree and $X = \{X_v \mid v \in V(T)\}$ is a set of subsets of $V(G)$ such that:
  - For every $xy \in E(G)$, there is a $v \in V(T)$ with $\{x, y\} \subseteq X_v$.
  - For every $x \in V(G)$, the subgraph of T induced by $X^{-1}(x) = \{v \in V(T) \mid x \in X_v\}$ is non-empty and connected.

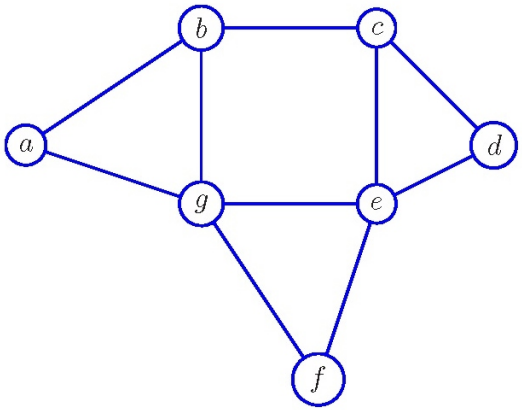


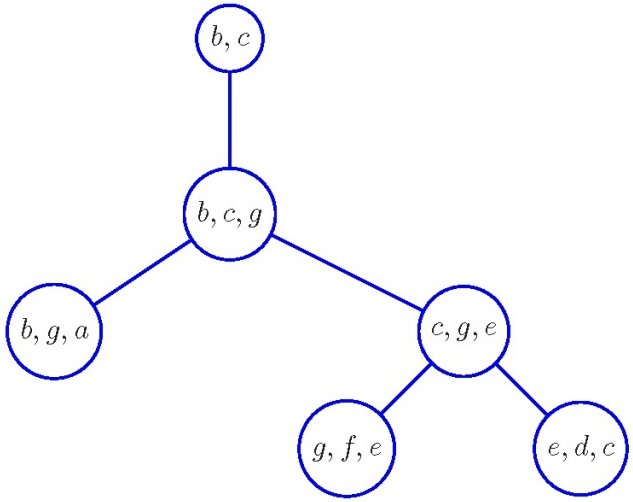


Tree decomposition

- A tree decomposition of a graph G is a tuple (T, X) where T is a tree and $X = \{X_v \mid v \in V(T)\}$ is a set of subsets of $V(G)$ such that:
  - For every $xy \in E(G)$, there is a $v \in V(T)$ with $\{x, y\} \subseteq X_v$.
  - For every $x \in V(G)$, the subgraph of T induced by $X^{-1}(x) = \{v \in V(T) \mid x \in X_v\}$ is non-empty and connected.
- Equivalently the second condition can be expressed as:
  - For every $u, v \in V(T)$ and every vertex w on the path between u and v, $X_u \cap X_v \subseteq X_w$, and every vertex of G appears in at least one X_v .

Tree decomposition



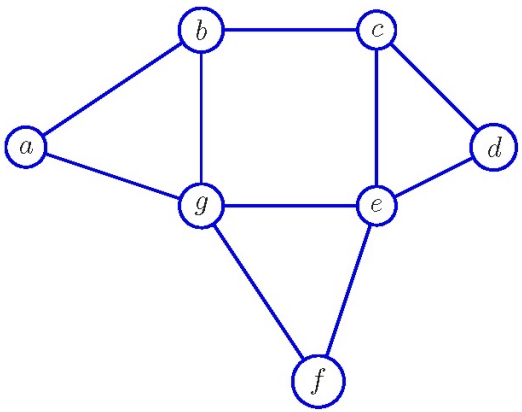


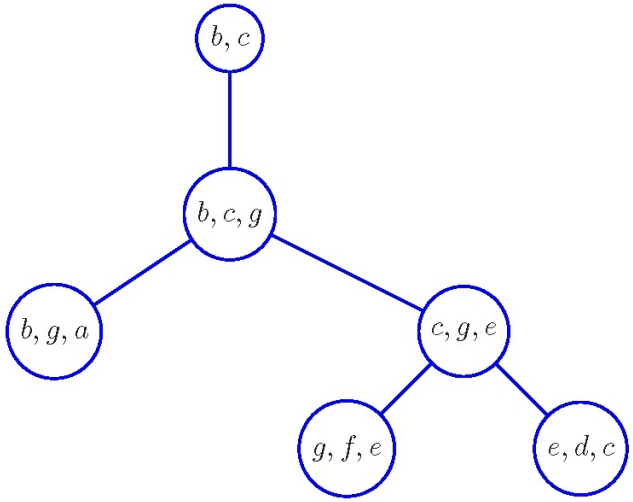
- To distinguish between vertices of G and T, the vertices of T are called nodes.
- The sets X_v are called the bags of the tree decomposition.

Tree width

- The width of a tree decomposition (T, X) for G is $\max_{v \in V(T)} |X_v| - 1$.
- The tree width ($tw(G)$) of a graph G is the minimum width over all tree decompositions of G.

A graph with tree width 2





This graph is an outerplanar graph.

Tree width of some graphs

- $tw(G) = 0$ iff $E(G) = \emptyset$.
- If G is a forest $tw(G) \leq 1$.
  - Consider the tree obtained from G by subdividing every edge $uv \in E(G)$ with a new vertex w_{uv}.
  - Set $X_u = \{u\}$ for all $u \in V(G)$, and $X_{w_{uv}} = \{u, v\}$ for every $uv \in E(G)$.
- If G is outerplanar then $tw(G) \leq 2$.
  - Let G' be a graph obtained after triangulating arbitrarily the face of G with more than 3 sides preserving outerplanarity .
  - T is the dual of G': a node per face and connecting two nodes if their faces share an edge.
  - Associate to every node the three vertices in the corresponding face.
- For K_n, the complete graph on n vertices, $tw(K_n) = n - 1$.

Tree width complexity

- Deciding if a graph has trewidth k is NP-complete.
- Computing a tree decomposition with width at most k (if it exists) takes $O(f(k)n)$ time.
- How big is such a tree decomposition?

Small tree decomposition

- A tree decomposition (T, X) is small if for distinct $u, v \in V(T)$, $X_u \not\subseteq X_v$ and $X_v \not\subseteq X_u$.
- When given a tree decomposition of G, in polynomial time we can construct a small tree decomposition of G with the same width.
  - If a tree decomposition is not small there should be two adjacent nodes $u, v \in V(T)$ with $X_u \subseteq X_v$.
  - Contracting uv into a new node w with $X_w = X_v$ gives a smaller tree decomposition for G.
  - repeating the above procedure we get a small tree decomposition.

Small tree decomposition

- If (T, X) is a small tree decomposition of G. Then $|V(T)| \leq |V(G)|$

  Exercise: proof by induction

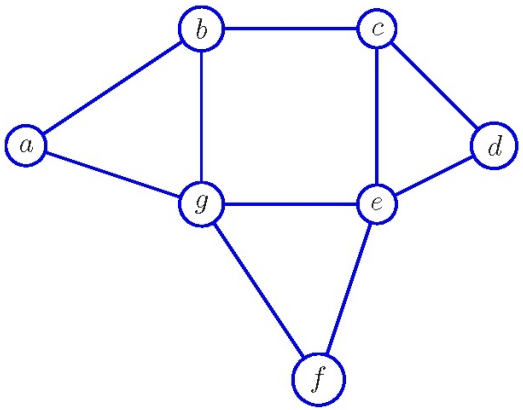
- We can assume that a tree decomposition of minimum width has polynomial size with respect to $|G|$ and that it is small.
- When needed we can make use of a polynomial time approximation algorithm that given a (G, k) decides in polynomial time whether $tw(G) \leq k$ and if so produces a small tree decomposition of width $\leq r \cdot k$, for some constant $r > 1$.

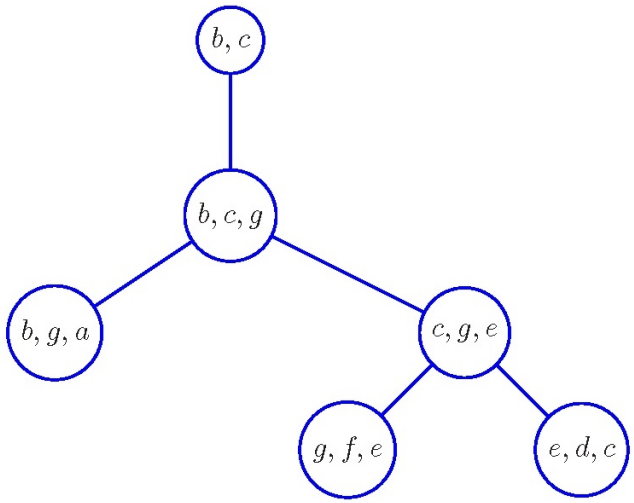
Notation

- Let (T, X) be a tree decomposition of G of width k.

  Such a tree decomposition can be computed by an FPT algorithm
- Make T into a rooted tree by choosing a root $r \in V(T)$, and replacing edges by arcs in such a way that every node points to its parent.
- For $v \in V(T)$, $R_T(v)$ denotes the nodes in the subtree rooted at v (including v).
- For $v \in V(T)$, $V(v) = X(R_T(v))$, and $G(v) = G[V(v)]$.

  Thus, we have an induced subgraph associated to each node.
- Notice that X_v is a separator in G.





Exercise For this rooted tree decomposition. Draw the graphs associated to each node in the tree. What are the differences among parent-child graphs?

A parameterization for Vertex Coloring?

TW-K-VERTEX COLORING
Input: A graph G and an integer k,
Parameter: $tw(G) + k$
Question: Is G k-colorable?

PROBLEM: $tw(G)$ cannot be computed in polynomial time (unless P=NP)!

Equivalent parameterizations for Vertex Coloring

TW-K-VERTEX COLORING
Input: A graph G, a tree decomposition (T, X) of G and an integer k,
Parameter: $width(T, X) + k$
Question: Is G k-colorable?

TW-K-VERTEX COLORING
Input: A graph G and integers w and k,
Parameter: $w + k$
Question: Is $tw(G) \leq w$ and is G k-colorable?

The same kind of tw parameterization applies to other graph problems.

For some graph properties, an upper bound in terms of treewidth can be found. For such problems the parameter might be only w.

tw-k-Vertex Coloring belongs to FPT

- To show that TW-K-VERTEX COLORING $\in$ FPT we use dynamic programing on the tree decomposition.
- Let H_1 and H_2 be two subgraphs of G, with valid k-colorings α_1 and α_2 respectively.
- α_2 is α_1-compatible if for all $v \in V(H_1) \cap V(H_2)$, $\alpha_2(v) = \alpha_1(v)$.
- Let (T, X) be a rooted tree decomposition of G.
  - For every $v \in V(T)$ and every proper k-coloring α of $G[Xv]$, define $P_v(\alpha) = 1$ iff $G(v)$ has an α-compatible k-coloring β.
- Our algorithm computes $P_v(\alpha)$, for each node in T, from leaves to root.

tw-k-Vertex Coloring belongs to FPT

Lemma

$P_u(\alpha) = 1$ iff for all children v of u, there is an α-compatible coloring β of $G[X_v]$ with $P_v(\beta) = 1$.

Proof.

- ($\Rightarrow$) Let γ be an α-compatible coloring of $G(u)$. $G(v)$ is a subgraph of $G(u)$, so restricting to X_v gives the desired coloring β.

tw-k-Vertex Coloring belongs to FPT

Proof.

- ($\Leftarrow$) Consider two children v and w of u, and suppose they have α-compatible colorings β and γ respectively.
  - Since (T, X) is a tree decomposition, $V(v) \cap V(w) \subset X_u$, so β is γ-compatible.
  - Combining β and γ gives $\delta : V(u) \to \{1, \ldots, k\}$.
  - Since (T, X) is a tree decomposition, there are no edges $xy \in E(G)$ with $x \in V(v) - X_u$ and $y \in V(w) - X_u$, so δ is a proper k-coloring of $G(u)$.
  - The same can be done for all children of u simultaneously.

tw-k-Vertex Coloring belongs to FPT

Theorem

Let (T, X) be a rooted small tree decomposition of G of width w. In time $k^{w+1}n^{O(1)}$ we can decide whether G is k-colorable.

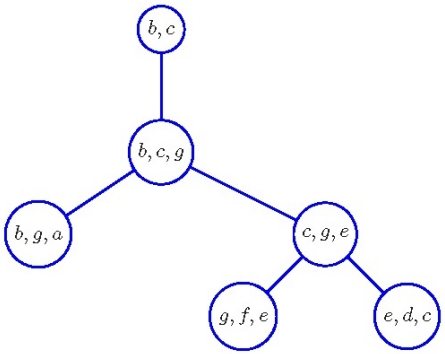
Proof.

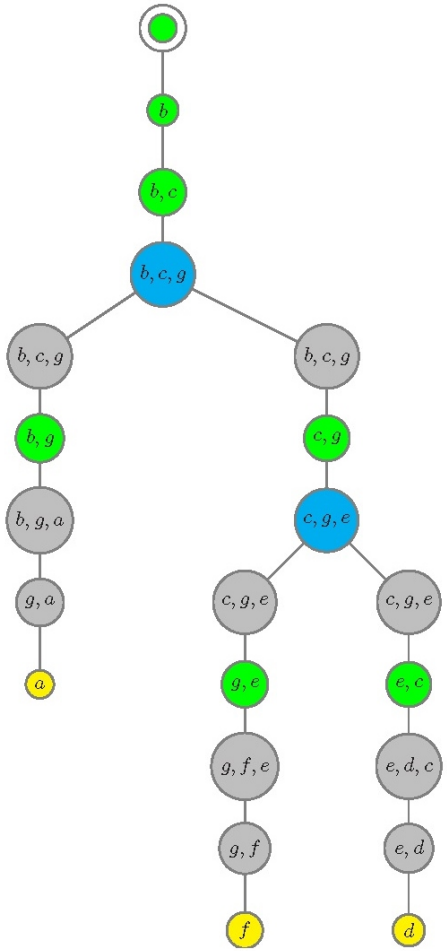
- For $v \in V(T)$ and a k-coloring α of $G[X_v]$, we compute $P_v(\alpha)$ starting at the leaves of T, and using the recurrence.
- $G = G(r)$ is k-colorable iff $Pr(\alpha) = 1$, for some α.
- testing whether α is a $G[Xv]$ coloring can be done in $O(w^2)$.
- computing $Pv(\alpha)$ for a valid α can be done in $n^{O(1)}$
- the total complexity is mainly determined by the number of candidates for α which is $k^{|X_v|}$: $|V(T)|k^{w+1}n^{O(1)}$.

Nice tree decomposition

- A nice tree decomposition is a variant in which the structure of the nodes is simpler.
- A rooted tree decomposition (T, X) is nice if for every $u \in V(T)$
  - $|X_u| = 1$ (start)
  - u has one child v
    with $X_u \subseteq X_v$ and $|X_u| = |Xv| - 1$ (forget)
    with $X_v \subseteq X_u$ and $|Xu| = |Xv| + 1$ (introduce)
  - u has two children v and w with $X_u = X_v = X_w$ (join)

Nice tree decomposition





Nice tree decomposition

Lemma

Computing a rooted nice tree decomposition with width at most k, given a small tree decomposition of width at most k takes $O(kn)$ time.

Nice tree decomposition

- Nodes in the tree
  node u holds a subset of vertices X_u, and has a subgraph G_u associated to it.
- the root r has $X_r = \emptyset$ and $G_r = G$.
- nodes can be of four types:
  Start Introduce Forget Join

A parameterization for Min Vertex Cover

TW-VERTEX COVER

Input: A graph G, a tree decomposition (T, X),

Parameter: $width(T, X)$

Question: Compute a minimum size vertex cover of G?

The algorithm for tw-Vertex Cover

- Obtain a rooted nice tree decomposition (T, X) of G
- For each node $v \in V(T)$ we keep a table $s_v(C)$ for each $C \subseteq X_v$
  holding the minimum size of a vertex cover C' of $G(v)$ with $C' \cap X_v = C$ if such a C' exists, and $s_v(C) = \infty$ otherwise.
- The value of $s_r(\emptyset)$ is the size of a minimum vertex cover of G.

- We deal with each type of node separately

Start node

Claim

Let u be a leaf of T with $X_u = \{x\}$.
$s_u(\{x\}) = 1$ and $s_u(\emptyset) = 0$.

Introduce node

Claim

Let u be an *introduce* node, let v be its unique child and assume that $\{x\} = X_u - X_v$.
Then for all $C \subseteq X_u$

- If C is not a vertex cover of $G[X_u]$ then $s_u(C) = \infty$.
- If C is a vertex cover of $G[X_u]$ and $x \in C$ then $s_u(C) = s_v(C - x) + 1$.
- If C is a vertex cover of $G[X_u]$ and $x \notin C$ then $s_u(C) = s_v(C)$.

  All neighbors of x in $G(u)$ are in X_v and therefore in C since C is a vertex cover of $G[X_u]$.

Forget node

Claim

Let u be a *forget* node, let v be its unique child and assume that $\{v\} = X_v - X_u$.
Then for all $C \subseteq X_u$, $s_u(C) = \min\{s_v(C), s_v(C + x)\}$.

Proof.

- ($\geq$) Let C' be a minVC of $G(u) = G(v)$ with $C' \cap X_u = C$.
  - If $x \notin C'$ then $C' \cap X_u = C$ so $|C'| \geq s_v(C)$.
  - If $x \in C'$ then similarly $|C'| \geq s_v(C + x)$.
- ($\leq$) Let C_1 and C_2 be the VCs that determine $s_v(C)$ and $s_v(C + x)$ respectively. C_1, C_2 are VC of $G(u)$ compatible with C, so $s_v(C) \leq \min\{|C_1|, |C_2|\}$.

Join node

Claim

Let u be a join node of T with children v and w.
Then for all $C \subseteq X_u$: $s_u(C) = s_v(C) + s_w(C) - |C|$.

Proof.

- ($\geq$) If C' is a vertex cover of $G(u)$ with $C' \cap X_u = C$, then $C' \cap V(v)$ is a vertex cover of $G(v)$ and $C' \cap V(w)$ is a vertex cover of $G(w)$, which share $|C|$ vertices.
- ($\leq$) Two C-compatible vertex covers of $G(v)$ and $G(w)$ of size $s_v(C)$ and $s_w(C)$ can be combined to a vertex cover of $G(u)$ of size $s_v(C) + s_w(C) - |C|$.

Min Vertex cover parameterized by treewidth

Theorem

Let (T, X) be a rooted nice tree decomposition of width w of a graph G on n vertices. In time $2^{w+1}n^{O(1)}$ the size of a minimum vertex cover of G can be computed.

Proof.

- We can construct a minimum vertex cover as well, by tracing back through the tree decomposition.
- $+ O(f(k)n)$ to get the tree decomposition if needed.