Approximate Turing Kernels

for Problems Parameterized by Treewidth

Astrid Pieterse

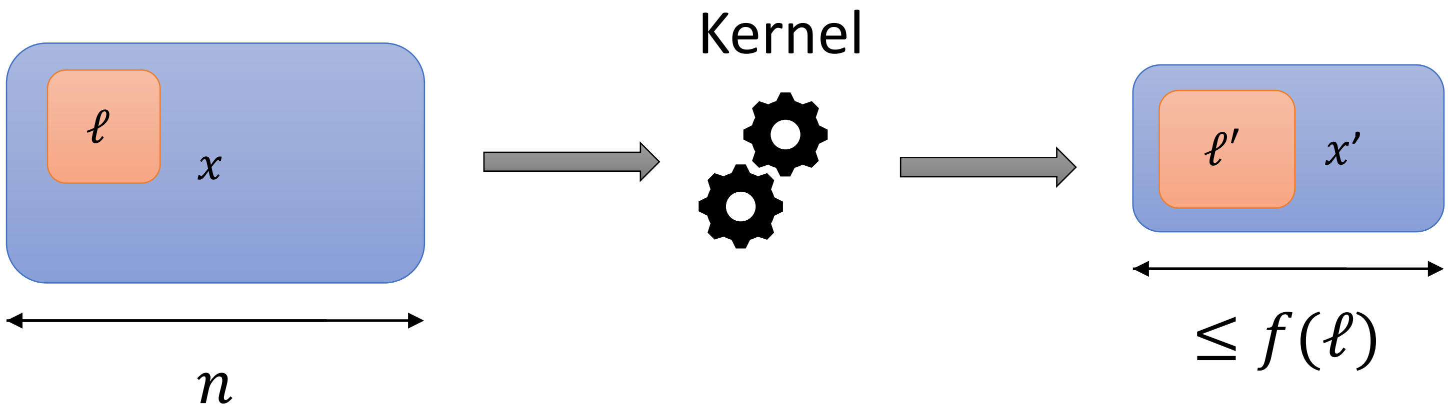
Based on joint work with

Eva-Maria C. Hols and Stefan Kratsch

https://arxiv.org/abs/2004.12683

Kernelization

Polynomial time preprocessing



Goal: obtain kernels that are small

- Every problem that is FPT has a kernel
- But only some problems have polynomial-size kernels
  - Under some complexity-theoretic assumptions

Beyond kernelization

Turing kernelization

- Allow creation of multiple instances

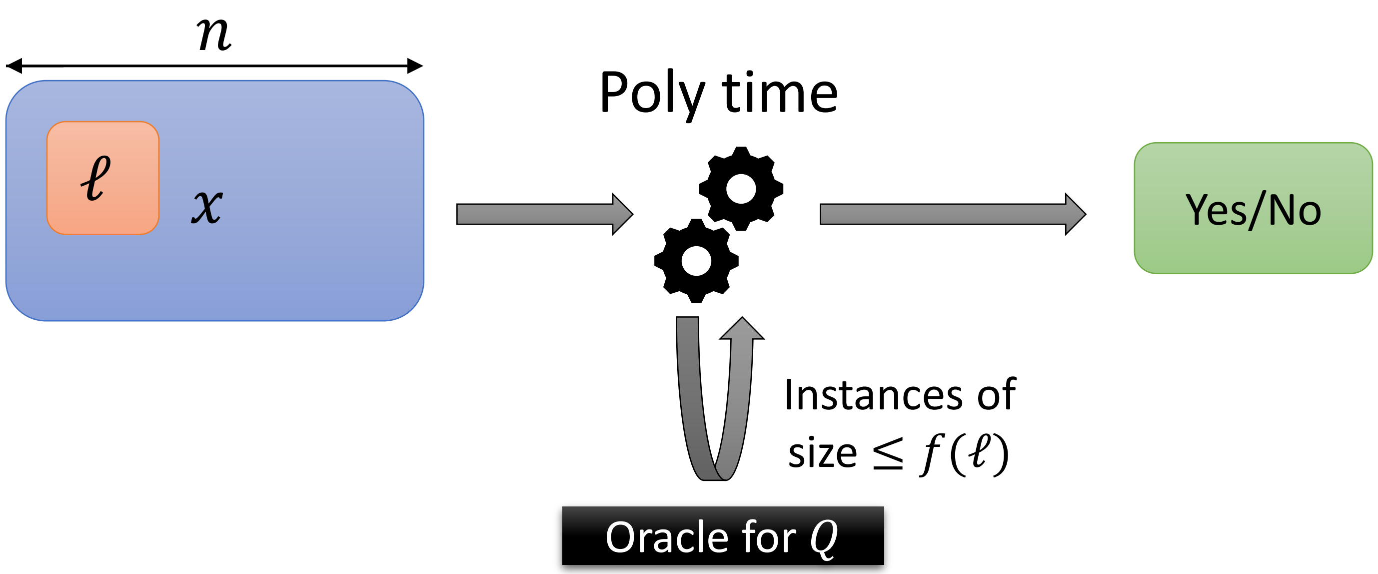
Approximate kernelization

- Relax the equivalence constraint

This talk: Approximate Turing Kernelization

Turing Kernelization

A Turing Kernel of size f for a problem Q is an algorithm that solves a given instance (x, ℓ) in time polynomial in $|x| + \ell$, when given access to an oracle that decides membership of Q for any instance with size at most $f(\ell)$ in a single step.



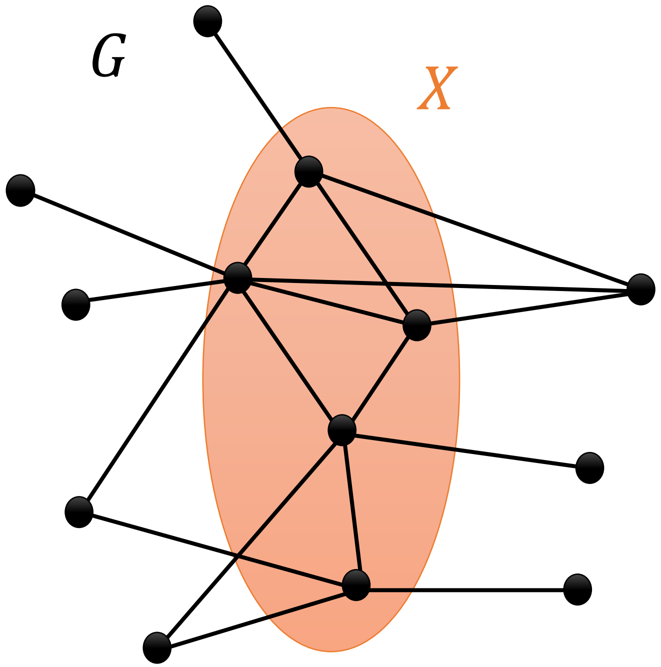
Turing Kernelization: Example

CLIQUE parameterized by vertex cover

Input Graph G, with vertex cover X of size ℓ, integer k

Question Does G have a clique of size k?

Parameter ℓ



No polynomial kernel [Bodlaender, Jansen, Kratsch 2012]

- Simple Turing kernel with $\ell + 1$ vertices

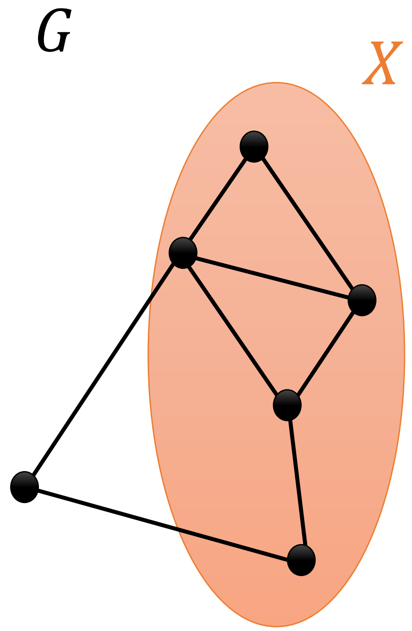
Turing Kernelization: Example

Clique parameterized by vertex cover

Input Graph G, with vertex cover X of size ℓ, integer k

Question Does G have a clique of size k?

Parameter ℓ



No polynomial kernel [Bodlaender, Jansen, Kratsch 2012]

- Simple Turing kernel with $\ell + 1$ vertices

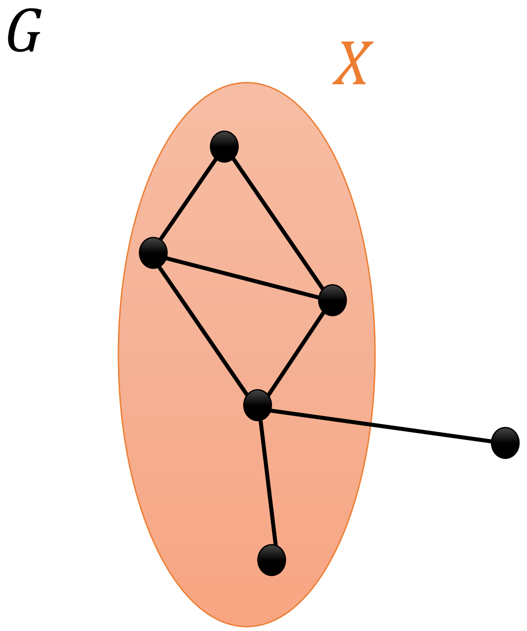
Turing Kernelization: Example

CLIQUE parameterized by vertex cover

Input Graph G, with vertex cover X of size ℓ, integer k

Question Does G have a clique of size k?

Parameter ℓ



No polynomial kernel [Bodlaender, Jansen, Kratsch 2012]

- Simple Turing kernel with $\ell + 1$ vertices

Turing Kernelization: Example

CLIQUE parameterized by vertex cover

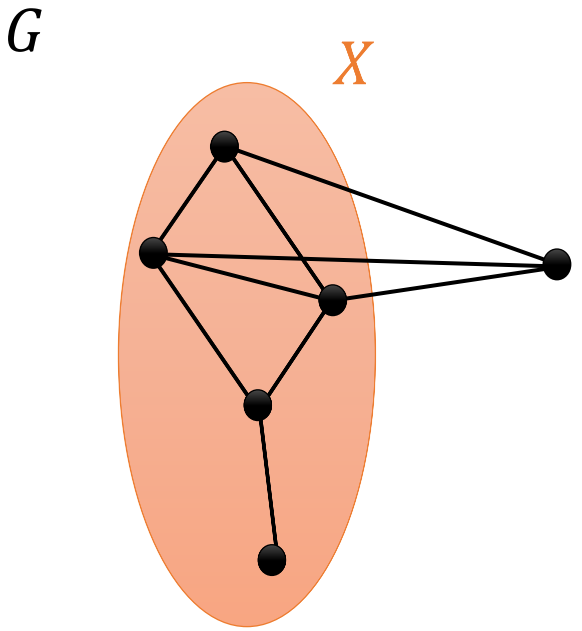
Input Graph G, with vertex cover X of size ℓ, integer k

Question Does G have a clique of size k?

Parameter ℓ

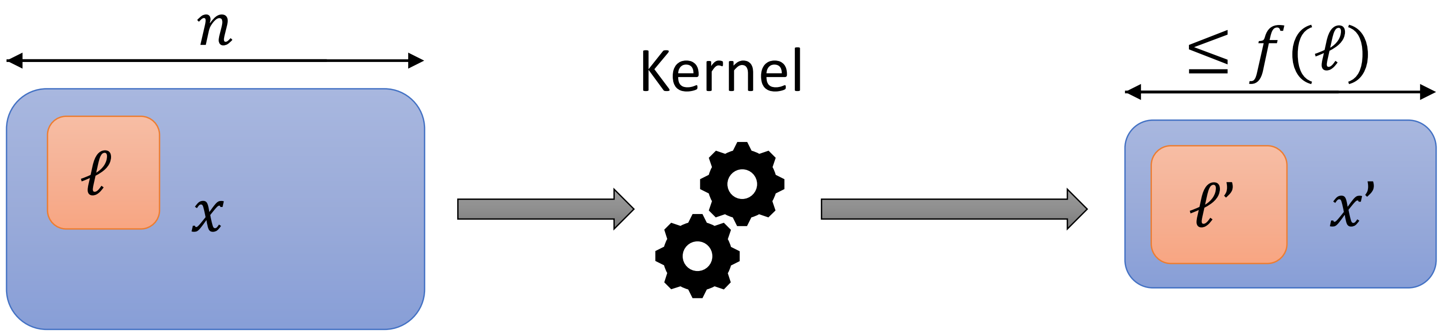
No polynomial kernel [Bodlaender, Jansen, Kratsch 2012]

- Simple Turing kernel with $\ell + 1$ vertices



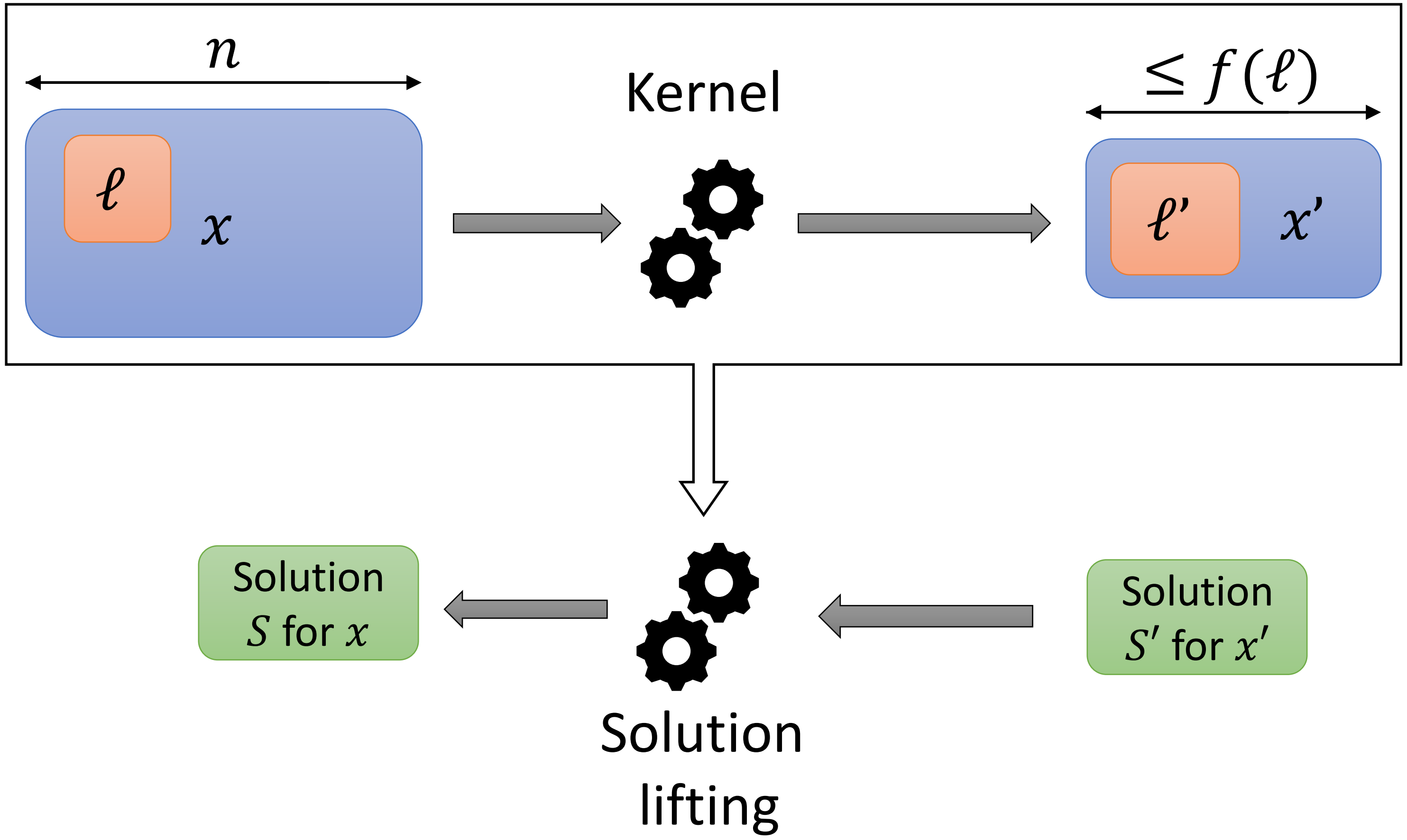
Towards approximate kernelization

Move from decision problems to optimization problems



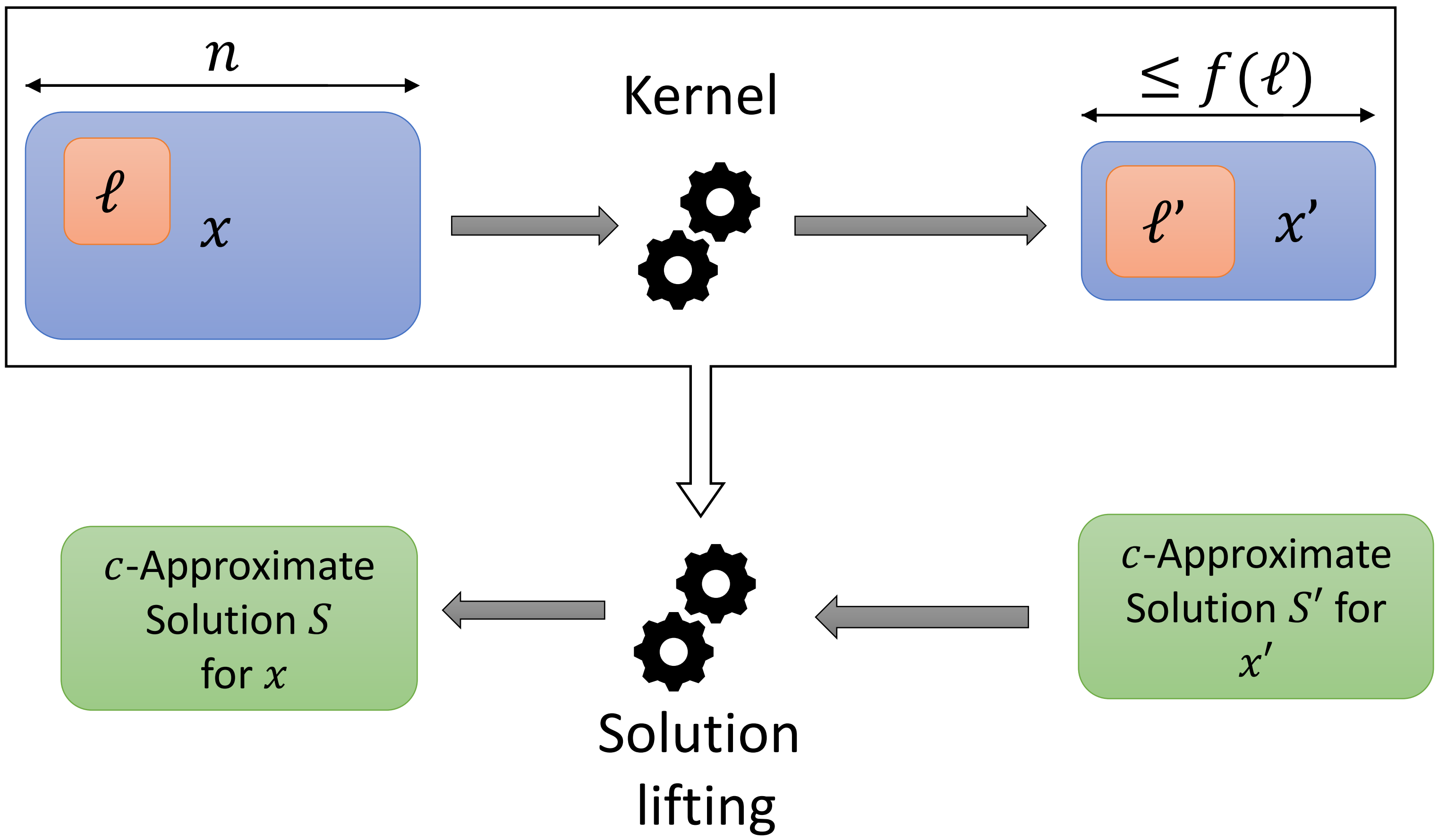
Towards approximate kernelization

Move from decision problems to optimization problems



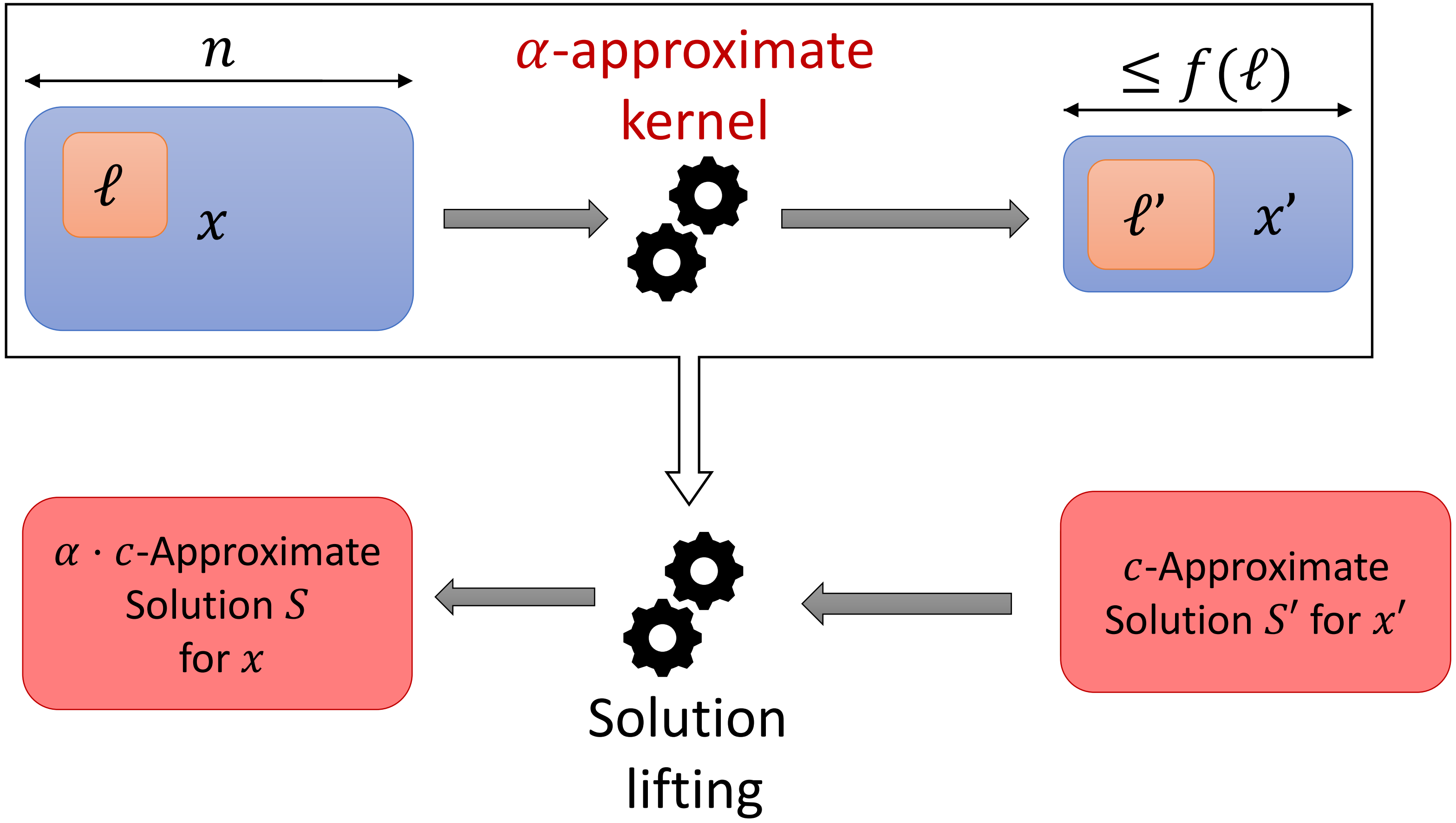
Towards approximate kernelization

Move from decision problems to optimization problems



Towards approximate kernelization

Move from decision problems to optimization problems



Approximate kernelization

Parameterized optimization problem Q

- Instances are pairs (x, ℓ), solutions are strings. A problem is a function Q, where $Q(x, \ell, s)$ is the value of solution s
- **Goal** find $\mathrm{OPT}_Q(x, \ell) = \min\{Q(x, \ell, s)\}$ for minimization problems

Subtlety

- If the parameter is also the optimized value, so $\ell = k$

  Vertex Cover by solution size: $Q(x, k, s) = \begin{cases} \infty & \text{if } s \text{ is not a vertex cover} \\ \min(|s|, k+1) & \text{otherwise} \end{cases}$

α-approximate kernel

Solution lifting algorithm satisfies $\frac{Q(x,\ell,s)}{OPT_Q(x,\ell)} \leq \alpha \frac{Q(x',\ell',s')}{OPT_Q(x',\ell')}$ for minimization problem

Approximate kernelization

Parameterized optimization problem Q

- Instances are pairs (x, ℓ), solutions are strings. A problem is a function Q, where $Q(x, \ell, s)$ is the value of solution s
- **Goal** find $\mathrm{OPT}_Q(x, \ell) = \min\{Q(x, \ell, s)\}$ for minimization problems

$\max\{..\}$

Always minimum, also for maximization problems

Subtlety

- If the parameter is also the optimized valu… o $\ell = k$
  - …x Cover by solution size: $Q(x, k, s) = \begin{cases} \infty & \text{if } s \text{ is not a vertex cover} \\ \min(|s|, k+1) & \text{otherwise} \end{cases}$

$-\infty$

Independent set

α-approximate kernel

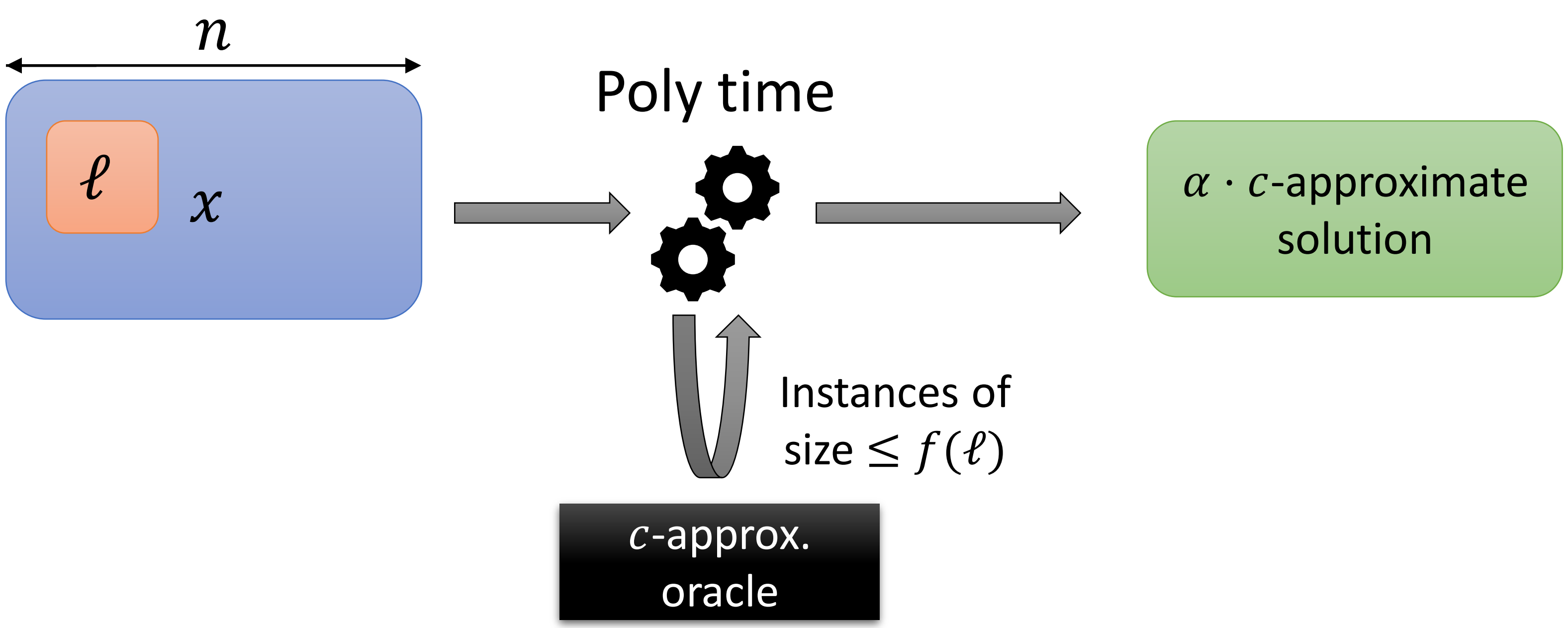
Solution lifting algorithm satisfies $\frac{Q(x,\ell,s)}{OPT_Q(x,\ell)} \leq \alpha \frac{Q(x',\ell',s')}{OPT_Q(x',\ell')}$ for minimization problem

$$\frac{Q(x, \ell, s)}{OPT_Q(x, \ell)} \geq \frac{1}{\alpha} \cdot \frac{Q(x', \ell', s')}{OPT_Q(x', \ell)}$$

Approximate Turing Kernelization

α-approximate Turing Kernel

- Turing kernel, but
  - The oracle is c-approximate for some (unknown) c
  - The output must be guaranteed to be $\alpha \cdot c$-approximate



Approximate Turing Kernels, when?

When is it possible to aim for a α-approximate Turing kernel

- The problem is α-FPT-approximable
- -approximable in polynomial time

It is only useful, when

- The best-known Turing kernel is too large
  - Ideally, evidence that no polynomial Turing kernel exists
- The best-known α-approximate kernel is also large
  - Ideally, proof of nonexistence, but this seems much harder to come by

Approximate Turing Kernels, when?

When is it possible to aim for a α-approximate Turing kernel

- The problem is α-FPT-approximable
- -approximable in polynomial time

Theorem

If a decidable problem has an α-approximate Turing kernel, it has an α-approximation algorithm that runs in FPT time.

Proof

Simply run the α-approximate Turing kernel, replacing oracle calls by calls to any algorithm solving the problem. Running time is bounded by

f(size of TK)·running time of approxTK $= f(\ell) \cdot \text{poly}(n)$

Approximate Turing Kernels, when?

When is it possible to aim for a α-approximate Turing kernel

- The problem is α-FPT-approximable
- But not α-approximable in polynomial time

It is only useful, when

- The best-known Turing kernel is too large
  - Ideally, evidence that no polynomial Turing kernel exists
- The best-known α-approximate kernel is also large
  - Ideally, proof of nonexistence, but this seems much harder to come by

Our results

Problem	#Vertices in kernel
Independent Set	$O\left(\frac{\ell^2}{\varepsilon}\right)$
Vertex Cover	$O\left(\frac{\ell}{\varepsilon}\right)$
Connected Vertex Cover	$O\left(\left(\frac{\ell^2}{\varepsilon}\right)^{\left\lceil\frac{3+\varepsilon}{\varepsilon}\right\rceil}\right)$
Edge Clique Cover	$O\left(\frac{\ell^4}{\varepsilon}\right)$
Edge-Disjoint Triangle Packing	$O\left(\frac{\ell^2}{\varepsilon}\right)$
Vertex-Disjoint H-Packing for connected H	$O\left(\left(\frac{\ell}{\varepsilon}\right)^{\lvert V(H)\rvert-1}\right)$
Clique Cover	$O\left(\frac{\ell^4}{\varepsilon^2}\right)$
Feedback vertex Set	$O\left(\frac{\ell^2}{\varepsilon^2}\right)$
Edge Dominating Set	$O\left(\frac{\ell^2}{\varepsilon^2}\right)$

These problems parameterized by treewidth ℓ have $(1+\varepsilon)$-approximate Turing Kernels

- Assuming tree decomposition on input
- For all $0 < \varepsilon \leq 1$

Plus a general statement concerning "sufficiently friendly" problems

Our results

Problem	#Vertices in kernel
Independent Set	$O\left(\frac{\ell^2}{\varepsilon}\right)$
Vertex Cover	$O\left(\frac{\ell}{\varepsilon}\right)$
Connected Vertex Cover	$O\left(\left(\frac{\ell^2}{\varepsilon}\right)^{\left\lceil\frac{3+\varepsilon}{\varepsilon}\right\rceil}\right)$
Edge Clique Cover	$O\left(\frac{\ell^4}{\varepsilon}\right)$
Edge-Disjoint Triangle Packing	$O\left(\frac{\ell^2}{\varepsilon}\right)$
Vertex-Disjoint H-Packing for connected H	$O\left(\left(\frac{\ell}{\varepsilon}\right)^{\lvert V(H)\rvert-1}\right)$
Clique Cover	$O\left(\frac{\ell^4}{\varepsilon^2}\right)$
Feedback vertex Set	$O\left(\frac{\ell^2}{\varepsilon^2}\right)$
Edge Dominating Set	$O\left(\frac{\ell^2}{\varepsilon^2}\right)$

These problems parameterized by treewidth ℓ have $(1+\varepsilon)$-approximate Turing Kernels

- Assuming tree decomposition on input
- For all $0 < \varepsilon \leq 1$

Plus a general statement concerning "sufficiently friendly" problems

Approximate Turing kernel checklist

Considered problems are FPT (and hence, FPT-approximable)

Polynomial kernels rare, parameterized by treewidth

- Vertex Cover and Independent Set are $MK[2]$ hard

- No good approximate kernels known
  - Explicitly asked open question [Lokshtanov, Panolan, Ramanujan, Saurabh STOC 2017]

Approximate Turing kernel checklist

Considered problems are FPT (and hence, FPT-approximable) ✓

Polynomial kernels rare, parameterized by treewidth

- Vertex Cover and Independent Set are $MK[2]$ hard

- No good approximate kernels known
  - Explicitly asked open question [Lokshtanov, Panolan, Ramanujan, Saurabh STOC 2017]

Approximate Turing kernel checklist

Considered problems are FPT (and hence, FPT-approximable) ✓

Polynomial kernels rare, parameterized by treewidth ✓

- VERTEX COVER and INDEPENDENT SET are $MK[2]$ hard

- No good approximate kernels known
  - Explicitly asked open question [Lokshtanov, Panolan, Ramanujan, Saurabh STOC 2017]

Approximate Turing kernel checklist

Considered problems are FPT (and hence, FPT-approximable) ✓

Polynomial kernels rare, parameterized by treewidth

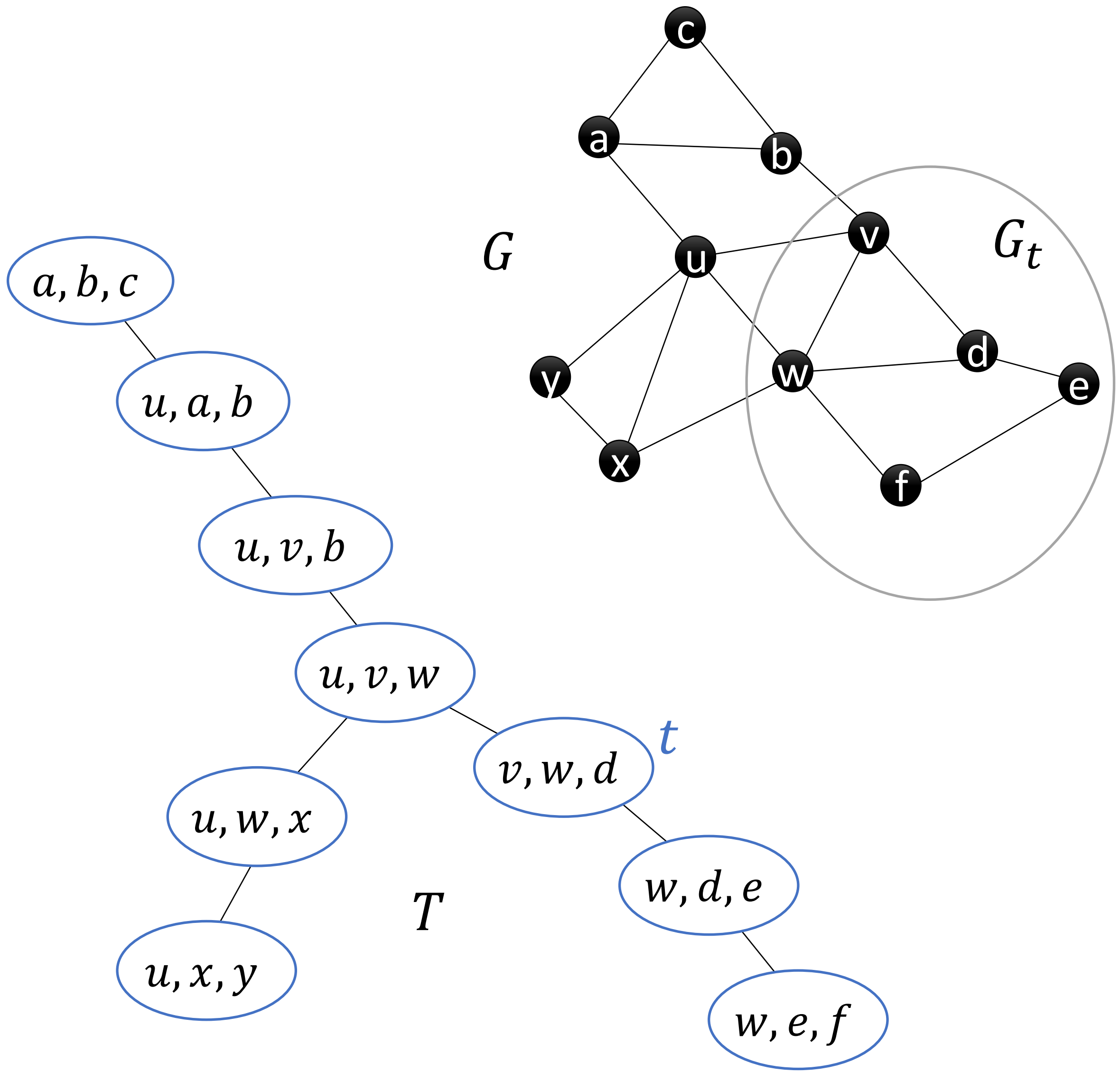
- VERTEX COVER and INDEPENDENT SET are $MK[2]$ hard ✓

- No good approximate kernels known ?
  - Explicitly asked open question [Lokshtanov, Panolan, Ramanujan, Saurabh STOC 2017]

Treewidth

Tree decomposition of G

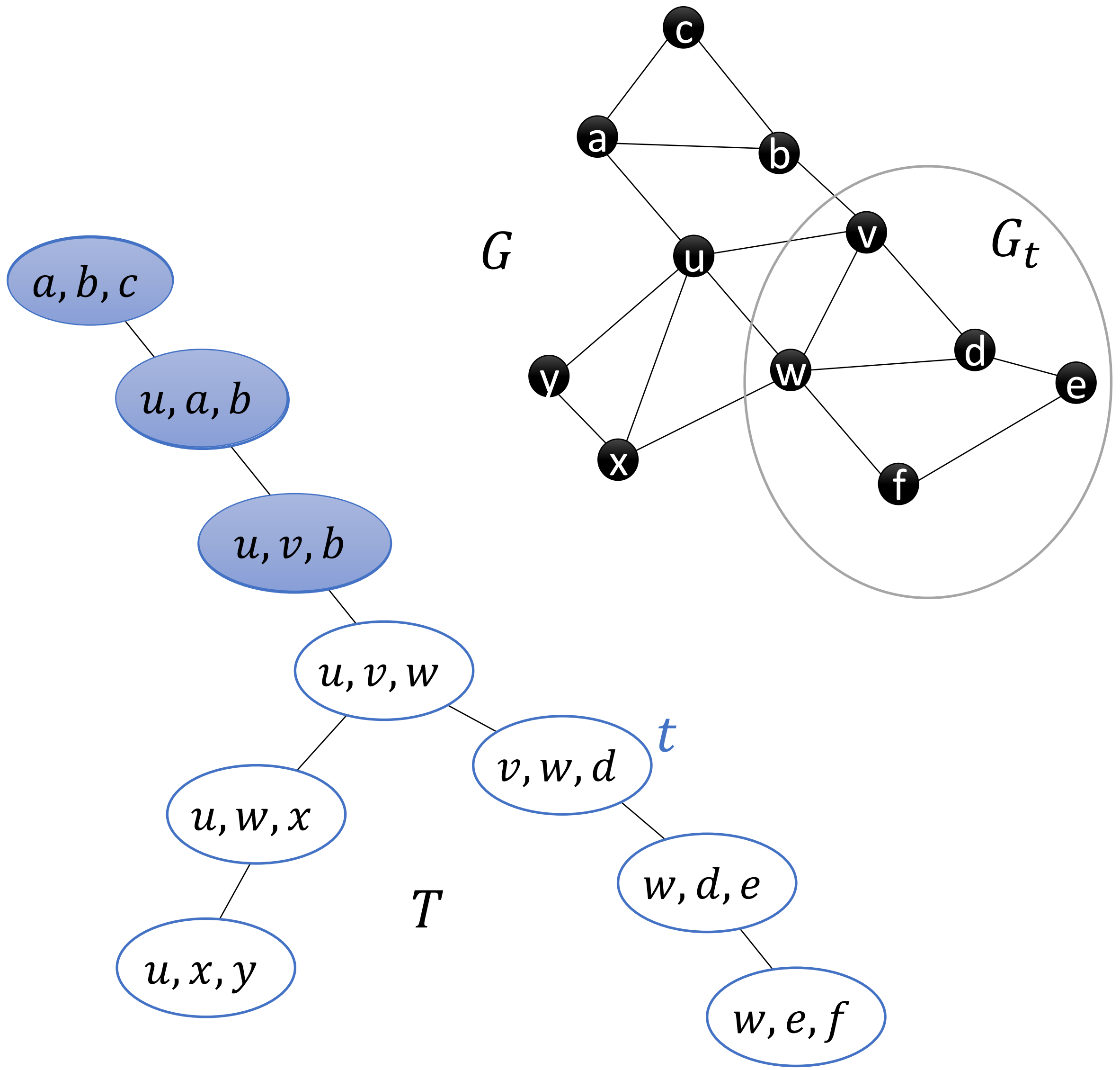
- Tree T with nodes each node t has bag $X_t \subseteq V(G)$
  - For each edge uv in G, exists bag such that $u \in X_t, v \in X_t$
  - For each $u \in V(G)$, bags in which u occurs form connected subgraph of T
  - Each $u \in V(G)$ occurs in at least one bag
- Width: size largest bag – 1



Treewidth

Tree decomposition of G

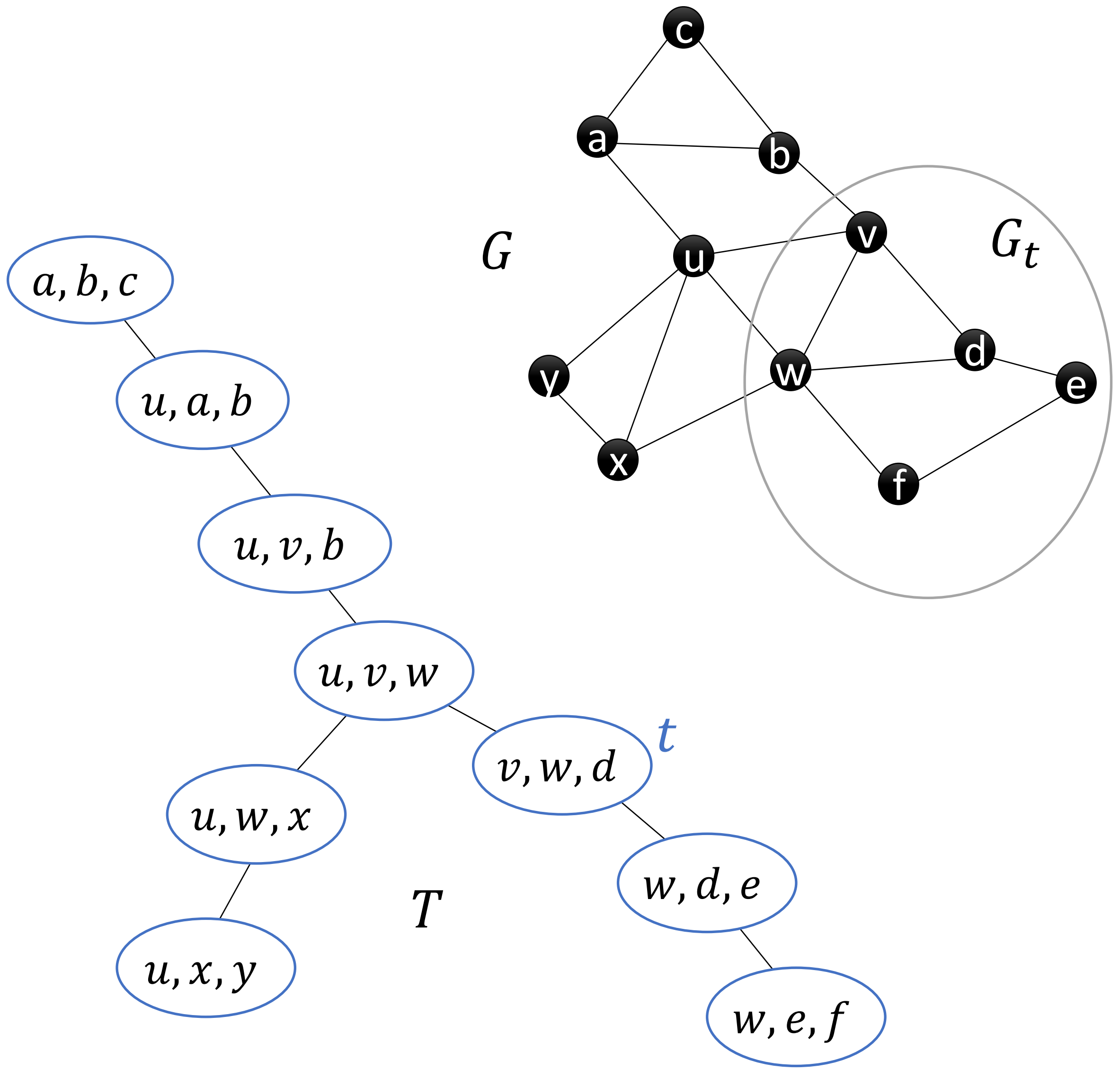
- Tree T with nodes each node t has bag $X_t \subseteq V(G)$
  - For each edge uv in G, exists bag such that $u \in X_t, v \in X_t$
  - For each $u \in V(G)$, bags in which u occurs form connected subgraph of T
  - Each $u \in V(G)$ occurs in at least one bag
- Width: size largest bag – 1



Treewidth

Tree decomposition of G

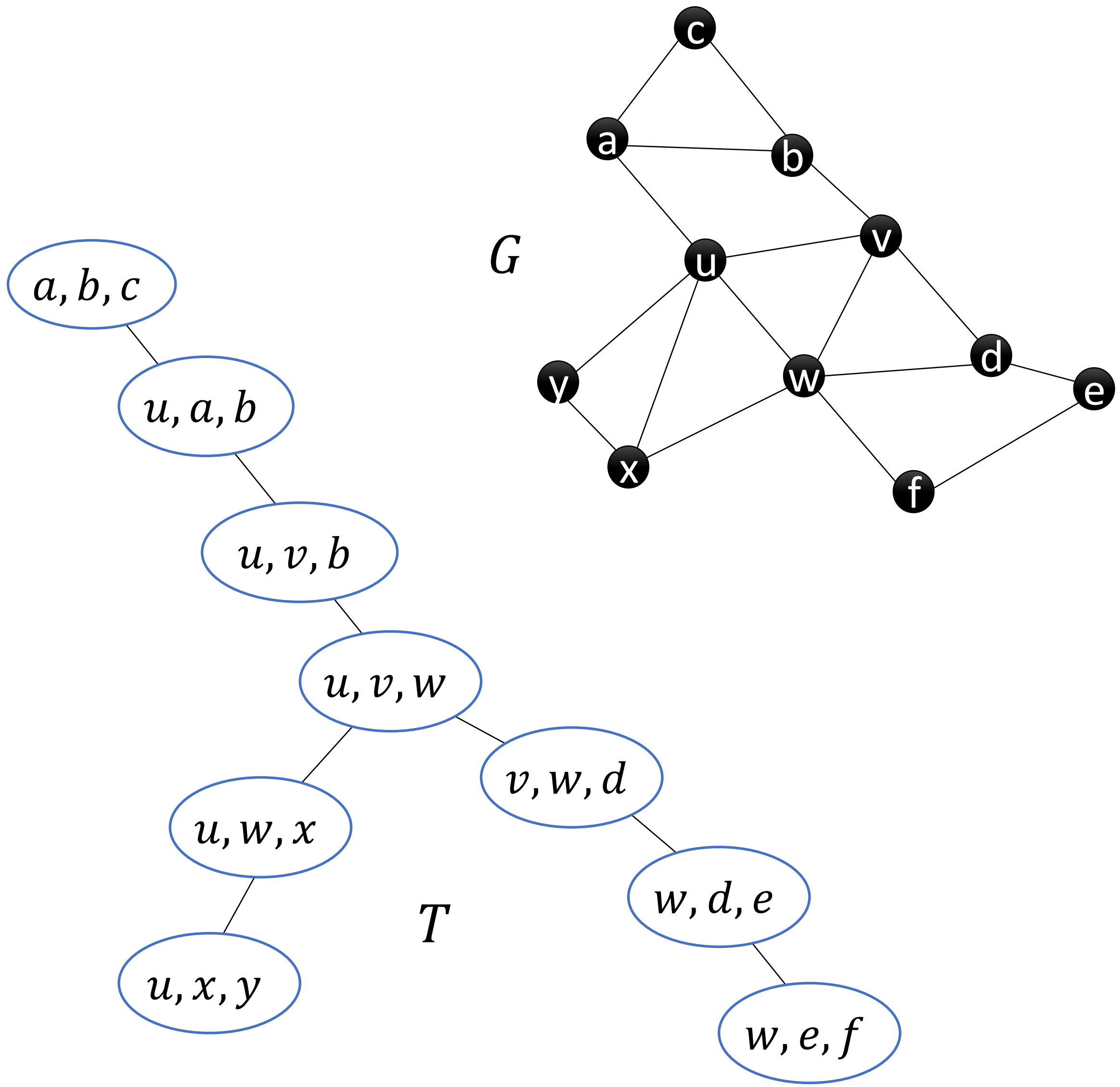
- Tree T with nodes each node t has bag $X_t \subseteq V(G)$
  - For each edge uv in G, exists bag such that $u \in X_t, v \in X_t$
  - For each $u \in V(G)$, bags in which u occurs form connected subgraph of T
  - Each $u \in V(G)$ occurs in at least one bag
- Width: size largest bag – 1



Treewidth

Tree decomposition of G

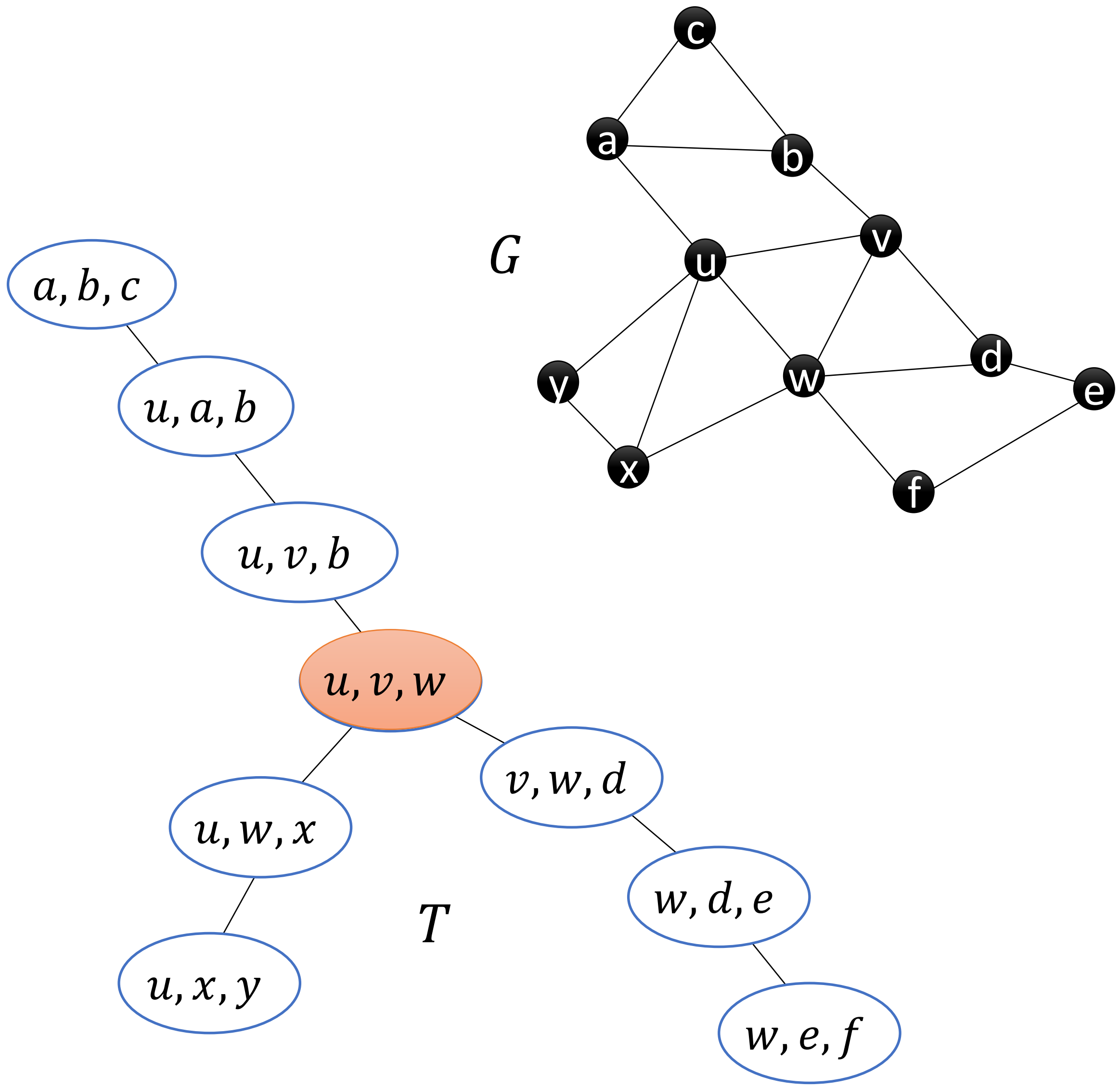
- Tree T with nodes each node t has bag $X_t \subseteq V(G)$
  - For each edge uv in G, exists bag such that $u \in X_t, v \in X_t$
  - For each $u \in V(G)$, bags in which u occurs form connected subgraph of T
  - Each $u \in V(G)$ occurs in at least one bag
- Width: size largest bag – 1



Treewidth

Tree decomposition of G

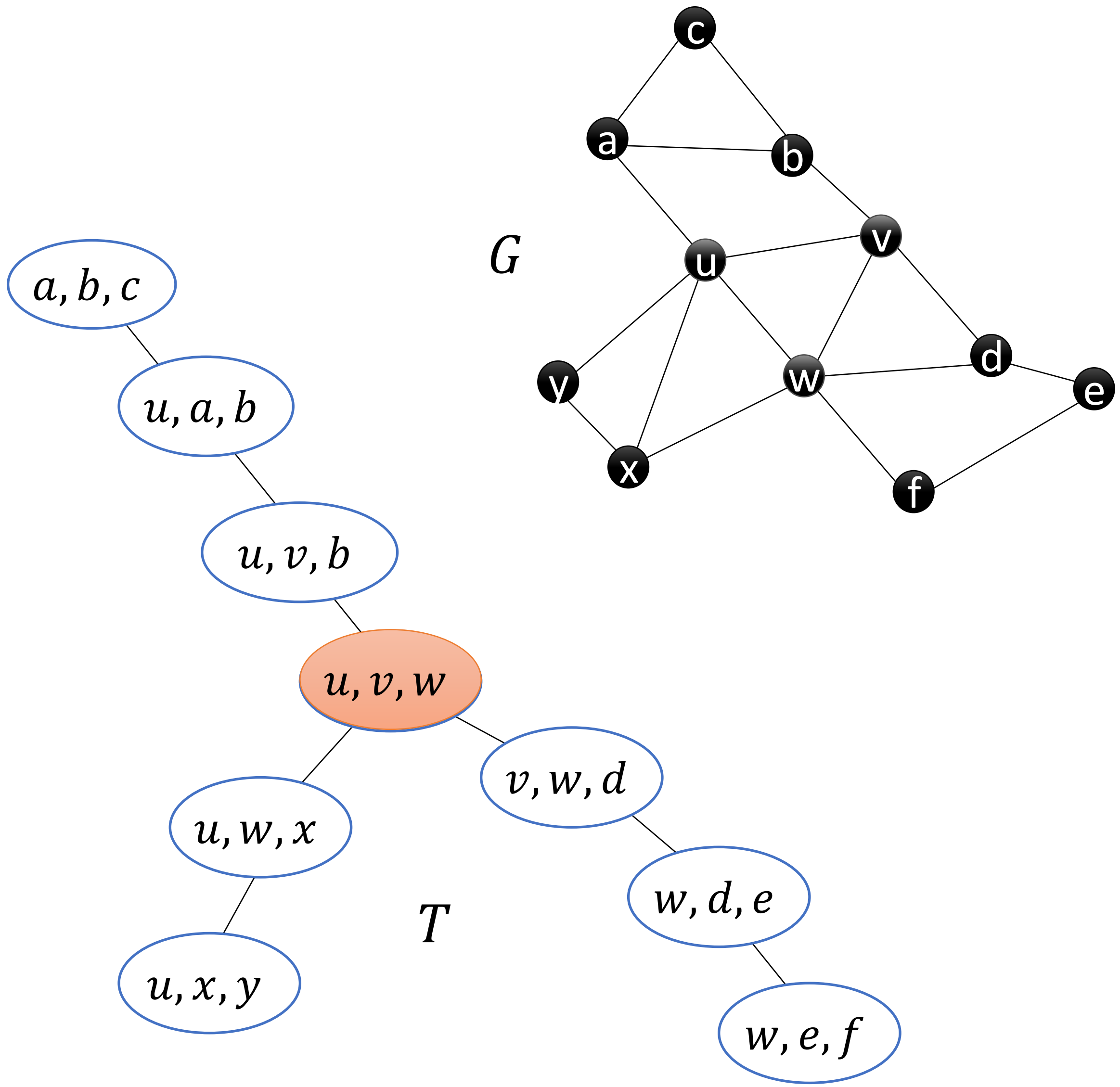
- Tree T with nodes each node t has bag $X_t \subseteq V(G)$
  - For each edge uv in G, exists bag such that $u \in X_t, v \in X_t$
  - For each $u \in V(G)$, bags in which u occurs form connected subgraph of T
  - Each $u \in V(G)$ occurs in at least one bag
- Width: size largest bag – 1



Treewidth

Tree decomposition of G

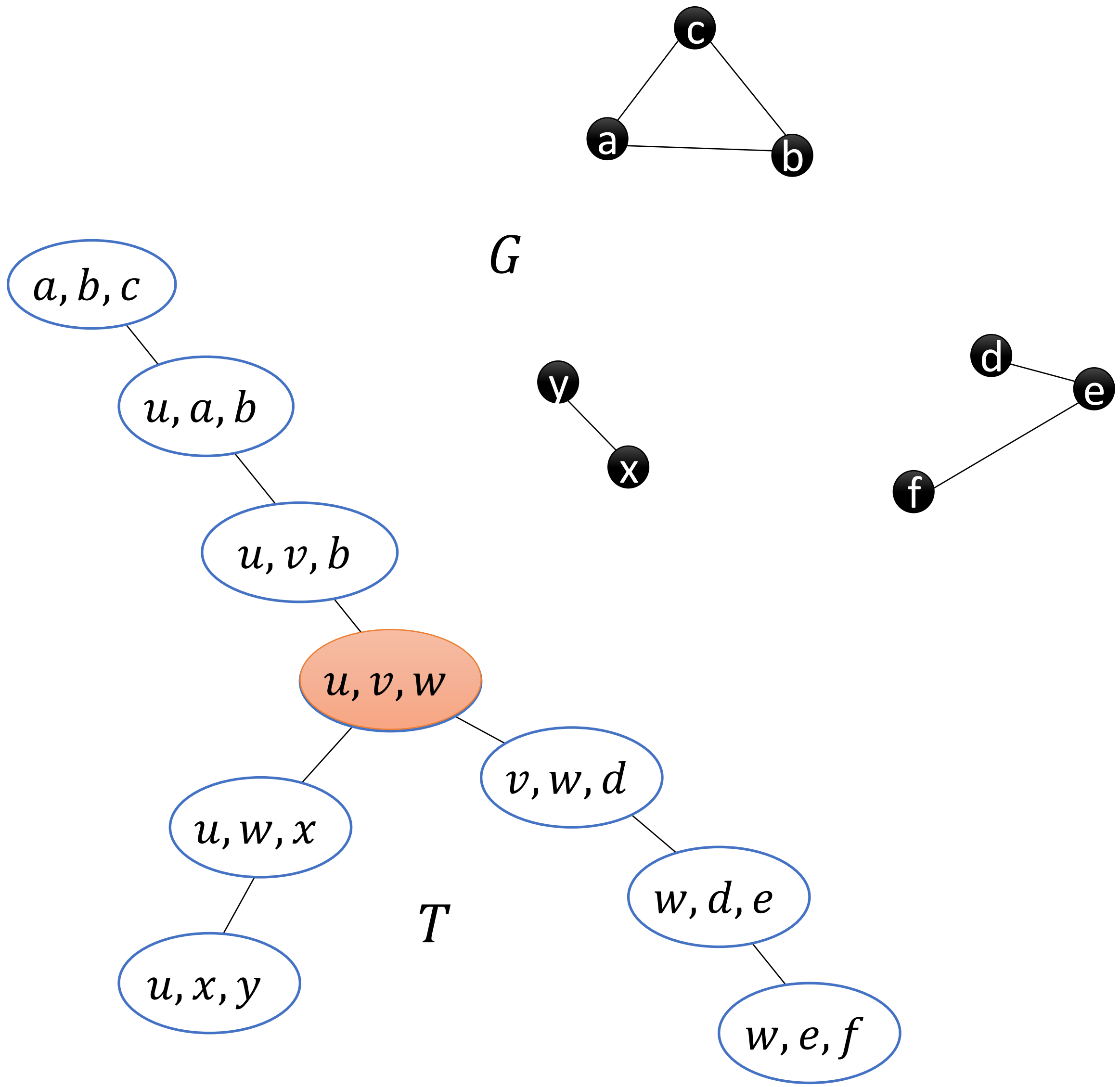
- Tree T with nodes each node t has bag $X_t \subseteq V(G)$
  - For each edge uv in G, exists bag such that $u \in X_t, v \in X_t$
  - For each $u \in V(G)$, bags in which u occurs form connected subgraph of T
  - Each $u \in V(G)$ occurs in at least one bag
- Width: size largest bag – 1



Treewidth

Tree decomposition of G

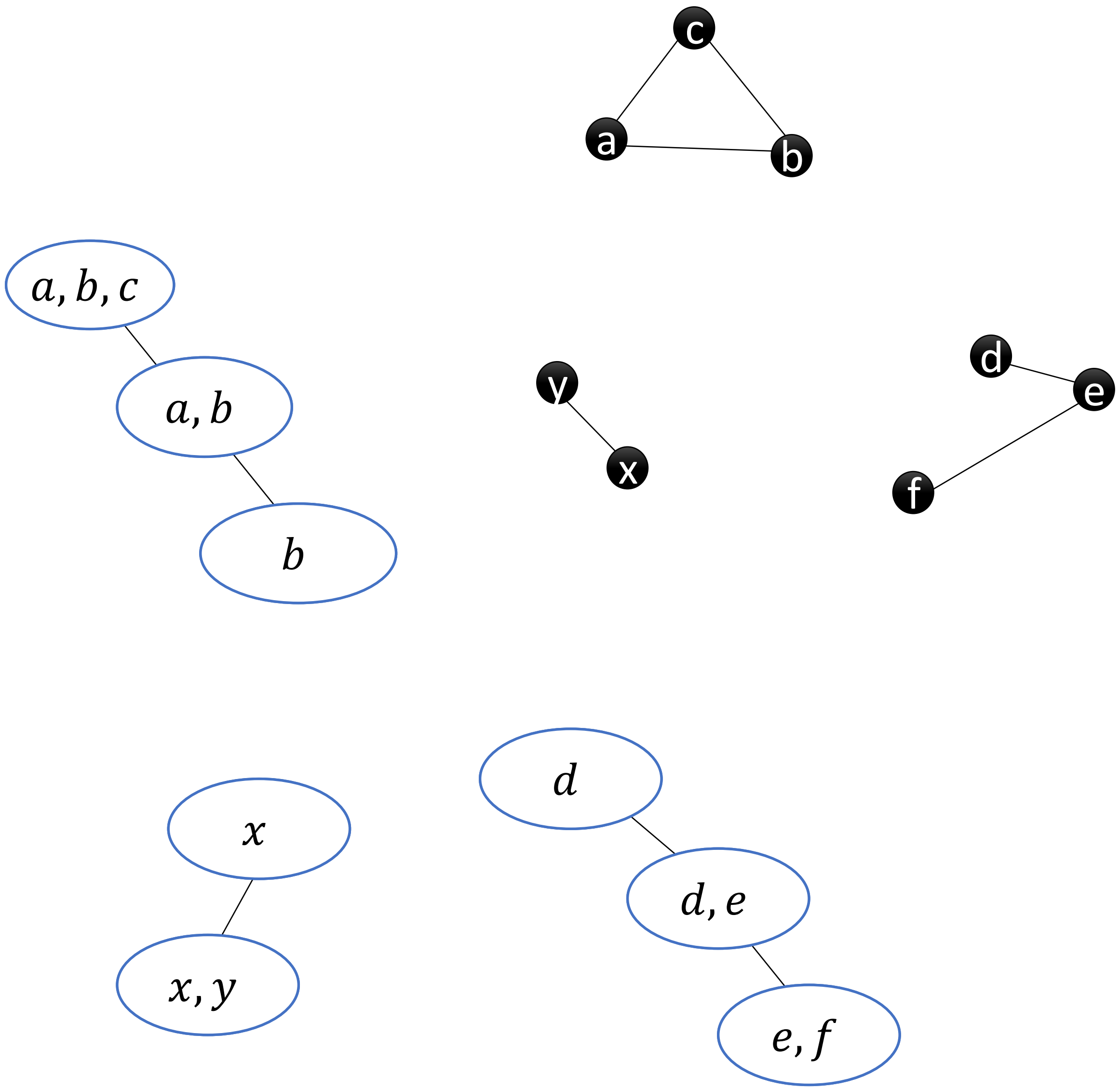
- Tree T with nodes each node t has bag $X_t \subseteq V(G)$
  - For each edge uv in G, exists bag such that $u \in X_t, v \in X_t$
  - For each $u \in V(G)$, bags in which u occurs form connected subgraph of T
  - Each $u \in V(G)$ occurs in at least one bag
- Width: size largest bag – 1



Treewidth

Tree decomposition of G

- Tree T with nodes each node t has bag $X_t \subseteq V(G)$
  - For each edge uv in G, exists bag such that $u \in X_t, v \in X_t$
  - For each $u \in V(G)$, bags in which u occurs form connected subgraph of T
  - Each $u \in V(G)$ occurs in at least one bag
- Width: size largest bag – 1



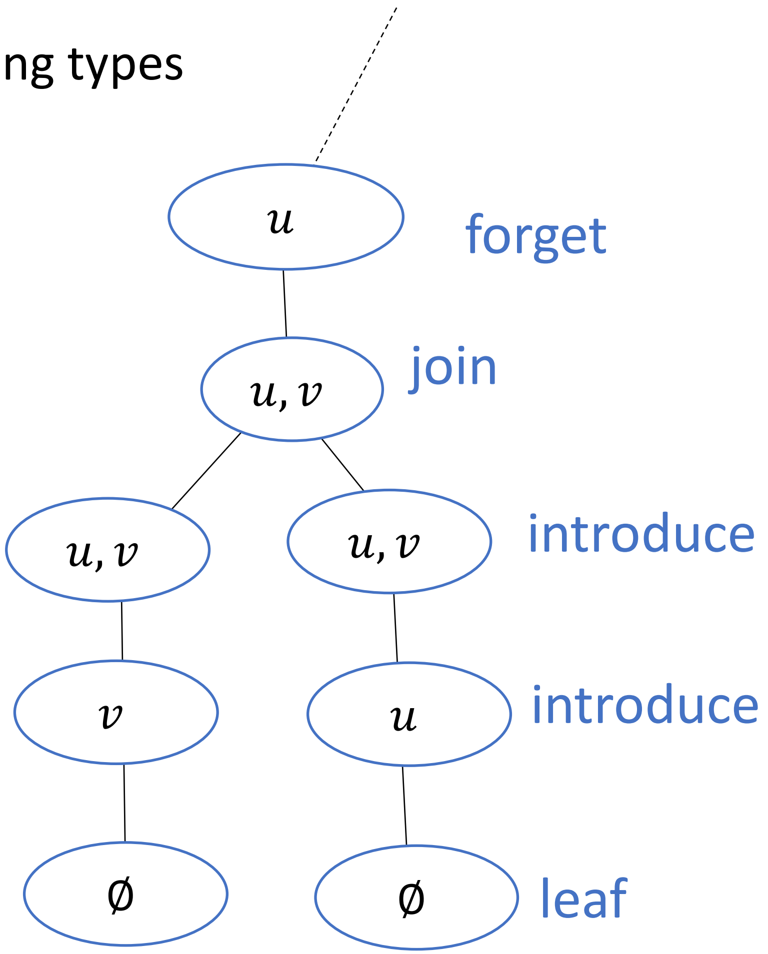
Nice tree decompositions

A rooted tree decomposition is nice if each node t is one of the following types

- **Leaf** – The node has no children, and $X_t = \emptyset$
  - $V(G_t) = V(G_t - X_t) = \emptyset$
- **Join** – The node has children t_1 and t_2 and $X_{t_1} = X_{t_2} = X_t$
  - $V(G_t) = V(G_{t_1}) \cup V(G_{t_2})$ and $V(G_{t_1}) \setminus X_t$, $V(G_{t_2}) \setminus X_t$ disjoint
- **Introduce** – The node has one child t_1 and $X_t = X_{t_1} \cup \{v\}$
  - $V(G_t) = V\left(G_{t_1}\right) \cup \{v\}$, but $V(G_{t_1}) \setminus X_t = V(G_{t_2}) \setminus X_t$
- **Forget** – The node has one child t_1 and $X_t = X_{t_1} \setminus \{v\}$
  - $V(G_t) = V\left(G_{t_1}\right)$, but $V(G_{t_1}) \setminus X_t = \{v\} \cup V(G_{t_2}) \setminus X_t$

Every tree decomposition can efficiently be made nice, without increasing its width

- We will assume $X_r = \emptyset$



Approximate Turing kernel for
Independent Set

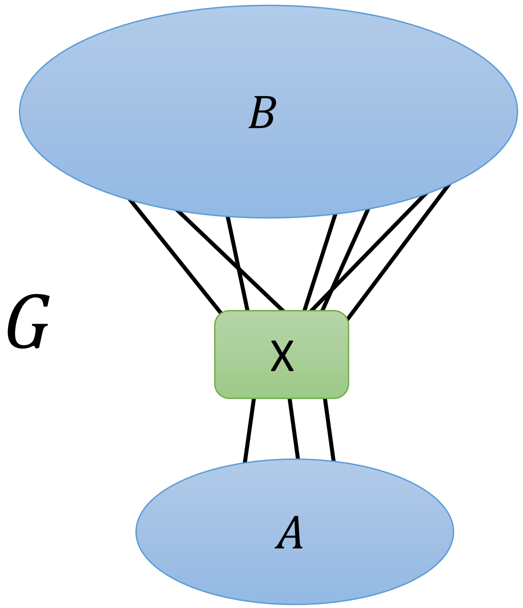
Independent Set

Theorem

Independent Set has a $(1+\varepsilon)$-approximate Turing Kernel with $O\left(\frac{\ell^2}{\varepsilon}\right)$ vertices.

Overview

1. Find a good separator X, separate the graph into (small) A and B
2. Ask the oracle for a solution S_A of part A
3. Recurse to find an approximate solution S_B for part B
4. Show $S_A \cup S_B$ is a c$(1+\varepsilon)$-approximate solution



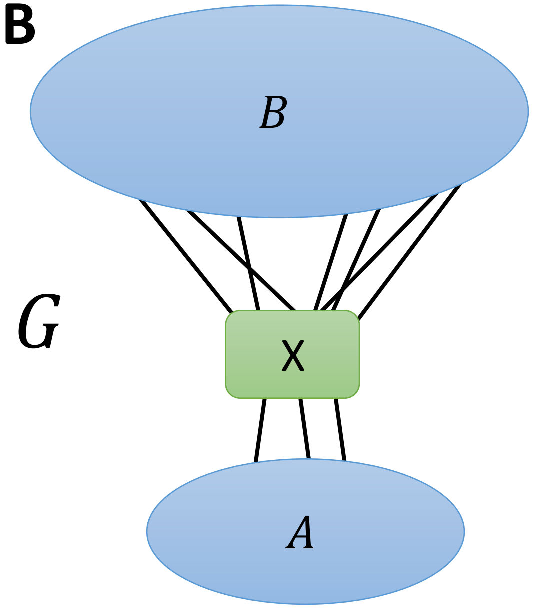
Independent Set

Theorem

Independent Set has a $(1+\varepsilon)$-approximate Turing Kernel with $O\left(\frac{\ell^2}{\varepsilon}\right)$ vertices.

Overview

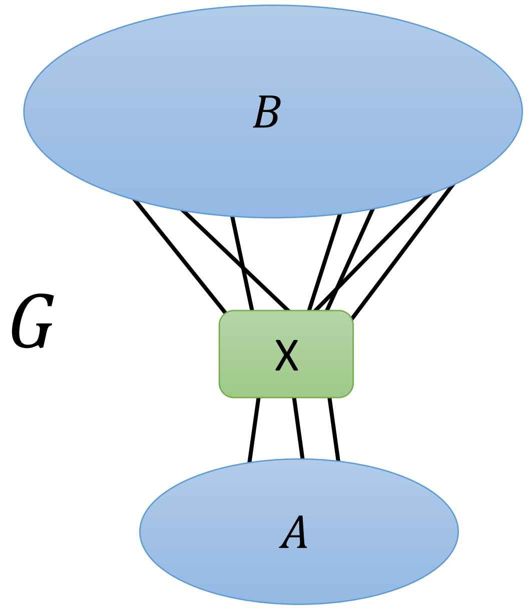
1. **Find a good separator X, separate the graph into (small) A and B**
2. Ask the oracle for a solution S_A of part A
3. Recurse to find an approximate solution S_B for part B
4. Show $S_A \cup S_B$ is a c$(1+\varepsilon)$-approximate solution



Finding a separator

What is a good separator? Separate the graph into X, A and B, such that

- $|X| \leq \ell + 1$
  - Use a bag in the tree decomposition!
- $|A|$ is small
  - $|A|$ will determine the size of the kernel
  - $|A| = O\left(\frac{\ell^2}{\varepsilon}\right)$
- The part of an optimal solution in $G[A]$ is sufficiently large
  - By discarding X, we loose out on value at most $|X|$
  - $|X|$ should be small, compared to IS(G[A])



Size of A

Theorem

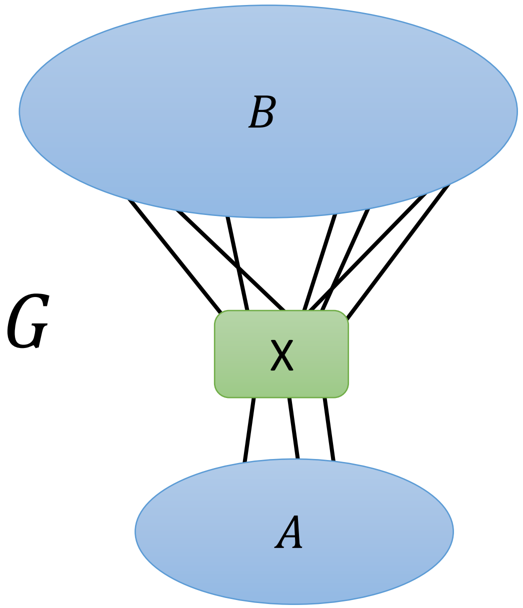
A graph with n vertices and treewidth ℓ, has an independent set of size at least $\frac{n}{\ell+1}$

Proof

Various options, immediate from alternative definition of TW

Conclusion

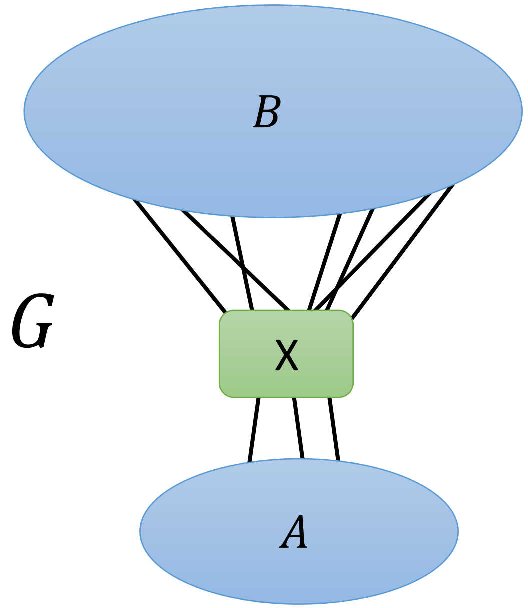
If $|A| \geq \frac{(\ell+1)^2}{\varepsilon}$, then $IS(A) \geq \frac{\ell+1}{\varepsilon} \geq \frac{|X|}{\varepsilon}$



Finding a separator

Find a node t in T such that $\frac{(\ell+1)^2}{\varepsilon} \leq |G_t - \mathrm{X}_t| \leq \frac{10(\ell+1)^2}{\varepsilon}$

- Let $A := G_t - X_t$, $X := X_t$
- Recurse as long as $G_t - \mathrm{X}_t$ too large
  - **Join node** – Recurse on subtree with at least half the vertices
  - **Introduce/forget node** – Recurse on subtree
  - **Leaf node** – Contradicts $G_t - \mathrm{X}_t$ large



Various separator theorems for treewidth are known; we show this one for completeness.

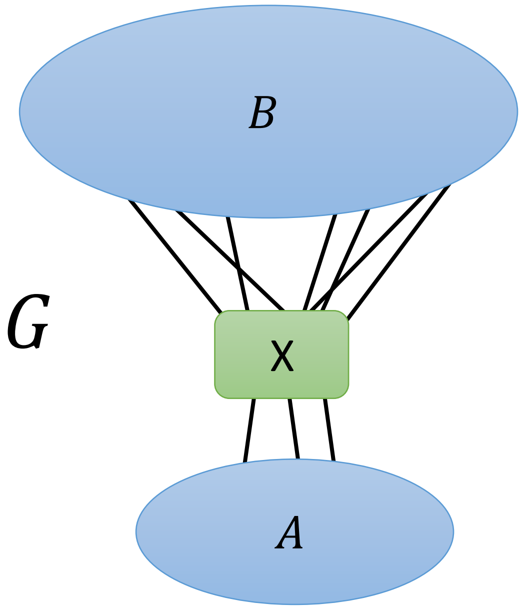
Independent Set

Theorem

Independent Set has a $(1+\varepsilon)$-approximate Turing Kernel with $O\left(\frac{\ell^2}{\varepsilon}\right)$ vertices.

Overview

1. Find a good separator X, separate the graph into (small) A and B
2. **Ask the oracle for a solution S_A of part A**
3. **Recurse to find an approximate solution S_B for part B**
4. Show $S_A \cup S_B$ is a c$(1+\varepsilon)$-approximate solution



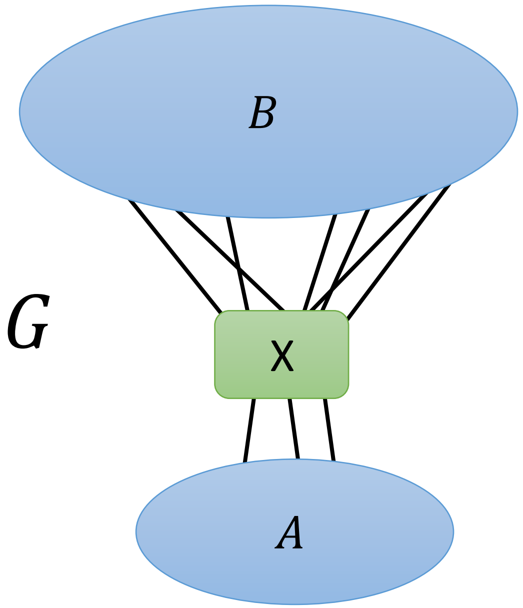
Independent Set

Theorem

Independent Set has a $(1+\varepsilon)$-approximate Turing Kernel with $O\left(\frac{\ell^2}{\varepsilon}\right)$ vertices.

Overview

1. Find a good separator X, separate the graph into (small) A and B
2. Ask the oracle for a solution S_A of part A
3. Recurse to find an approximate solution S_B for part B
4. **Show $S_A \cup S_B$ is a c$(1+\varepsilon)$-approximate solution**



Independent Set: Correctness

Consider an optimal solution S, then

$$|S| = |S \cap A| + |S \cap B| + |S \cap \mathrm{X}| \leq opt(G[A]) + opt(G[B]) + |X|$$

$$\leq c|S_A| + c(1+\varepsilon)|S_B| + \varepsilon|S_A|$$

$$\leq c(1+\varepsilon)(S_A + S_B)$$

Crucial point: Lower bound for IS on graphs of low treewidth

Independent Set: Correctness

Consider an optimal solution S, then

$$|S| = |S \cap A| + |S \cap \ldots X| \leq opt(G[A]) + opt(G[B]) + |X|$$

By the oracle

Induction

$$\leq c|S_A| + c(1+\varepsilon)|S_B| + \varepsilon|S_A|$$

$$\leq c(1+\varepsilon)(S_A + S_B)$$

Crucial point: Lower bound for IS on graphs of low treewidth

Approximate Turing kernel for Vertex Cover

Parameterized by treewidth

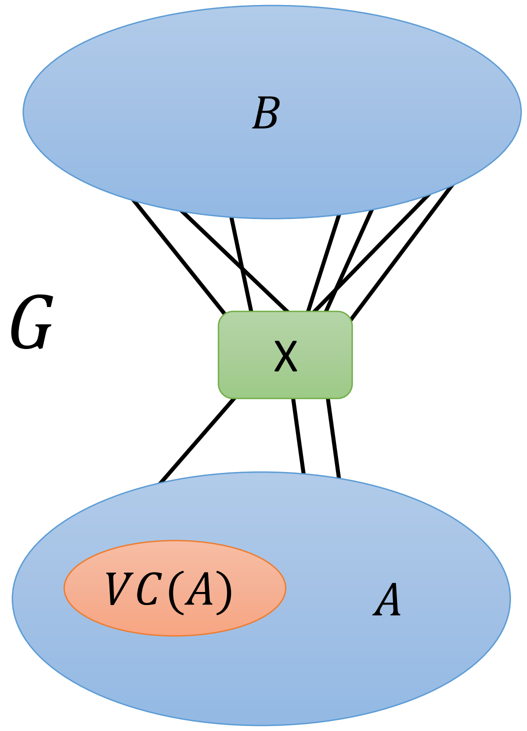
Vertex Cover

Theorem

Vertex Cover has a $(1+\varepsilon)$-approximate Turing Kernel with $O\left(\frac{\ell}{\varepsilon}\right)$ vertices.

Overview

1. Find a good separator X, separate the graph into A and B
   - Such that $VC(G[A])$ small
2. Apply the kernel for vertex cover to $G[A]$
3. Ask the oracle for a solution $S_A{}'$ of A'
4. Use this to obtain a solution S_A of $G[A]$
5. Recurse to find an approximate solution S_B for part B
6. Show $S_A \cup S_B \cup X$ is a $(1+\varepsilon)$-approximate solution

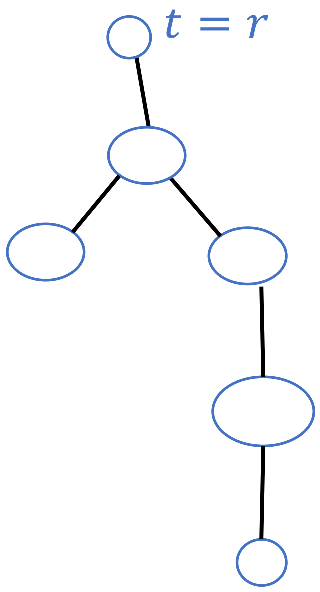


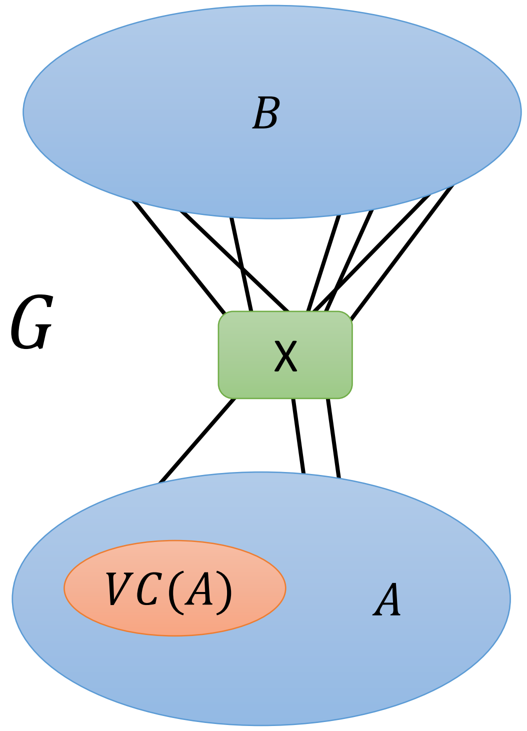
Finding a separator

Find a node t such that

$$\frac{\ell+1}{\varepsilon} \leq VC(G_t - \mathrm{X}_t) \leq \frac{10(\ell+1)}{\varepsilon}$$

Start from the root r



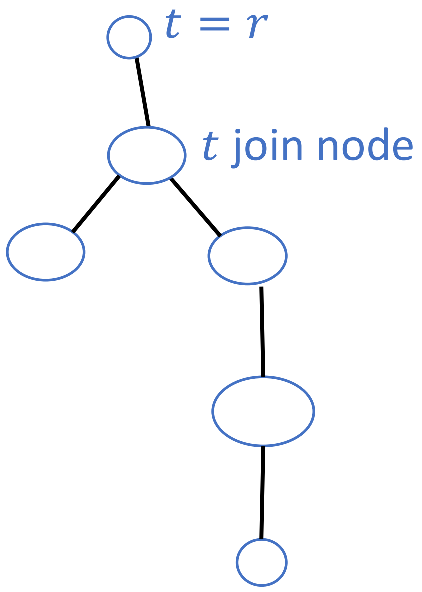


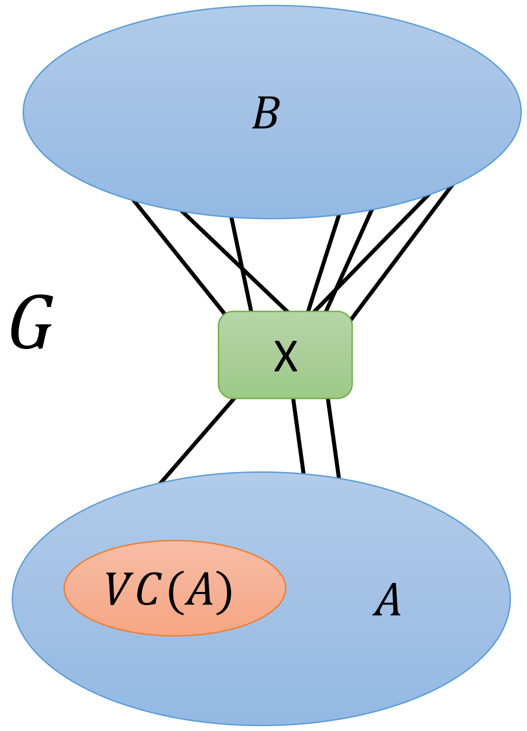
Finding a separator

Find a node t such that

$$\frac{\ell+1}{\varepsilon} \leq VC(G_t - \mathrm{X}_t) \leq \frac{10(\ell+1)}{\varepsilon}$$

Start from the root r



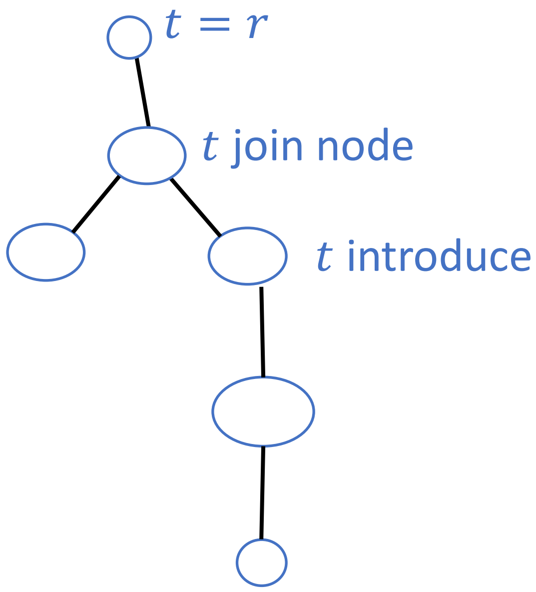


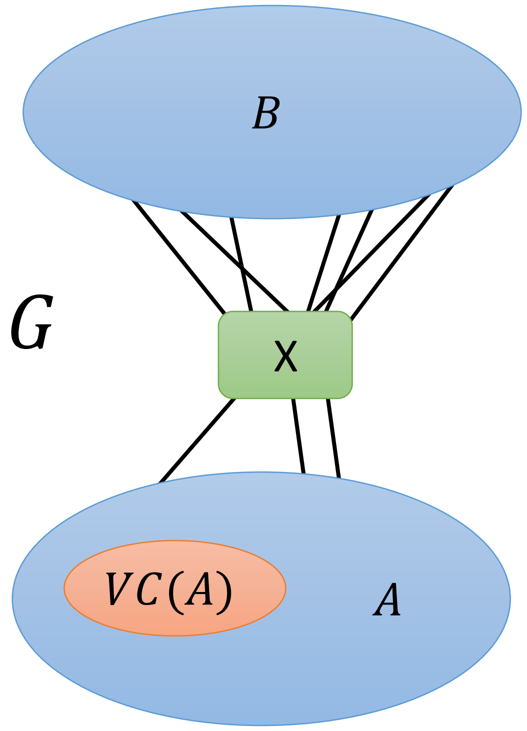
Finding a separator

Find a node t such that

$$\frac{\ell+1}{\varepsilon} \leq VC(G_t - X_t) \leq \frac{10(\ell+1)}{\varepsilon}$$

Start from the root r



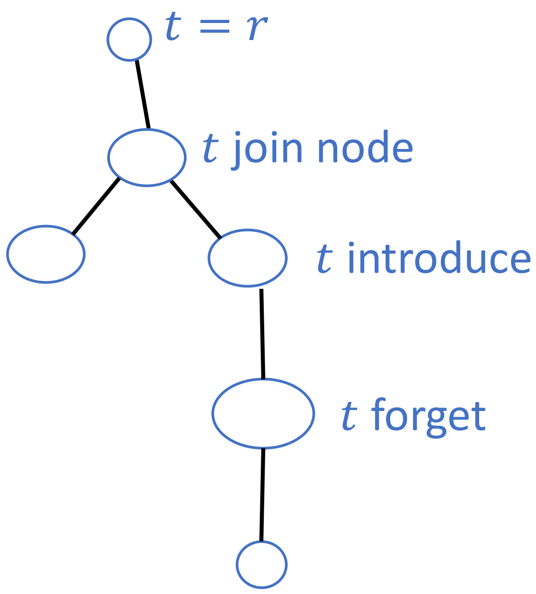


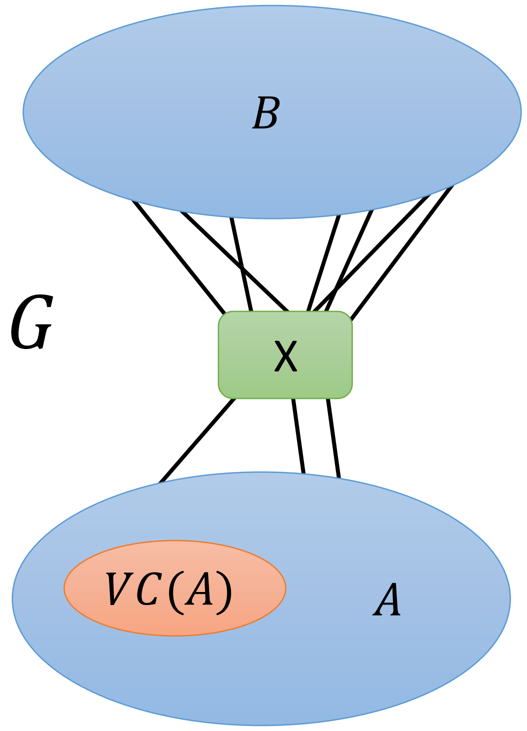
Finding a separator

Find a node t such that

$$\frac{\ell+1}{\varepsilon} \leq VC(G_t - \mathrm{X}_t) \leq \frac{10(\ell+1)}{\varepsilon}$$

Start from the root r



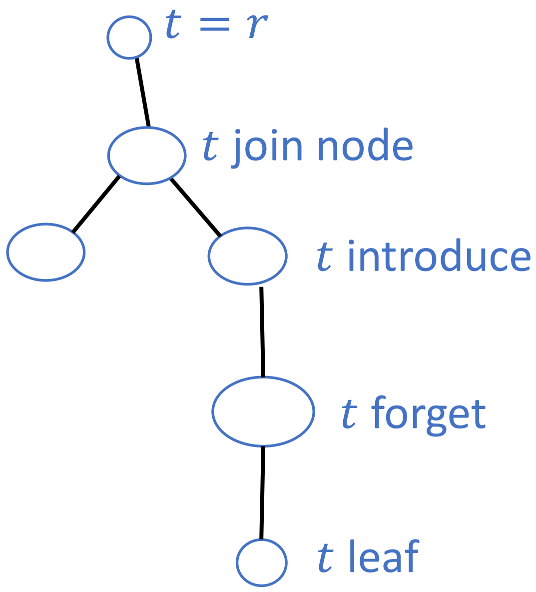


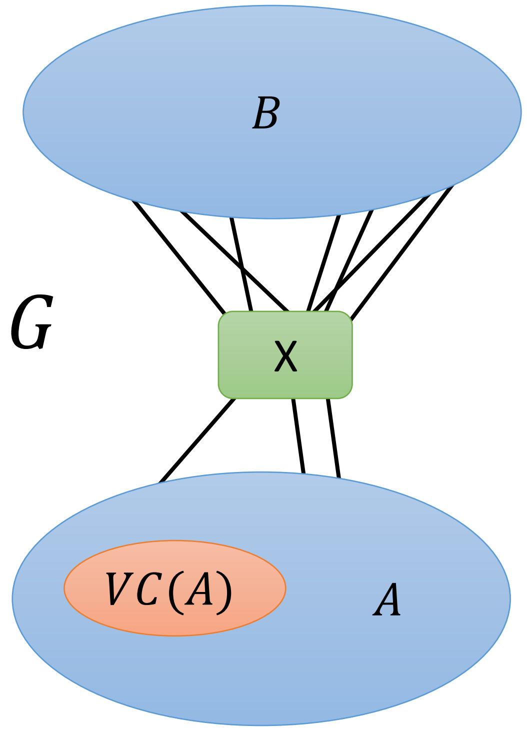
Finding a separator

Find a node t such that

$$\frac{\ell+1}{\varepsilon} \leq VC(G_t - X_t) \leq \frac{10(\ell+1)}{\varepsilon}$$

Start from the root r



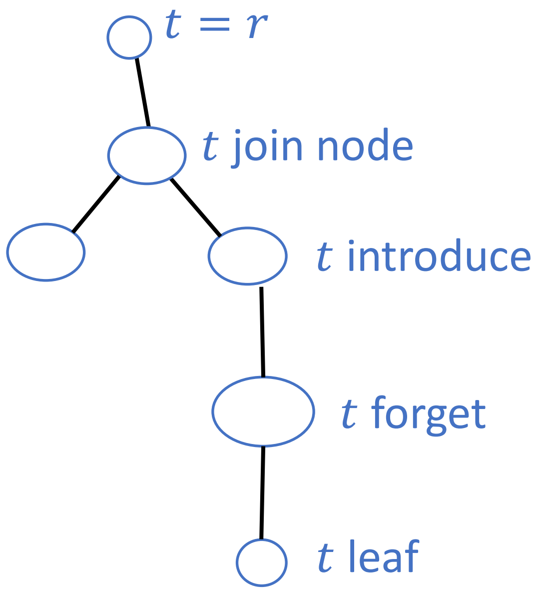


Finding a separator

Find a node t such that

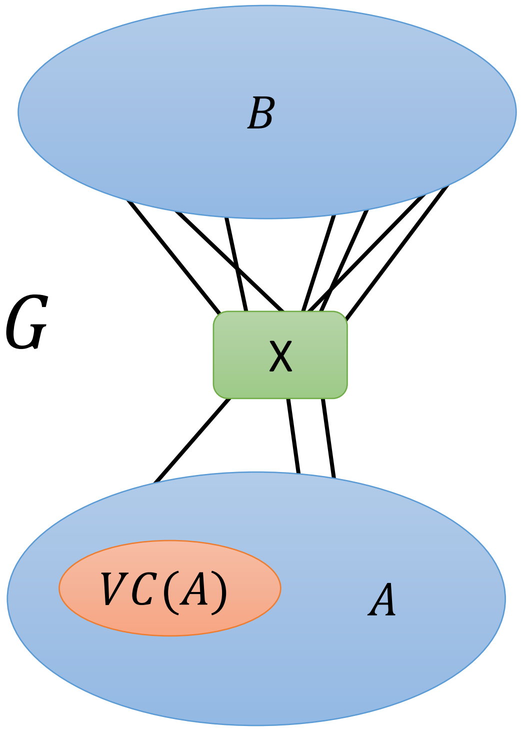
$$\frac{\ell + 1}{\varepsilon} \leq VC(G_t - X_t) \leq \frac{10(\ell + 1)}{\varepsilon}$$

Start from the root r



One issue:
Computing Vertex Cover is NP-hard, so how to find t?

Solution
Approximate!



NO Turing kernel for

Vertex Cover

Parameterized by treewidth

Turing kernel lower bound

Theorem

Vertex Cover parameterized by treewidth is MK[2]-hard

If Q is MK[2]-hard, then a poly Turing kernel for Q implies a poly Turing kernel for CNF-SAT(n)

- Believed to not exist

Lower bound proof

Reduction from CNF-SAT

$$F = (x_1 \lor x_2 \lor \cdots \lor x_m) \land (\neg x_3 \lor x_5) \land (x_1 \lor \neg x_2 \lor x_4 \lor x_8) \land \cdots$$

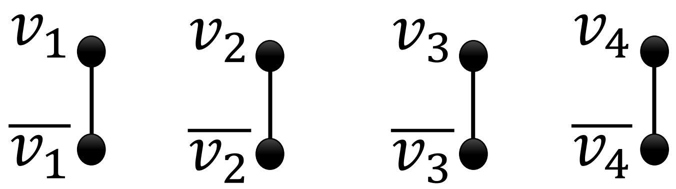
Unbounded clause length

Turing kernel lower bound

$$F = \underbrace{(x_1 \vee x_2)}^{C_1} \wedge \underbrace{(\neg x_1 \vee x_3 \vee x_4)}^{C_2} \wedge \cdots$$

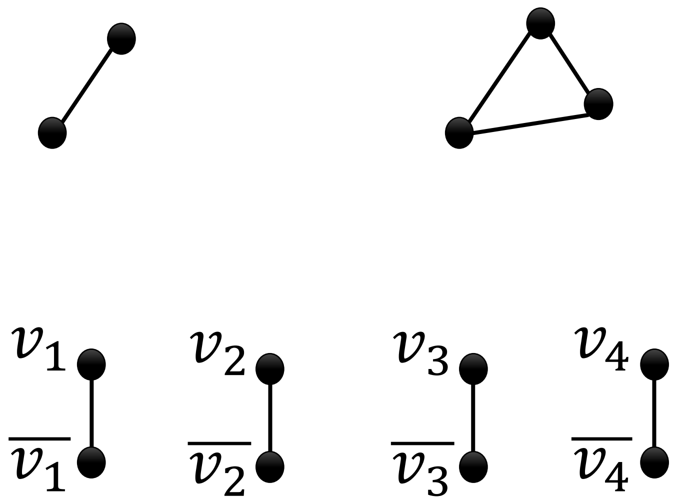
Turing kernel lower bound

$$F = \overset{C_1}{(x_1 \vee x_2)} \wedge \overset{C_2}{(\neg x_1 \vee x_3 \vee x_4)} \wedge \cdots$$



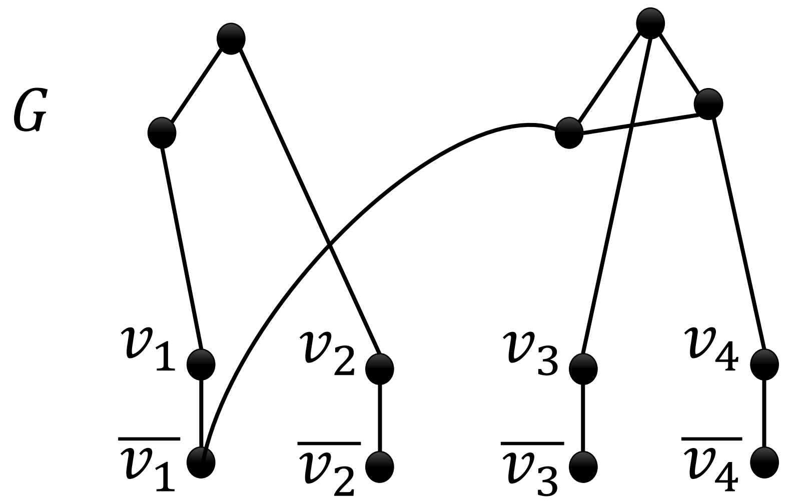
Turing kernel lower bound

$$F = \overset{C_1}{(x_1 \lor x_2)} \land \overset{C_2}{(\neg x_1 \lor x_3 \lor x_4)} \land \cdots$$



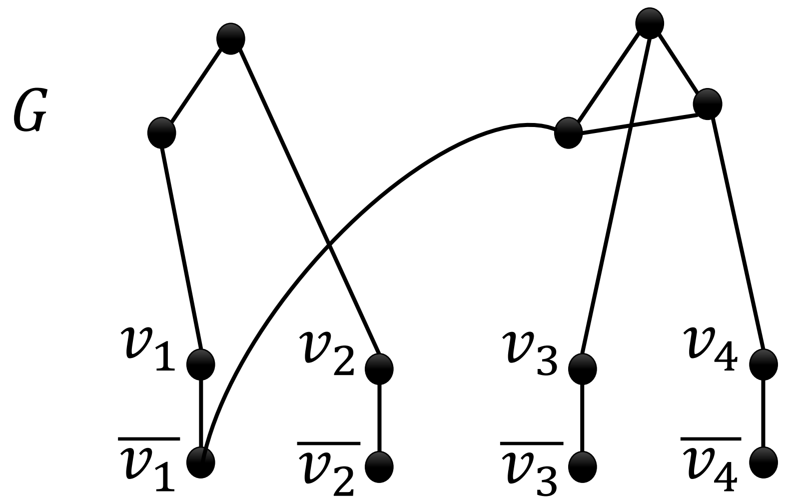
Turing kernel lower bound

$$F = \underbrace{(x_1 \vee x_2)}_{C_1} \wedge \underbrace{(\neg x_1 \vee x_3 \vee x_4)}_{C_2} \wedge \cdots$$



Turing kernel lower bound

$$F = \overset{C_1}{(x_1 \vee x_2)} \wedge \overset{C_2}{(\neg x_1 \vee x_3 \vee x_4)} \wedge \cdots$$

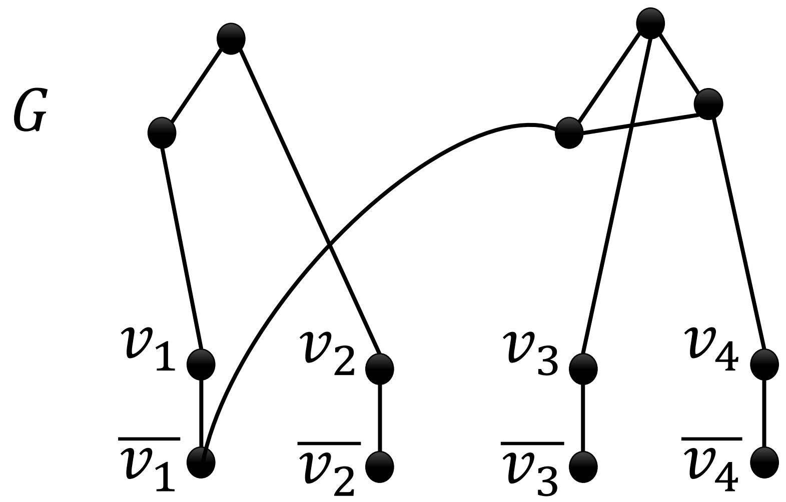


G has a vertex cover of size $n + \sum(|C_i| - 1)$ if and only if F is satisfiable

Turing kernel lower bound

$$\overset{C_1}{F = (x_1 \lor x_2)} \land \overset{C_2}{(\neg x_1 \lor x_3 \lor x_4)} \land \cdots$$

G has treewidth $O(n)$



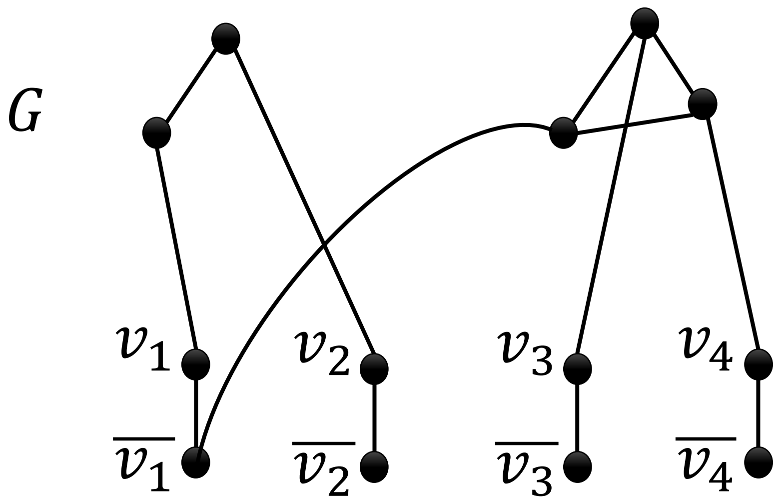
G has a vertex cover of size $n + \sum(|C_i| - 1)$ if and only if F is satisfiable

Turing kernel lower bound

$$\overset{C_1}{} \quad \overset{C_2}{}$$

$$F = \underbrace{(x_1 \vee x_2)}_{C_1} \wedge \underbrace{(\neg x_1 \vee x_3 \vee x_4)}_{C_2} \wedge \cdots$$

G has treewidth $O(n)$



G has a vertex cover of size $n + \sum(|C_i| - 1)$ if and only if F is satisfiable

$\Rightarrow$ If VC(tw) has a polynomial (Turing) kernel, then so does CNF-SAT(n)

Approximate Turing kernel for

Connected Vertex Cover

Parameterized by treewidth

Connected Vertex Cover

Given a graph G (and tree decomposition T) find minimum vertex cover S such that $G[S]$ is connected

Cannot apply earlier idea immediately

- No lower bound based on treewidth
- No polynomial kernel with parameter k
- Combining solutions is complex
  - Need to ensure connectivity

Connected Vertex Cover

No polynomial kernel parameterized by solution size, but

- A $(1+\delta)$-approximate kernel for all $\delta > 0$
  [Lokshtanov, Panolan, Ramanujan, Saurabh STOC 2017]

No good bounds on optimal solution depending on $CVC(G[A])$, $CVC(G[B])$, and X

- Recall for vertex cover we implicitly used

$$\mathrm{VC}(\mathrm{G}[\mathrm{A}]) + VC(G[B]) \leq VC(G) \leq VC(G[A]) + VC(G[B]) + |X|$$

Connected Vertex Cover

No polynomial kernel parameterized by solution size, but

- A $(1 + \delta)$-approximate kernel for all $\delta > 0$
  [Lokshtanov, Panolan, Ramanujan, Saurabh STOC 2017]

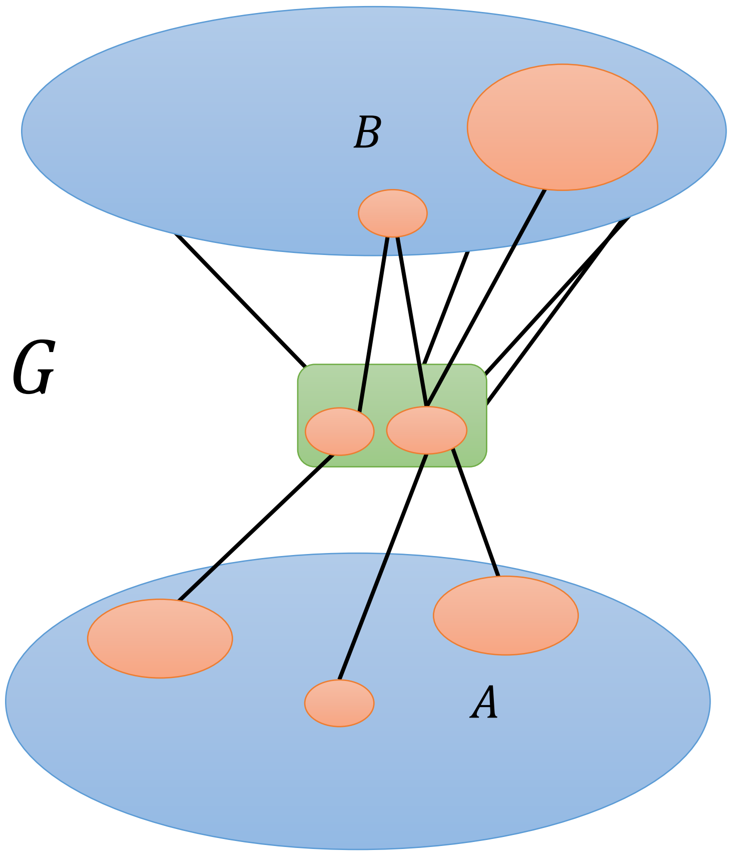
No good bounds on optimal solution depending on $CVC(G[A])$, $CVC(G[B])$, and X

- Recall for vertex cover we implicitly used

$$\mathrm{VC}(\mathrm{G}[\mathrm{A}]) + VC(G[B]) \leq VC(G) \leq VC(G[A]) + VC(G[B]) + |X|$$

False for CVC, even when $G[A]$ connected

Also problematic

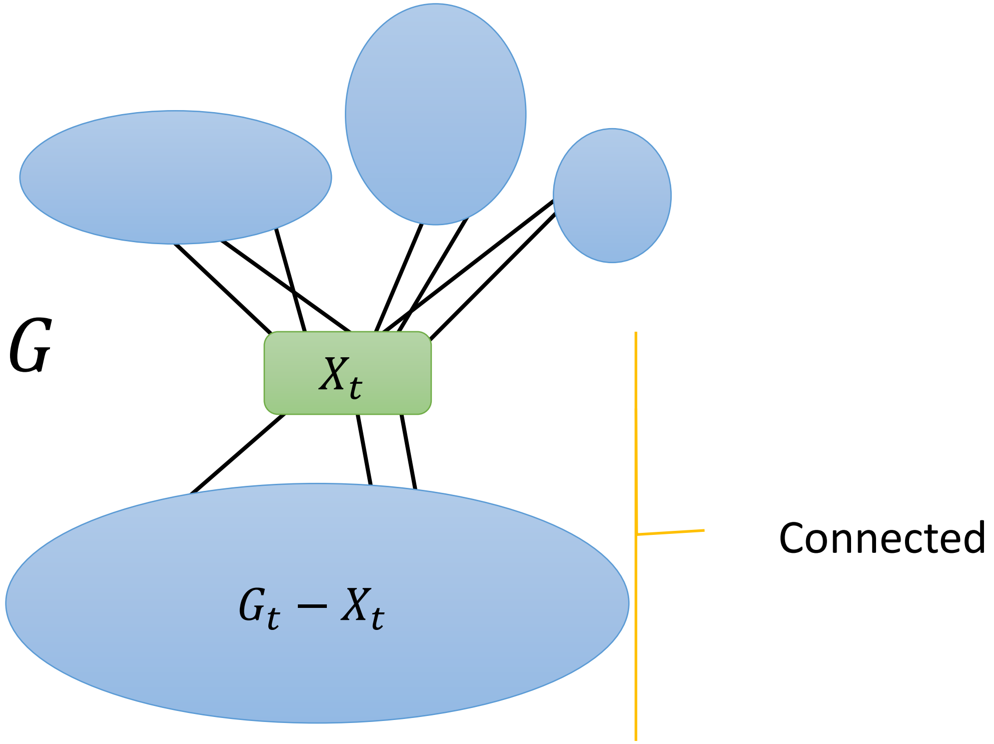


Subconnected tree decompositions

Tree decomposition such that G_t is connected for all t

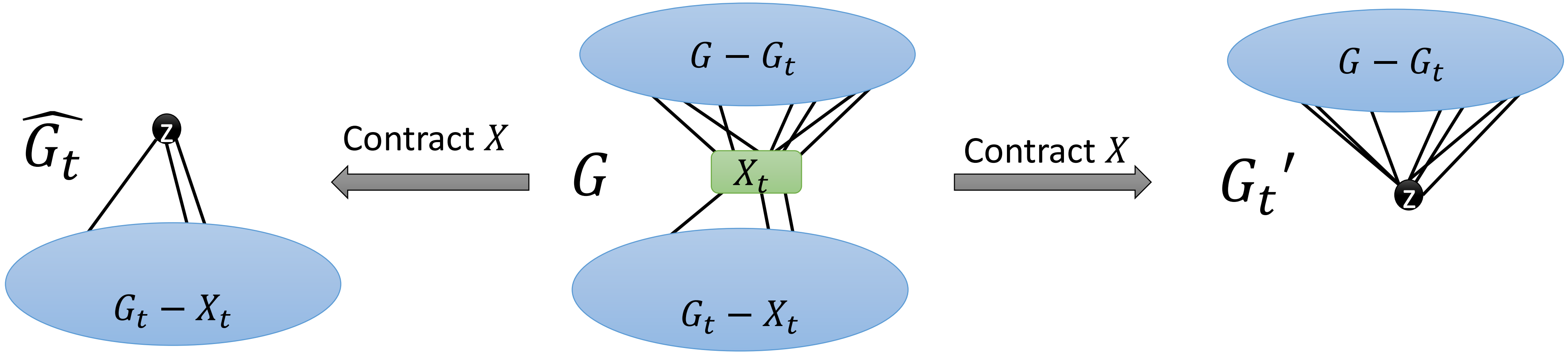
- A given tree decomposition can be made subconnected in polynomial time
  - Without increasing its width

    [Fraigniaud, Nisse, LATIN 2006]



Connected Vertex Cover

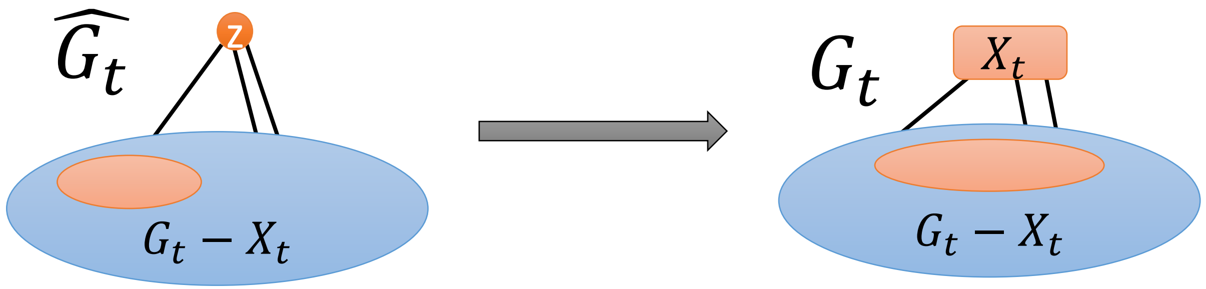
1. If our graph has a small CVC
   - Apply $(1+\varepsilon)$-approximate kernel, obtain (G', k')
   - Feed (G', k') to oracle, obtain solution S'
   - Lift S' to a solution S of (G, k)
2. Else, obtain tree decomposition such that G_t connected for all t
   - For all t, define the following graphs



Connected Vertex Cover

Lemma

Given a connected vertex cover S in $\widehat{G_t}$, we can in polynomial time find a connected vertex cover S' in G_t such that $|S'| \leq |S| + 2|X|$. Furthermore, $X \subseteq S'$.

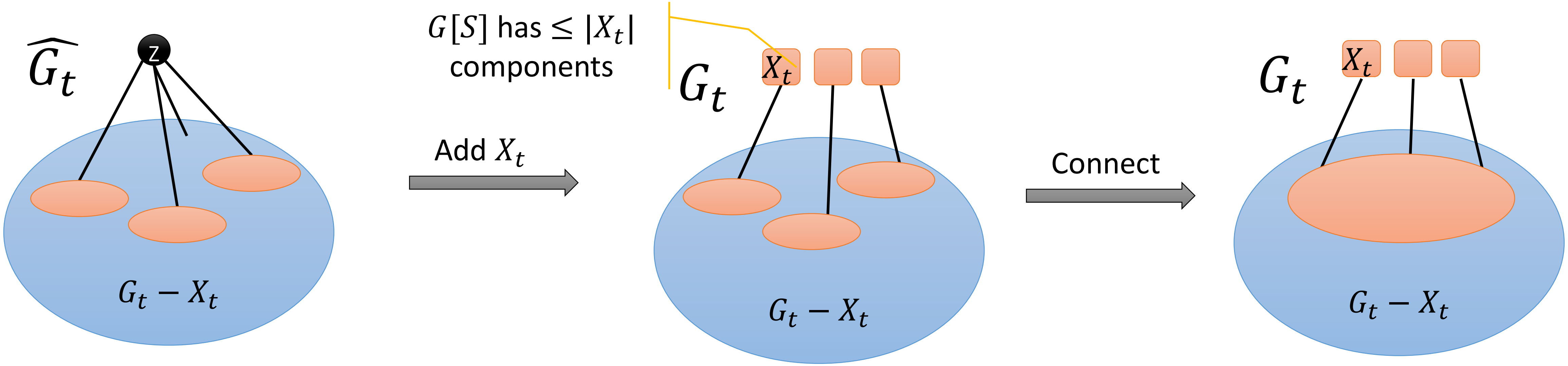


Connected Vertex Cover

Lemma

Given a connected vertex cover S in $\widehat{G_t}$, we can in polynomial time find a connected vertex cover S' in G_t such that $|S'| \leq |S| + 2|X|$. Furthermore, $X \subseteq S'$.

(recall G_t is connected)

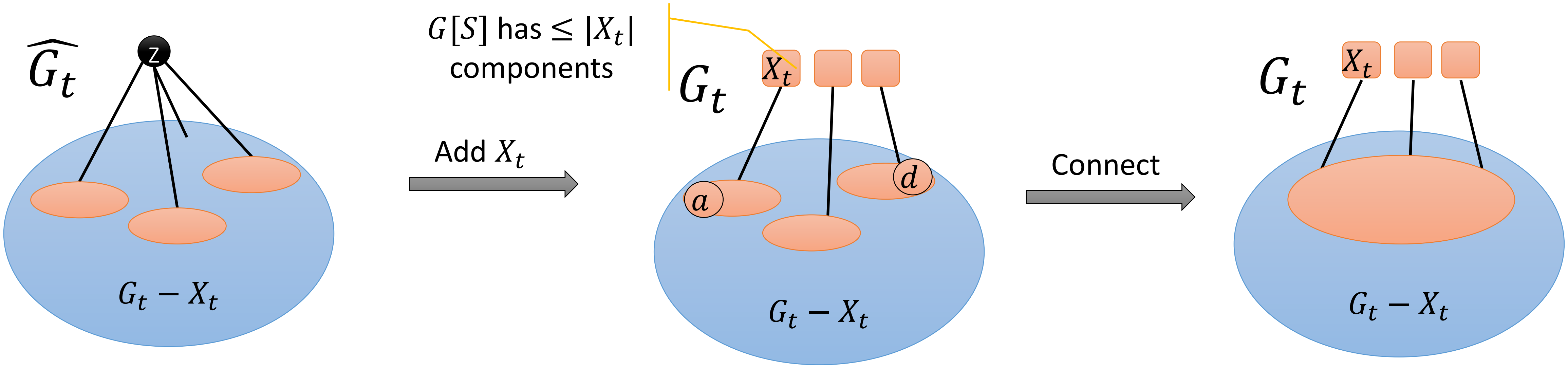


Connected Vertex Cover

Lemma

Given a connected vertex cover S in $\widehat{G_t}$, we can in polynomial time find a connected vertex cover S' in G_t such that $|S'| \leq |S| + 2|X|$. Furthermore, $X \subseteq S'$.

(recall G_t is connected)

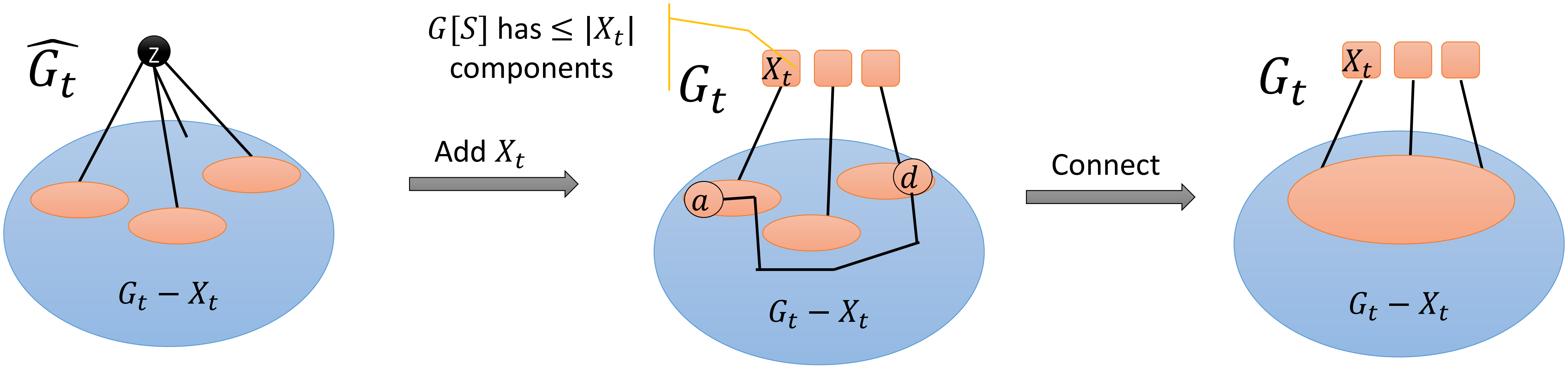


Connected Vertex Cover

Lemma

Given a connected vertex cover S in $\widehat{G_t}$, we can in polynomial time find a connected vertex cover S' in G_t such that $|S'| \leq |S| + 2|X|$. Furthermore, $X \subseteq S'$.

(recall G_t is connected)

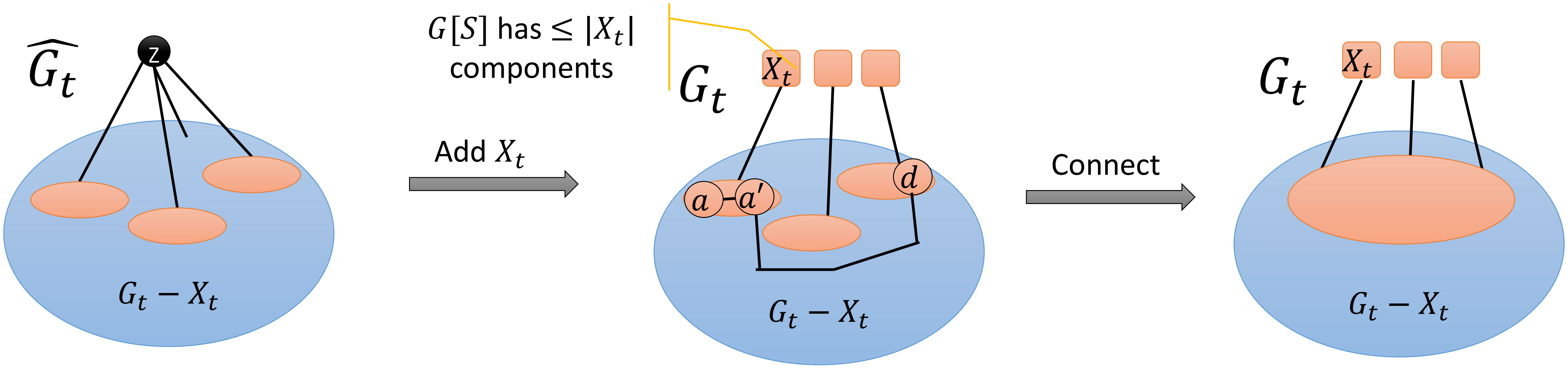


Connected Vertex Cover

Lemma

Given a connected vertex cover S in $\widehat{G_t}$, we can in polynomial time find a connected vertex cover S' in G_t such that $|S'| \leq |S| + 2|X|$. Furthermore, $X \subseteq S'$.

(recall G_t is connected)

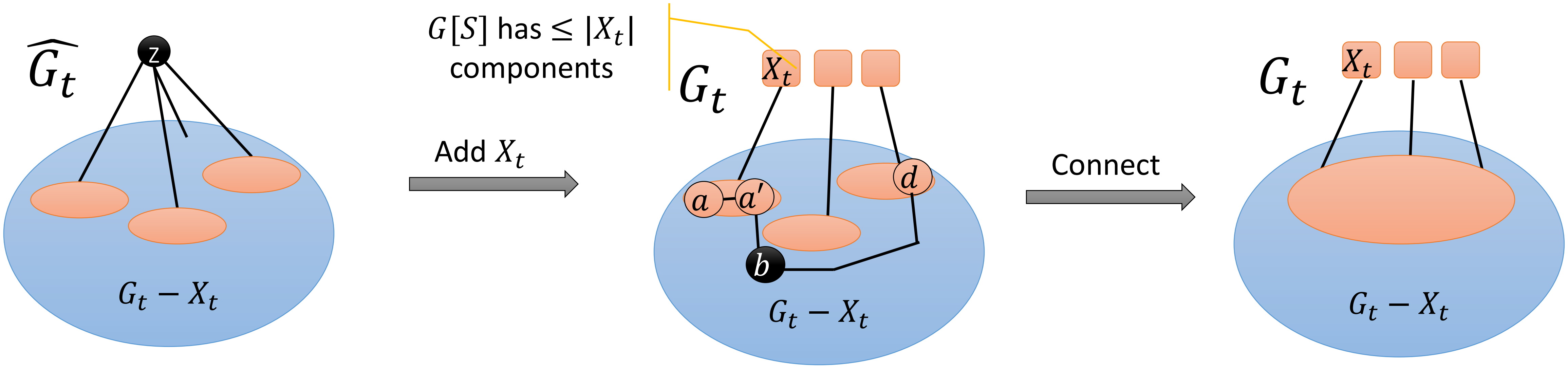


Connected Vertex Cover

Lemma

Given a connected vertex cover S in $\widehat{G_t}$, we can in polynomial time find a connected vertex cover S' in G_t such that $|S'| \leq |S| + 2|X|$. Furthermore, $X \subseteq S'$.

(recall G_t is connected)

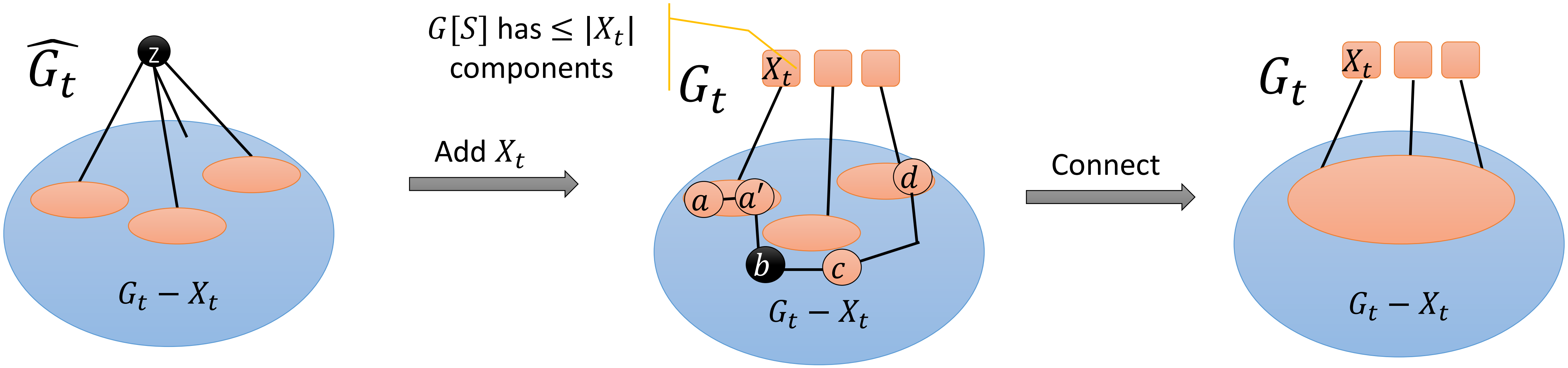


Connected Vertex Cover

Lemma

Given a connected vertex cover S in $\widehat{G_t}$, we can in polynomial time find a connected vertex cover S' in G_t such that $|S'| \leq |S| + 2|X|$. Furthermore, $X \subseteq S'$.

(recall G_t is connected)

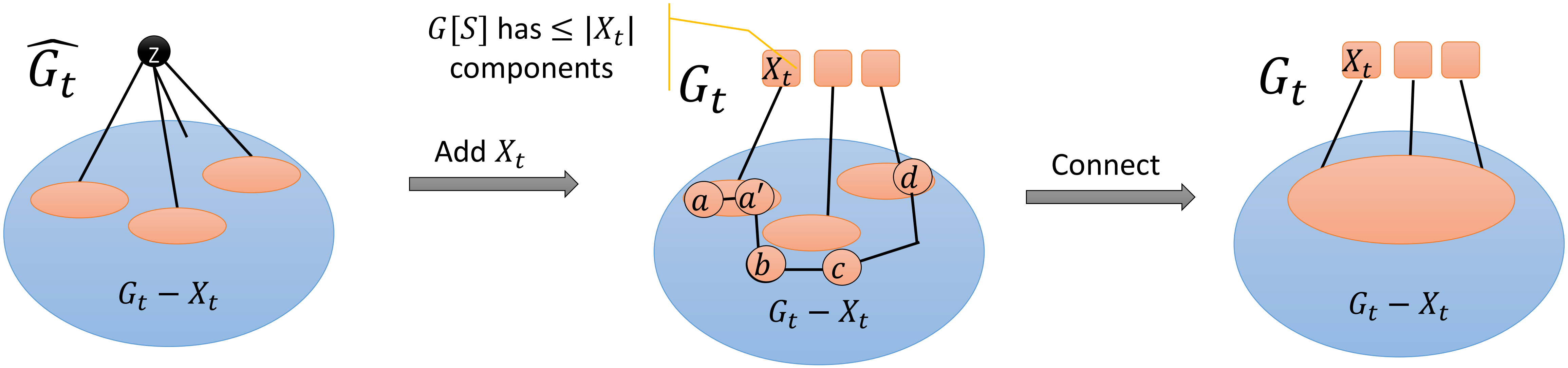


Connected Vertex Cover

Lemma

Given a connected vertex cover S in $\widehat{G_t}$, we can in polynomial time find a connected vertex cover S' in G_t such that $|S'| \leq |S| + 2|X|$. Furthermore, $X \subseteq S'$.

(recall G_t is connected)



Connected Vertex Cover

1. If G has small CVC
   - Use the $(1+\varepsilon)$-approximate kernel & oracle to obtain $c(1+\varepsilon)$-approx. solution
2. Otherwise, find t such that $\widehat{G_t}$ has CVC of size between $\frac{\ell}{\delta}$ and $\frac{100\ell^2}{\delta}$ for $\delta = \frac{\varepsilon}{3}$
3. Obtain $c(1+\delta)$-approximate CVC $\hat{S}$ in $\widehat{G_t}$
   - Use the $(1+\delta)$-approximate kernel & oracle
4. By lemma, obtain CVC $\tilde{S}$ in G_t, with $X \subseteq \tilde{S}$ and $|\tilde{S}| \leq |\hat{S}| + 2|X|$
5. Obtain $c(1+\varepsilon)$-approximate CVC S' in G'_t
6. Output $S' \cup \tilde{S} \setminus \{z\}$

Approximate Turing kernel for
Friendly Problems

Parameterized by treewidth

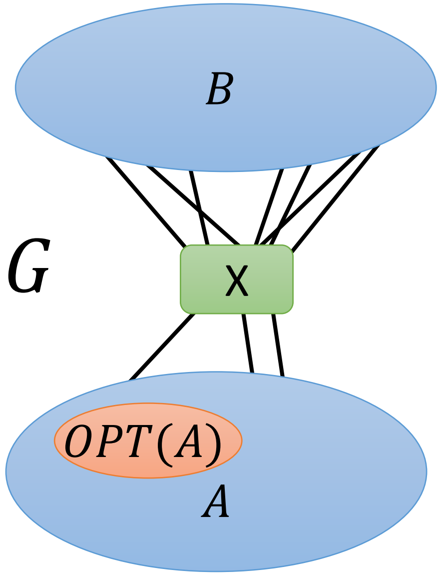
Ingredients for our ATK

A friendly problem

1. Has poly-size $(1+\varepsilon)$-Approximate kernel when parameterized by solution size

2. Has constant-factor approximation algorithm

3. Has very good behavior with respect to separators
   - Construct "good" solution for G based on solutions for A, B



Ingredients for our ATK

A friendly problem

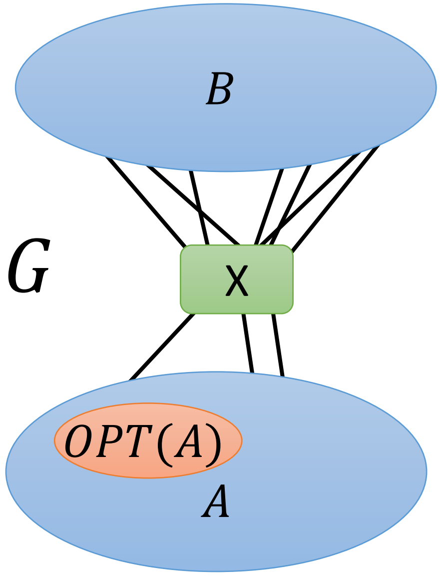
1. Has poly-size $(1+\varepsilon)$-Approximate kernel when parameterized by solution size

Can be relaxed to $k+\ell$

2. Has constant-factor approximation algorithm

Can be relaxed to function of $k+\ell$

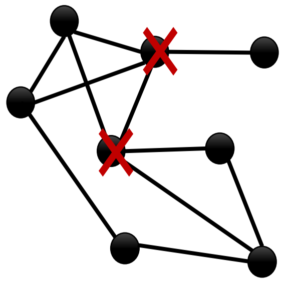
3. Has very good behavior with respect to separators
   - Construct "good" solution for G based on solutions for A, B



Example: Feedback vertex set

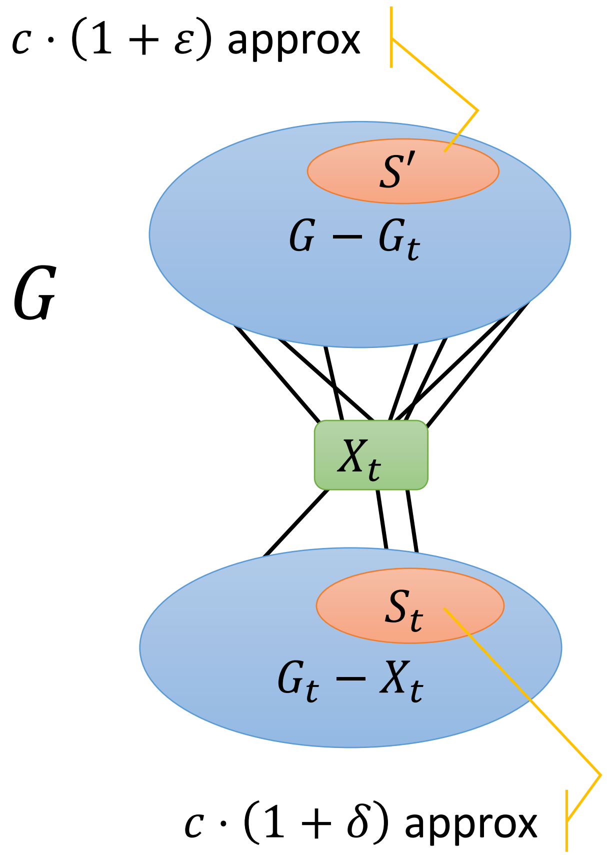
#Vertices we need to remove to make a graph acyclic

- Has kernel parameterized by k
  - Verify: 1-approximate
- Has 2-approximation
- Is otherwise well-behaved
  - If S is a FVS in $G - X$, then $S \cup X$ is a FVS in G



A general strategy (minimization)

1. If our graph has a small optimal solution
   - Apply $(1+\varepsilon)$-approximate kernel, obtain (G', k')
   - Feed (G', k') to oracle, obtain solution S'
   - Lift S' to a solution S of (G, k)
2. Else, find t such that $G_t - \mathrm{X_t}$ has solution size $\geq \frac{g(\ell+1)}{\delta}$
   - Use approximation algorithm
3. Obtain a $c(1+\delta)$-approximate solution S_t for $G_t - X_t$
   - See point 1, use $\delta < \varepsilon$
4. Recurse to obtain $c(1+\varepsilon)$-approximate solution S' of $G - G_t$
5. Combine S_t, S', and info about X_t to a solution in G



Approximate Turing kernels

Conclusions and future work

Summary

Problem	#Vertices in kernel
Independent Set	$O\left(\frac{\ell^2}{\varepsilon}\right)$
Vertex Cover	$O\left(\frac{\ell}{\varepsilon}\right)$
Connected Vertex Cover	$O\left(\left(\frac{\ell^2}{\varepsilon}\right)^{\left\lceil\frac{3+\varepsilon}{\varepsilon}\right\rceil}\right)$
Edge Clique Cover	$O\left(\frac{\ell^4}{\varepsilon}\right)$
Edge-Disjoint Triangle Packing	$O\left(\frac{\ell^2}{\varepsilon}\right)$
Vertex-Disjoint H-Packing for connected H	$O\left(\left(\frac{\ell}{\varepsilon}\right)^{\lvert V(H)\rvert-1}\right)$
Clique Cover	$O\left(\frac{\ell^4}{\varepsilon^2}\right)$
Feedback vertex Set	$O\left(\frac{\ell^2}{\varepsilon^2}\right)$
Edge Dominating Set	$O\left(\frac{\ell^2}{\varepsilon^2}\right)$

These problems parameterized by treewidth ℓ have $(1+\varepsilon)$-approximate Turing Kernels

- Assuming tree decomposition on input
- For all $0 < \varepsilon \leq 1$

Friendly problems have a $(1+\varepsilon)$-approximate Turing kernel with

$$h\left(\frac{\varepsilon}{3}, \varphi\left(\frac{6 \cdot \mathrm{g}(\ell+1)}{\varepsilon} + g(1), \ell\right) + \ell\right)$$

vertices

Summary

Problem	#Vertices in kernel
Independent Set	$O\left(\frac{\ell^2}{\varepsilon}\right)$
Vertex Cover	$O\left(\frac{\ell}{\varepsilon}\right)$
Connected Vertex Cover	$O\left(\left(\frac{\ell^2}{\varepsilon}\right)^{\left\lceil\frac{3+\varepsilon}{\varepsilon}\right\rceil}\right)$
Edge Clique Cover	$O\left(\frac{\ell^4}{\varepsilon}\right)$
Edge-Disjoint Triangle Packing	$O\left(\frac{\ell^2}{\varepsilon}\right)$
Vertex-Disjoint H-Packing for connected H	$O\left(\left(\frac{\ell}{\varepsilon}\right)^{\lvert V(H)\rvert-1}\right)$
Clique Cover	$O\left(\frac{\ell^4}{\varepsilon^2}\right)$
Feedback vertex Set	$O\left(\frac{\ell^2}{\varepsilon^2}\right)$
Edge Dominating Set	$O\left(\frac{\ell^2}{\varepsilon^2}\right)$

These problems parameterized by treewidth ℓ have $(1+\varepsilon)$-approximate Turing Kernels

- Assuming tree decomposition on input
- For all $0 < \varepsilon \leq 1$

Size of $(1+\varepsilon)$-approx. kernel

…blems have a $(1+\varepsilon)$-approximate …g kernel with

$$h\left(\frac{\varepsilon}{3}, \varphi\left(\frac{6 \cdot \mathrm{g}(\ell+1)}{\varepsilon} + g(1), \ell\right) + \ell\right)$$

vertices

Summary

Problem	**#Vertices in kernel**
INDEPENDENT SET	$O\left(\frac{\ell^2}{\varepsilon}\right)$
VERTEX COVER	$O\left(\frac{\ell}{\varepsilon}\right)$
CONNECTED VERTEX COVER	$O\left(\left(\frac{\ell^2}{\varepsilon}\right)^{\left\lceil\frac{3+\varepsilon}{\varepsilon}\right\rceil}\right)$
EDGE CLIQUE COVER	$O\left(\frac{\ell^4}{\varepsilon}\right)$
EDGE-DISJOINT TRIANGLE PACKING	$O\left(\frac{\ell^2}{\varepsilon}\right)$
VERTEX-DISJOINT H-PACKING FOR CONNECTED H	$O\left(\left(\frac{\ell}{\varepsilon}\right)^{\lvert V(H)\rvert-1}\right)$
CLIQUE COVER	$O\left(\frac{\ell^4}{\varepsilon^2}\right)$
FEEDBACK VERTEX SET	$O\left(\frac{\ell^2}{\varepsilon^2}\right)$
EDGE DOMINATING SET	$O\left(\frac{\ell^2}{\varepsilon^2}\right)$

These problems parameterized by treewidth ℓ have $(1+\varepsilon)$-approximate Turing Kernels

- Assuming tree decomposition on input
- For all $0 < \varepsilon \leq 1$

Size of $(1+\varepsilon)$-approx. kernel

…blems have a $(1+\varepsilon)$-approximate …g kernel with

$$h\left(\frac{\varepsilon}{3}, \varphi\left(\frac{6 \cdot g(\ell+1)}{\varepsilon} + g(1), \ell\right) + \ell\right)$$

Approximation factor of approximation algorithm

Summary

Problem	#Vertices in kernel
Independent Set	$O\left(\frac{\ell^2}{\varepsilon}\right)$
Vertex Cover	$O\left(\frac{\ell}{\varepsilon}\right)$
Connected Vertex Cover	$O\left(\left(\frac{\ell^2}{\varepsilon}\right)^{\left\lceil\frac{3+\varepsilon}{\varepsilon}\right\rceil}\right)$
Edge Clique Cover	$O\left(\frac{\ell^4}{\varepsilon}\right)$
Edge-Disjoint Triangle Packing	$O\left(\frac{\ell^2}{\varepsilon}\right)$
Vertex-Disjoint H-Packing for connected H	$O\left(\left(\frac{\ell}{\varepsilon}\right)^{\lvert V(H)\rvert-1}\right)$
Clique Cover	$O\left(\frac{\ell^4}{\varepsilon^2}\right)$
Feedback vertex Set	$O\left(\frac{\ell^2}{\varepsilon^2}\right)$
Edge Dominating Set	$O\left(\frac{\ell^2}{\varepsilon^2}\right)$

These problems parameterized by treewidth ℓ have $(1+\varepsilon)$-approximate Turing Kernels

- Assuming tree decomposition on input
- For all $0 < \varepsilon \leq 1$

…blem… …ximate …g kernel with

$$h\left(\frac{\varepsilon}{3}, \varphi\left(\frac{6\cdot g(\ell+1)}{\varepsilon} + g(1), \ell\right) + \ell\right)$$

Size of $(1+\varepsilon)$-approx. kernel

“Friendlyness” (usually $\ell+1$)

Approximation factor of approximation algorithm

Open questions

Approximate Turing kernels for other problems

- Many graph problems are not “friendly”
  - Constant-factor approximate Turing kernel for DOMINATING SET parameterized by treewidth ?
- Extend to other parameters
  - Other width parameters

More lower bounds

- Problems without $(1+\varepsilon)$-approximate (Turing) kernels

Open questions

Approximate Turing kernels for other problems

- Many graph problems are not “friendly”
  - Constant-factor approximate Turing kernel for DOMINATING SET parameterized by treewidth ?
- Extend to other parameters
  - Other width parameters

More lower bounds

- Problems without $(1 + \varepsilon)$-approximate (Turing) kernels

Thank you!