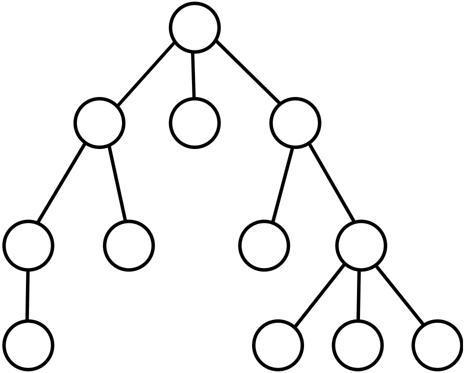
A Faster Parameterized Algorithm for Treedepth

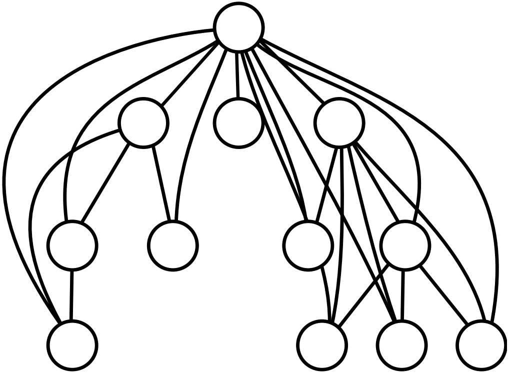
Felix Reidl, Peter Rossmanith, **Fernando Sánchez Villaamil**
Somnath Sikdar

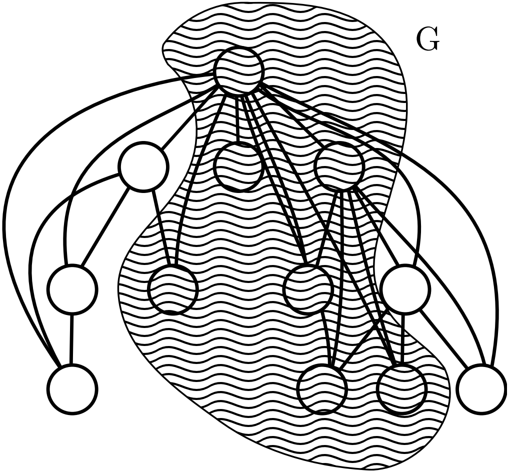
RWTH Aachen University

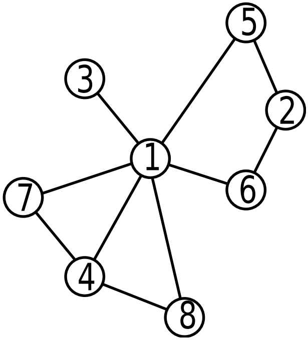
July 11, 2014

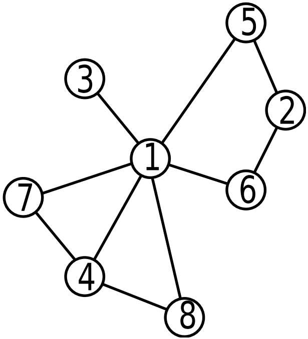
Treedepth is a width measure.

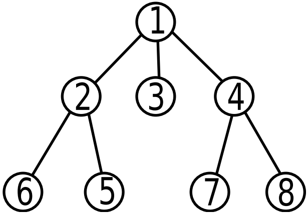


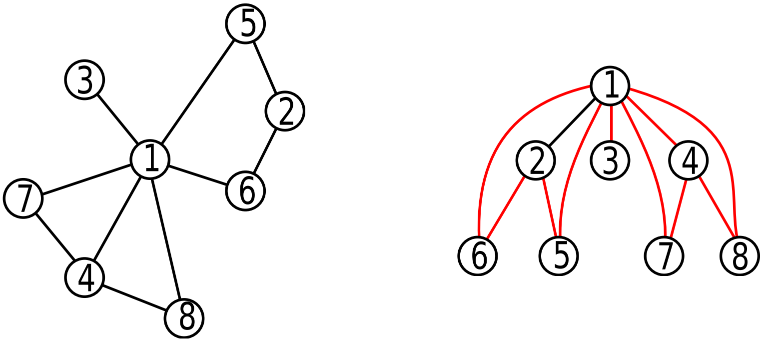



G


5
3
2
1
7
6
4
8


5
3
2
1
7
6
4
8


1
2
3
4
6
5
7
8


5
3
2
1
7
6
4
8
1
2
3
4
6
5
7
8

Definition (Treedepth decomposition)

A *treedepth decomposition* of a graph G is a rooted forest F such that $V(G) \subseteq V(F)$ and $E(G) \subseteq E(clos(F))$.

Definition (Treedepth)

The *treedepth* $\mathbf{td}(G)$ of a graph G is the minimum height of any treedepth decomposition of G.

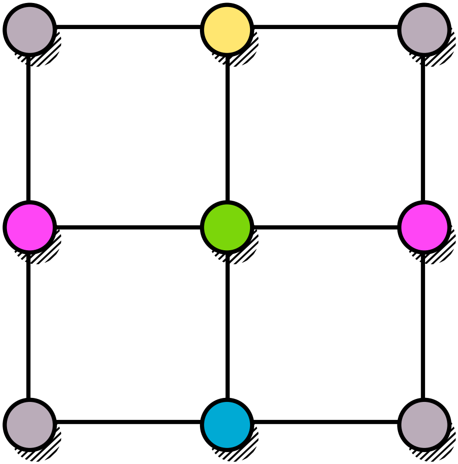
A strange width measure...

> "So many choices"
>
> —Dr. Dre

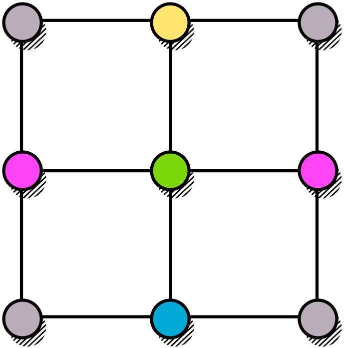
A graph G has *treedepth* at most t if

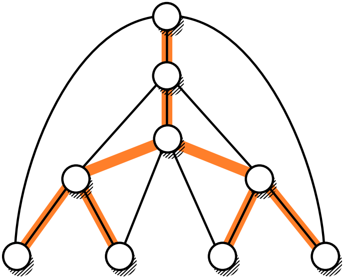
- G is a subgraph the closure of a tree (forest) of height $\leq t$
- G has a *centered coloring* with t colors
- G has a *ranked coloring* with t colors
- G is the subgraph of a *trivially perfect graph* with clique size $\leq t$

Centered Coloring

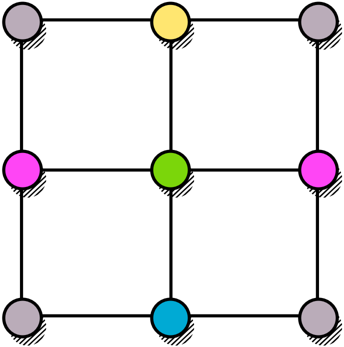


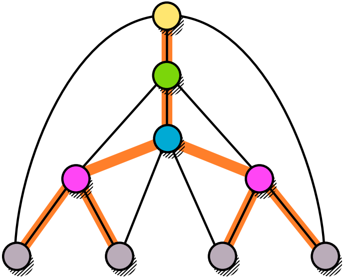
Centered Coloring



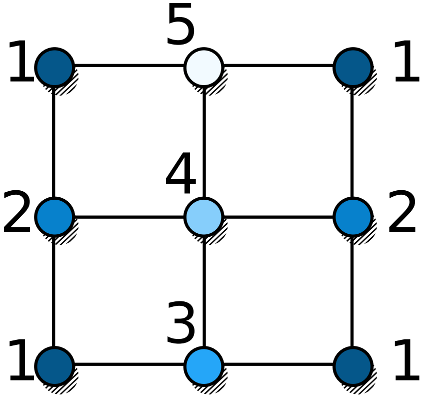


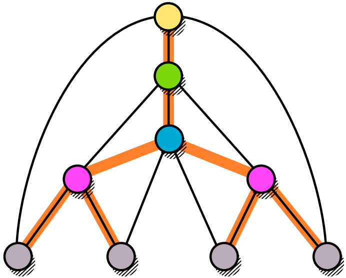
Centered Coloring



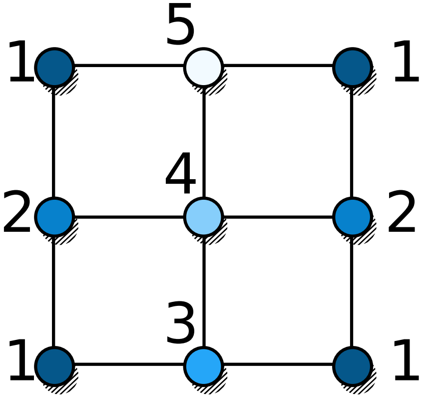


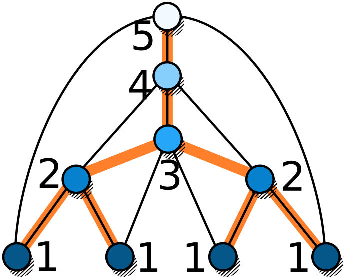
Ranked Coloring





Ranked Coloring



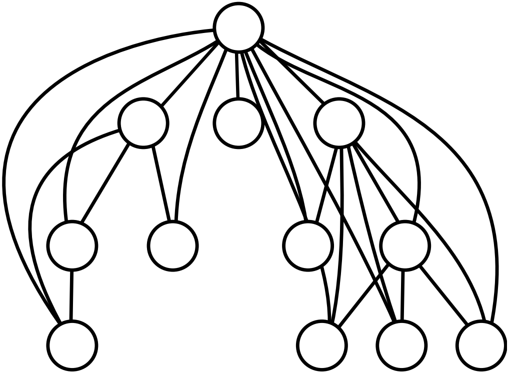


Trivially Perfect Graphs

G is the subgraph of a *trivially perfect graph* with clique size at most t.

Trivially Perfect Graphs

G is the subgraph of a *trivially perfect graph* with clique size at most t.



Arises again and again

Introduced as...

- *minimum elimination tree* by Pothen [1988]
- *ordered coloring* by Katchalski et al. [1995]
- *vertex ranking* by Bodlaender et al. [1998]
- **again** as treedepth by Nešetřil and Ossona de Mendez [2008]

Arises again and again

Introduced as...

- *minimum elimination tree* by Pothen [1988]
- *ordered coloring* by Katchalski et al. [1995]
- *vertex ranking* by Bodlaender et al. [1998]
- **again** as treedepth by Nešetřil and Ossona de Mendez [2008]

Related to...

- layouting of VLSI chips
- star height of regular languages
- characterizing bounded expansion graph classes
- counting subgraphs [New results coming]

Arises again and again

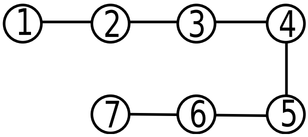
Introduced as...

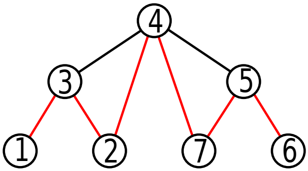
- *minimum elimination tree* by Pothen [1988]
- *ordered coloring* by Katchalski et al. [1995]
- *vertex ranking* by Bodlaender et al. [1998]
- **again** as treedepth by Nešetřil and Ossona de Mendez [2008]

Related to...

- layouting of VLSI chips
- star height of regular languages
- characterizing bounded expansion graph classes
- counting subgraphs [New results coming]

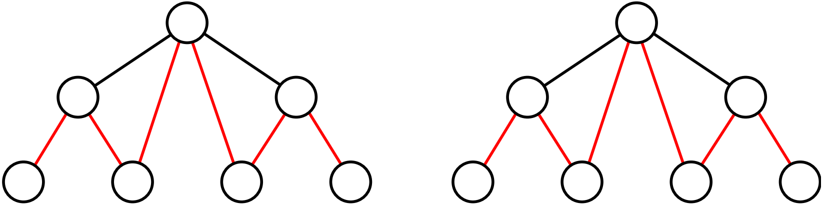
Personal opinion: Treedepth is the most useful definition.



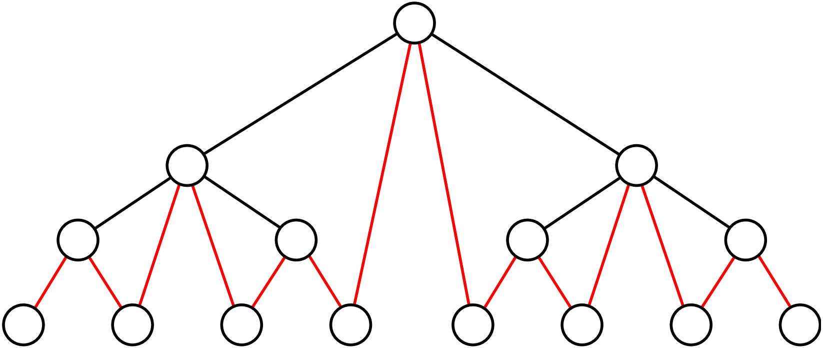


Treedepth $t \rightarrow$ Maximal path length $2^t - 1$.

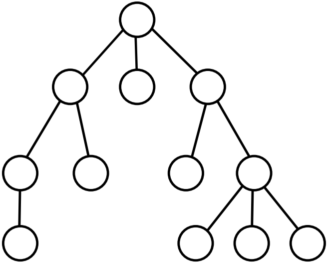
Treedepth $t \rightarrow$ Maximal path length $2^t - 1$.



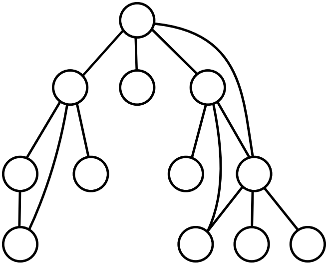
Treedepth $t \rightarrow$ Maximal path length $2^t - 1$.



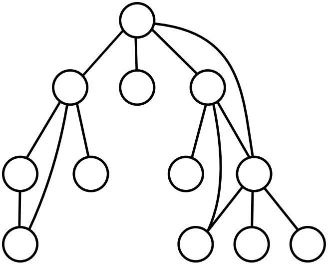
A DFS is a Treedepth decomposition



A DFS is a Treedepth decomposition

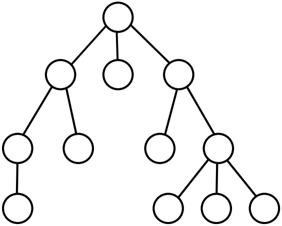


A DFS is a Treedepth decomposition

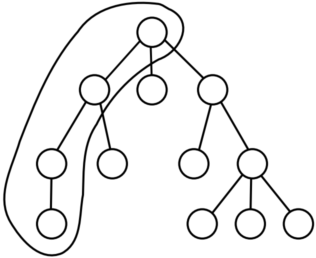


Treedepth $t \Rightarrow$ Maximal path length $2^t - 1 \Rightarrow 2^t$-approximation

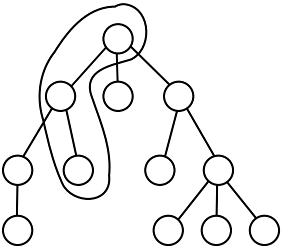
Treedepth to pathwidth



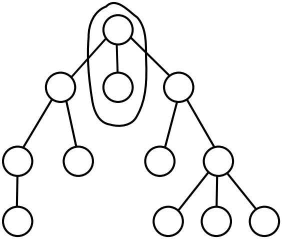
Treedepth to pathwidth



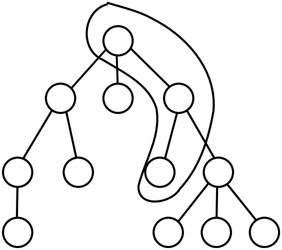
Treedepth to pathwidth



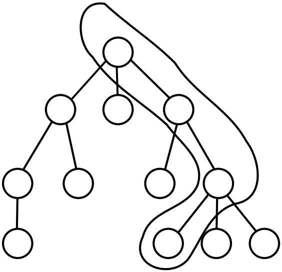
Treedepth to pathwidth



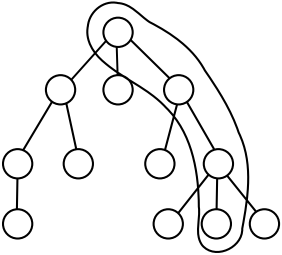
Treedepth to pathwidth



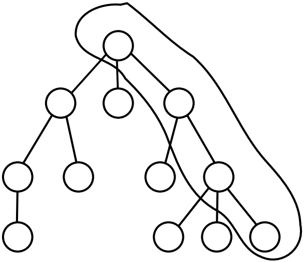
Treedepth to pathwidth



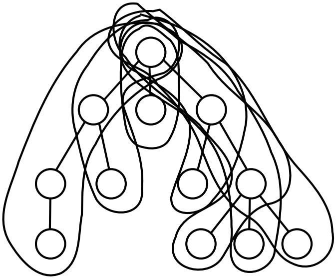
Treedepth to pathwidth



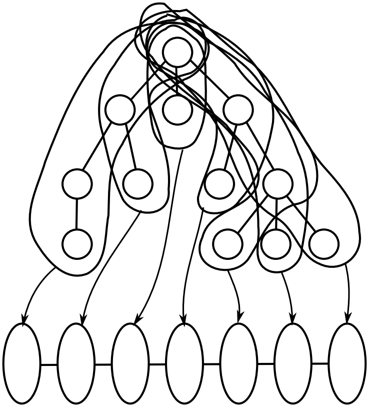
Treedepth to pathwidth



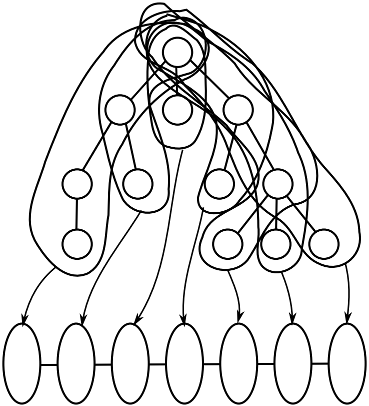
Treedepth to pathwidth



Treedepth to pathwidth

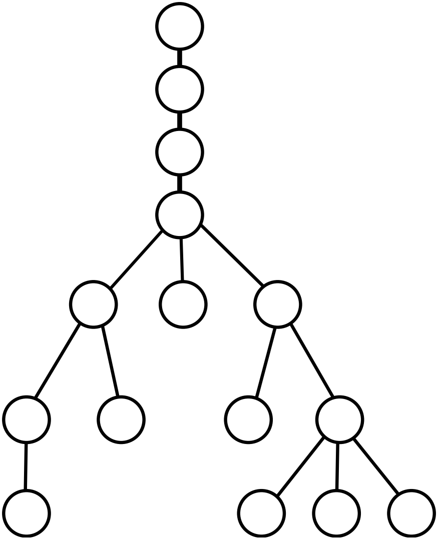


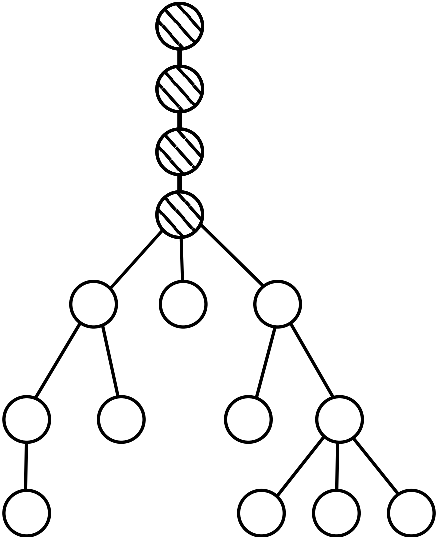
Treedepth to pathwidth

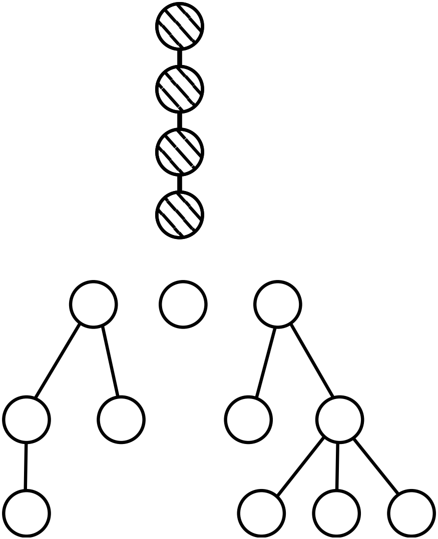


$$\mathbf{tw}(G) \leq \mathbf{pw}(G) \leq \mathbf{td}(G) - 1$$

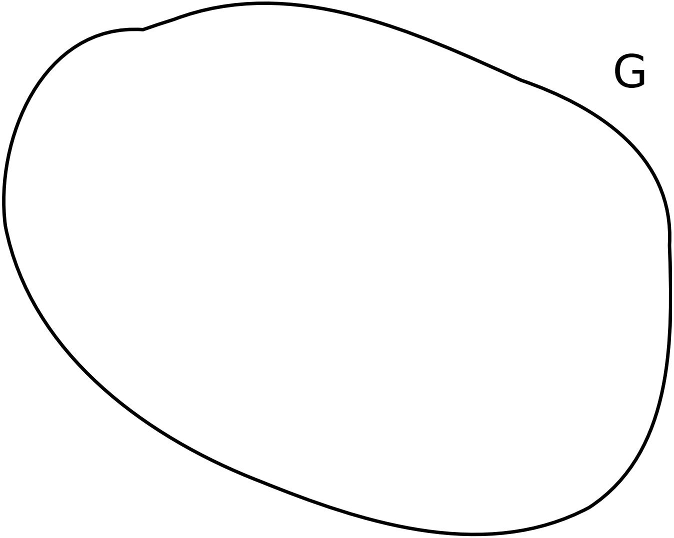
Treedepth $t \Rightarrow$ Path decomposition of width $2^t - 2$



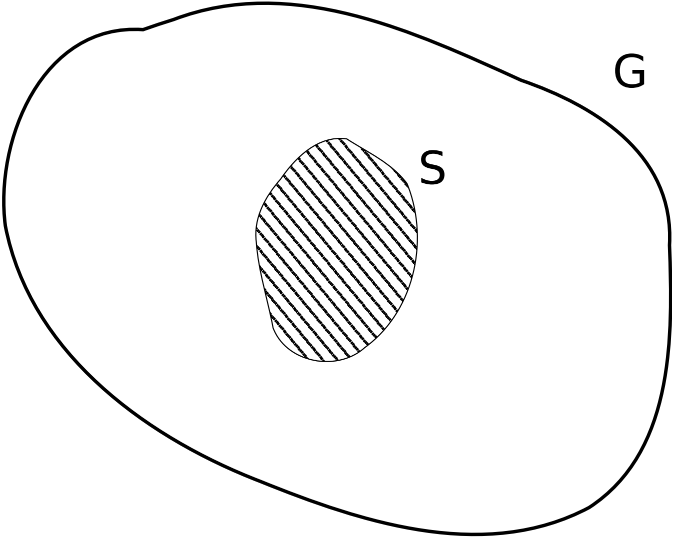




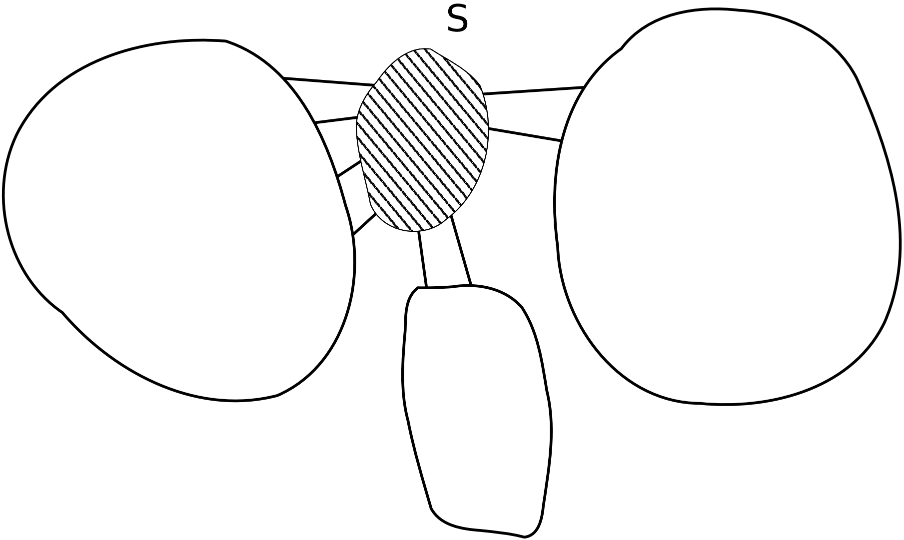
Treedepth by bruteforce



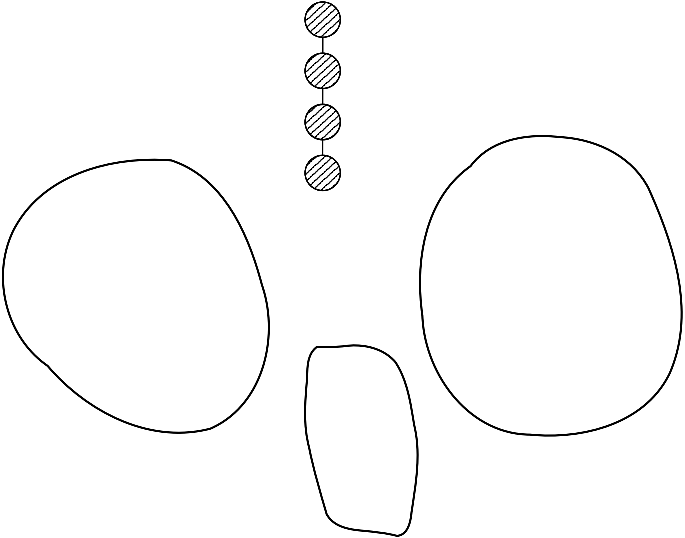
Treedepth by bruteforce



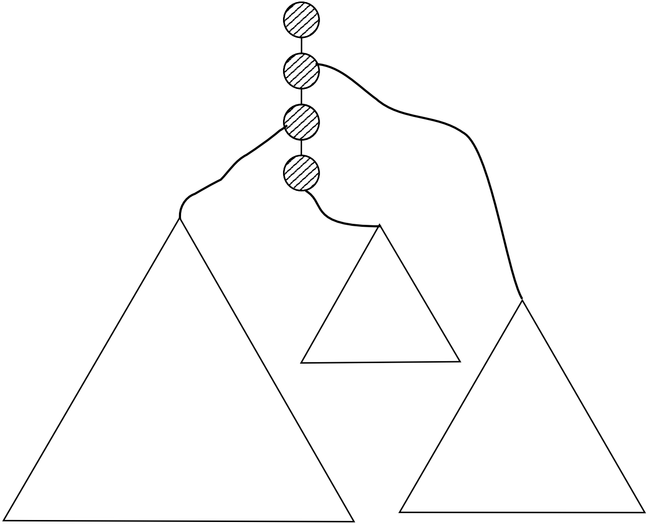
Treedepth by bruteforce



Treedepth by bruteforce



Treedepth by bruteforce



Parameterized algorithms

Open problem by Nešetřil and Ossona de Mendez [2012]

Is there a simple linear time algorithm to check $\mathbf{td}(G) \leq t$ for fixed t?

Parameterized algorithms

Open problem by Nešetřil and Ossona de Mendez [2012]

Is there a simple linear time algorithm to check $\mathbf{td}(G) \leq t$ for fixed t?

- In $f(t) \cdot n^3$ time by Robertson and Seymour.
- $\mathbf{tw}(G) \leq \mathbf{td}(G) - 1 \Rightarrow$ By Courcelle's Theorem $2^{2^{2^{\cdot^{\cdot^{\cdot^{t}}}}}} \cdot n$.
- Algorithm by Bodlaender et. al. with running time $2^{O(w^2 t)} \cdot n^2$.

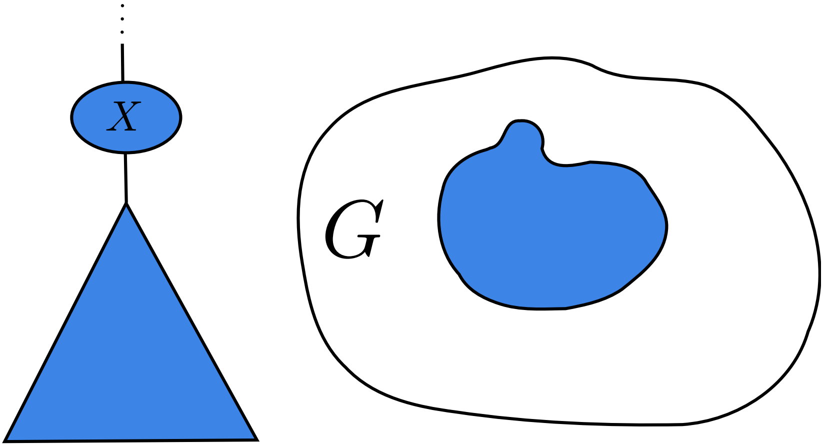
Our results:

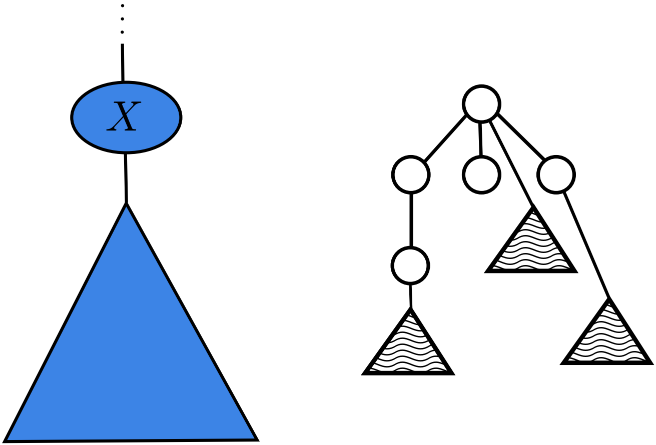
- A (relatively) simple direct algorithm in time $2^{2^{O(t)}} \cdot n$.
- A fast algorithm in time $2^{O(t^2)} \cdot n$.

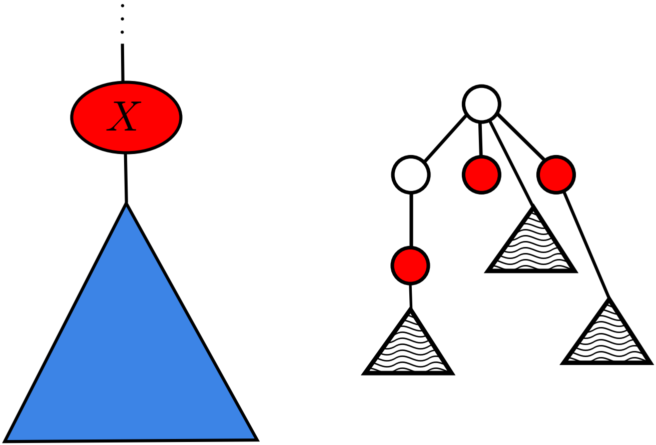
Our results:

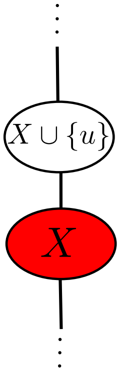
- A (relatively) simple direct algorithm in time $2^{2^{O(t)}} \cdot n$.
- A fast algorithm in time $2^{O(t^2)} \cdot n$.

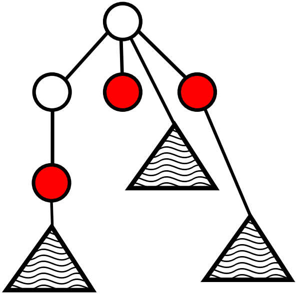
Both results follow from an algorithm on tree decompositions which runs in time $2^{O(wt)} \cdot n$.

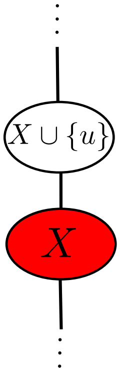

X
G

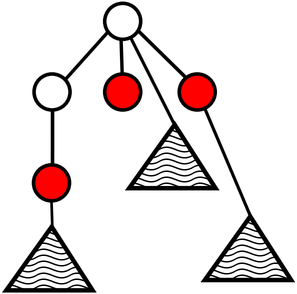

X


X

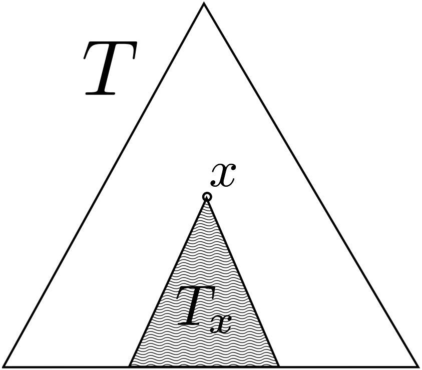

$X \cup \{u\}$
X

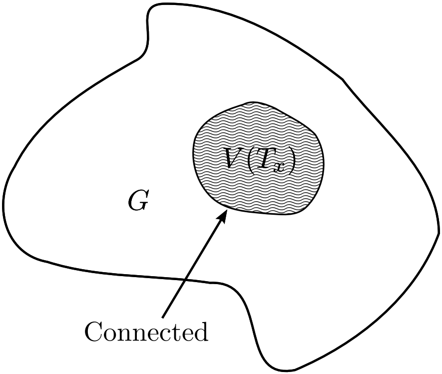


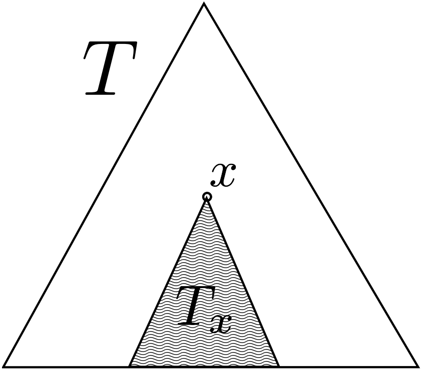


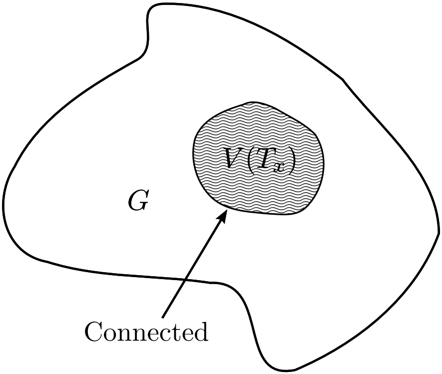


Where could the introduced node u be?


T
x
T_x

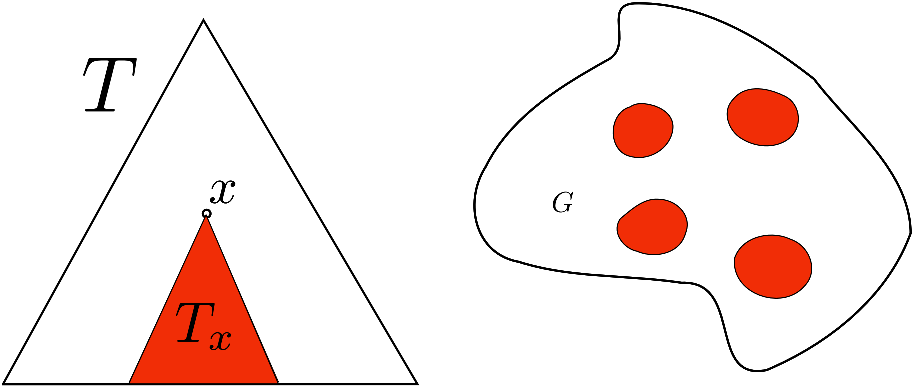

$V(T_x)$
G
Connected

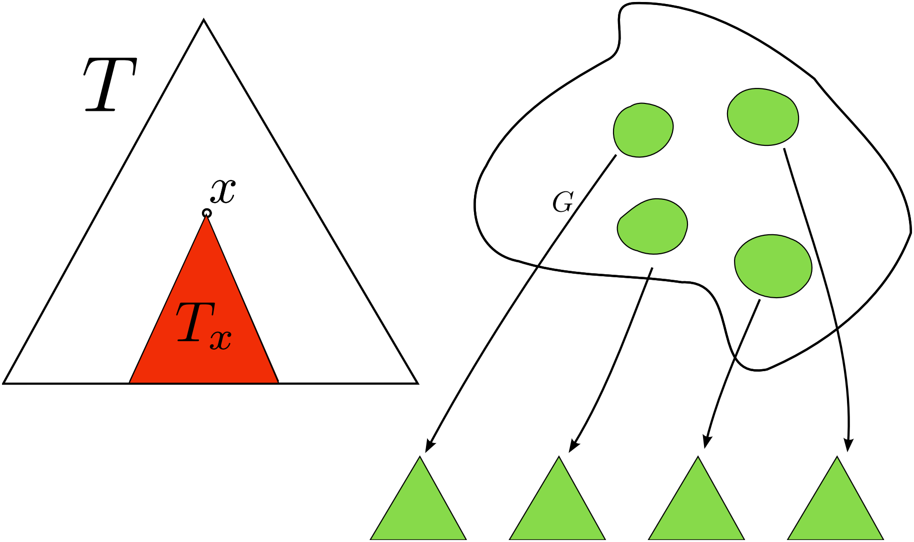


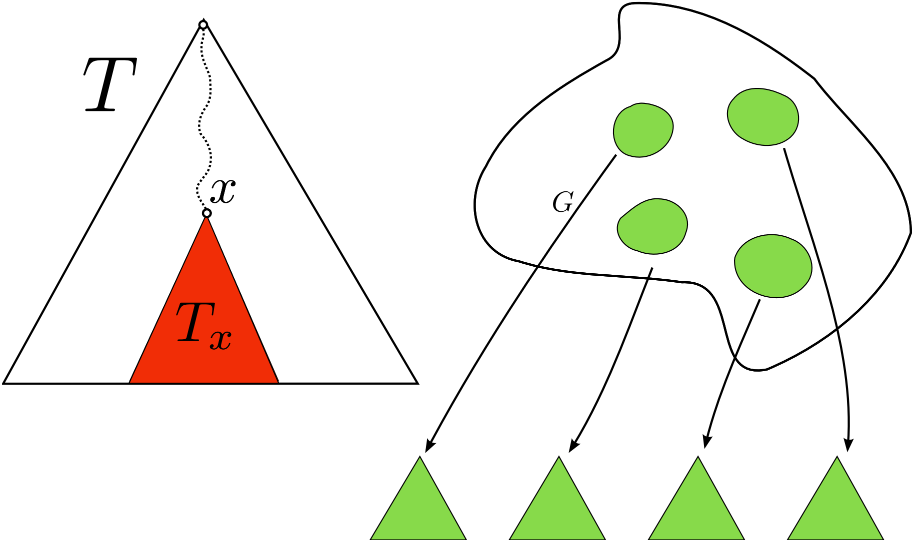


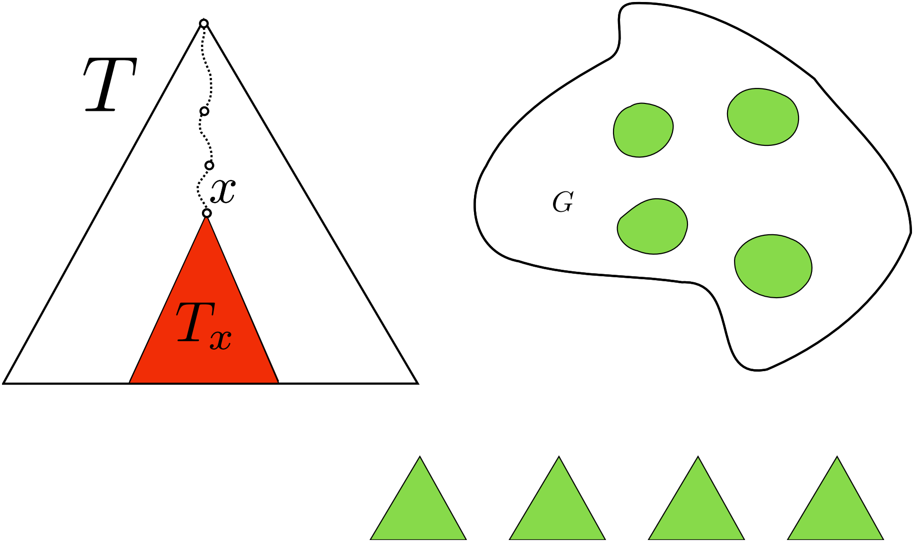
Definition (Nice treedepth decomposition)

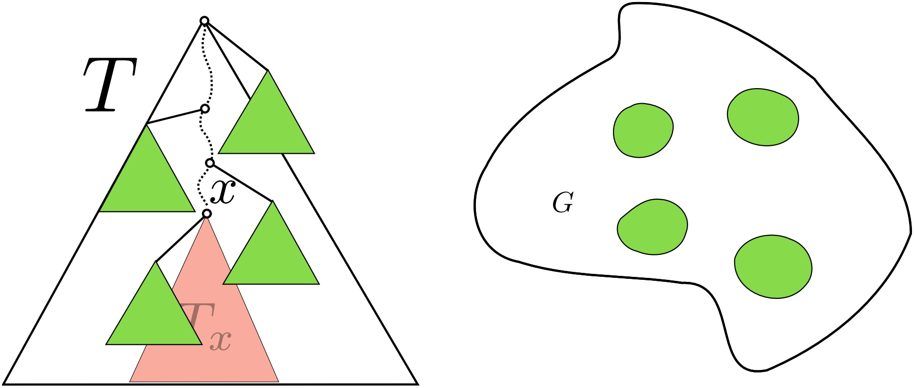
We say that T is *nice* if for every vertex $x \in V(T)$, the subgraph of G induced by the vertices in T_x is connected.


T
x
T_x
G


T
x
T_x
G

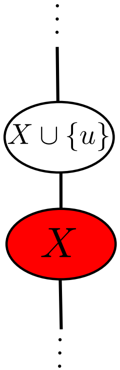

T
x
T_x
G

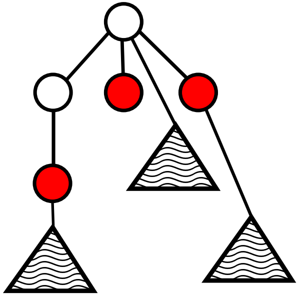

T
x
T_x
G


T
x
T_x
G

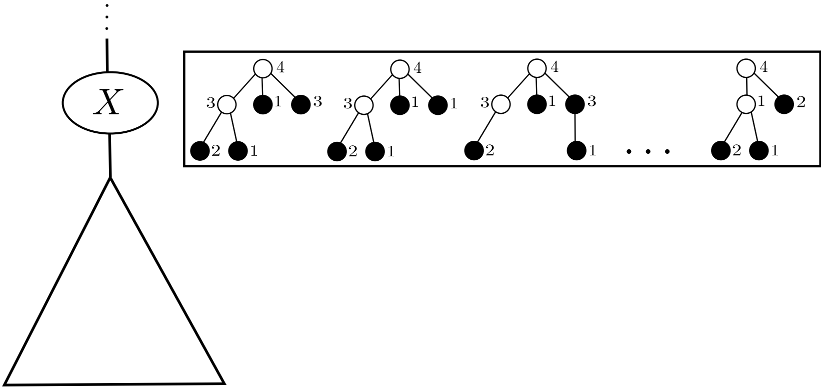
Lemma

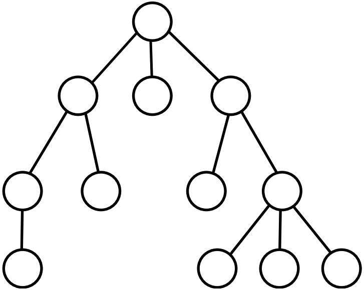
For any graph there exists a treedepth decomposition of minimal depth which is nice.

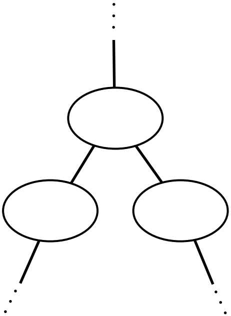


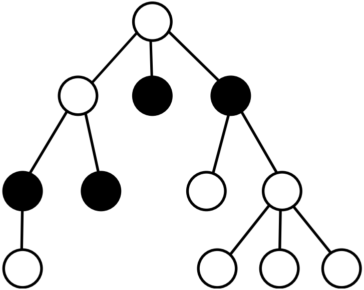


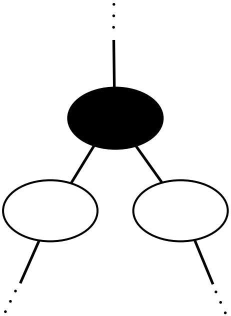
Where could the introduced node u be?

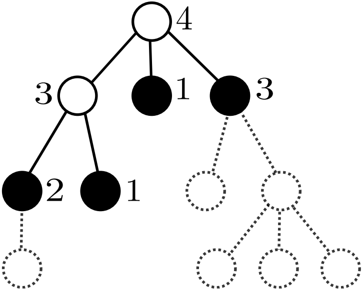

X
4
3
1
3
2
1
4
3
1
1
2
1
4
3
1
3
2
1
4
1
2
2
1

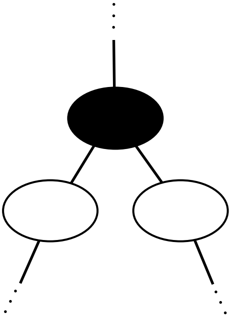


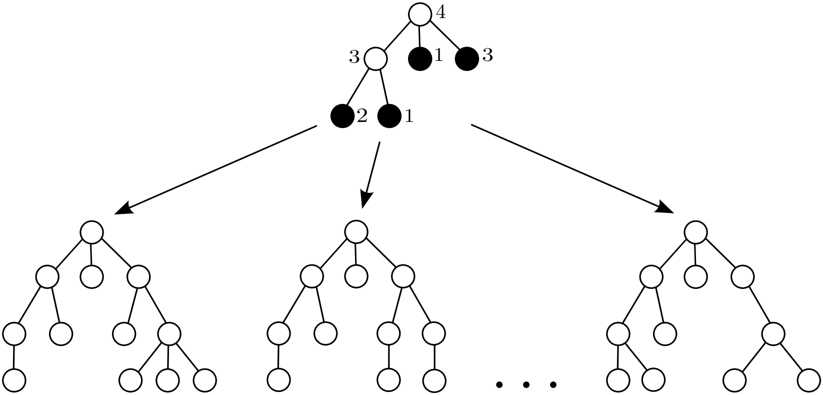







4
3
1
3
2
1




4
3
1
3
2
1

Theorem

Given a graph G with n nodes and a tree decomposition of G of width w the treedepth t of G can be decided in time $2^{O(wt)} \cdot n$.

Simple algorithm

1. Depth-first-search to construct treedepth decomposition T.
2. If depth greater than $2^t - 1$ say NO.
3. Construct path decomposition $\mathcal{P}$ from T of width 2^t.
4. Run algorithm on $\mathcal{P}$.

Theorem

There is a (simple) algorithm to decide if a graph G with n nodes has treedepth t which runs in time $2^{2^{O(t)}} \cdot n$.

Fast algorithm

1. Use single exponential 5-approximation for treewidth[1].
2. Remember $\mathbf{tw}(G) \leq \mathbf{pw}(G) \leq \mathbf{td}(G) - 1$.
3. If width is greater than $5t$ say NO.
4. Else run algorithm on tree decomposition.

Theorem

There is a algorithm to decide if a graph G with n nodes has treedepth t which runs in time $2^{O(t^2)} \cdot n$.

[1] Very useful result by Hans Bodlaender, Pål G. Drange, Markus S. Dregi, Fedor V. Fomin, Daniel Lokshtanov and Michał Pilipczuk

Thank you for listening.

Questions?