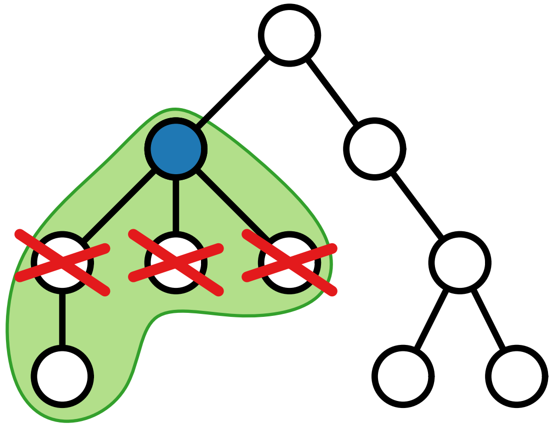
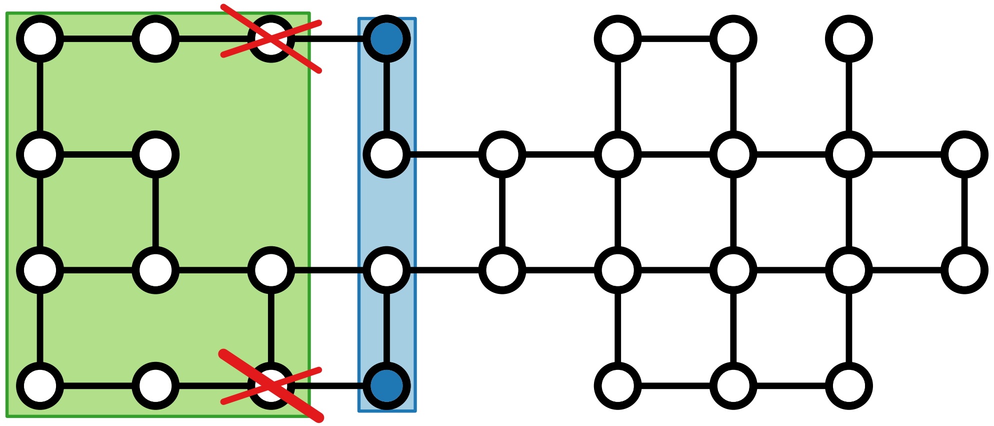
Advanced Algorithms

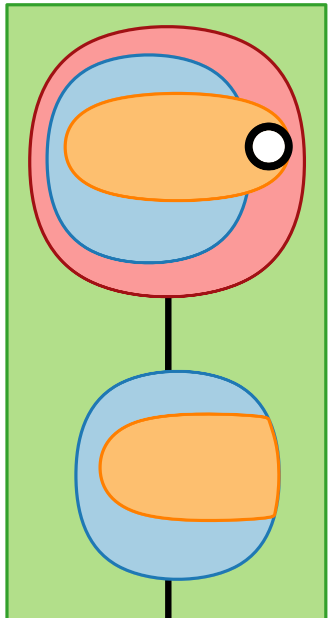
Parameterized Algorithms

Structural Parametrization

Johannes Zink · WS22





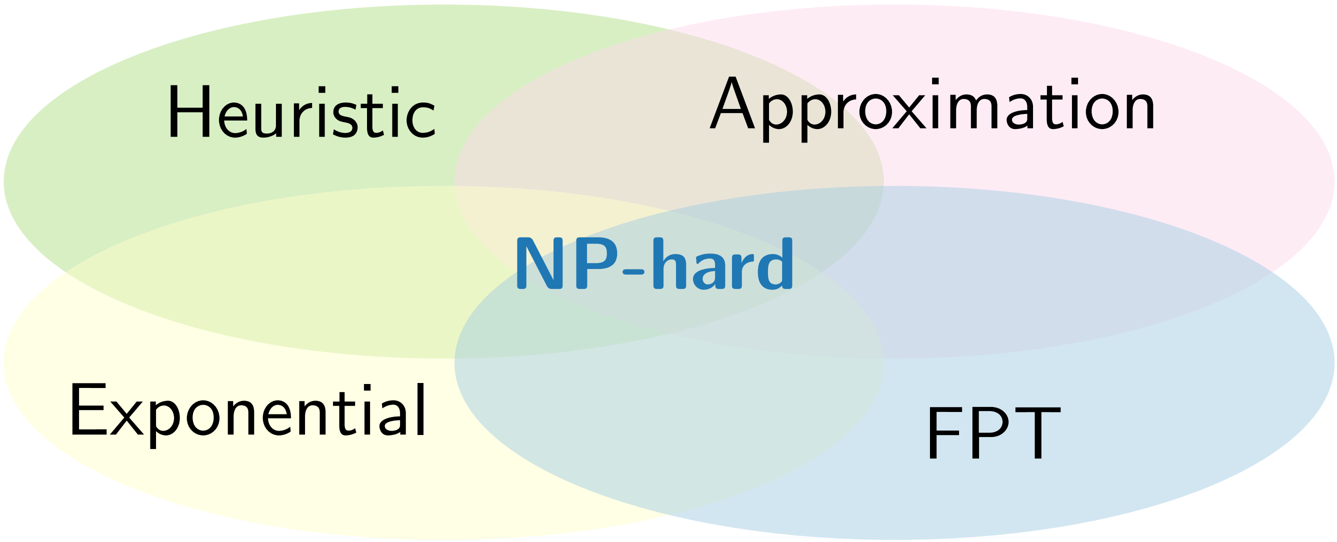


Dealing with NP-Hard Problems

What should we do?

- Sacrifice optimality for speed
  - Heuristics
  - Approximation Algorithms
- Optimal Solutions
  - Exact exponential-time algorithms
  - Fine-grained analysis – parameterized algorithms

this lecture



Parameterized Algorithms

Classical complexity theory:

Running time is expressed as a function in the input size

Parameterized algorithmics:

Running time is expressed as a function in the input size, as well as one or more additional parameter(s)

Example: (recall from AGT)

k-Vertex Cover

Input Graph $G = (V, E)$, $k \in \mathbb{N}$

Question Is there a set $C \subseteq V$ with $|C| \leq k$ s.t. $\forall \{u, v\} \in E \colon \{u, v\} \cap C \neq \emptyset$?

- NP-complete,
- but there is an algorithm with runtime $\mathcal{O}(2^k \cdot k \cdot (|V| + |E|))$.

Idea: If $k \in \mathcal{O}(1)$, then $\mathcal{O}(2^k \cdot k \cdot (|V| + |E|)) \subseteq \mathcal{O}(|V| + |E|)$, in other words, if we assume the **parameter** k to be **fixed**, k-Vertex Cover becomes **tractable**.

Parameterized Complexity Classes

Definition.

Let Π be a decision problem. If there is

- an algorithm $\mathcal{A}$ and
- a computable function f

such that, given an instance I of Π and a parameter $k \in \mathbb{N}$,
the algorithm $\mathcal{A}$ provides the correct answer to I in time $f(k) \cdot |I|^{\mathcal{O}(1)}$,
then $\mathcal{A}$ (and Π) are called **fixed-parameter tractable (FPT)** with respect to k.

If $\mathcal{A}$ provides the correct answer to I in time $|I|^{f(k)}$, then $\mathcal{A}$ (and Π) are called **slice-wise polynomial (XP)** with respect to k. (Note that FPT $\subsetneq$ XP.)

Example. k-Vertex Cover can be solved in time $\mathcal{O}(\underbrace{2^k \cdot k}_{f(k)} \cdot \underbrace{(|V| + |E|)}_{|I|^{\mathcal{O}(1)}})$.

$\Rightarrow$ k-Vertex Cover is FPT (and therefore also XP) with respect to k.

Examples and Counterexamples

k-Vertex Cover

- NP-complete
- but FPT with respect to k

k-Clique

- NP-complete
- but XP with respect to k
- Under common assumptions, k-Clique is not FPT with respect to k (namely, k-Clique is $W[1]$-complete with respect to k; $\rightarrow$ Section 13 in [1])
- There is an $\mathcal{O}(2^{\Delta} \cdot \Delta^2 \cdot (|V| + |E|))$ time algorithm for k-Clique, where Δ is the maximum degree of the input graph $\Rightarrow$ k-Clique is FPT with respect to Δ.

Vertex k-Coloring

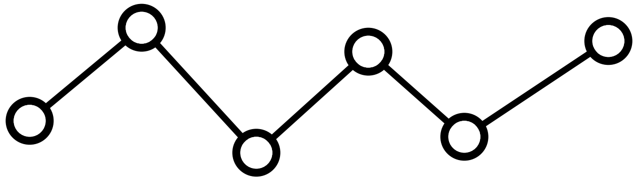
- NP-complete for every $k \geq 3$
- $\Rightarrow$ neither FPT nor XP with respect to k, unless $P = NP$

In all these examples, k is the *natural* parameter that comes with the decision problem.

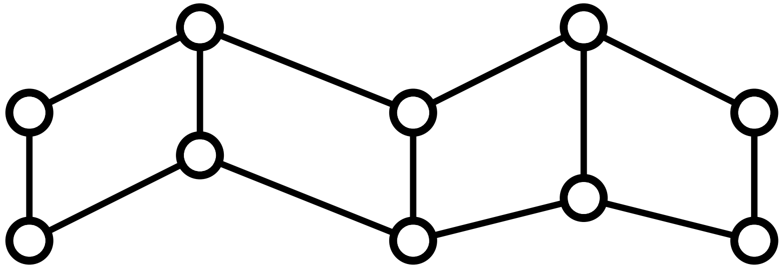
We can also study other types of parameters!

Pathwidth and Treewidth (Intuition)

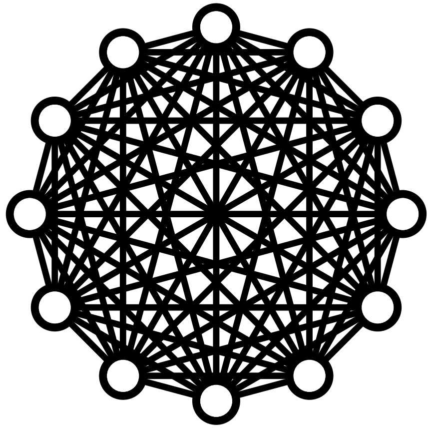
Pathwidth describes how *path-like* a graph is.



1

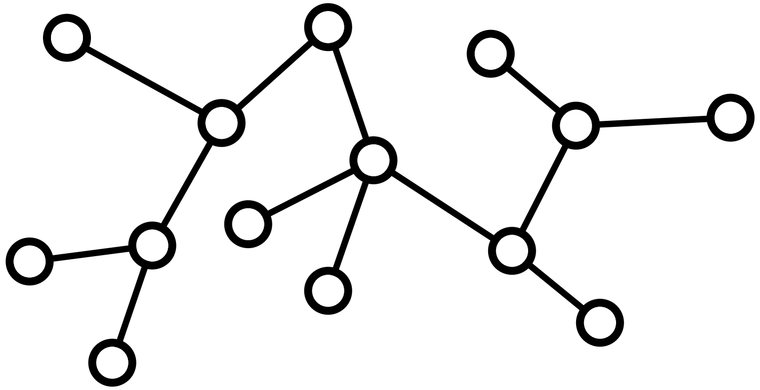


2

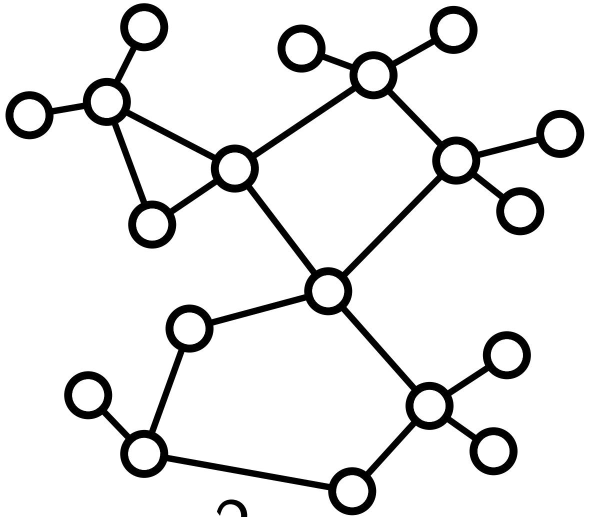


$n-1$

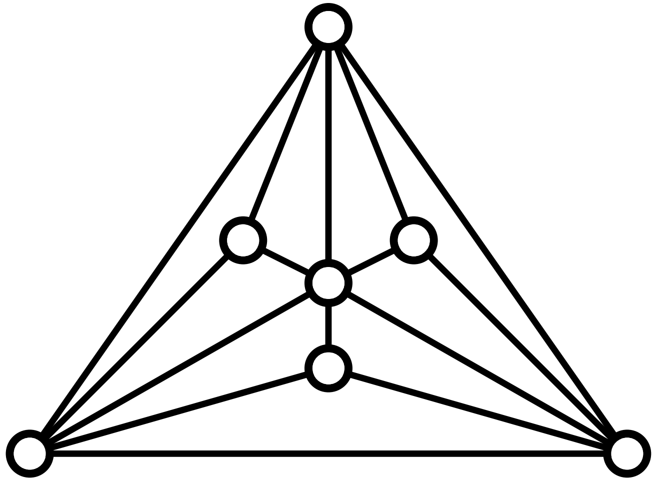
Treewidth describes how *tree-like* a graph is.



1



2



3

Tree-like structure is useful for designing dynamic programming algorithms.

(Weighted) Independent Set

Input. A graph $G = (V, E)$. Weight function $w : V \to \mathbb{N}$.

Output. A set $I \subseteq V$ that is **independent**, i.e., $\forall u, v \in I \colon \{u, v\} \notin E$, and has **maximum weight**, i.e., $w(I) := \sum_{v \in I} w(v)$ is maximized.

- (Already unweighted) Independent Set is NP-complete,
- but can be solved efficiently on tree-like graphs (also when weighted).
- On trees, (Weighted) Independent Set can be solved in linear time.

Independent Set in Trees

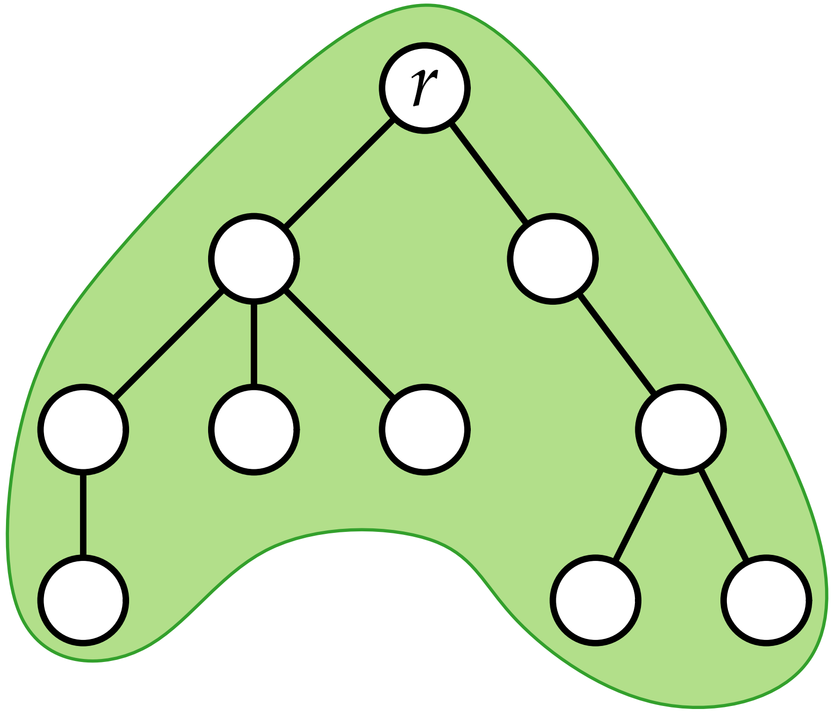
$A(r)$ = solution

Choose an arbitrary root r.

Let $T(v)$:= subtree rooted at v

Let $A(v)$:= maximum weight of an independent set I in $T(v)$

Let $B(v)$:= maximum weight of an independent set I in $T(v)$ where $v \notin I$



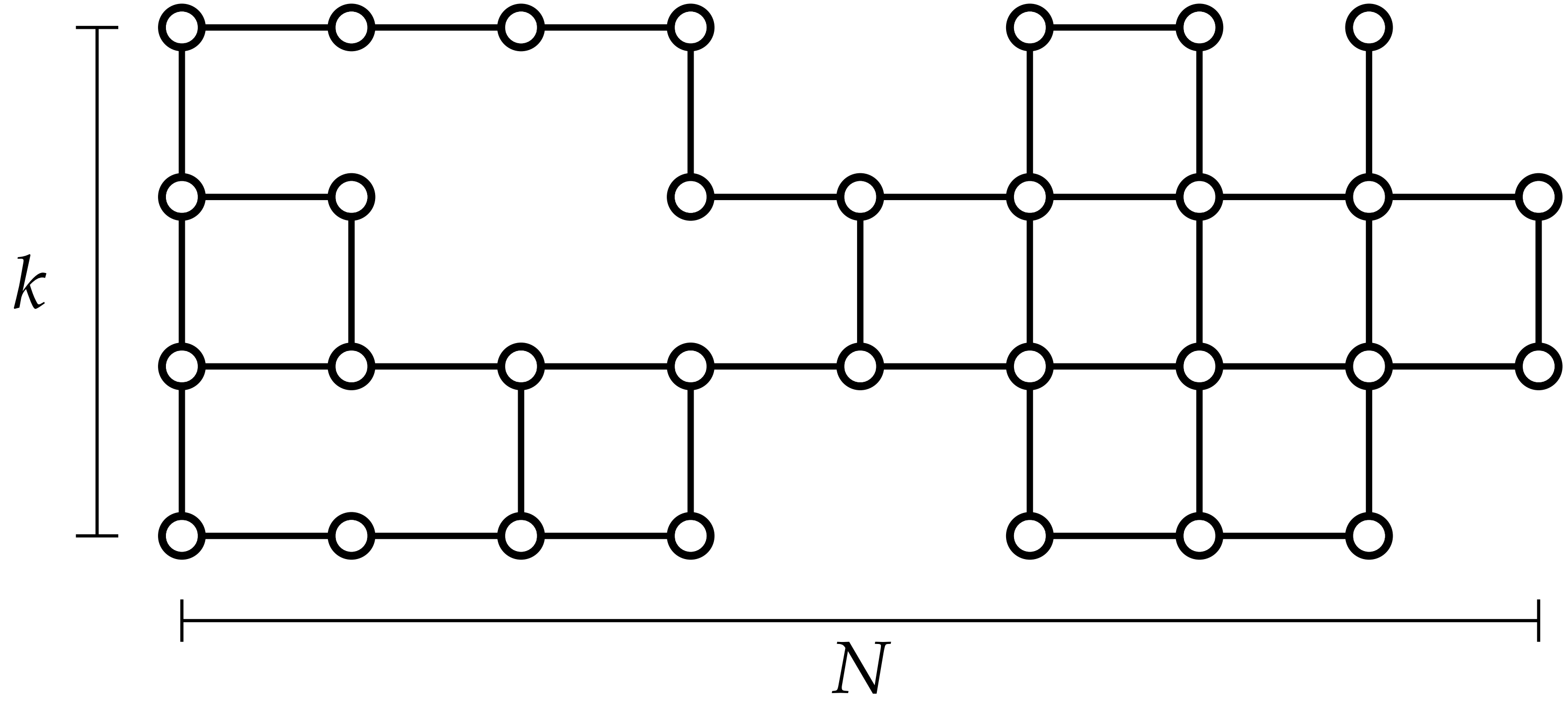
- If v is a leaf: $B(v) = 0$ and $A(v) = w(v)$
- If v has children $x_1, \ldots, x_\ell$:
  $B(v) = \sum_{i=1}^{\ell} A(x_i)$; $A(v) = \max\{B(v), w(v) + \sum_{i=1}^{\ell} B(x_i)\}$

Algorithm: Compute $A(\cdot)$ and $B(\cdot)$ bottom-up, return $A(r)$.

Grid Graphs

In a $k \times N$ **grid graph**

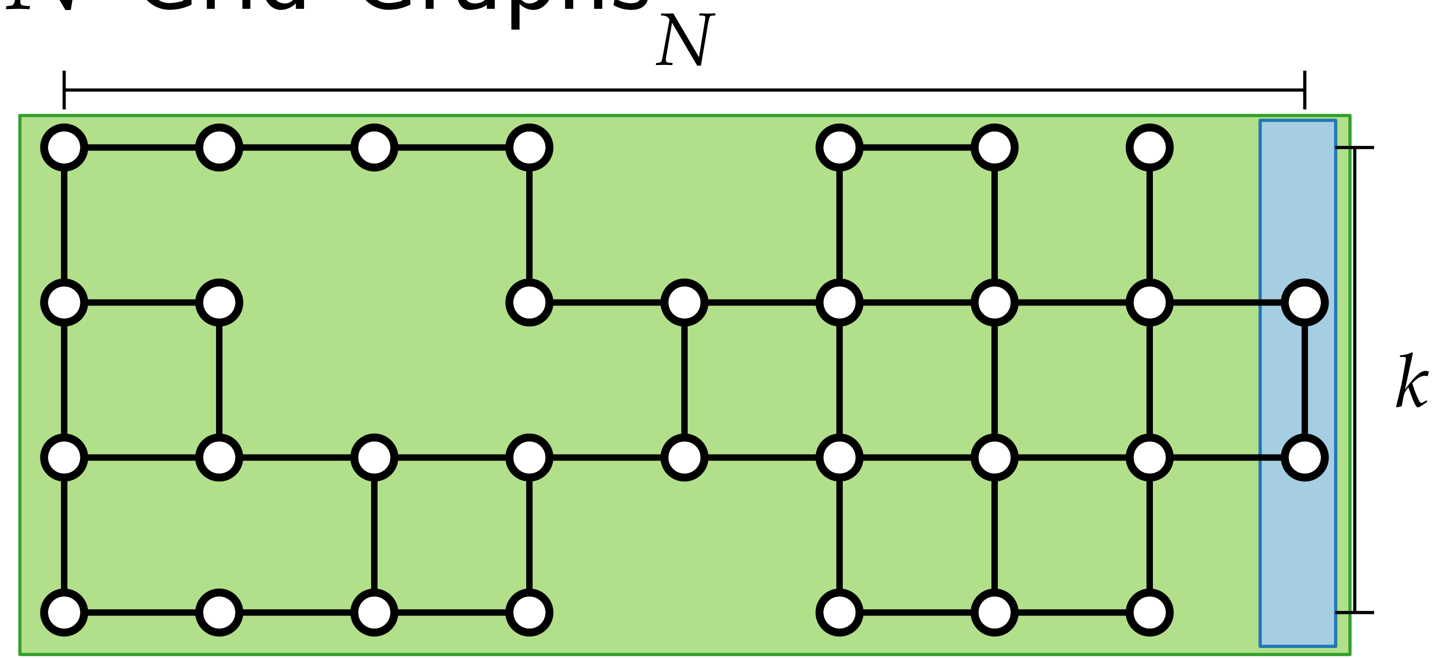
- the vertex set consist of all pairs (i, j) where $1 \leq i \leq k$ and $1 \leq j \leq N$, and
- two vertices (i_1, j_1) and (i_2, j_2) are adjacent if and only if $|i_1 - i_2| + |j_1 - j_2| = 1$.



We will study INDEPENDENT SET in subgraphs of $k \times N$ grid graphs.

Goal: An FTP algorithms with respect to parameter k.

Indenpendent Set in $k \times N$ Grid Graphs



Let X_j be the **j-th column**, that is, $X_j = \mathsf{V}(G) \cap \{(i,j) \mid 1 \leq i \leq k\}$.

Let G_j be the graph induced by the first j columns $X_1 \cup X_2 \cup \ldots X_j$.

Let $1 \leq j \leq N$. For each $Y \subseteq X_j$ let

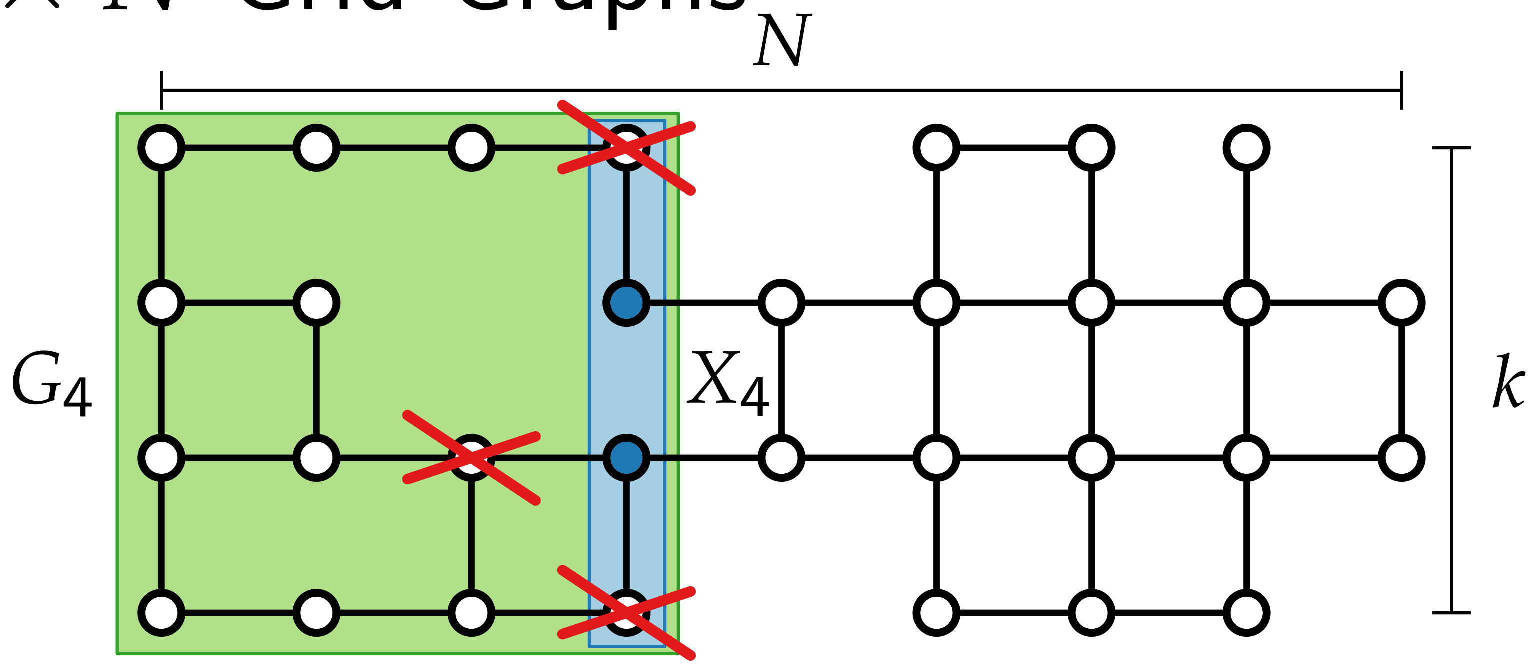
$C[j, Y] :=$ maximum weight of an independent set I in G_j such that $I \cap Y = \emptyset$

$C[N, \emptyset] =$ solution

INDENPENDENT SET in $k \times N$ Grid Graphs

Let X_j be the **j-th column**, that is, $X_j = \mathsf{V}(G) \cap \{(i,j) \mid 1 \leq i \leq k\}$.

Let G_j be the graph induced by the first j columns $X_1 \cup X_2 \cup \ldots X_j$.



Let $1 \leq j \leq N$. For each $Y \subseteq X_j$ let

$C[j, Y] :=$ maximum weight of an independent set I in G_j such that $I \cap Y = \emptyset$

$$C[1, Y] = \max_{I \subseteq X_1 \setminus Y \text{ where } I \text{ independent}} \{w(I)\}$$

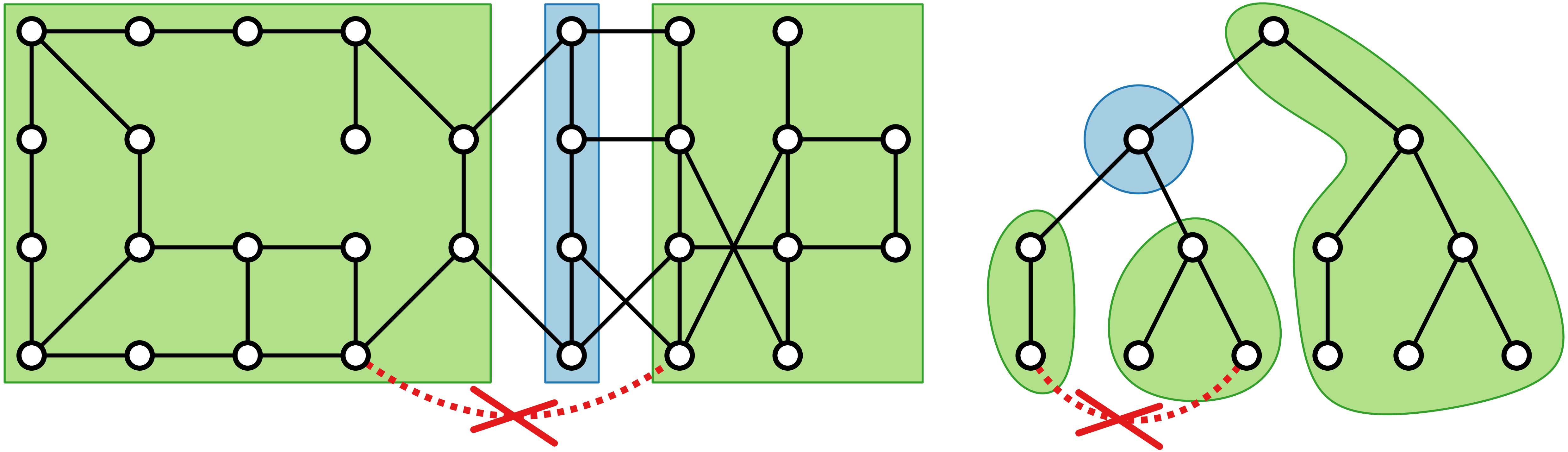
$$C[j, Y] = \max_{I \subseteq X_j \setminus Y \text{ where } I \text{ independent}} \{w(I) + C[j-1, X_{j-1} \cap \mathsf{N}(I)]\}$$

For each j there are $\leq 2^k$ choices of Y, and for each Y there are $2^{|X_j \setminus Y|}$ choices of I.

For each of these $\leq N3^k$ choices of I, we need to test if I is independent.

$\rightarrow$ total running time $\leq 3^k k^{\mathcal{O}(1)} N$.

Can We Apply This Approach to Other Graphs?



Yes!

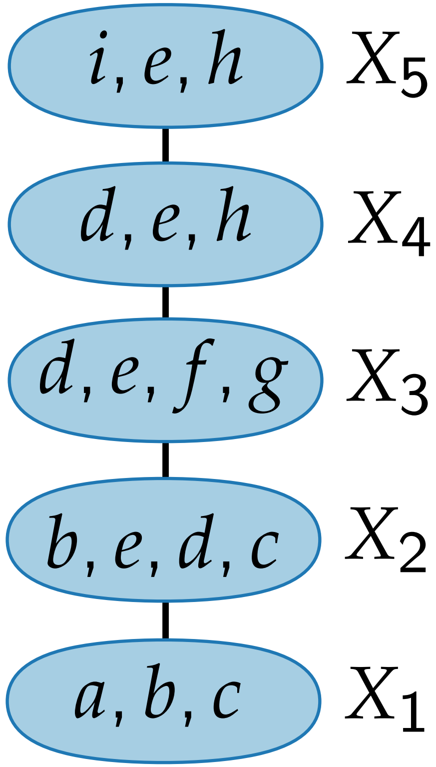
We mainly used the fact that the graph consists of a **sequence of small separators**.

A similiar fact was used in the algorithm for trees.

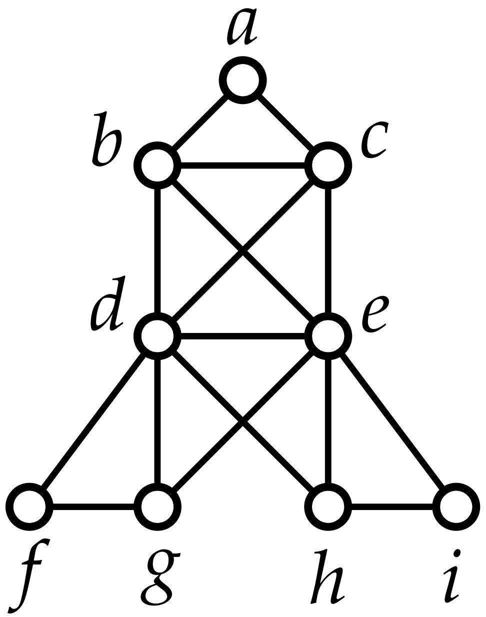
Goal: Define a more general graph class featuring a structure that is suited for this kind of dynamic programming approach.

Path Decompositions

$w(P) = 3$



$\mathrm{pw}(G) \le 3$



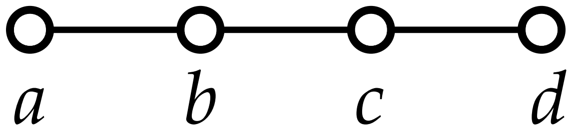
Let $G = (V, E)$ be a graph.

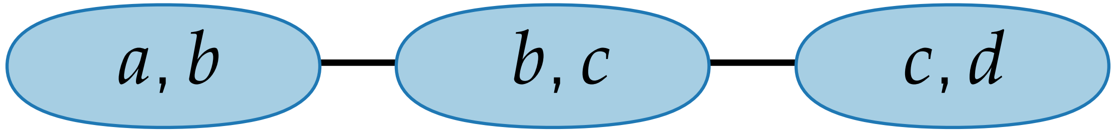
A **path decomposition** of G is a sequence $P = (X_1, X_2, \dots, X_r)$ of **bags**, where $X_i \subseteq V$, such that

- **(P1)** $\bigcup_{i=1}^{r} X_i = V$
- **(P2)** $\forall \{u, v\} \in E \; \exists i \in \{1, 2, \dots, r\} : u, v \in X_i$
- **(P3)** $\forall v \in V$, if $v \in X_i \cap X_j$ with $i \le j$, then $v \in X_i \cap X_{i+1} \cap \dots \cap X_j$

The **width** of P is $w(P) = \max_{1 \le I \le r} |X_i| - 1$.

The **pathwidth** $\mathrm{pw}(G)$ of G is the minimum width of a path decomposition of G.





$\mathrm{pw}(G) = 1$

Okay – But Where Are the Separators?

Lemma. Let $i < r$. Then there is no edge between
$A = (X_1 \cup X_2 \cup \cdots \cup X_i) \setminus (X_i \cap X_{i+1})$ and
$B = (X_{i+1} \cup X_{i+2} \cup \cdots \cup X_r) \setminus (X_i \cap X_{i+1})$.

Proof. Assume there are $a \in A$ and $b \in B$ s.t. $\{a, b\} \in E$.

Let $j \leq i$ s.t. $a \in X_j$ and let $k \geq i+1$ s.t. $b \in X_k$.

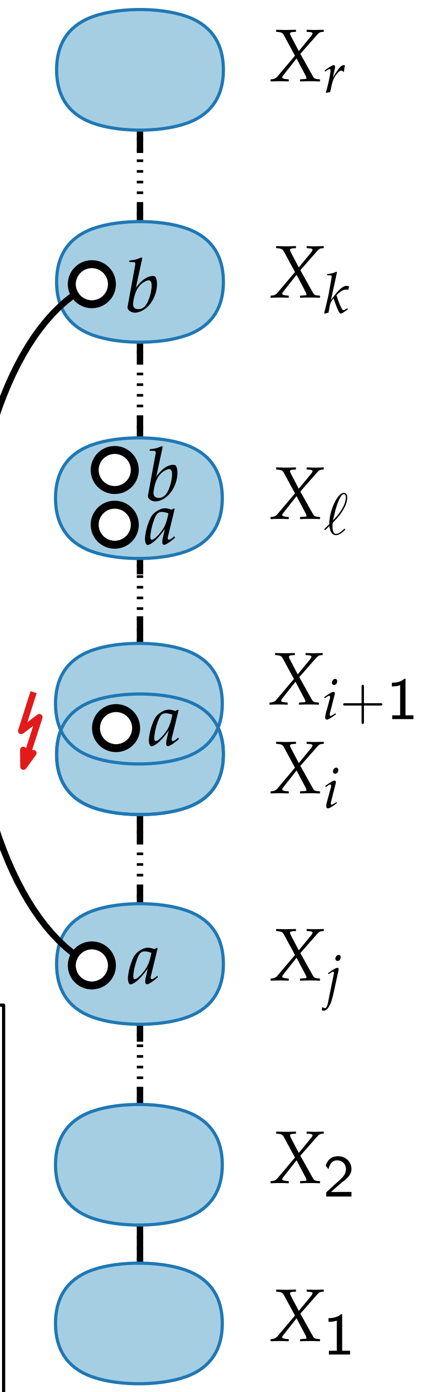
(P2) $\Rightarrow$ there is a bag X_ℓ s.t. $a, b \in X_\ell$, w.l.o.g. let $\ell \geq i+1$.

(P3) $\Rightarrow$ $a \in X_i \cap X_{i+1}$; contradiction to $a \in A$. $\square$

(P1) $\bigcup_{i=1}^{r} X_i = V$

(P2) $\forall \{u, v\} \in E \; \exists i \in \{1, 2, \ldots, r\} : u, v \in X_i$

(P3) $\forall v \in V$, if $v \in X_i \cap X_j$ with $i \leq j$, then $v \in X_i \cap X_{i+1} \cap \cdots \cap X_j$



Computing Path Decompositions

k-PATHWIDTH

Input. Graph $G = (V, E)$, $k \in \mathbb{N}$

Question. Is the pathwidth of G at most k?

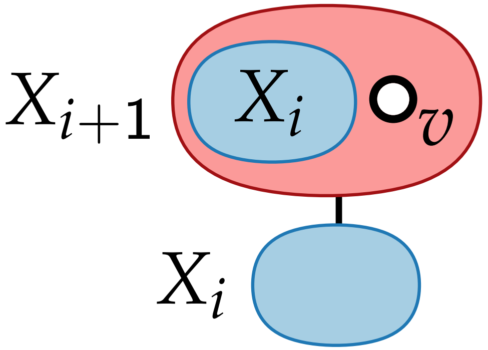
- NP-complete
- FPT in k
  - The algorithm constructs a path decomposition of width $\leq k$.
  - Its runtime depends linearly on $|V| + |E|$.

$\Rightarrow$ When designing FPT algorithms with respect to the pathwidth, we may assume to be given a path decomposition!

Nice Path Decompositions

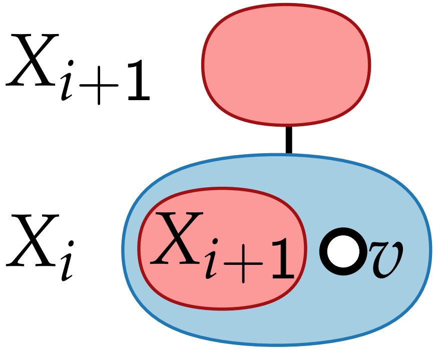
A path decomposition is **nice** if $|X_1| = 1$ and each other bag has one of two **types**:

X_{i+1} is of type **Introduce** if

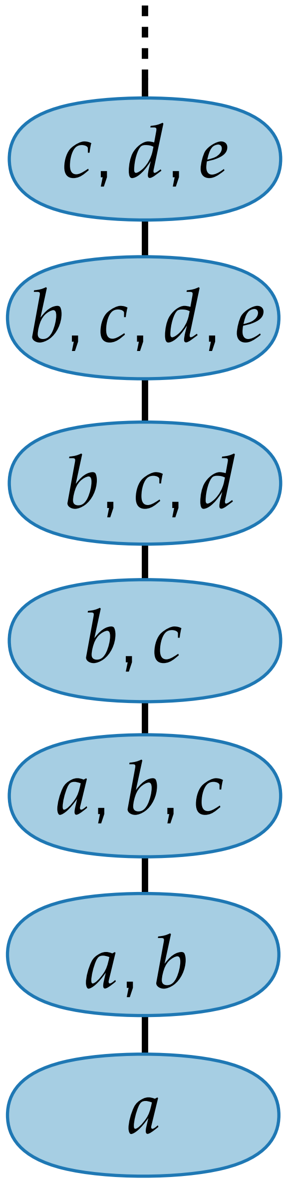


$X_{i+1} = X_i \cup \{v\}$ where $v \notin X_i$

X_{i+1} is of type **Forget** if



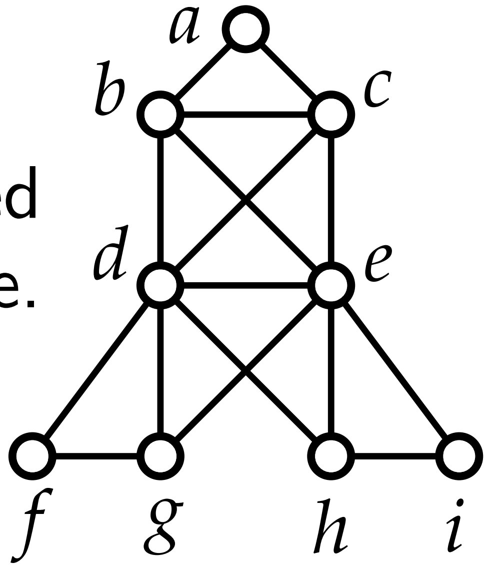
$X_i = X_{i+1} \cup \{v\}$ where $v \notin X_{i+1}$



Observation. The number of bags is $r \leq 2|V| - 1$.

Lemma. A path decomposition of width k can be transformed into a nice path decomposition of width k in polynomial time.

$\Rightarrow$ When designing FPT algorithms w.r.t. the pathwidth, we may assume to be given a *nice* path decomposition.



INDEPENDENT SET in Graphs of Bounded Pathwidth

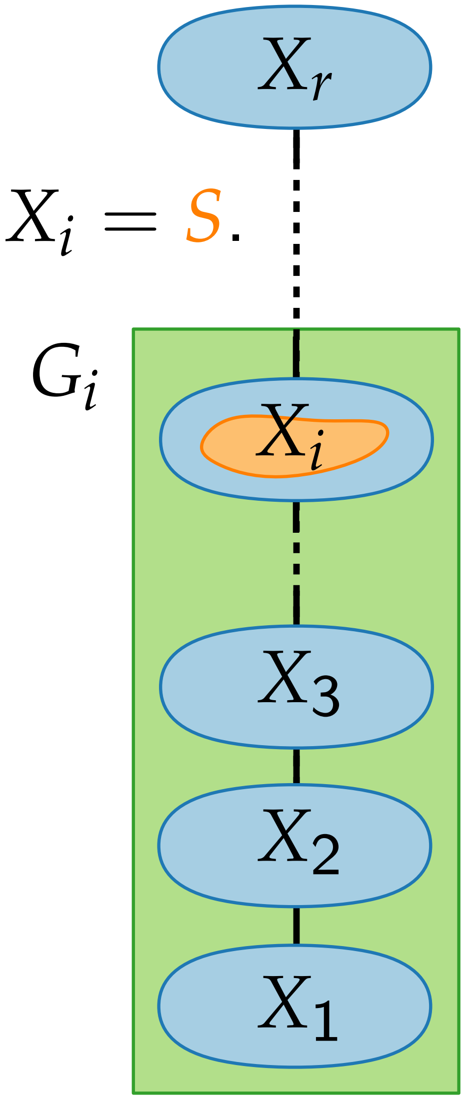
Assume we are given a nice path decomposition $P = (X_1, X_2, \ldots, X_r)$ of width k.

Let G_i be the graph induced by $X_1 \cup X_2 \cup \cdots \cup X_i$ for some $i \in \{1, 2, \ldots, r\}$.

For each $S \subseteq X_i$ let

$D[i, S]$:= maximum weight of an independent set I in G_i such that $I \cap X_i = S$.

(P1) $\Rightarrow$ $G_r = G$ $\Rightarrow$ solution $= \max_{S \subseteq X_r} D[r, S]$



Independent Set in Graphs of Bounded Pathwidth

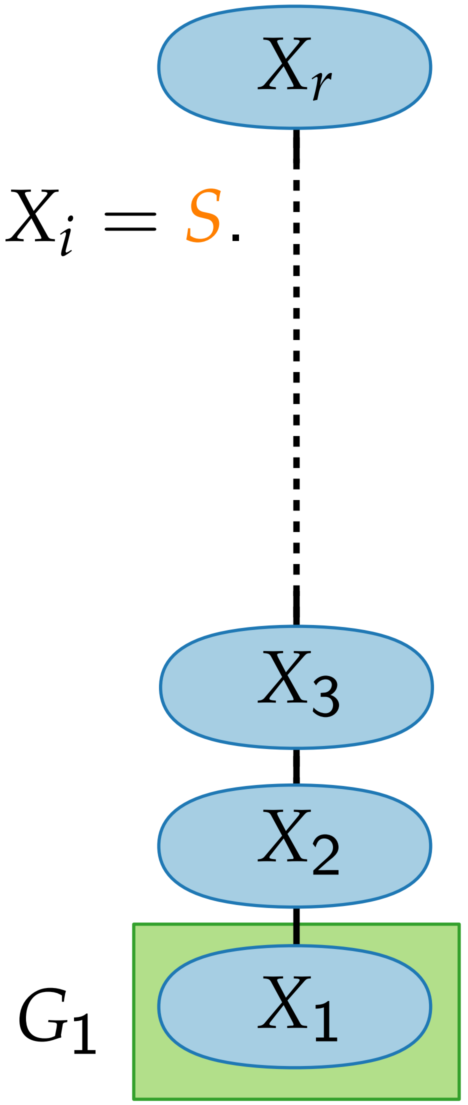
Assume we are given a nice path decomposition $P = (X_1, X_2, \ldots, X_r)$ of width k.

Let G_i be the graph induced by $X_1 \cup X_2 \cup \cdots \cup X_i$ for some $i \in \{1, 2, \ldots, r\}$.

For each $S \subseteq X_i$ let

$D[i, S] :=$ maximum weight of an independent set I in G_i such that $I \cap X_i = S$.

$$D[1, S] = \begin{cases} 0 & \text{, if } S = \emptyset \\ w(v) & \text{, if } S = \{v\} \end{cases}$$



Independent Set in Graphs of Bounded Pathwidth

Assume we are given a nice path decomposition $P = (X_1, X_2, \dots, X_r)$ of width k.

Let G_i be the graph induced by $X_1 \cup X_2 \cup \dots \cup X_i$ for some $i \in \{1, 2, \dots, r\}$.

For each $S \subseteq X_i$ let

$D[i, S]$:= maximum weight of an independent set I in G_i such that $I \cap X_i = S$.

Assume that $i > 1$. If S is not independent, $D[i, S] = -\infty$.

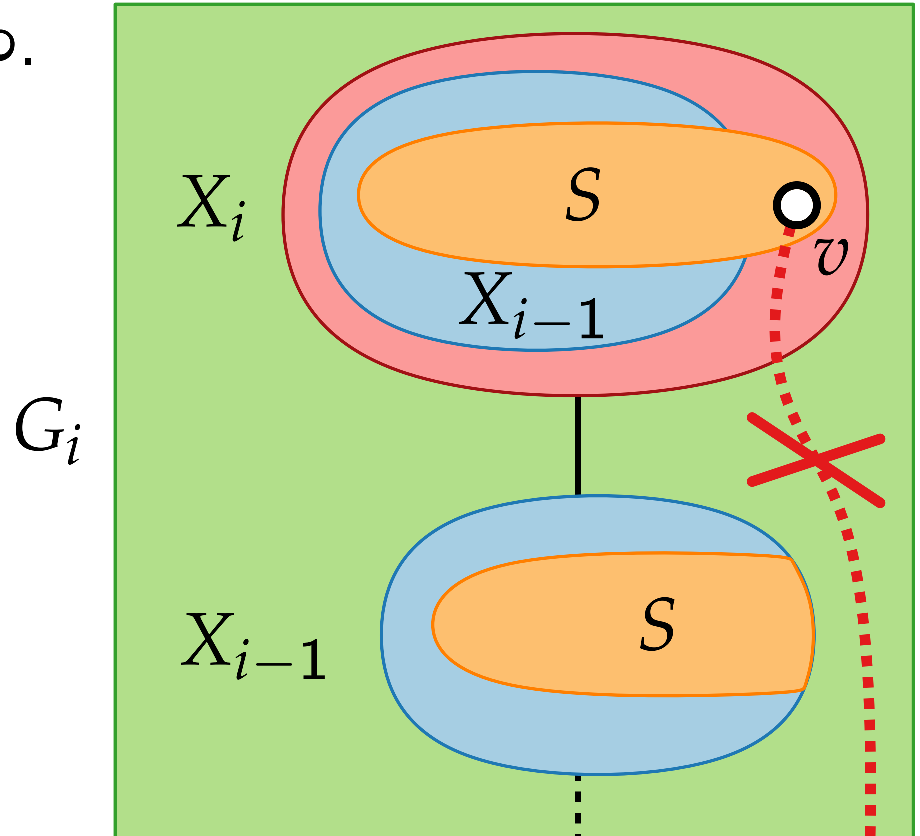
Otherwise, we distinguish between the two types of X_i.

If X_i is **Introduce**, then

$$D[i, S] = \begin{cases} D[i-1, S] & \text{, if } v \notin S \\ w(v) + D[i-1, S \setminus \{v\}] & \text{, if } v \in S \end{cases}$$

Let I' denote the independent set corresponding to

Why is $I' \cup \{v\}$ independent? due to Lemma 1!



INDEPENDENT SET in Graphs of Bounded Pathwidth

Assume we are given a nice path decomposition $P = (X_1, X_2, \ldots, X_r)$ of width k.

Let G_i be the graph induced by $X_1 \cup X_2 \cup \cdots \cup X_i$ for some $i \in \{1, 2, \ldots, r\}$.

For each $S \subseteq X_i$ let

$D[i, S] :=$ maximum weight of an independent set I in G_i such that $I \cap X_i = S$.

Assume that $i > 1$. If S is not independent, $D[i, S] = -\infty$.

Otherwise, we distinguish between the two types of X_i.

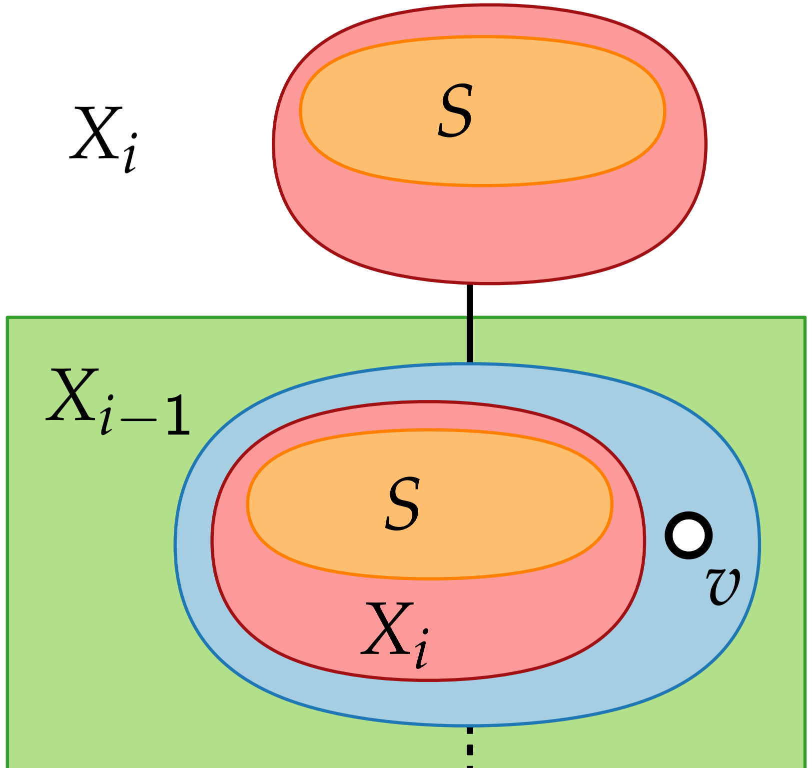
If X_i is **Forget**, then

$$D[i, S] = \max\{\ D[i-1, S]\ ,\ D[i-1, S \cup \{v\}]\ \}$$

$v \notin I \Rightarrow I \cap X_{i-1} = S$

$v \in I \Rightarrow I \cap X_{i-1} = S \cup \{v\}$

$G_i = G_{i-1}$



INDEPENDENT SET in Graphs of Bounded Pathwidth

Assume we are given a nice path decomposition $P = (X_1, X_2, \ldots, X_r)$ of width k.

Let G_i be the graph induced by $X_1 \cup X_2 \cup \cdots \cup X_i$ for some $i \in \{1, 2, \ldots, r\}$.

For each $S \subseteq X_i$ let

$D[i, S] :=$ maximum weight of an independent set I in G_i such that $I \cap X_i = S$.

Assume that $i > 1$. If S is not independent, $D[i, S] = -\infty$.

Otherwise, we distinguish between the two types of X_i.

For each of the $\mathcal{O}(|V|)$ many bags, there are $\leq 2^{k+1}$ choices for S.

For each of these choices, we need to test if S is independent, which can be done in $k^{\mathcal{O}(1)}$ time ($\rightarrow$ Section 7.3.1 in [1]).

$\Rightarrow$ total running time $\leq 2^k k^{\mathcal{O}(1)} |V|$

Theorem. INDEPENDENT SET is FPT with respect to the pathwidth.

Discussion

- The fixed-parameter tractability of a problem may be studied with respect to various structural parameters.
- The assumption that the chosen parameter is small should be plausible!
- **Treewidth** is among the most studied parameters.
  - It is defined like pathwidth, except that the bags form a tree instead of a path.
  - **Nice** tree decomposition only have one additional bag type ...
  - ... and can be constructed efficiently from a tree decomposition.
- Our $\leq 2^{\text{pw}(G)}\text{pw}(G)^{\mathcal{O}(1)}|V|$ time Algorithm for Independent Set can easily be turned into an algorithm with running time $\leq 2^{\text{tw}(G)}\text{tw}(G)^{\mathcal{O}(1)}|V|$.

Theorem. Independent Set is FPT with respect to the treewidth.

References and Literature

[1] Parameterized Algorithms,
M. Cygan, F. Fomin, Ł. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk,
M. Pilipczuk, S. Saurabh, Springer International Publishing 2015.

Sections 1, 7.1, 7.2, 7.3