Treewidth: Vol. 3

Dániel Marx

Lecture #13
February 1, 2022
**Treewidth — a measure of “tree-likeness”**

**Tree decomposition:** Vertices are arranged in a tree structure satisfying the following properties:

1. If \( u \) and \( v \) are neighbors, then there is a bag containing both of them.
2. For every \( v \), the bags containing \( v \) form a connected subtree.
Treewidth — a measure of “tree-likeness”

**Tree decomposition:** Vertices are arranged in a tree structure satisfying the following properties:

1. If $u$ and $v$ are neighbors, then there is a bag containing both of them.
2. For every $v$, the bags containing $v$ form a connected subtree.
Treewidth — a measure of “tree-likeness”

**Tree decomposition:** Vertices are arranged in a tree structure satisfying the following properties:

1. If $u$ and $v$ are neighbors, then there is a bag containing both of them.
2. For every $v$, the bags containing $v$ form a connected subtree.
Treewidth — a measure of “tree-likeness”

**Tree decomposition:** Vertices are arranged in a tree structure satisfying the following properties:

1. If $u$ and $v$ are neighbors, then there is a bag containing both of them.
2. For every $v$, the bags containing $v$ form a connected subtree.

**Width of the decomposition:** largest bag size $-1$.

**Treewidth:** width of the best decomposition.
Treewidth — a measure of “tree-likeness”

**Tree decomposition:** Vertices are arranged in a tree structure satisfying the following properties:
1. If \( u \) and \( v \) are neighbors, then there is a bag containing both of them.
2. For every \( v \), the bags containing \( v \) form a connected subtree.

**Width of the decomposition:** largest bag size \(-1\).

**Treewidth:** width of the best decomposition.

A subtree communicates with the outside world only via the root of the subtree.
Weighted Max Independent Set and treewidth

Theorem

Given a tree decomposition of width $w$, Weighted Max Independent Set can be solved in time $O(2^w \cdot w^{O(1)} \cdot n)$.

$B_x$: vertices appearing in node $x$.
$V_x$: vertices appearing in the subtree rooted at $x$.

Generalizing our solution for trees:

Instead of computing 2 values $A[v], B[v]$ for each vertex of the graph, we compute $2^{|B_x|} \leq 2^{w+1}$ values for each bag $B_x$.

$M[x, S]$: the max. weight of an independent set $I \subseteq V_x$ with $I \cap B_x = S$. 
**Weighted Max Independent Set** and treewidth

**Theorem**

Given a tree decomposition of width $w$, Weighted Max Independent Set can be solved in time $O(2^w \cdot w^{O(1)} \cdot n)$.

$B_x$: vertices appearing in node $x$.

$V_x$: vertices appearing in the subtree rooted at $x$.

Generalizing our solution for trees:

Instead of computing 2 values $A[v], B[v]$ for each vertex of the graph, we compute $2^{|B_x|} \leq 2^{w+1}$ values for each bag $B_x$.

$M[x, S]$: the max. weight of an independent set $I \subseteq V_x$ with $I \cap B_x = S$.

How to determine $M[x, S]$ if all the values are known for the children of $x$?
Monadic Second Order Logic

Extended Monadic Second Order Logic (EMSO)

A logical language on graphs consisting of the following:

- Logical connectives $\land$, $\lor$, $\rightarrow$, $\neg$, $=$, $\neq$
- Quantifiers $\forall$, $\exists$ over vertex/edge variables
- Predicate $\text{adj}(u, v)$: vertices $u$ and $v$ are adjacent
- Predicate $\text{inc}(e, v)$: edge $e$ is incident to vertex $v$
- Quantifiers $\forall$, $\exists$ over vertex/edge set variables
- $\in$, $\subseteq$ for vertex/edge sets

Example:
The formula

$$\exists C \subseteq V \forall v \in C \exists u_1, u_2 \in C (u_1 \neq u_2 \land \text{adj}(u_1, v) \land \text{adj}(u_2, v))$$

is true on graph $G$ if and only if $G$ has a cycle.
Courcelle’s Theorem

There exists an algorithm that, given a width-$w$ tree decomposition of an $n$-vertex graph $G$ and an EMSO formula $\phi$, decides whether $G$ satisfies $\phi$ in time $f(w, |\phi|) \cdot n$.

If we can express a property in EMSO, then we immediately get that testing this property is FPT parameterized by the treewidth $w$ of the input graph.

⇒ The following problem are FPT parameterized by treewidth:

- $c$-Coloring
- Hamiltonian Cycle
- Partition into Triangles
- ...
Input: graphs $H$ and $G$
Find: a subgraph of $G$ isomorphic to $H$. 
Subgraph Isomorphism

Input: graphs $H$ and $G$
Find: a subgraph of $G$ isomorphic to $H$.

For each $H$, we can construct a formula $\phi_H$ that expresses “$G$ has a subgraph isomorphic to $H$”.

⇒ By Courcelle’s Theorem, Subgraph Isomorphism can be solved in time $f(H, w) \cdot n$ if $G$ has treewidth at most $w$.

Theorem
Subgraph Isomorphism is FPT parameterized by combined parameter $k := |V(H)|$ and the treewidth $w$ of $G$. 
Finding tree decompositions

**Fixed-parameter tractability:**

**Theorem** [Bodlaender 1996]

There is a $2^{O(w^3)} \cdot n$ time algorithm that finds a tree decomposition of width $w$ (if exists).

Sometimes we can get better dependence on treewidth using approximation.

**FPT approximation:**

**Theorem**

There is a $O(3^{3w} \cdot w \cdot n^2)$ time algorithm that finds a tree decomposition of width $4w + 1$, if the treewidth of the graph is at most $w$. 
Minor

An operation similar to taking subgraphs:

Definition

Graph $H$ is a minor of $G$ ($H \leq G$) if $H$ can be obtained from $G$ by deleting edges, deleting vertices, and contracting edges.
A classical result

Theorem [Kuratowski 1930]
A graph $G$ is planar if and only if $G$ does not contain a subdivision of $K_5$ or $K_{3,3}$. 

\[ \begin{array}{c}
K_5 \\
\end{array} \quad \begin{array}{c}
K_{3,3} \\
\end{array} \]
A classical result

Theorem [Kuratowski 1930]
A graph $G$ is planar if and only if $G$ does not contain a subdivision of $K_5$ or $K_{3,3}$.

Theorem [Wagner 1937]
A graph $G$ is planar if and only if $G$ does not contain $K_5$ or $K_{3,3}$ as minor.
Graph Minors Theory

Neil Robertson  Paul Seymour

Theory of graph minors developed in the monumental series

Graph Minors I–XXIII.
J. Combin. Theory, Ser. B
1983–2012

- Structure theory of graphs excluding minors (and much more).
- Galactic combinatorial bounds and running times.
- Important early influence for parameterized algorithms.
Properties of treewidth

**Fact:** Treewidth does not increase if we delete edges, delete vertices, or contract edges. 

⇒ If $F$ is a minor of $G$, then the treewidth of $F$ is at most the treewidth of $G$. 
Properties of treewidth

**Fact:** Treewidth does not increase if we delete edges, delete vertices, or contract edges.

⇒ If $F$ is a minor of $G$, then the treewidth of $F$ is at most the treewidth of $G$.

**Fact:** For every clique $K$, there is a bag $B$ with $K \subseteq B$.

**Fact:** The treewidth of the $k$-clique is $k - 1$. 
Properties of treewidth

**Fact:** Treewidth does not increase if we delete edges, delete vertices, or contract edges.  
⇒ If $F$ is a minor of $G$, then the treewidth of $F$ is at most the treewidth of $G$.

**Fact:** For every clique $K$, there is a bag $B$ with $K \subseteq B$.

**Fact:** The treewidth of the $k$-clique is $k - 1$.

**Fact:** For every $k \geq 2$, the treewidth of the $k \times k$ grid is exactly $k$.  

![Diagram of a $k \times k$ grid]
The Cops and Robber game

**Game:** $k$ cops try to capture a robber in the graph.

- In each step, (a subset of) the cops can move from vertex to vertex arbitrarily with helicopters.
- The robber moves infinitely fast on the edges, cannot move through the cops staying on the ground, and sees where the cops will land.
The Cops and Robber game

Example: 2 cops have a winning strategy in a tree.
The Cops and Robber game

**Example:** 2 cops have a winning strategy in a tree.
The Cops and Robber game

Example: 2 cops have a winning strategy in a tree.
The Cops and Robber game

Example: 2 cops have a winning strategy in a tree.
The Cops and Robber game

Example: 2 cops have a winning strategy in a tree.
The Cops and Robber game

Example: 2 cops have a winning strategy in a tree.
The Cops and Robber game

Example: 2 cops have a winning strategy in a tree.
The Cops and Robber game

Example: 2 cops have a winning strategy in a tree.
The Cops and Robber game

Example: 2 cops have a winning strategy in a tree.
The Cops and Robber game

**Theorem** [Seymour and Thomas 1993]

\[ k + 1 \text{ cops can win the game} \iff \text{the treewidth of the graph is at most } k. \]
The Cops and Robber game

**Theorem** [Seymour and Thomas 1993]

\[ k + 1 \text{ cops can win the game } \iff \text{the treewidth of the graph is at most } k. \]

**Consequence 1: Algorithms**

The winner of the game can be determined in time \( n^{O(k)} \) using standard techniques (there are at most \( n^k \) positions for the cops)

\[ \Downarrow \]

For every fixed \( k \), it can be checked in polynomial time if treewidth is at most \( k \).

(But \( f(k) \cdot n^{O(1)} \) algorithms are also known with different techniques!)
The Cops and Robber game

Theorem [Seymour and Thomas 1993]

\[ k + 1 \text{ cops can win the game} \iff \text{the treewidth of the graph is at most } k. \]

Consequence 2: Lower bounds

Exercise 1:
Show that the treewidth of the \( k \times k \) grid is at least \( k - 1 \).
(E.g., robber can win against \( k - 1 \) cops.)

Exercise 2:
Show that the treewidth of the \( k \times k \) grid is at least \( k \).
(E.g., robber can win against \( k \) cops.)
Excluded Grid Theorem

If the treewidth of $G$ is $\Omega(k^9 \log k)$, then $G$ has a $k \times k$ grid minor.
Excluded Grid Theorem

If the treewidth of $G$ is $\Omega(k^9 \log k)$, then $G$ has a $k \times k$ grid minor.

A large grid minor is a “witness” that treewidth is large, but the relation is approximate:
Excluded Grid Theorem

**Excluded Grid Theorem**

If the treewidth of $G$ is $\Omega(k^9 \log k)$, then $G$ has a $k \times k$ grid minor.

**Observation:** Every planar graph is the minor of a sufficiently large grid.

**Consequence**

If $H$ is planar, then every $H$-minor free graph has treewidth at most $f(H)$. 
Planar Excluded Grid Theorem

For planar graphs, we get linear instead of exponential dependence:

**Theorem**

Every *planar graph* with treewidth at least $5k$ has a $k \times k$ grid minor.
Planar Excluded Grid Theorem

For planar graphs, we get linear instead of exponential dependence:

**Theorem**

Every **planar graph** with treewidth at least $5k$ has a $k \times k$ grid minor.

**Theorem**

An $n$-vertex planar graph has treewidth $O(\sqrt{n})$. 
Outerplanar graphs

**Definition**
A planar graph is **outerplanar** if it has a planar embedding where every vertex is on the infinite face.

**Fact**
Every outerplanar graph has treewidth at most 2.
$k$-outerplanar graphs

Given a planar embedding, we can define **layers** by iteratively removing the vertices on the infinite face.

**Definition**

A planar graph is **$k$-outerplanar** if it has a planar embedding having at most $k$ layers.

**Fact**

Every $k$-outerplanar graph has treewidth at most $3k + 1$. 
**k-outerplanar graphs**

Given a planar embedding, we can define *layers* by iteratively removing the vertices on the infinite face.

**Definition**

A planar graph is **k-outerplanar** if it has a planar embedding having at most $k$ layers.

**Fact**

Every $k$-outerplanar graph has treewidth at most $3k + 1$. 
$k$-outerplanar graphs

Given a planar embedding, we can define **layers** by iteratively removing the vertices on the infinite face.

**Definition**

A planar graph is **$k$-outerplanar** if it has a planar embedding having at most $k$ layers.

**Fact**

Every $k$-outerplanar graph has treewidth at most $3k + 1$. 
**$k$-outerplanar graphs**

Given a planar embedding, we can define **layers** by iteratively removing the vertices on the infinite face.

**Definition**

A planar graph is **$k$-outerplanar** if it has a planar embedding having at most $k$ layers.

**Fact**

Every $k$-outerplanar graph has treewidth at most $3k + 1$. 
**$k$-outerplanar graphs**

Given a planar embedding, we can define **layers** by iteratively removing the vertices on the infinite face.

**Definition**

A planar graph is **$k$-outerplanar** if it has a planar embedding having at most $k$ layers.

**Fact**

Every $k$-outerplanar graph has treewidth at most $3k + 1$. 
Treewidth — outline

1 Basic algorithms
2 Complexity results
3 Combinatorial properties
4 Applications
   • The shifting technique
   • Bidimensionality
Approximation schemes

**Definition**

A polynomial-time approximation scheme (PTAS) for a problem $P$ is an algorithm that takes an instance of $P$ and a rational number $\epsilon > 0$,

- always finds a $(1 + \epsilon)$-approximate solution,
- the running time is polynomial in $n$ for every fixed $\epsilon > 0$.

Typical running times: $2^{1/\epsilon} \cdot n$, $n^{1/\epsilon}$, $(n/\epsilon)^2$, $n^{1/\epsilon^2}$. 

Some classical problems that have a PTAS:
- Independent Set for planar graphs
- TSP in the Euclidean plane
- Steiner Tree in planar graphs
- Knapsack
Approximation schemes

Definition

A polynomial-time approximation scheme (PTAS) for a problem $P$ is an algorithm that takes an instance of $P$ and a rational number $\epsilon > 0$,

- always finds a $(1 + \epsilon)$-approximate solution,
- the running time is polynomial in $n$ for every fixed $\epsilon > 0$.

Typical running times: $2^{1/\epsilon} \cdot n$, $n^{1/\epsilon}$, $(n/\epsilon)^2$, $n^{1/\epsilon^2}$.

Some classical problems that have a PTAS:

- **Independent Set** for planar graphs
- **TSP** in the Euclidean plane
- **Steiner Tree** in planar graphs
- **Knapsack**
Baker’s shifting strategy for PTAS

Theorem

There is a $2^{O(1/\epsilon)} \cdot n$ time PTAS for \textsc{Independent Set} for planar graphs.

Let $D := 1/\epsilon$. For a fixed $0 \leq s < D$, delete every layer $L_i$ with $i = s \pmod{D}$.
Theorem

There is a $2^{O(1/\epsilon)} \cdot n$ time PTAS for \textsc{Independent Set} for planar graphs.

- Let $D := 1/\epsilon$. For a fixed $0 \leq s < D$, delete every layer $L_i$ with $i = s \pmod{D}$.
Baker’s shifting strategy for PTAS

**Theorem**

There is a $2^{O(1/\epsilon)} \cdot n$ time PTAS for **INDEPENDENT SET** for planar graphs.

Let $D := 1/\epsilon$. For a fixed $0 \leq s < D$, delete every layer $L_i$ with $i = s \pmod{D}$
Baker’s shifting strategy for PTAS

Theorem

There is a $2^{O(1/\epsilon)} \cdot n$ time PTAS for **Independent Set** for planar graphs.

- Let $D := 1/\epsilon$. For a fixed $0 \leq s < D$, delete every layer $L_i$ with $i = s \pmod{D}$.
Baker’s shifting strategy for PTAS

**Theorem**

There is a $2^{O(1/\epsilon)} \cdot n$ time PTAS for **Independent Set** for planar graphs.

- Let $D := 1/\epsilon$. For a fixed $0 \leq s < D$, delete every layer $L_i$ with $i = s \pmod{D}$
Baker’s shifting strategy for PTAS

**Theorem**

There is a $2^{O(1/\epsilon)} \cdot n$ time PTAS for **Independent Set** for planar graphs.

- Let $D := 1/\epsilon$. For a fixed $0 \leq s < D$, delete every layer $L_i$ with $i = s \pmod{D}$.
- The resulting graph is $D$-outerplanar, hence it has treewidth at most $3D + 1 = O(1/\epsilon)$.
- Using the $2^{O(\text{tw})} \cdot n$ time algorithm for **Independent Set**, the problem on the $D$-outerplanar graph can be solved in time $2^{O(1/\epsilon)} \cdot n$. 
Baker’s shifting strategy for PTAS

**Theorem**
There is a $2^{O(1/\epsilon)} \cdot n$ time PTAS for **Independent Set** for planar graphs.

We do this for every $0 \leq s < D$:
for at least one value of $s$, we delete
at most $1/D = \epsilon$ fraction of the solution

\[ \downarrow \]
We get a $(1 + \epsilon)$-approximate solution.
Baker’s shifting strategy for FPT

**Subgraph Isomorphism**

Input: graphs $H$ and $G$
Find: a subgraph $G$ isomorphic to $H$. 
Baker’s shifting strategy for FPT

**Subgraph Isomorphism**

- **Input:** graphs $H$ and $G$
- **Find:** a subgraph $G$ isomorphic to $H$.

- For a fixed $0 \leq s < k + 1$, delete every layer $L_i$ with $i = s \pmod{k + 1}$

The resulting graph is $k$-outerplanar, hence it has treewidth at most $3k + 1$.

Using the $f(k, \text{tw}) \cdot n$ time algorithm for Subgraph Isomorphism, the problem can be solved in time $f(k, 3k + 1) \cdot n$. 
Baker’s shifting strategy for FPT

**Subgraph Isomorphism**

Input: graphs $H$ and $G$
Find: a subgraph $G$ isomorphic to $H$.

- For a fixed $0 \leq s < k + 1$, delete every layer $L_i$ with $i = s \pmod{k + 1}$

The resulting graph is $k$-outerplanar, hence it has treewidth at most $3k + 1$. Using the $f(k, \text{tw}) \cdot n$ time algorithm for Subgraph Isomorphism, the problem can be solved in time $f(k, 3k + 1) \cdot n$. 

22
Baker's shifting strategy for FPT

**Subgraph Isomorphism**

**Input:** graphs $H$ and $G$

**Find:** a subgraph $G$ isomorphic to $H$.

- For a fixed $0 \leq s < k + 1$, delete every layer $L_i$ with $i = s \pmod {k + 1}$

The resulting graph is $k$-outerplanar, hence it has treewidth at most $3k + 1$. Using the $f(k, \text{tw}) \cdot n$ time algorithm for Subgraph Isomorphism, the problem can be solved in time $f(k, 3k + 1) \cdot n$. 
Baker’s shifting strategy for FPT

**Subgraph Isomorphism**

Input: graphs $H$ and $G$
Find: a subgraph $G$ isomorphic to $H$.

- For a fixed $0 \leq s < k + 1$, delete every layer $L_i$ with $i = s \pmod{k + 1}$
Baker’s shifting strategy for FPT

**Subgraph Isomorphism**

- **Input:** graphs $H$ and $G$
- **Find:** a subgraph $G$ isomorphic to $H$.

- For a fixed $0 \leq s < k + 1$, delete every layer $L_i$ with $i = s \pmod{k + 1}$
- The resulting graph is $k$-outerplanar, hence it has treewidth at most $3k + 1$.
- Using the $f(k, tw) \cdot n$ time algorithm for **Subgraph Isomorphism**, the problem can be solved in time $f(k, 3k + 1) \cdot n$. 
Baker’s shifting strategy for FPT

**Subgraph Isomorphism**

**Input:** graphs $H$ and $G$  
**Find:** a subgraph $G$ isomorphic to $H$.

We do this for every $0 \leq s < k + 1$:

for at least one value of $s$, we do not delete any of the $k$ vertices of the solution

\[\downarrow\]

We find a copy of $H$ in $G$ if there is one.
Baker’s shifting strategy for FPT

**Subgraph Isomorphism**

Input: graphs $H$ and $G$

Find: a subgraph $G$ isomorphic to $H$.

We do this for every $0 \leq s < k + 1$:

for at least one value of $s$, we do not delete any of the $k$ vertices of the solution

↓

We find a copy of $H$ in $G$ if there is one.
Baker’s shifting strategy for FPT

**Subgraph Isomorphism**

Input: graphs $H$ and $G$
Find: a subgraph $G$ isomorphic to $H$.

We do this for every $0 \leq s < k + 1$:
for at least one value of $s$, we do not delete
any of the $k$ vertices of the solution

\[\downarrow\]

We find a copy of $H$ in $G$ if there is one.
Baker’s shifting strategy for FPT

**Subgraph Isomorphism**

- **Input:** graphs $H$ and $G$
- **Find:** a subgraph $G$ isomorphic to $H$.

We do this for every $0 \leq s < k + 1$:
for at least one value of $s$, we do not delete any of the $k$ vertices of the solution

\[\downarrow\]

We find a copy of $H$ in $G$ if there is one.
Baker’s shifting strategy for FPT

**Subgraph Isomorphism**

Input: graphs $H$ and $G$
Find: a subgraph $G$ isomorphic to $H$.

**Theorem**

*Subgraph Isomorphism* for planar graphs is FPT parameterized by $k := |V(H)|$. 
Baker’s shifting strategy for FPT

- The technique is very general, works for many problems on planar graphs:
  - Independent Set
  - Vertex Cover
  - Dominating Set
  - $k$-Path
  - ...

- More generally: First-Order Logic problems.

- But for some of these problems, much better techniques are known (see the following slides).
The race for better FPT algorithms

- Single exponential
- Double exponential
- Tower of exponentials
- "Slightly super-exponential"
- Subexponential

\[ 2^{O(k \log k)} \rightarrow 2^{O(\sqrt{k \log k})} \rightarrow 2^{O(\sqrt{k})} \rightarrow 2^{O(\frac{1}{3})} \rightarrow \ldots \]
Square root phenomenon

Most NP-hard problems (e.g., 3-Coloring, Independent Set, Hamiltonian Cycle, Steiner Tree, etc.) remain NP-hard on planar graphs.\(^1\)

---

\(^1\)Notable exception: Max Cut is in P for planar graphs.
Square root phenomenon

Most NP-hard problems (e.g., 3-Coloring, Independent Set, Hamiltonian Cycle, Steiner Tree, etc.) remain NP-hard on planar graphs.\(^1\)

The running time is still exponential, but significantly smaller:

\[
2^\Omega(n) \Rightarrow 2^{\Omega\left(\sqrt{n}\right)} \\
2^\Omega(k) \Rightarrow 2^{\Omega\left(\sqrt{k}\right)} \\
n^{\Omega(k)} \Rightarrow n^{\Omega\left(\sqrt{k}\right)} \\
2^{\Omega(k)} \cdot n^{\Omega(1)} \Rightarrow 2^{\Omega\left(\sqrt{k}\right)} \cdot n^{\Omega(1)}
\]

**Example:** A planar \(n\)-vertex graph has treewidth \(2^{\Omega\left(\sqrt{n}\right)} \Rightarrow 3\text{-Coloring}\) can be solved in time \(2^{\Omega\left(\sqrt{n}\right)}\) in planar graphs.

\(^{1}\)Notable exception: **Max Cut** is in P for planar graphs.
**Theorem**

**Vertex Cover** can be solved in time $2^{O(\sqrt{k})} \cdot n^{O(1)}$ in planar graphs.

We need two facts:

- Removing an edge, removing a vertex, contracting an edge cannot increase the vertex cover number.
- **Vertex Cover** can be solved in time $2^w \cdot n^{O(1)}$ if a tree decomposition of width $w$ is given.
**Vertex Cover**

**Observation:** If the treewidth of a planar graph $G$ is at least $5\sqrt{2k}$

$\Rightarrow$ It has a $\sqrt{2k} \times \sqrt{2k}$ grid minor (Planar Excluded Grid Theorem)

$\Rightarrow$ The grid has a matching of size $k$

$\Rightarrow$ Vertex cover size is at least $k$ in the grid.

$\Rightarrow$ Vertex cover size is at least $k$ in $G$. 

---

We use this observation to solve Vertex Cover on planar graphs:
**Vertex Cover**

**Observation:** If the treewidth of a planar graph $G$ is at least $5\sqrt{2k}$

- It has a $\sqrt{2k} \times \sqrt{2k}$ grid minor (Planar Excluded Grid Theorem)
- The grid has a matching of size $k$
- Vertex cover size is at least $k$ in the grid.
- Vertex cover size is at least $k$ in $G$.

We use this observation to solve Vertex Cover on planar graphs:

- If treewidth is at least $5\sqrt{2k}$: we answer “vertex cover is $\geq k$.”
- If treewidth is less than $5\sqrt{2k}$, then we can solve the problem in time
  $2^{O(5\sqrt{2k})} \cdot n^{O(1)} = 2^{O(\sqrt{k})} \cdot n^{O(1)}$. 


**Vertex Cover**

**Observation:** If the treewidth of a planar graph $G$ is at least $5\sqrt{2}k$

$\Rightarrow$ It has a $\sqrt{2}k \times \sqrt{2}k$ grid minor (Planar Excluded Grid Theorem)

$\Rightarrow$ The grid has a matching of size $k$

$\Rightarrow$ Vertex cover size is at least $k$ in the grid.

$\Rightarrow$ Vertex cover size is at least $k$ in $G$.

We use this observation to solve **Vertex Cover** on planar graphs:

- Set $w := 5\sqrt{2}k$.
- Find a 4-approximate tree decomposition.
  - If treewidth is at least $w$: we answer “vertex cover is $\geq k$.”
  - If we get a tree decomposition of width $4w$, then we can solve the problem in time $2^{O(w)} \cdot n^{O(1)} = 2^{O(\sqrt{k})} \cdot n^{O(1)}$. 
Bidimensionality

A powerful framework for efficient algorithms on planar graphs.

Setup:
- Let $x(G)$ be some graph invariant (i.e., an integer associated with each graph).
- Given $G$ and $k$, we want to decide if $x(G) \leq k$ (or $x(G) \geq k$).
- Typical examples:
  - Maximum independent set size.
  - Minimum vertex cover size.
  - Length of the longest path.
  - Minimum dominating set size.
  - Minimum feedback vertex set size.

For many natural invariants, we can do this in time $2^{O(\sqrt{k})} \cdot n^{O(1)}$ on planar graphs.
Bidimensionality

**Definition**

A graph invariant $x(G)$ is **minor-bidimensional** if

- $x(G') \leq x(G)$ for every minor $G'$ of $G$, and
- If $G_k$ is the $k \times k$ grid, then $x(G_k) \geq ck^2$ (for some constant $c > 0$).

**Examples:** minimum vertex cover, length of the longest path, feedback vertex set are minor-bidimensional.
Bidimensionality

Definition

A graph invariant $x(G)$ is **minor-bidimensional** if

- $x(G') \leq x(G)$ for every minor $G'$ of $G$, and
- If $G_k$ is the $k \times k$ grid, then $x(G_k) \geq ck^2$ (for some constant $c > 0$).

Examples: minimum vertex cover, **length of the longest path**, feedback vertex set are minor-bidimensional.
Bidimensionality

Definition

A graph invariant \(x(G)\) is **minor-bidimensional** if

- \(x(G') \leq x(G)\) for every minor \(G'\) of \(G\), and
- If \(G_k\) is the \(k \times k\) grid, then \(x(G_k) \geq ck^2\) (for some constant \(c > 0\)).

Examples: minimum vertex cover, length of the longest path, feedback vertex set are minor-bidimensional.
We can answer “$x(G) \geq k$?” for a minor-bidimensional invariant the following way:

- Set $w := c\sqrt{k}$ for an appropriate constant $c$.
- Use the 4-approximation tree decomposition algorithm.
  - If treewidth is at least $w$: $x(G)$ is at least $k$.
  - If we get a tree decomposition of width $4w$, then we can solve the problem using dynamic programming on the tree decomposition.

**Running time:**

- If we can solve the problem on tree decomposition of width $w$ in time $2^{O(w)} \cdot n^{O(1)}$, then the running time is $2^{O(\sqrt{k})} \cdot n^{O(1)}$.
- If we can solve the problem on tree decomposition of width $w$ in time $w^{O(w)} \cdot n^{O(1)}$, then the running time is $2^{O(\sqrt{k}\log k)} \cdot n^{O(1)}$. 
Contraction bidimensionality

**Definition**

A graph invariant $x(G)$ is **minor-bidimensional** if

- $x(G') \leq x(G)$ for every minor $G'$ of $G$, and
- If $G_k$ is the $k \times k$ grid, then $x(G_k) \geq ck^2$
  (for some constant $c > 0$).

**Exercise:** **Dominating Set** is **not** minor-bidimensional.
Contraction bidimensionality

**Definition**
A graph invariant $x(G)$ is **minor-bidimensional** if
- $x(G') \leq x(G)$ for every minor $G'$ of $G$, and
- If $G_k$ is the $k \times k$ grid, then $x(G_k) \geq ck^2$ (for some constant $c > 0$).

**Exercise:** **Dominating Set** is **not** minor-bidimensional.

We fix the problem by allowing only contractions but not edge/vertex deletions.
Contraction bidimensionality

**Theorem**

Every **planar graph** with treewidth at least $9k + 5$ can be contracted to a **triangulated** $k \times k$ grid $\Gamma_k$.

$5 \times 5$ triangulated grid $\Gamma_5$: 

![Diagram of a 5x5 triangulated grid](image_url)
Contraction bidimensionality

Definition

A graph invariant $x(G)$ is **contraction-bidimensional** if

- $x(G') \leq x(G)$ for every contraction $G'$ of $G$, and
- $x(\Gamma_k) \geq ck^2$ (for some $c > 0$).

$5 \times 5$ triangulated grid $\Gamma_5$: 

![Diagram of $5 \times 5$ triangulated grid $\Gamma_5$]
Contraction bidimensionality

Definition
A graph invariant $x(G)$ is contraction-bidimensional if
- $x(G') \leq x(G)$ for every contraction $G'$ of $G$, and
- $x(\Gamma_k) \geq ck^2$ (for some $c > 0$).

5 × 5 triangulated grid $\Gamma_5$:

Example: maximum independent set, minimum dominating set are contraction-bidimensional.
Definition

A graph invariant $x(G)$ is **contraction-bidimensional** if

- $x(G') \leq x(G)$ for every contraction $G'$ of $G$, and
- $x(\Gamma_k) \geq ck^2$ (for some $c > 0$).

**Example:** maximum independent set, *minimum dominating set* are contraction-bidimensional.
Bidimensionality for **Dominating Set**

The size of a minimum dominating set is a *contraction bidimensional* invariant: we need at least \((k - 2)^2 / 7\) vertices to dominate all the internal vertices of the triangulated \(k \times k\) grid \(\Gamma_k\) (since a vertex can dominate at most 7 internal vertices).

**Theorem**

Given a tree decomposition of width \(w\), **Dominating Set** can be solved in time \(3^w \cdot w^{O(1)} \cdot n^{O(1)}\).

Solving **Dominating Set** on planar graphs:

- Set \(w := 9(3\sqrt{k} + 2)\).
- Use the 4-approximation tree decomposition algorithm.
  - If treewidth is at least \(w\): we answer ’dominating set is \(\geq k\’
  - If we get a tree decomposition of width \(4w\), then we can solve the problem in time \(3^w \cdot n^{O(1)} = 2^{O(\sqrt{k})} \cdot n^{O(1)}\).
Beyond bidimensionality

The following problems are not minor/contraction-bidimensional, but admit $2^{O(\sqrt{k \cdot \text{polylog} k})} \cdot n^{O(1)}$ time algorithms on planar graphs using other techniques:

- **Odd Cycle Transversal**
- **$k$-Path** in directed planar graphs
- **Subset Feedback Vertex Set**
- **Multiway Cut**
- **Subset TSP** (parameterized by the number of terminals)
- ...
The race for better FPT algorithms

Double exponential

Tower of exponentials

"Slightly super-exponential"

Single exponential

Subexponential
Lower bounds based on ETH

**Exponential Time Hypothesis (ETH) + Sparsification Lemma**

There is no $2^{o(n+m)}$-time algorithm for $n$-variable $m$-clause 3SAT.

The textbook reduction from 3SAT to **Vertex Cover**:

\[ x_1 \bar{x}_1 \quad x_2 \bar{x}_2 \quad x_3 \bar{x}_3 \quad x_4 \bar{x}_4 \]

\[ \begin{array}{cccccc}
    \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
    \bullet & \bullet & \bullet & \bullet & \bullet & \bullet \\
\end{array} \]
Lower bounds based on ETH

Exponential Time Hypothesis (ETH) + Sparsification Lemma

There is no $2^{o(n+m)}$-time algorithm for $n$-variable $m$-clause 3SAT.

The textbook reduction from 3SAT to Vertex Cover:

formula is satisfiable $\iff$ there is a vertex cover of size $n + 2m$
Lower bounds based on ETH

Exponential Time Hypothesis (ETH) + Sparsification Lemma

There is no $2^{o(n+m)}$-time algorithm for $n$-variable $m$-clause 3SAT.

The textbook reduction from 3SAT to Vertex Cover:

3SAT formula $\phi$

- $n$ variables
- $m$ clauses

$\Rightarrow$

Graph $G$

- $O(n + m)$ vertices
- $O(n + m)$ edges
Lower bounds based on ETH

Exponential Time Hypothesis (ETH) + Sparsification Lemma

There is no $2^{o(n+m)}$-time algorithm for $n$-variable $m$-clause 3SAT.

The textbook reduction from 3SAT to Vertex Cover:

3SAT formula $\phi$
- $n$ variables
- $m$ clauses

$\Rightarrow$
Graph $G$
- $O(n + m)$ vertices
- $O(n + m)$ edges

Corollary
Assuming ETH, there is no $2^{o(n)}$ algorithm for Vertex Cover on an $n$-vertex graph.
Lower bounds based on ETH

Exponential Time Hypothesis (ETH) + Sparsification Lemma

There is no $2^{o(n+m)}$-time algorithm for $n$-variable $m$-clause 3SAT.

The textbook reduction from 3SAT to Vertex Cover:

3SAT formula $\phi$
- $n$ variables
- $m$ clauses

$\Rightarrow$

Graph $G$
- $O(n + m)$ vertices
- $O(n + m)$ edges

Corollary

Assuming ETH, there is no $2^{o(k)} \cdot n^{O(1)}$ algorithm for Vertex Cover.
Other problems

There are polytime reductions from 3SAT to many problems such that the reduction creates a graph with $O(n + m)$ vertices/edges.

**Consequence:** Assuming ETH, the following problems cannot be solved in time $2^{o(n)}$ and hence in time $2^{o(k)} \cdot n^{O(1)}$ (but $2^{O(k)} \cdot n^{O(1)}$ time algorithms are known):

- **Vertex Cover**
- **Longest Cycle**
- **Feedback Vertex Set**
- **Multiway Cut**
- **Odd Cycle Transversal**
- **Steiner Tree**
- ...
Lower bounds based on ETH

What about \texttt{3-Coloring} on planar graphs?

The textbook reduction from \texttt{3-Coloring} to \texttt{Planar 3-Coloring} uses a “crossover gadget” with 4 external connectors:

- In every 3-coloring of the gadget, opposite external connectors have the same color.
- Every coloring of the external connectors where the opposite vertices have the same color can be extended to the whole gadget.
- If two edges cross, replace them with a crossover gadget.
Lower bounds based on ETH

What about **3-Coloring** on planar graphs?

The textbook reduction from **3-Coloring** to **Planar 3-Coloring** uses a “crossover gadget” with 4 external connectors:

- In every 3-coloring of the gadget, opposite external connectors have the same color.
- Every coloring of the external connectors where the opposite vertices have the same color can be extended to the whole gadget.
- If two edges cross, replace them with a crossover gadget.
Lower bounds based on ETH

What about $3$-COLORING on planar graphs?

The textbook reduction from $3$-COLORING to PLANAR $3$-COLORING uses a “crossover gadget” with 4 external connectors:

- In every $3$-coloring of the gadget, opposite external connectors have the same color.
- Every coloring of the external connectors where the opposite vertices have the same color can be extended to the whole gadget.
- If two edges cross, replace them with a crossover gadget.
Lower bounds based on ETH

- The reduction from **3-Coloring** to **Planar 3-Coloring** introduces $O(1)$ new edges/vertices for each crossing.
- A graph with $m$ edges can be drawn with $O(m^2)$ crossings.

\[
\begin{array}{|c|c|c|}
\hline
3SAT \text{ formula } \phi & \text{Graph } G & \text{Planar graph } G' \\
\text{n variables} & O(m) \text{ vertices} & O(m^2) \text{ vertices} \\
\text{m clauses} & O(m) \text{ edges} & O(m^2) \text{ edges} \\
\hline
\end{array}
\]

Corollary

Assuming ETH, there is no $2^{o(\sqrt{n})}$ algorithm for **3-Coloring** on an $n$-vertex planar graph $G$. 
Lower bounds for planar problems

**Consequence:** Assuming ETH, there is no \(2^{o(\sqrt{n})}\) time algorithm on \(n\)-vertex planar graphs for

- Independent Set
- Dominating Set
- Vertex Cover
- Hamiltonian Path
- Feedback Vertex Set
- \ldots
Lower bounds for planar problems

**Consequence:** Assuming ETH, there is no $2^{o(\sqrt{k})} \cdot n^{O(1)}$ time algorithm on planar graphs for

- **Independent Set**
- **Dominating Set**
- **Vertex Cover**
- **$k$-Path**
- **Feedback Vertex Set**
- ...
A lower bound on Steiner Tree

**Steiner Tree**: Given a graph $G$ and set $T$ of terminals, find a tree of minimum size that contains $T$.

We have seen:
- Can be solved in time $3^{|T|} \cdot n^{O(1)}$ using dynamic programming.
- Can be solved in time $2^{|T|} \cdot n^{O(1)}$ using algebraic techniques.

Is there a subexponential FPT algorithm on planar graphs?
An exceptional lower bound

**Steiner Tree**: Given a graph $G$ and set $T$ of terminals, find a tree of minimum size that contains $T$.

We have seen:
- Can be solved in time $3^{|T|} \cdot n^{O(1)}$ using dynamic programming.
- Can be solved in time $2^{|T|} \cdot n^{O(1)}$ using algebraic techniques.

Is there a subexponential FPT algorithm on planar graphs?

**Theorem**

Assuming ETH, **Steiner Tree** on planar graphs with $k$ terminals cannot be solved in time $2^{o(k)} \cdot n^{O(1)}$. 
Treewidth — summary

- Notion of treewidth: widely used in graph theory and parameterized algorithms.
- Efficient algorithms parameterized by treewidth.
- Applications e.g. to planar graphs.
Tree decomposition: Vertices are arranged in a tree structure satisfying the following properties:

1. If \( u \) and \( v \) are neighbors, then there is a bag containing both of them.
2. For every \( v \), the bags containing \( v \) form a connected subtree.

Width of the decomposition: largest bag size \(-1\).

treewidth: width of the best decomposition.