Polynomial kernels for hitting forbidden minors using constant treedepth modulators

Astrid Pieterse

TACO day 2018

joint work with Bart M. P. Jansen

April 5, 2018

NETWORKS
THENETWORKCENTER.NL

TU/e Technische Universiteit Eindhoven University of Technology

Introduction

Kernelization (meta) theorems

- Obtain polynomial kernels for a wide variety of problems
- Small parameter

Solution size may be large

- Focus on structural parameters

Introduction

Courcelle's Theorem

All problems that can be expressed in a certain logic are FPT when parameterized by treewidth

Obtain similar results for kernelization?

- Dominating Set has no polynomial kernel when parameterized by Vertex Cover [Dom et al. ICALP'09]
  - Large structural parameter
- But is easy to express in most types of logic

Focus on another very general problem

Introduction

Courcelle's Theorem

All problems that can be expressed in a certain logic are FPT when parameterized by treewidth

Obtain similar results for kernelization?

- Dominating Set has no polynomial kernel when parameterized by Vertex Cover [Dom et al. ICALP'09]
  - Large structural parameter
- But is easy to express in most types of logic

Focus on another very general problem

$\mathcal{F}$-Minor-Free Deletion

Let $\mathcal{F}$ be a set of connected graphs

Definition

Input	A graph G and integer k
Question	Does there exist $S \subseteq V(G)$ with $\lvert S\rvert \leq k$, such that no graph in $\mathcal{F}$ is a minor of $G - S$?

We sometimes say S breaks $\mathcal{F}$

Generalizes

- $\mathcal{F} = \{K_3\}$: Feedback Vertex Set
- $\mathcal{F} = \{K_2\}$: Vertex Cover
- $\mathcal{F} = \{K_{3,3}, K_5\}$: Making a graph planar by vertex-deletions

$\mathcal{F}$-Minor-Free Deletion

Let $\mathcal{F}$ be a set of connected graphs

Definition

Input	A graph G and integer k
Question	Does there exist $S \subseteq V(G)$ with $\lvert S\rvert \leq k$, such that no graph in $\mathcal{F}$ is a minor of $G - S$?

We sometimes say S breaks $\mathcal{F}$

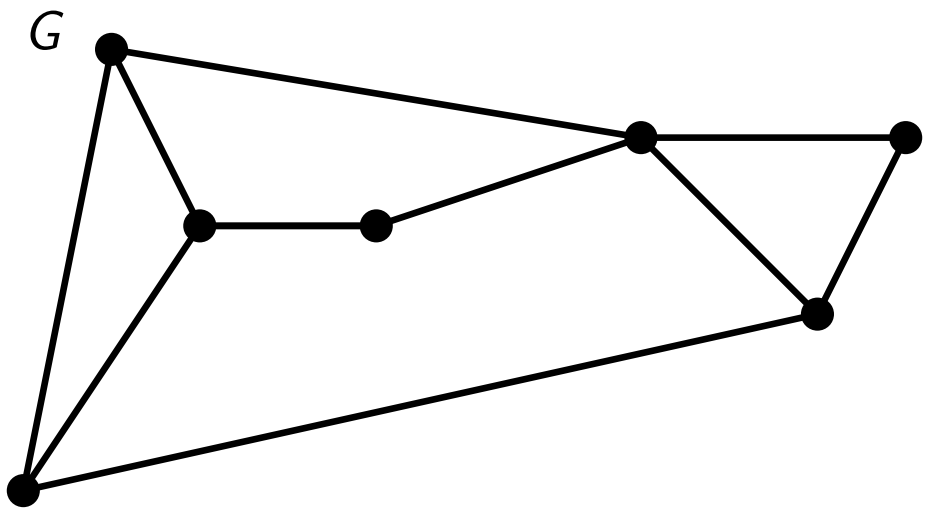
Generalizes

- $\mathcal{F} = \{K_3\}$: Feedback Vertex Set
- $\mathcal{F} = \{K_2\}$: Vertex Cover
- $\mathcal{F} = \{K_{3,3}, K_5\}$: Making a graph planar by vertex-deletions

Graph minors

H is a minor of G if H can be obtained by a sequence of

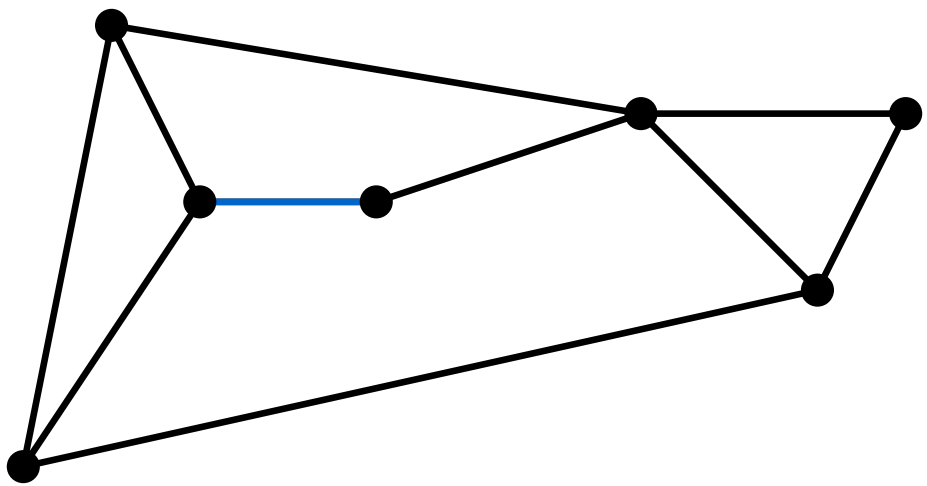
- Edge contractions
- Edge/vertex removals



Graph minors

H is a minor of G if H can be obtained by a sequence of

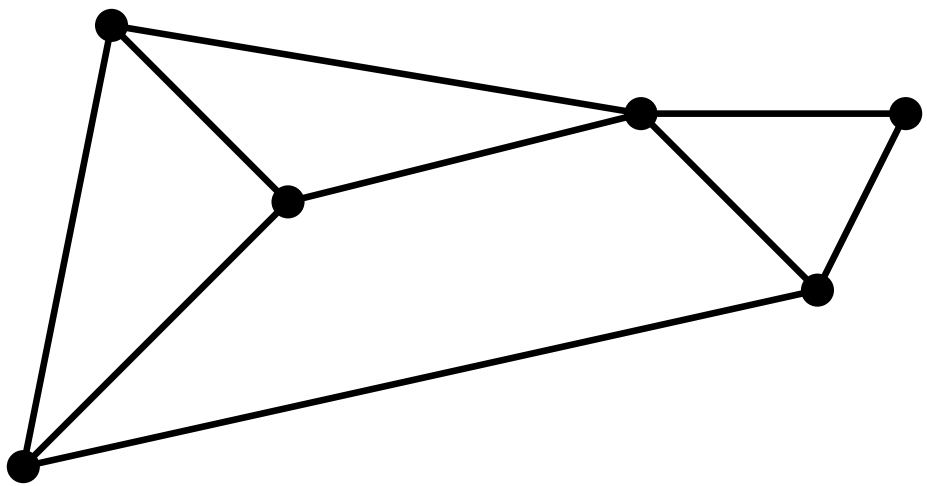
- Edge contractions
- Edge/vertex removals



Graph minors

H is a minor of G if H can be obtained by a sequence of

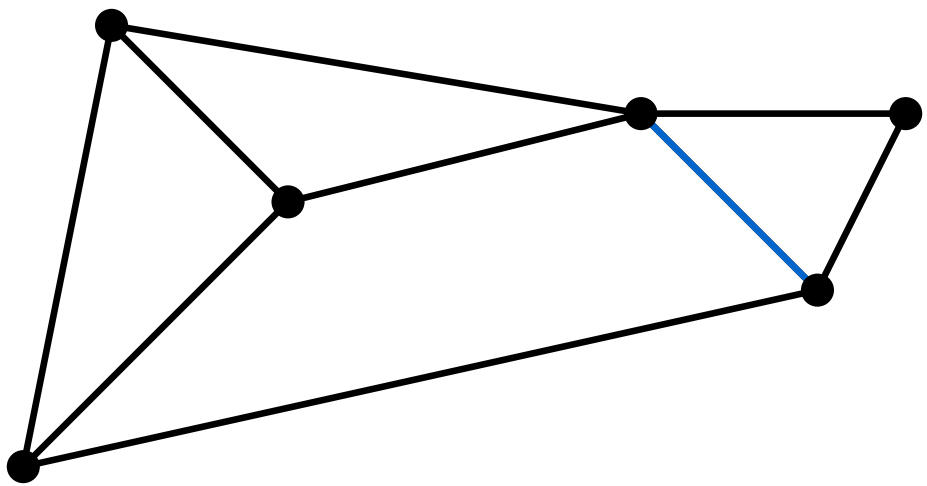
- Edge contractions
- Edge/vertex removals



Graph minors

H is a minor of G if H can be obtained by a sequence of

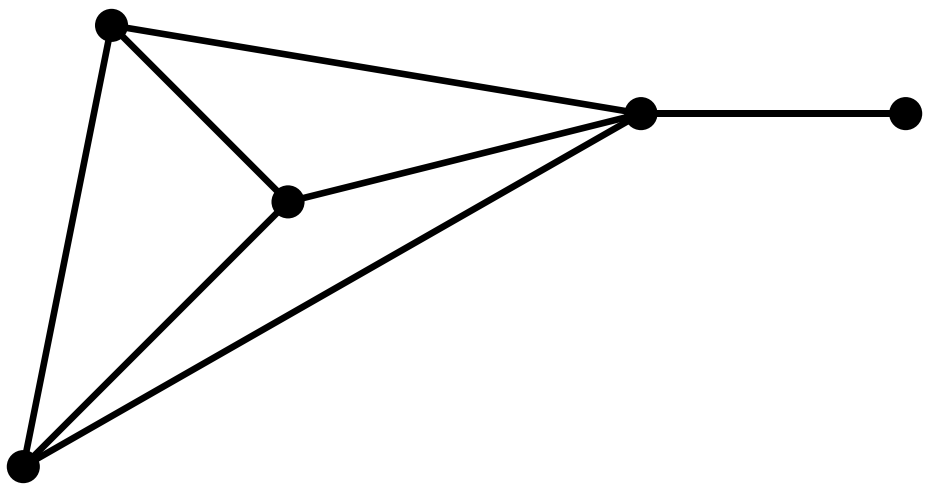
- Edge contractions
- Edge/vertex removals



Graph minors

H is a minor of G if H can be obtained by a sequence of

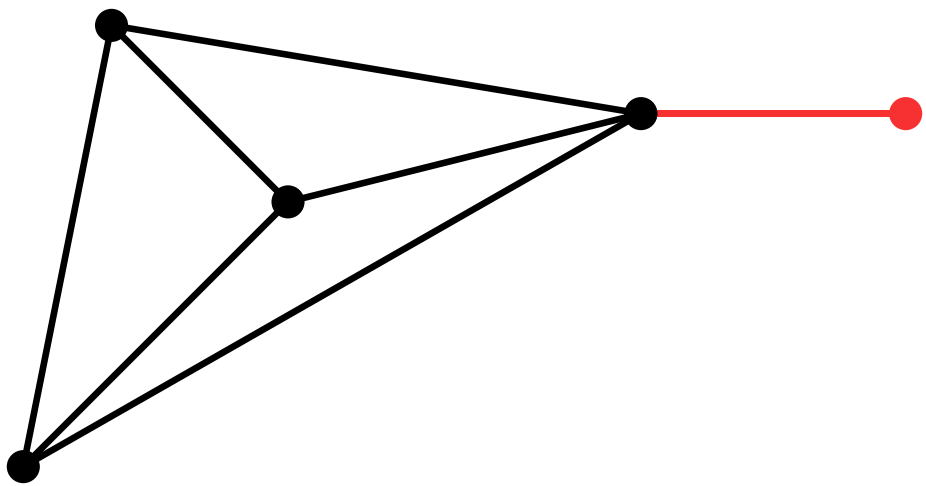
- Edge contractions
- Edge/vertex removals



Graph minors

H is a minor of G if H can be obtained by a sequence of

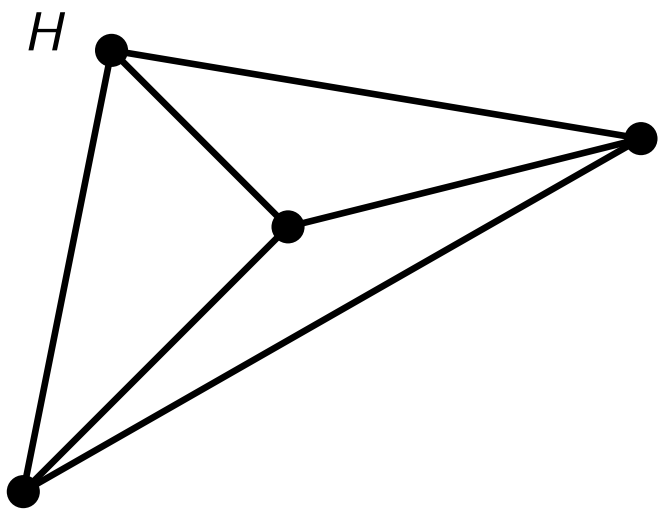
- Edge contractions
- Edge/vertex removals



Graph minors

H is a minor of G if H can be obtained by a sequence of

- Edge contractions
- Edge/vertex removals



Kernelization

Efficiently reduce the size of an input instance

- Resulting size depends on parameter value
- Provably small (polynomial in k)
- Provably correct

Polynomial kernel

Polynomial-time algorithm that, given instance (X, k), outputs (X', k')

- $|X'|$ and k' are bounded by $f(k)$
- X is a yes-instance if and only if X' is a yes-instance

Kernelization

Efficiently reduce the size of an input instance

- Resulting size depends on parameter value
- Provably small (polynomial in k)
- Provably correct

Polynomial kernel

Polynomial-time algorithm that, given instance (X, k), outputs (X', k')

- $|X'|$ and k' are bounded by $f(k)$
- X is a yes-instance if and only if X' is a yes-instance

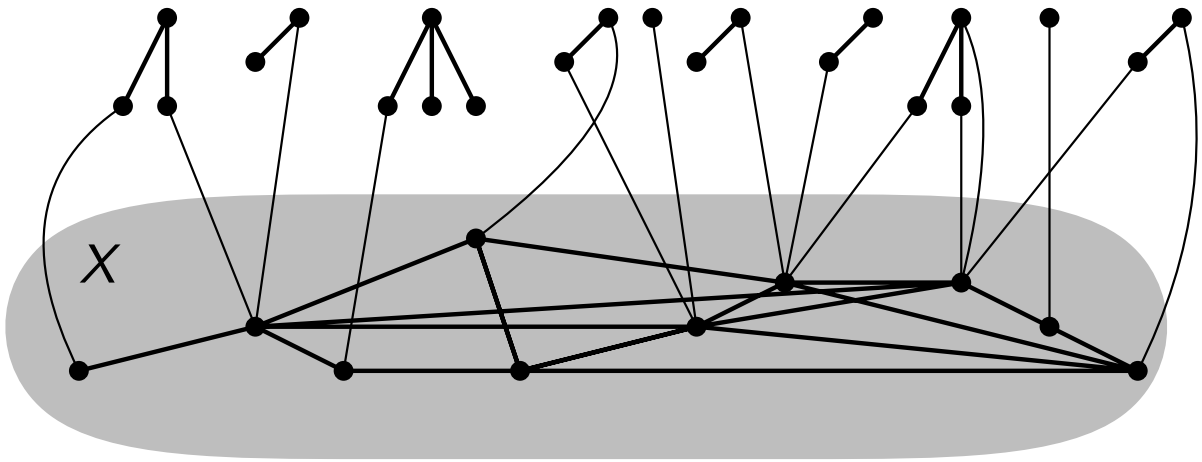
Parameter: modulator

Many problems are easy for some simple graph class $\mathcal{G}$

- Trees, cliques, paths, independent sets, forests, ...

Complexity measure for graphs

- Number of vertices we need to remove until $G \in \mathcal{G}$
- Also called a modulator



Parameter: modulator to constant treedepth

X is a treedepth-η modulator when $\mathrm{td}(G - X) \leq \eta$

- η is considered a fixed constant

Parameter: $|X|$ for optimal X

$\mathcal{F}$-Minor-Free Deletion is easy on graphs of constant treedepth

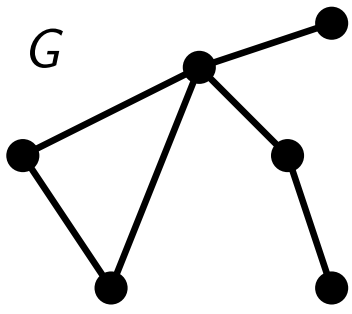
- Polynomial-time solvable

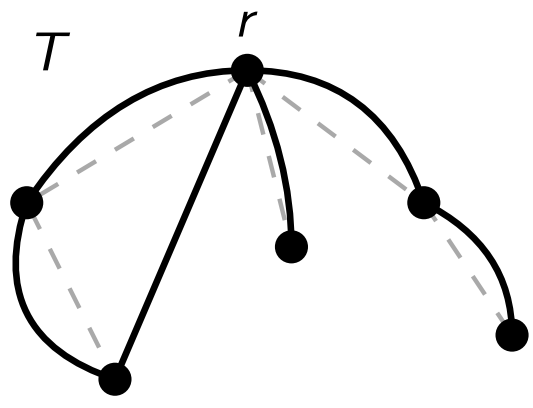
Treedepth

$\mathrm{td}(G)$ is the minimum depth of any treedepth decomposition

- Tree T on all vertices of G
- Any edge in G is between children/ancestors in T

Example of treedepth 3





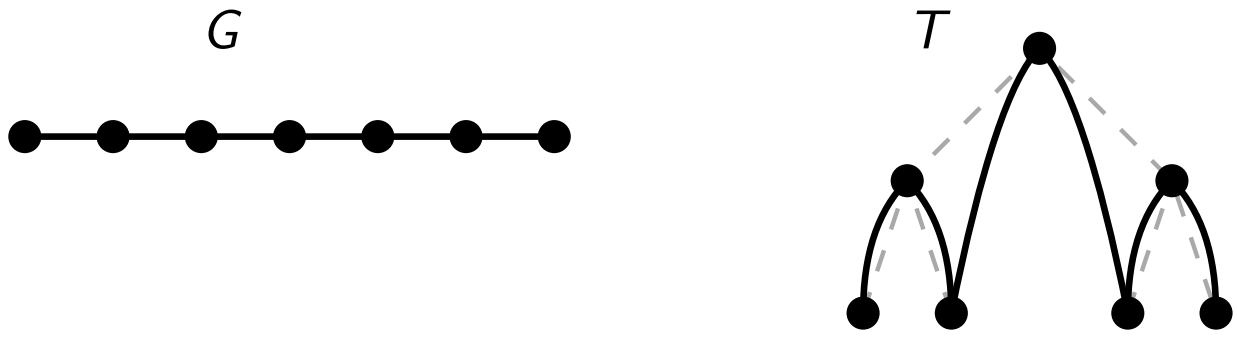
For any graph G, $\mathrm{tw}(G) \leq \mathrm{td}(G)$

Treedepth

$\mathrm{td}(G)$ is the minimum depth of any treedepth decomposition

- Tree T on all vertices of G
- Any edge in G is between children/ancestors in T

Treedepth of a path is logarithmic



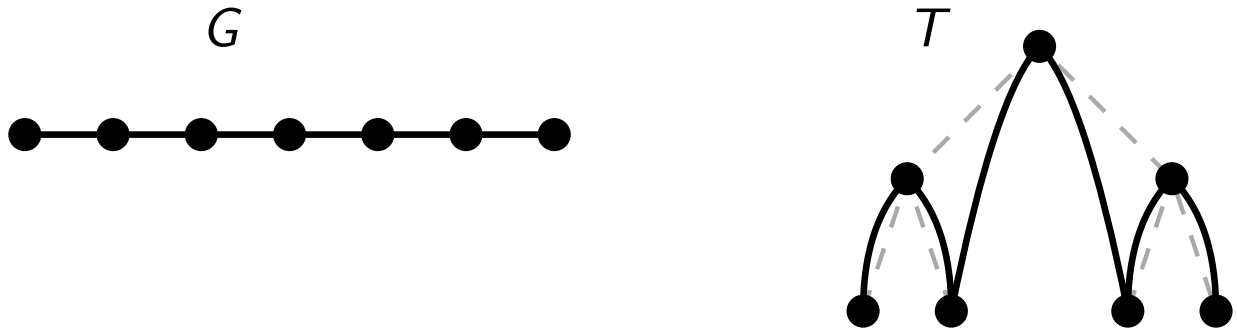
For any graph G, $\mathrm{tw}(G) \leq \mathrm{td}(G)$

Treedepth

$\mathrm{td}(G)$ is the minimum depth of any treedepth decomposition

- Tree T on all vertices of G
- Any edge in G is between children/ancestors in T

Treedepth of a path is logarithmic



For any graph G, $\mathrm{tw}(G) \leq \mathrm{td}(G)$

Previous work

Parameterized by solution size

- Whether a polynomial kernel exists is an open problem

Parameterized by treewidth

- Vertex Cover has no polynomial kernel [Bodlaender et.al. 2009]

Parameterized by pseudo forest modulator

- Feedback Vertex Set has a polynomial kernel [Jansen et al. 2014]

Parameterized by d-quasi forest modulator

- Vertex Cover has a polynomial kernel [Hols,Kratsch IPEC'17]

Previous work

Parameterized by treewidth-η modulator [Jansen,Bodlaender STACS'11]

- Polynomial kernel for Vertex Cover for $\eta = 1$
- No polynomial kernel for Vertex Cover for $\eta \geq 2$

Parameterized by treedepth-η modulator

- Polynomial kernel for Vertex Cover [Bougeret,Sau IPEC'17]
  - Kernelization of Feedback Vertex Set left open

Our results

Let $\mathcal{F}$ be a set of connected graphs

Theorem

$\mathcal{F}$-Minor-Free Deletion has a polynomial kernel parameterized by a treedepth-η modulator

- Kernel of size $O(|X|^{g(\eta,\mathcal{F})})$ for some function g
- Resolves the question about FVS

Lower bound

Vertex Cover parameterized by a treedepth-η modulator has no kernel of size $O(|X|^{2^{\eta-4}-\varepsilon})$, unless $NP \subseteq coNP/poly$

- g is exponential in η, and this cannot be avoided

Our results

Let $\mathcal{F}$ be a set of connected graphs

Theorem

$\mathcal{F}$-Minor-Free Deletion has a polynomial kernel parameterized by a treedepth-η modulator

- Kernel of size $O(|X|^{g(\eta,\mathcal{F})})$ for some function g
- Resolves the question about FVS

Lower bound

Vertex Cover parameterized by a treedepth-η modulator has no kernel of size $O(|X|^{2^{\eta-4}-\varepsilon})$, unless $NP \subseteq coNP/poly$

- g is exponential in η, and this cannot be avoided

Kernel for $\mathcal{F}$-Minor-Free Deletion

Kernel: general idea

Given G with modulator X

- If X not given, use approximation

First reduce the number of connected components of $G - X$

- Components could still be large

Then apply induction on η

Lemma

There is a polynomial-time algorithm that transforms G into induced subgraph G', and returns an integer Δ such that

- $\mathrm{opt}(G') + \Delta = \mathrm{opt}(G)$
- $G' - X$ has at most $|X|^{O(1)}$ connected components

Kernel: general idea

Given G with modulator X

- If X not given, use approximation

First reduce the number of connected components of $G - X$

- Components could still be large

Then apply induction on η

Lemma

There is a polynomial-time algorithm that transforms G into induced subgraph G', and returns an integer Δ such that

- $\mathrm{opt}(G') + \Delta = \mathrm{opt}(G)$
- $G' - X$ has at most $|X|^{O(1)}$ connected components

Kernel: using Lemma

Theorem

$\mathcal{F}$-Minor-Free Deletion has a polynomial kernel parameterized by a treedepth-η modulator

We do induction on η, using the Lemma

Base case

If $\eta = 1$, every connected component is a single vertex

- Given G and budget k, apply lemma to obtain G' and Δ
- Let the kernel be G' with budget $k - \Delta$
- G' has at most $|X| + |X|^{O(1)} = |X|^{O(1)}$ vertices

Kernel: using Lemma

Theorem

$\mathcal{F}$-Minor-Free Deletion has a polynomial kernel parameterized by a treedepth-η modulator

We do induction on η, using the Lemma

Base case

If $\eta = 1$, every connected component is a single vertex

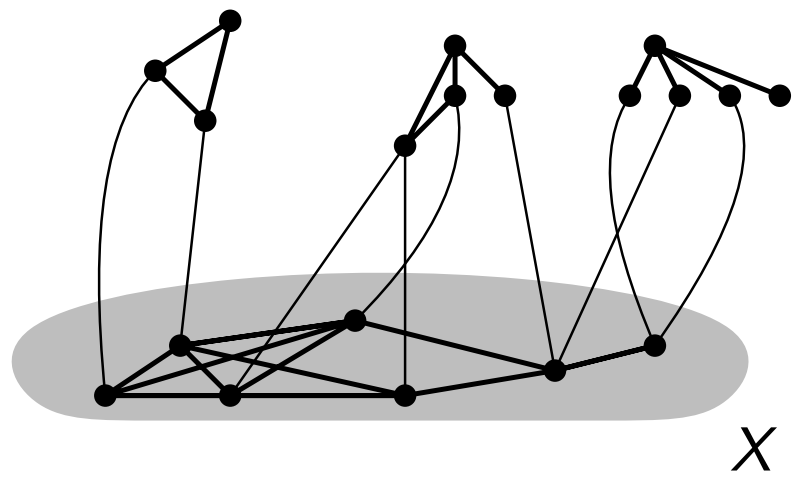
- Given G and budget k, apply lemma to obtain G' and Δ
- Let the kernel be G' with budget $k - \Delta$
- G' has at most $|X| + |X|^{O(1)} = |X|^{O(1)}$ vertices

Kernel: using Lemma

Step: $\eta > 1$

Apply lemma, find G' with only few components in $G' - X$

- For each component of $G' - X$, select root vertex r
  - Can be found efficiently

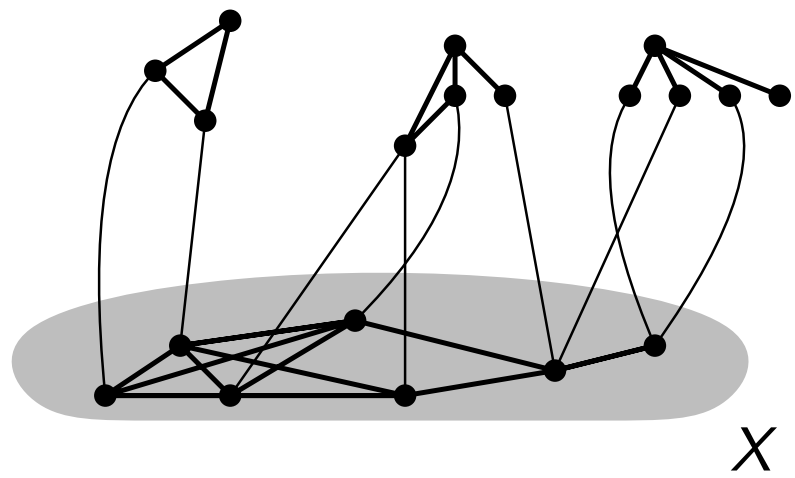


Kernel: using Lemma

Step: $\eta > 1$

Apply lemma, find G' with only few components in $G' - X$

- For each component of $G' - X$, select root vertex r
  - Can be found efficiently

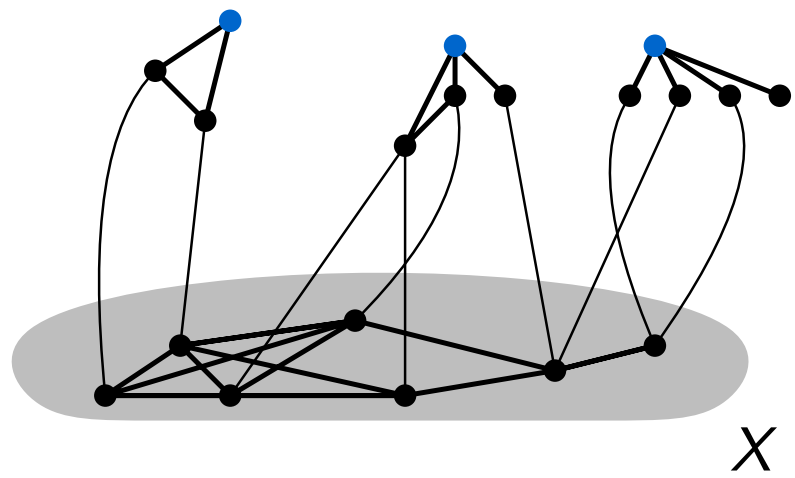


Kernel: using Lemma

Step: $\eta > 1$

Apply lemma, find G' with only few components in $G' - X$

- For each component of $G' - X$, select root vertex r
  - Can be found efficiently
- Add r to X, obtain X'

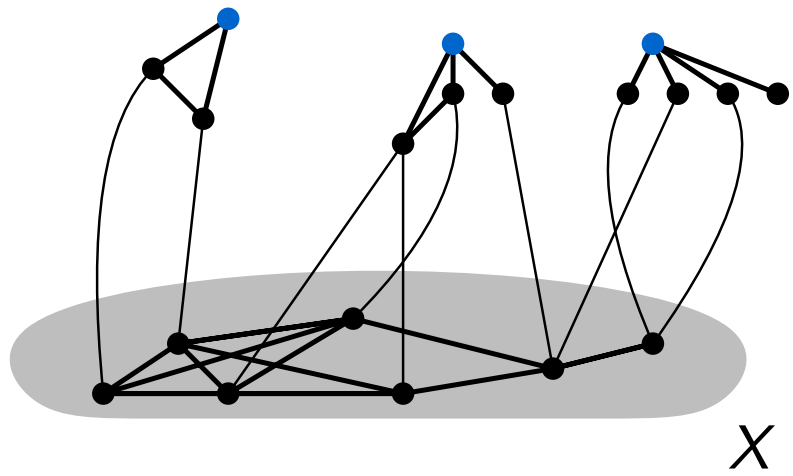


Kernel: using Lemma

Step: $\eta > 1$

Apply lemma, find G' with only few components in $G' - X$

- For each component of $G' - X$, select root vertex r
  - Can be found efficiently
- Add r to X, obtain X'



Kernel: using Lemma

Step: $\eta > 1$

Apply lemma, find G' with only few components in $G' - X$

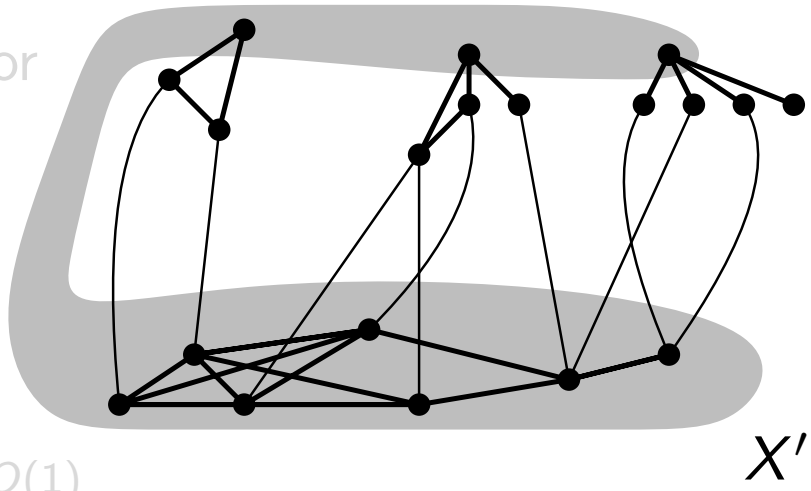
- For each component of $G' - X$, select root vertex r
  - Can be found efficiently
- Add r to X, obtain X'

Now X' is a treedepth-$(\eta - 1)$ modulator

- $|X'| = |X|^{O(1)}$

Apply IH with X'

- Gives kernel of size $|X'|^{O(1)} = |X|^{O(1)}$



Kernel: using Lemma

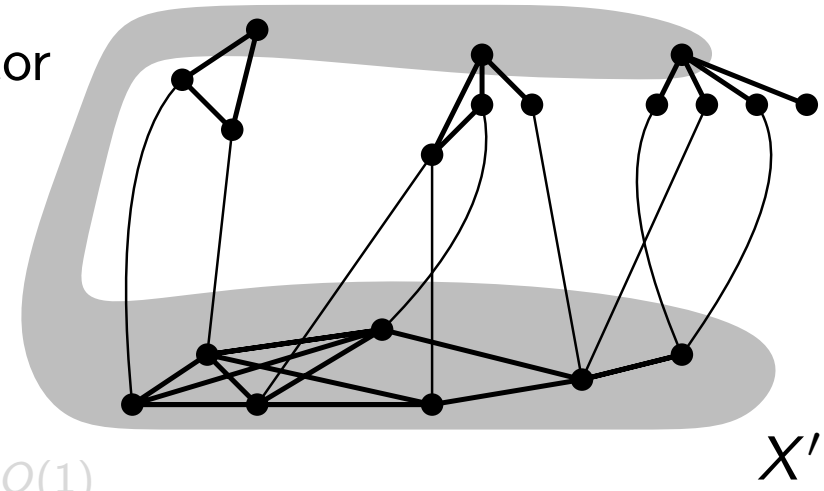
Step: $\eta > 1$

Apply lemma, find G' with only few components in $G' - X$

- For each component of $G' - X$, select root vertex r
  - Can be found efficiently
- Add r to X, obtain X'

Now X' is a treedepth-$(\eta - 1)$ modulator

- $|X'| = |X|^{O(1)}$



Apply IH with X'

- Gives kernel of size $|X'|^{O(1)} = |X|^{O(1)}$

Kernel: using Lemma

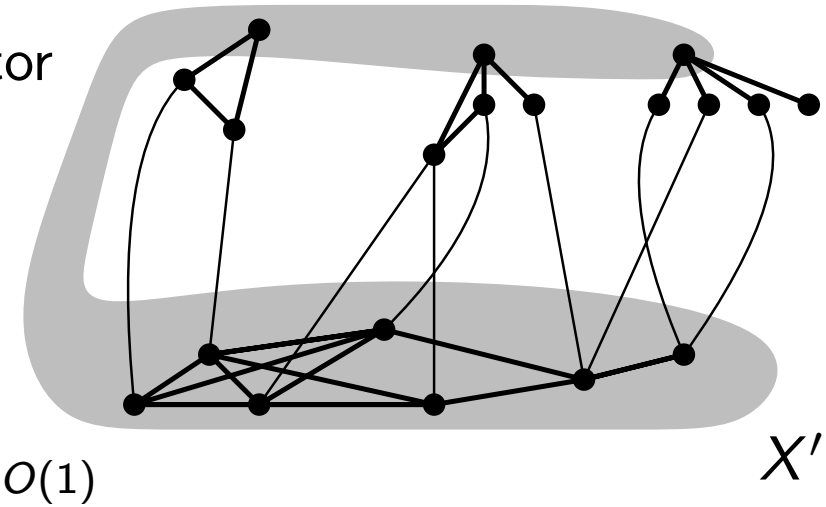
Step: $\eta > 1$

Apply lemma, find G' with only few components in $G' - X$

- For each component of $G' - X$, select root vertex r
  - Can be found efficiently
- Add r to X, obtain X'

Now X' is a treedepth-$(\eta - 1)$ modulator

- $|X'| = |X|^{O(1)}$



Apply IH with X'

- Gives kernel of size $|X'|^{O(1)} = |X|^{O(1)}$

Theorem

$\mathcal{F}$-Minor-free Deletion has a polynomial kernel parameterized by a treedepth-η modulator

using

Lemma

There is a polynomial-time algorithm that transforms G into induced subgraph G', and returns an integer Δ such that

- $\mathsf{opt}(G') + \Delta = \mathsf{opt}(G)$
- $G' - X$ has at most $|X|^{O(1)}$ connected components

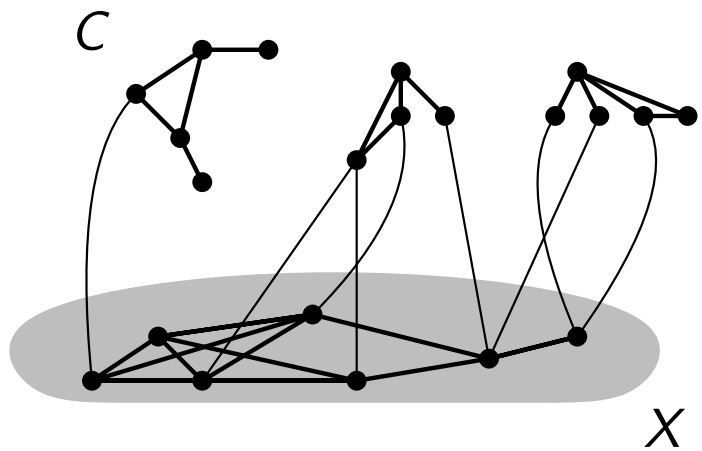
Towards a proof sketch of the Lemma

Example: $\mathcal{F} = \{K_3\}$.

- Feedback Vertex Set (abbreviated as FVS)

Try to remove connected components of $G - X$

- Exists an FVS in $G[C]$ disconnecting C from X
  - Remove C
  - Reduce budget by opt(C)



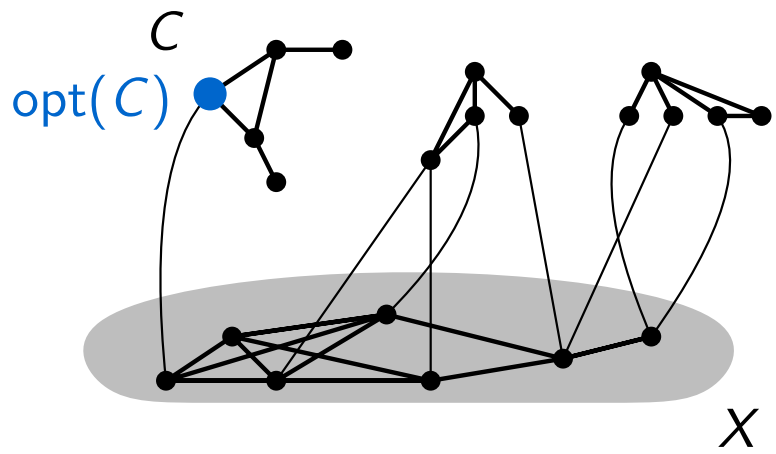
Towards a proof sketch of the Lemma

Example: $\mathcal{F} = \{K_3\}$.

- Feedback Vertex Set (abbreviated as FVS)

Try to remove connected components of $G - X$

- Exists an FVS in $G[C]$ disconnecting C from X
  - Remove C
  - Reduce budget by $\mathrm{opt}(C)$



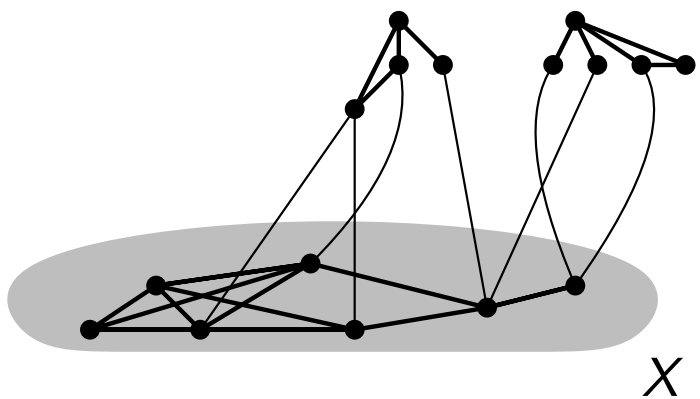
Towards a proof sketch of the Lemma

Example: $\mathcal{F} = \{K_3\}$.

- Feedback Vertex Set (abbreviated as FVS)

Try to remove connected components of $G - X$

- Exists an FVS in $G[C]$ disconnecting C from X
  - Remove C
  - Reduce budget by opt(C)



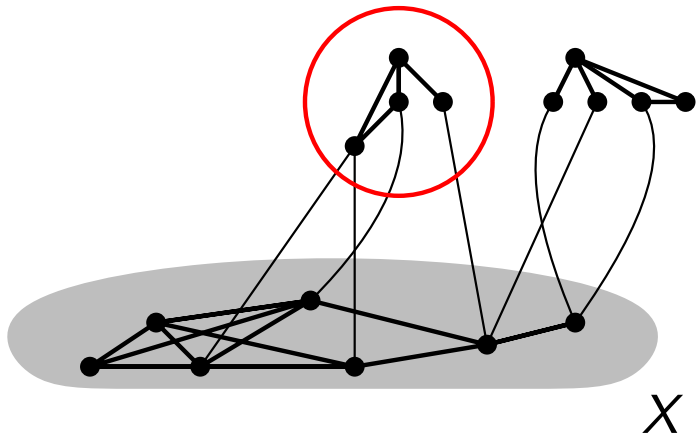
Towards a proof sketch of the Lemma

Example: $\mathcal{F} = \{K_3\}$.

- Feedback Vertex Set (abbreviated as FVS)

Try to remove connected components of $G - X$

- Exists an FVS in $G[C]$ disconnecting C from X
  - Remove C
  - Reduce budget by opt(C)



Solutions in G: property

Let S be an optimal Feedback Vertex Set in G

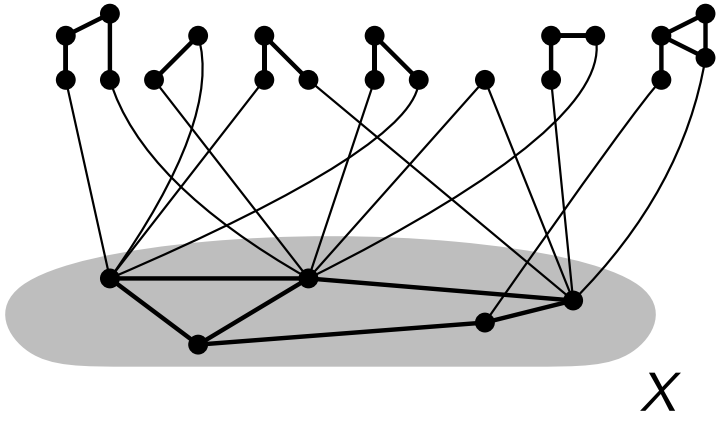
- Also works if S is an $\mathcal{F}$-deletion

Lemma

There exist $\leq |X|$ components C in $G - X$ such that

- S is not locally optimal in C

Any optimal FVS is locally optimal for most components!



Solutions in G: property

Let S be an optimal Feedback Vertex Set in G

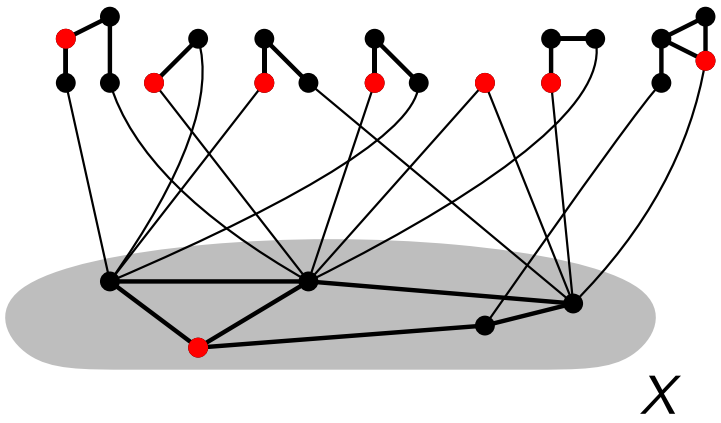
- Also works if S is an $\mathcal{F}$-deletion

Lemma

There exist $\leq |X|$ components C in $G - X$ such that

- S is not locally optimal in C

Any optimal FVS is locally optimal for most components!



Solutions in G: property

Let S be an optimal Feedback Vertex Set in G

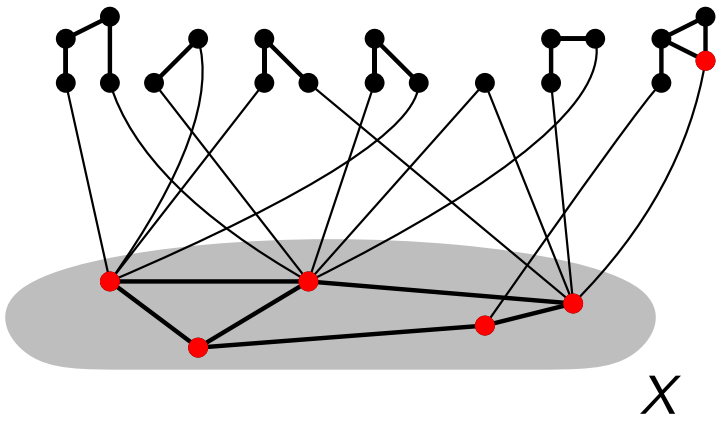
- Also works if S is an $\mathcal{F}$-deletion

Lemma

There exist $\leq |X|$ components C in $G - X$ such that

- S is not locally optimal in C

Any optimal FVS is locally optimal for most components!



Solutions in G: property

Let S be an optimal Feedback Vertex Set in G

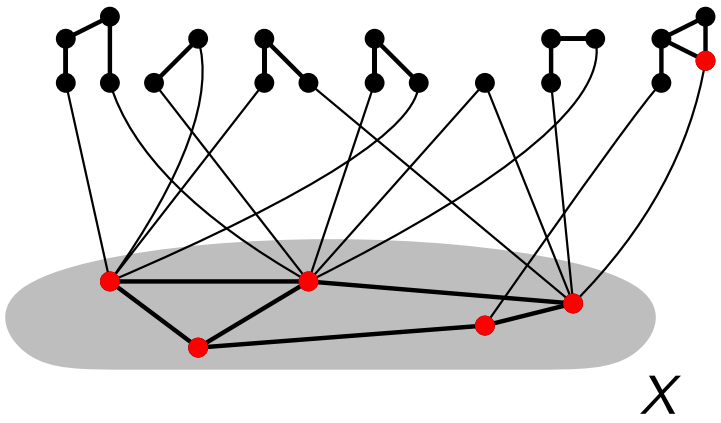
- Also works if S is an $\mathcal{F}$-deletion

Lemma

There exist $\leq |X|$ components C in $G - X$ such that

- S is not locally optimal in C

Any optimal FVS is locally optimal for most components!

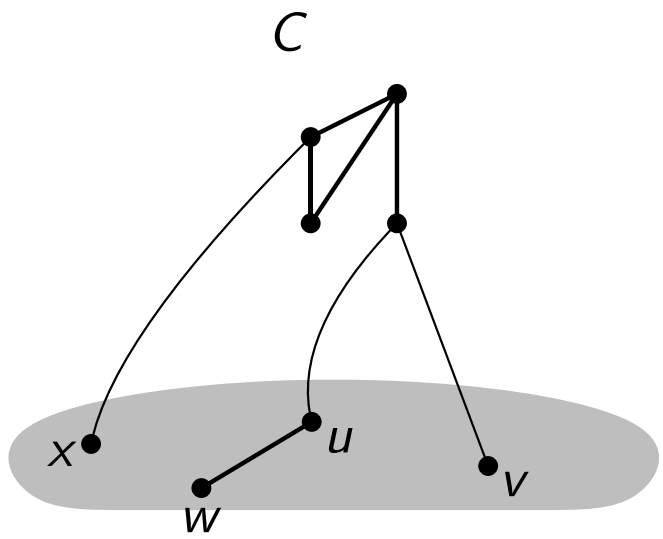


Removing components of $G - X$

Consider which $x \rightsquigarrow x'$-connections are made by C for $x, x' \in X$

- After removing some optimal FVS

No optimal FVS breaks $u \rightsquigarrow v$ in C

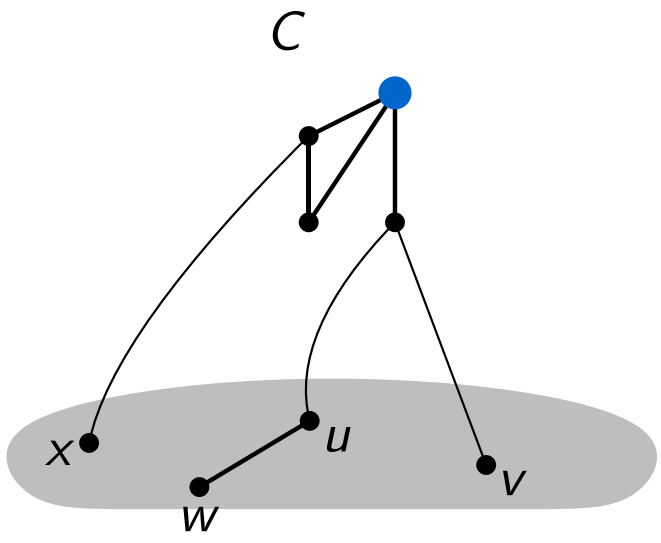


Removing components of $G - X$

Consider which $x \rightsquigarrow x'$-connections are made by C for $x, x' \in X$

- After removing some optimal FVS

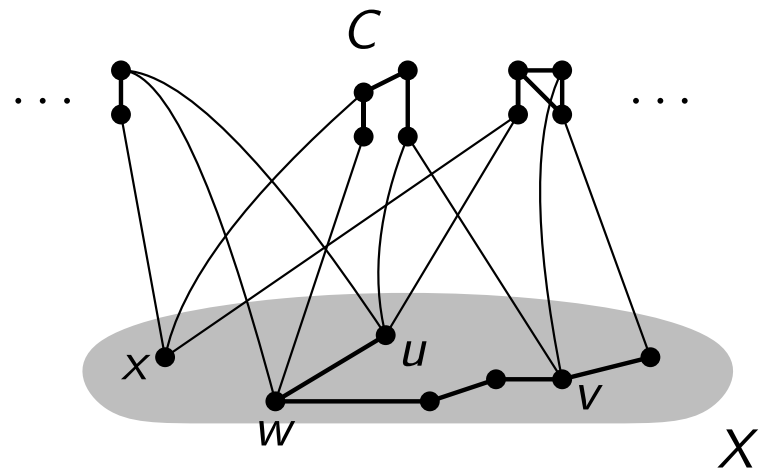
Some optimal FVS breaks $x \rightsquigarrow v$ and $x \rightsquigarrow u$ in C



Removing components of $G - X$

Suppose $\mathrm{opt}(C)$ never breaks $u \rightsquigarrow v$, $v \rightsquigarrow w$, $u \rightsquigarrow w$, $w \rightsquigarrow x$, ...

- Select a number of representative components
  - Mark $|X|^c$ other components that do not break $u \rightsquigarrow v$
  - Mark $|X|^c$ other components that do not break $v \rightsquigarrow w$
  - ...
- Remove C, decrease budget by $\mathrm{opt}(C)$

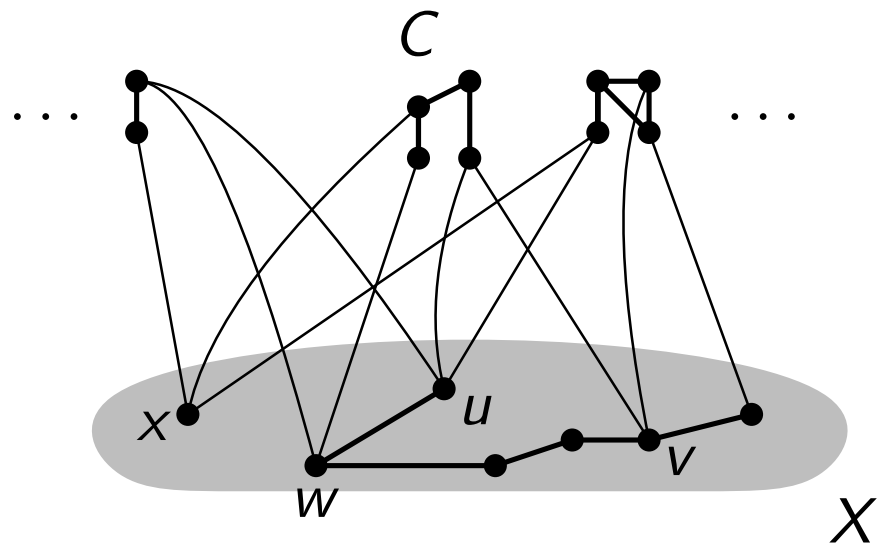


Removing components of $G - X$

Suppose C was removed by this rule

- If S is a FVS in G', $S \cup S_C$ is a FVS in G

G

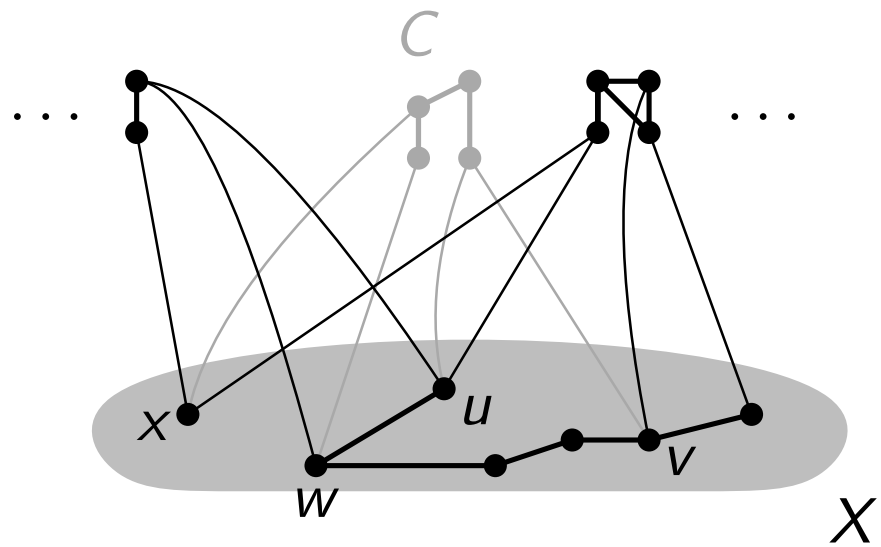


Removing components of $G - X$

Suppose C was removed by this rule

- If S is a FVS in G', $S \cup S_C$ is a FVS in G

G'

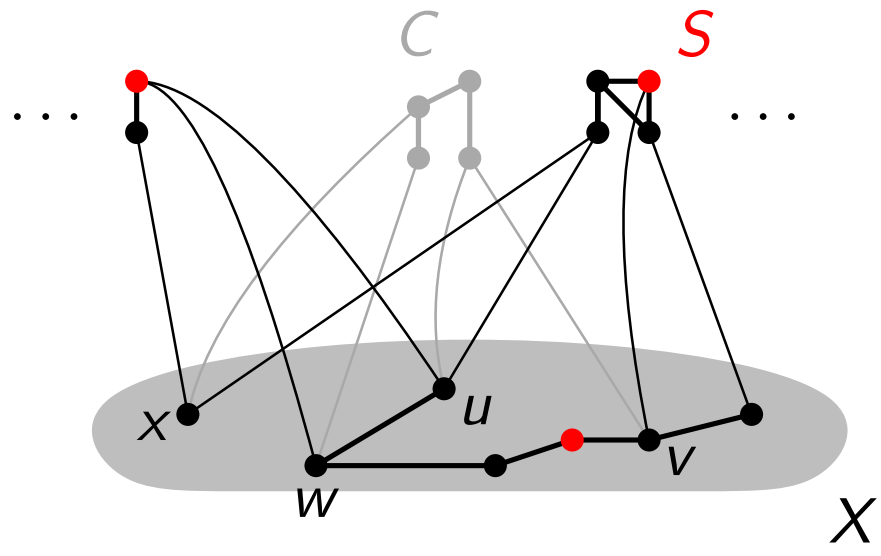


Removing components of $G - X$

Suppose C was removed by this rule

- If S is a FVS in G', $S \cup S_C$ is a FVS in G

G'

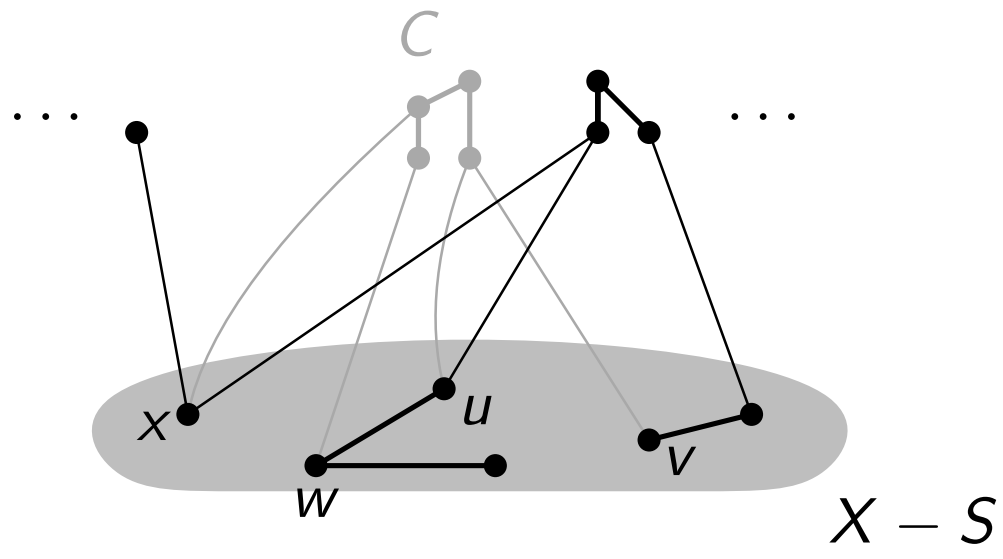


Removing components of $G - X$

Suppose C was removed by this rule

- If S is a FVS in G', $S \cup S_C$ is a FVS in G

$G' - S$

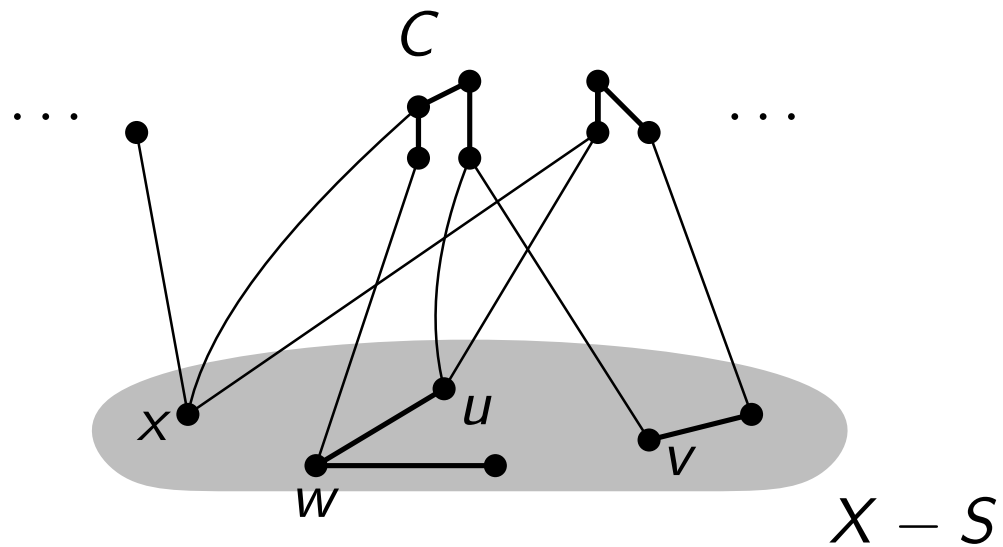


Removing components of $G - X$

Suppose C was removed by this rule

- If S is a FVS in G', $S \cup S_C$ is a FVS in G

$G - (S \cup S_C)$

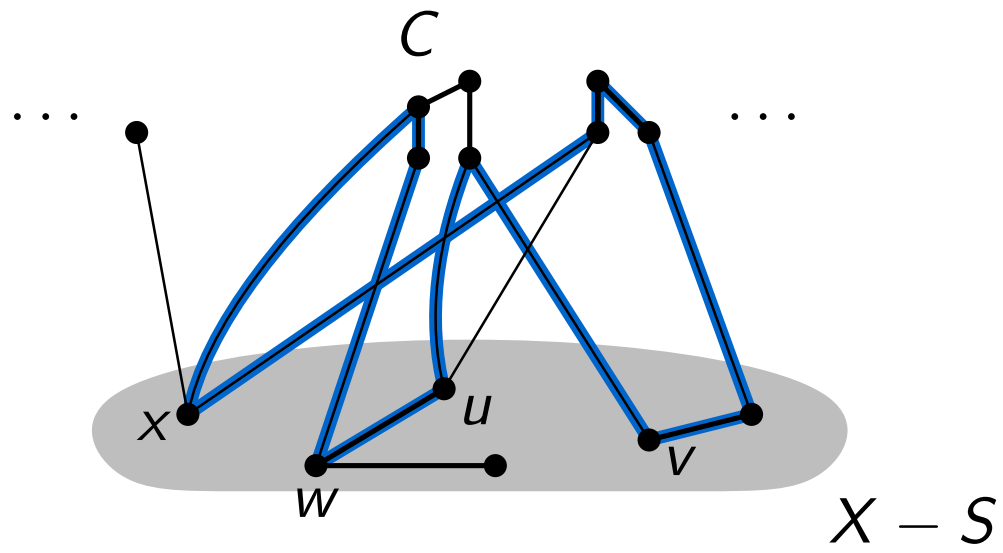


Removing components of $G - X$

Suppose C was removed by this rule

- If S is a FVS in G', $S \cup S_C$ is a FVS in G

$G - (S \cup S_C)$

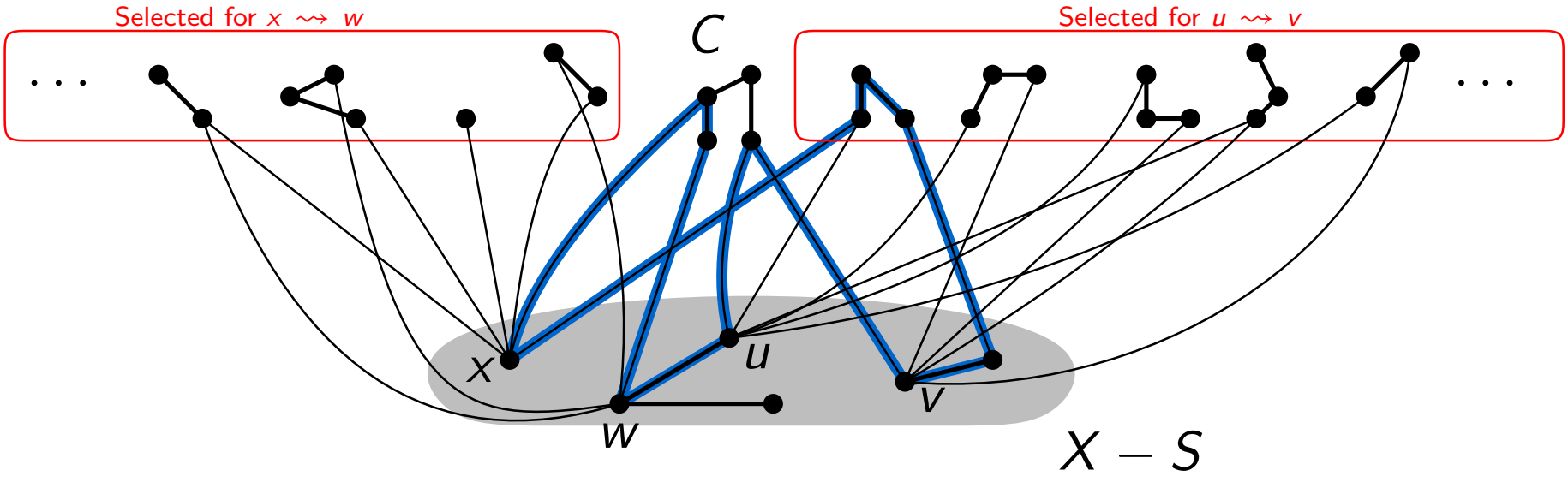


Removing components of $G - X$

Suppose C was removed by this rule

- If S is a FVS in G', $S \cup S_C$ is a FVS in G

$G - (S \cup S_C)$

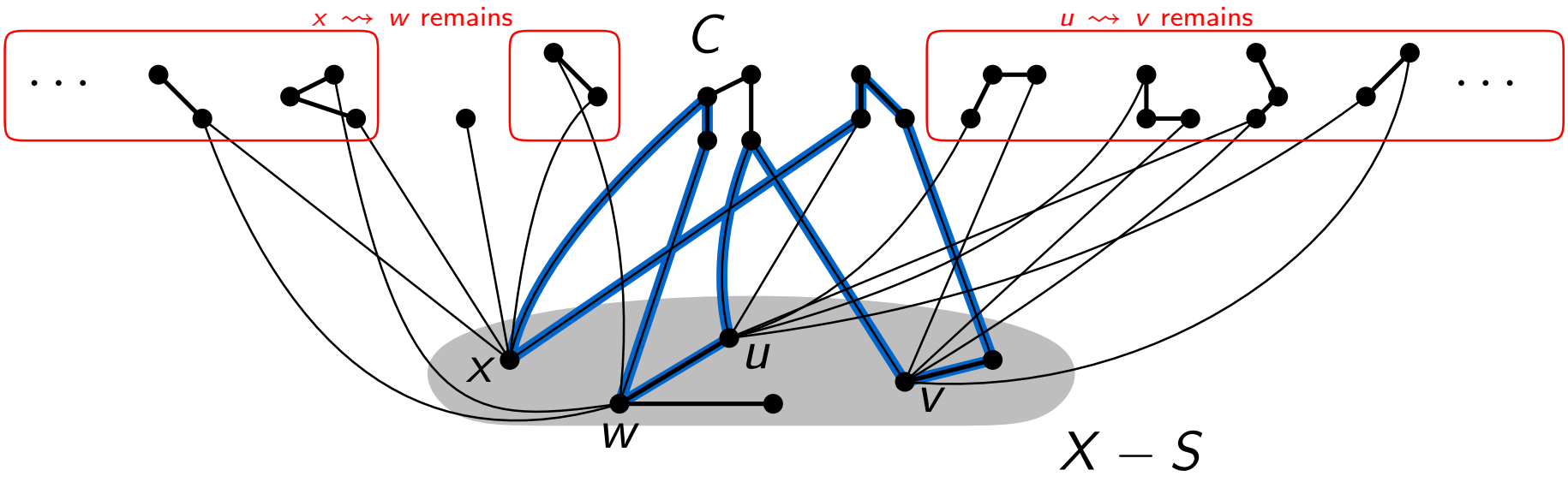


Removing components of $G - X$

Suppose C was removed by this rule

- If S is a FVS in G', $S \cup S_C$ is a FVS in G

$G - (S \cup S_C)$

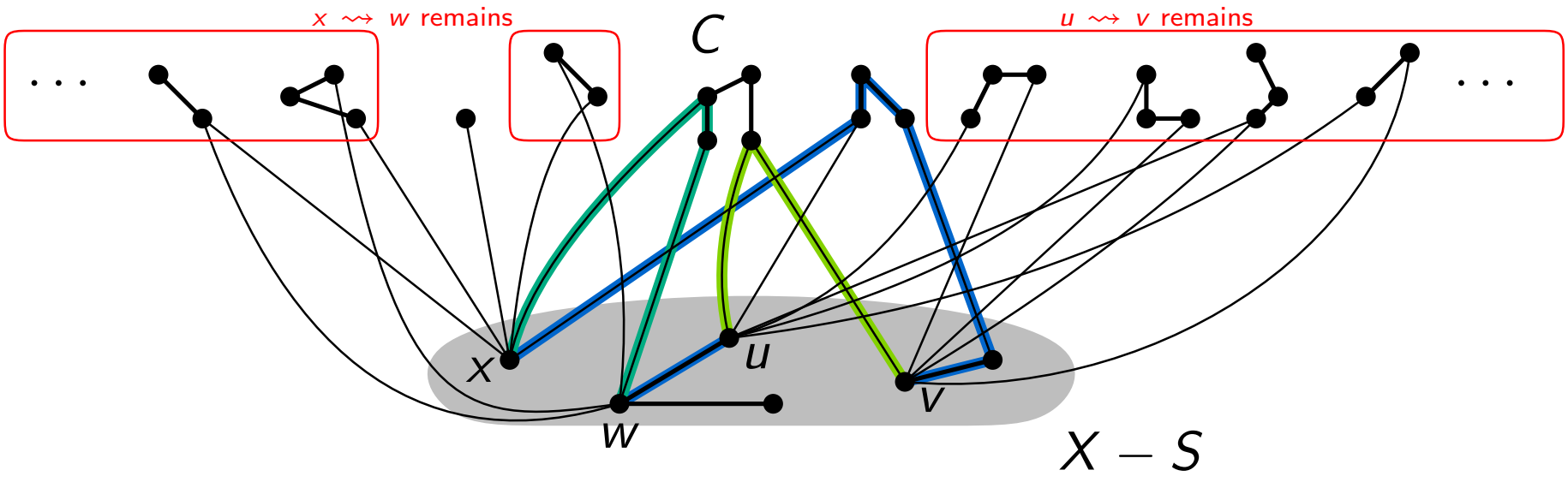


Removing components of $G - X$

Suppose C was removed by this rule

- If S is a FVS in G', $S \cup S_C$ is a FVS in G

$G - (S \cup S_C)$

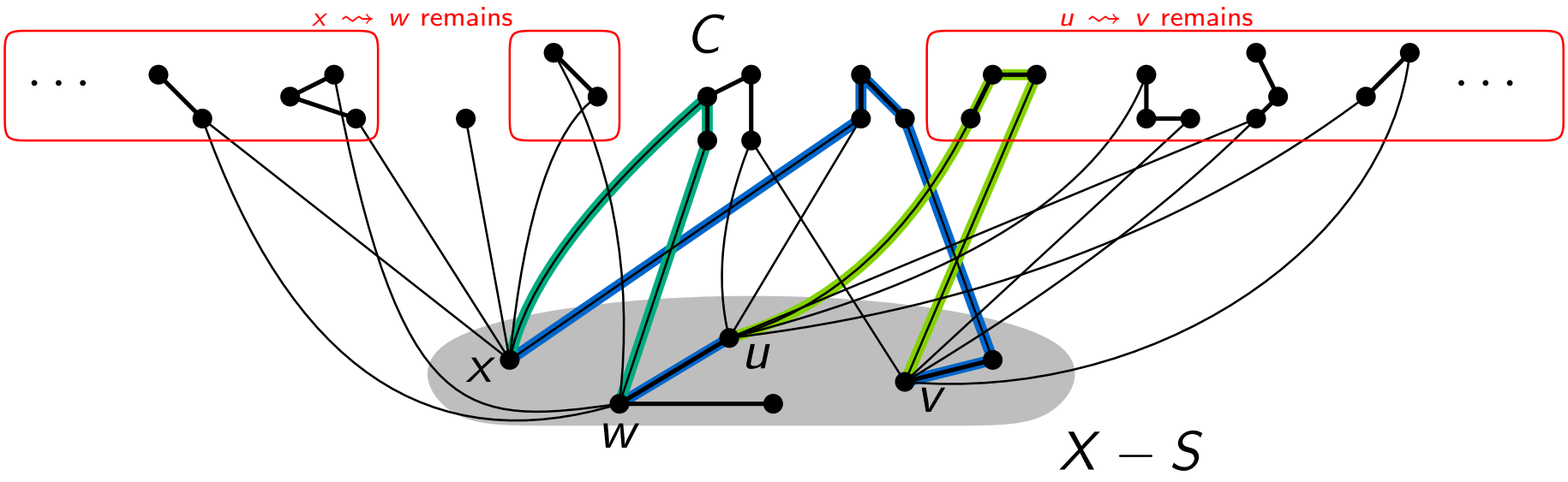


Removing components of $G - X$

Suppose C was removed by this rule

- If S is a FVS in G', $S \cup S_C$ is a FVS in G

$G - (S \cup S_C)$

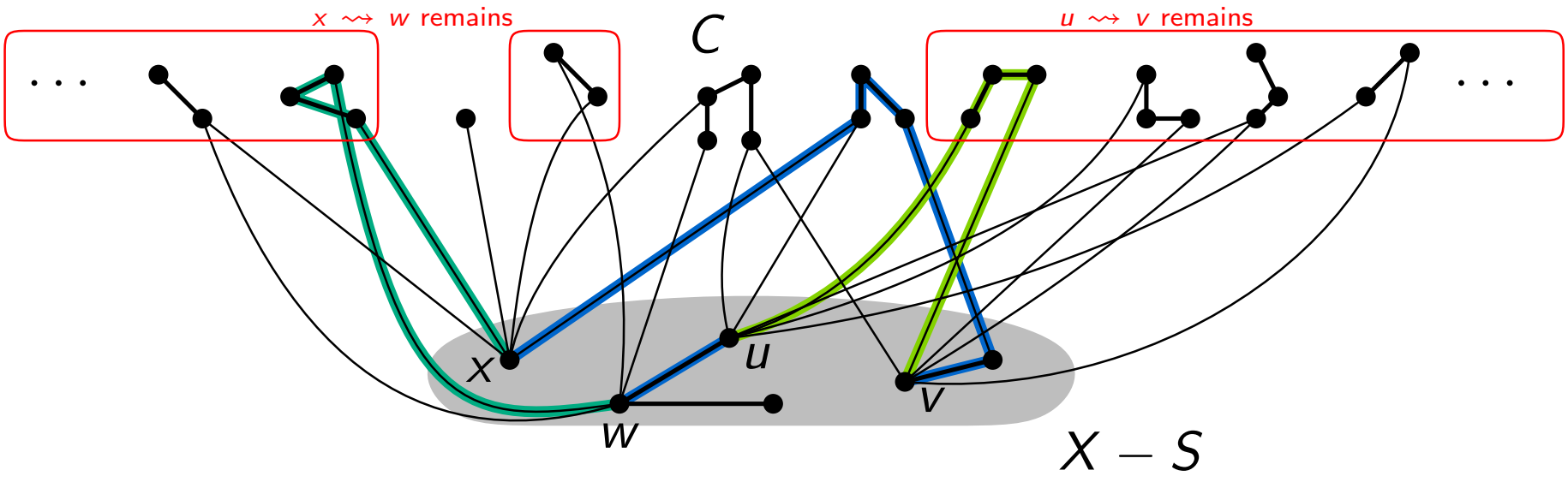


Removing components of $G - X$

Suppose C was removed by this rule

- If S is a FVS in G', $S \cup S_C$ is a FVS in G

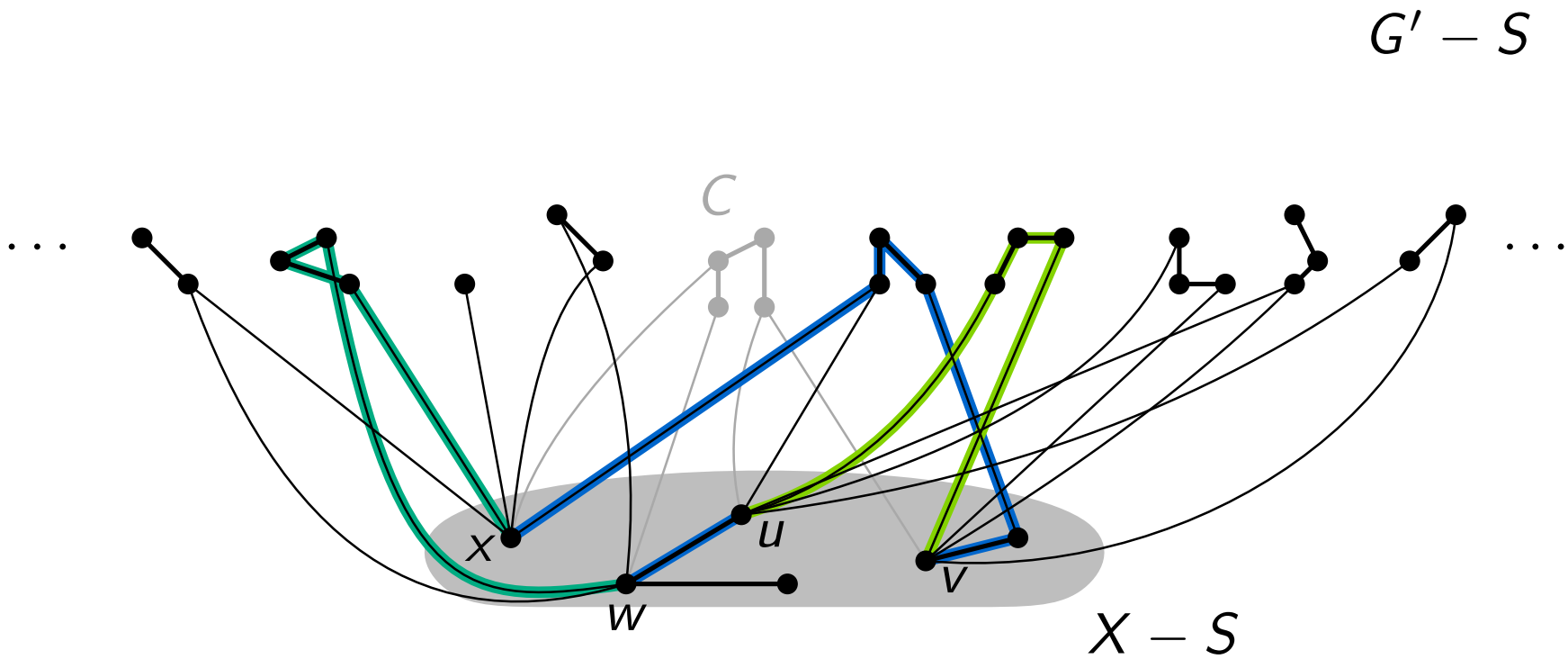
$G - (S \cup S_C)$



Removing components of $G - X$

Suppose C was removed by this rule

- If S is a FVS in G', $S \cup S_C$ is a FVS in G



Removing components of $G - X$: rules so far

Cases we handled

- Some optimal FVS in C breaks all connections of C to X
- Any optimal FVS in C leaves connections $u \rightsquigarrow v, v \rightsquigarrow w, \ldots$
  - and no others

Any other options to consider?

Removing components of $G - X$: rules so far

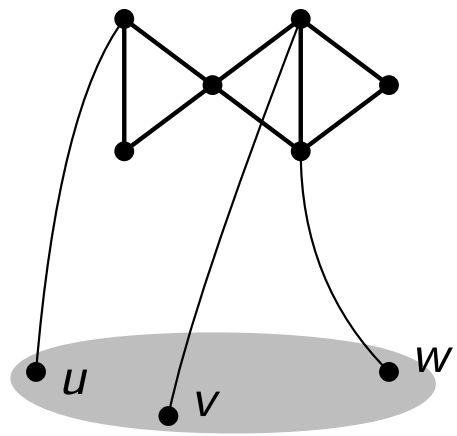
Cases we handled

- Some optimal FVS in C breaks all connections of C to X
- Any optimal FVS in C leaves connections $u \rightsquigarrow v, v \rightsquigarrow w, \ldots$
  - and no others

Any other options to consider?

Classifying components

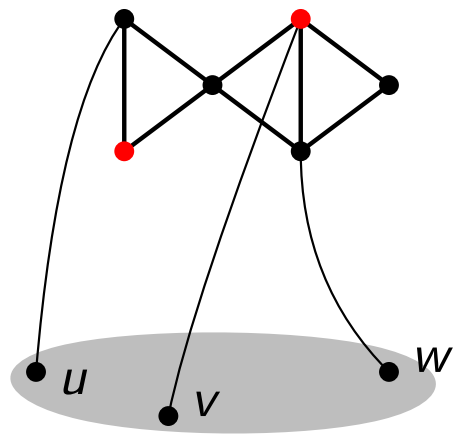
- Some solution may break $u \rightsquigarrow v$ and $v \rightsquigarrow w$
- Another breaks $u \rightsquigarrow w$ and $u \rightsquigarrow v$
- ...



For each set L of possible connections, indicate if there exists an optimal FVS in C that breaks all connections in L.

Classifying components

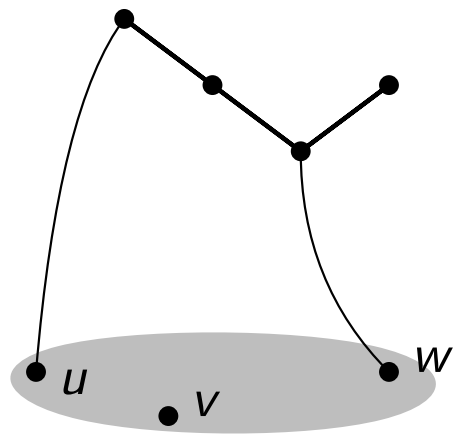
- Some solution may break $u \rightsquigarrow v$ and $v \rightsquigarrow w$
- Another breaks $u \rightsquigarrow w$ and $u \rightsquigarrow v$
- ...



For each set L of possible connections, indicate if there exists an optimal FVS in C that breaks all connections in L.

Classifying components

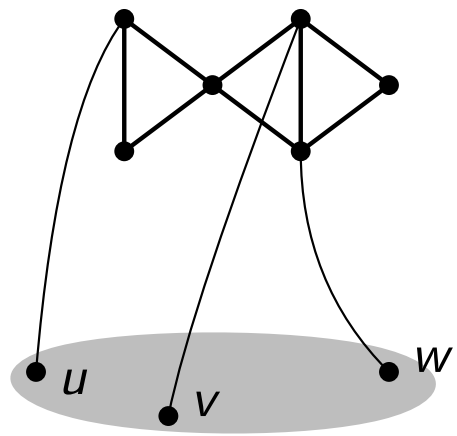
- Some solution may break $u \rightsquigarrow v$ and $v \rightsquigarrow w$
- Another breaks $u \rightsquigarrow w$ and $u \rightsquigarrow v$
- ...



For each set L of possible connections, indicate if there exists an optimal FVS in C that breaks all connections in L.

Classifying components

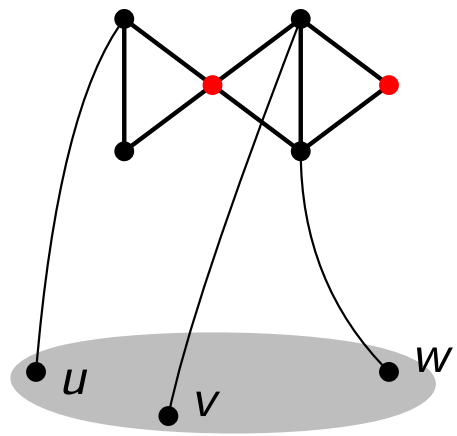
- Some solution may break $u \rightsquigarrow v$ and $v \rightsquigarrow w$
- Another breaks $u \rightsquigarrow w$ and $u \rightsquigarrow v$
- ...



For each set L of possible connections, indicate if there exists an optimal FVS in C that breaks all connections in L.

Classifying components

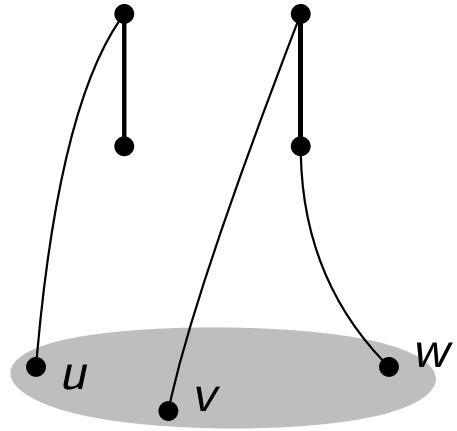
- Some solution may break $u \rightsquigarrow v$ and $v \rightsquigarrow w$
- Another breaks $u \rightsquigarrow w$ and $u \rightsquigarrow v$
- ...



For each set L of possible connections, indicate if there exists an optimal FVS in C that breaks all connections in L.

Classifying components

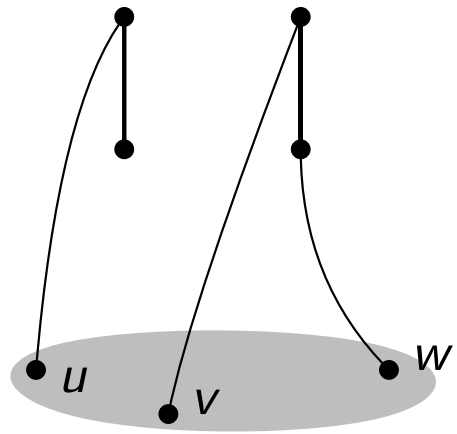
- Some solution may break $u \rightsquigarrow v$ and $v \rightsquigarrow w$
- Another breaks $u \rightsquigarrow w$ and $u \rightsquigarrow v$
- ...



For each set L of possible connections, indicate if there exists an optimal FVS in C that breaks all connections in L.

Classifying components

- Some solution may break $u \rightsquigarrow v$ and $v \rightsquigarrow w$
- Another breaks $u \rightsquigarrow w$ and $u \rightsquigarrow v$
- ...

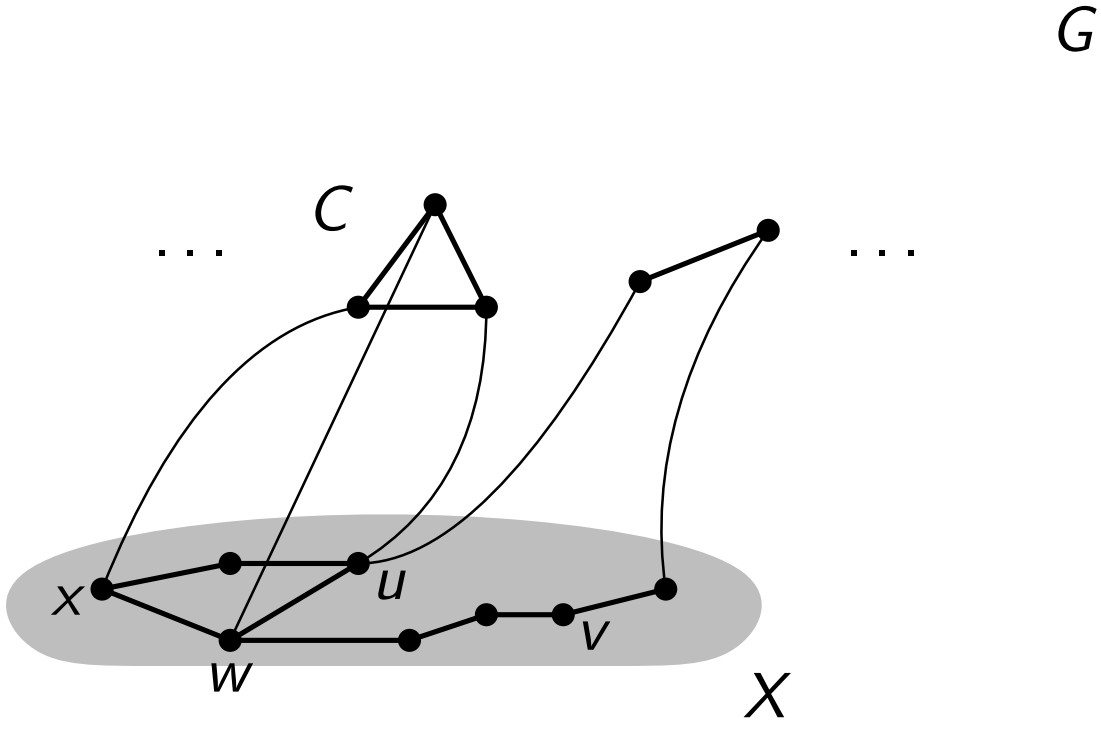


For each set L of possible connections, indicate if there exists an optimal FVS in C that breaks all connections in L.

Removing components of $G - X$: using classification

Let $L = \{u \rightsquigarrow x, u \rightsquigarrow w, x \rightsquigarrow w\}$

- Suppose no optimal FVS in C breaks all connections in L
  - but any proper subset of L can be broken
- If there are many other such components
  - Remove C, decrease budget by $\mathrm{opt}(C)$.

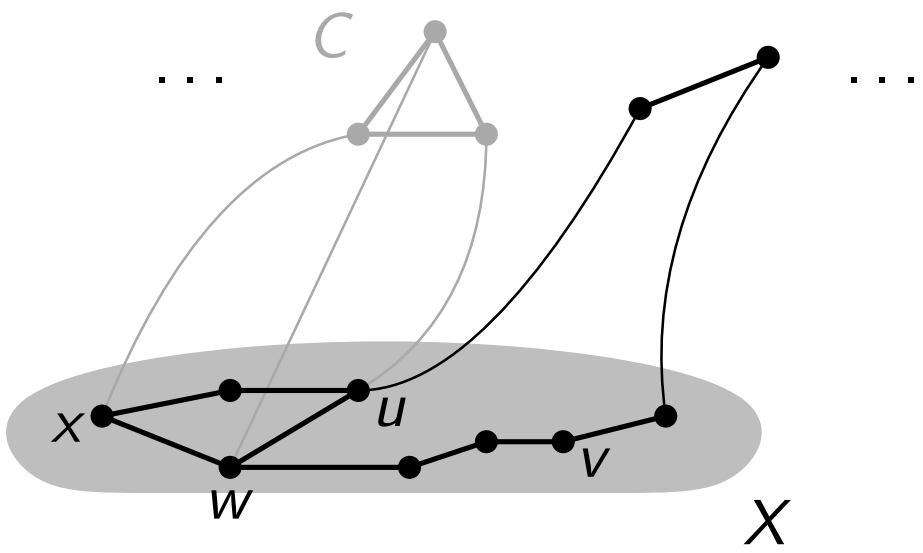


Removing components of $G - X$: using classification

Let $L = \{u \rightsquigarrow x, u \rightsquigarrow w, x \rightsquigarrow w\}$

- Suppose no optimal FVS in C breaks all connections in L
  - but any proper subset of L can be broken
- If there are many other such components
  - Remove C, decrease budget by $\mathrm{opt}(C)$.

G'

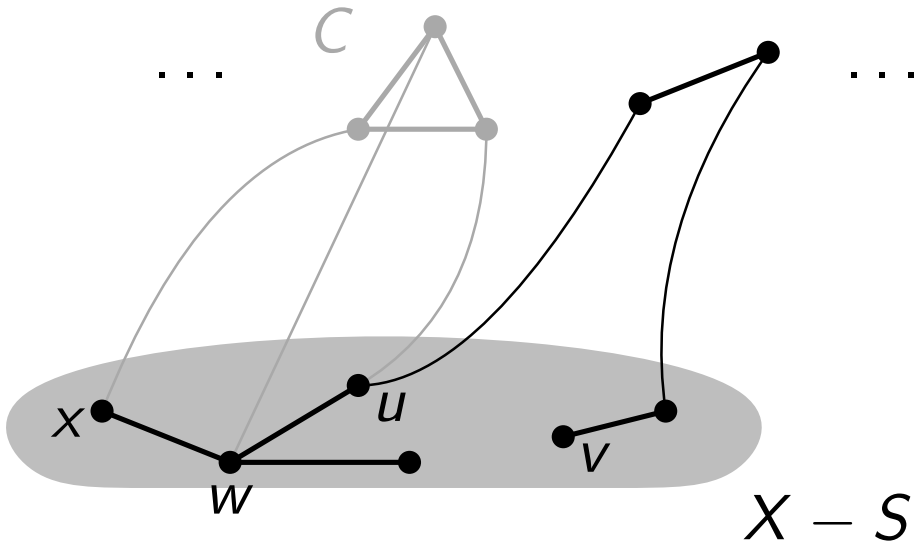


Removing components of $G - X$: using classification

Let $L = \{u \rightsquigarrow x, u \rightsquigarrow w, x \rightsquigarrow w\}$

- Suppose no optimal FVS in C breaks all connections in L
  - but any proper subset of L can be broken
- If there are many other such components
  - Remove C, decrease budget by $\mathrm{opt}(C)$.

$G' - S$



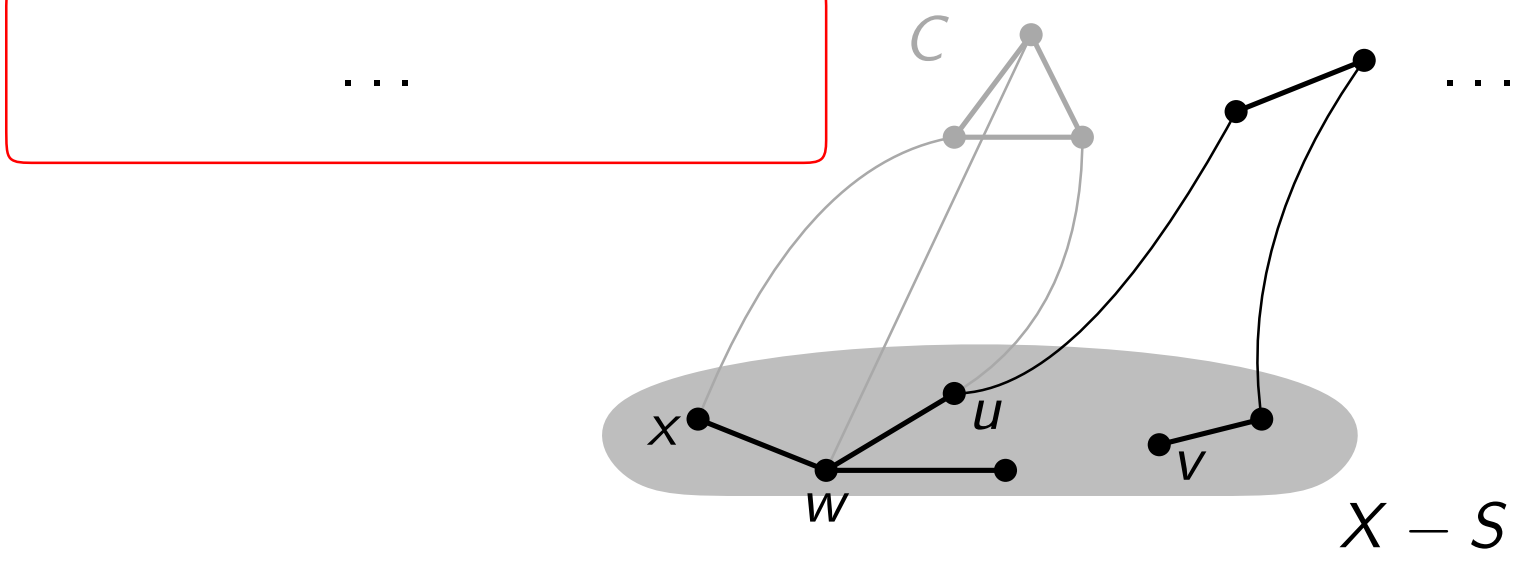
Removing components of $G - X$: using classification

Let $L = \{u \rightsquigarrow x, u \rightsquigarrow w, x \rightsquigarrow w\}$

- Suppose no optimal FVS in C breaks all connections in L
  - but any proper subset of L can be broken
- If there are many other such components
  - Remove C, decrease budget by $\mathrm{opt}(C)$.

$G' - S$

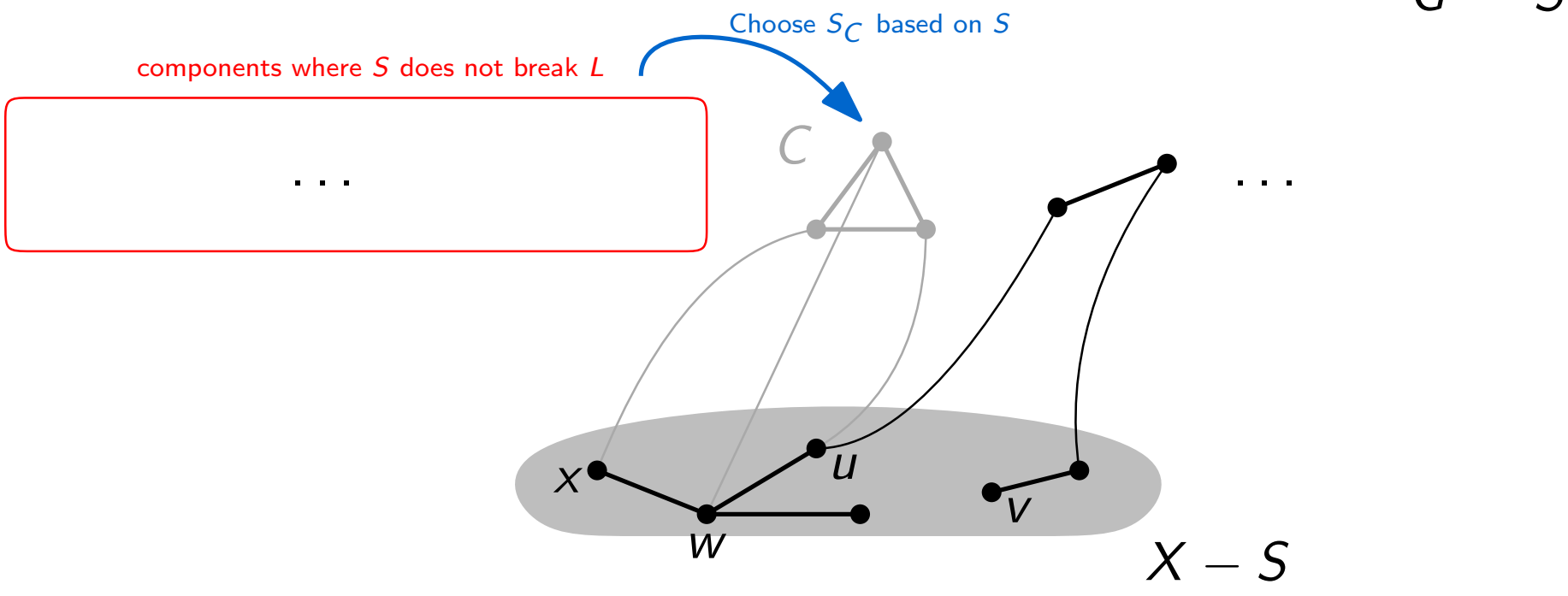
components where S does not break L



Removing components of $G - X$: using classification

Let $L = \{u \rightsquigarrow x, u \rightsquigarrow w, x \rightsquigarrow w\}$

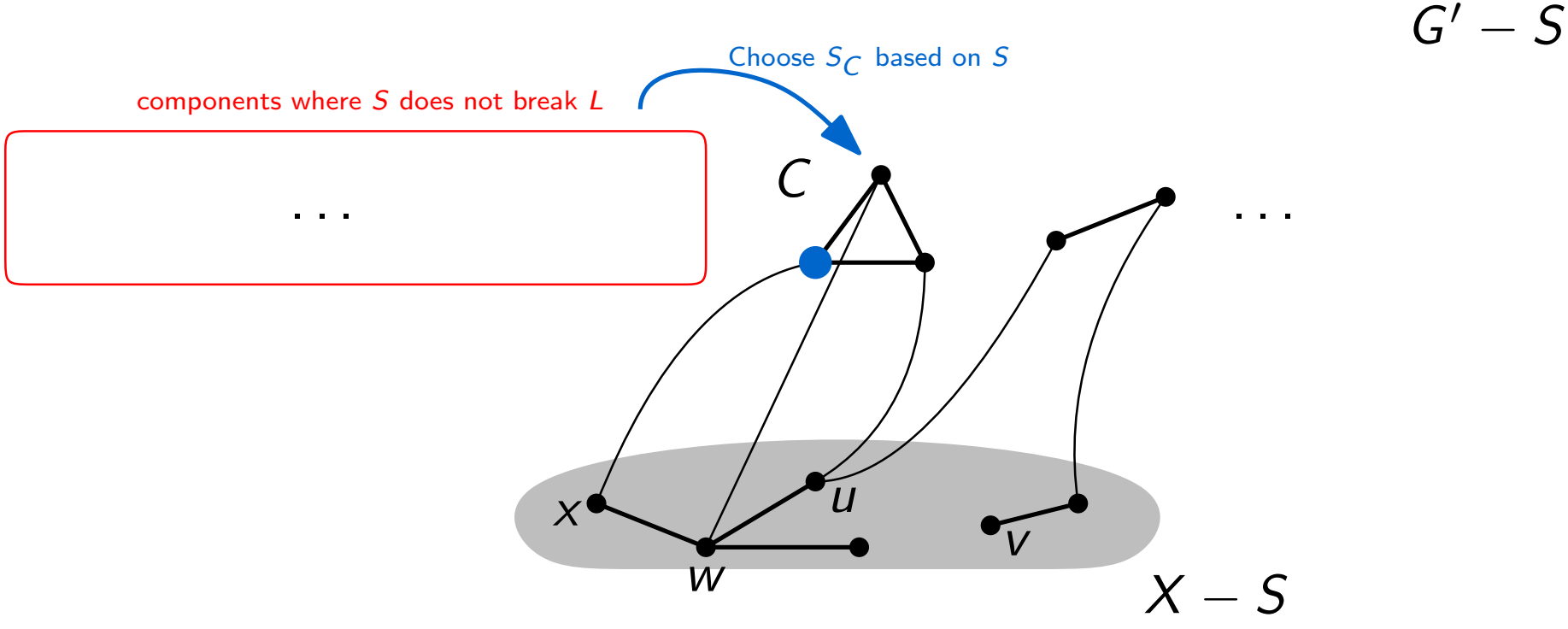
- Suppose no optimal FVS in C breaks all connections in L
  - but any proper subset of L can be broken
- If there are many other such components
  - Remove C, decrease budget by opt(C).



Removing components of $G - X$: using classification

Let $L = \{u \rightsquigarrow x, u \rightsquigarrow w, x \rightsquigarrow w\}$

- Suppose no optimal FVS in C breaks all connections in L
  - but any proper subset of L can be broken
- If there are many other such components
  - Remove C, decrease budget by opt(C).



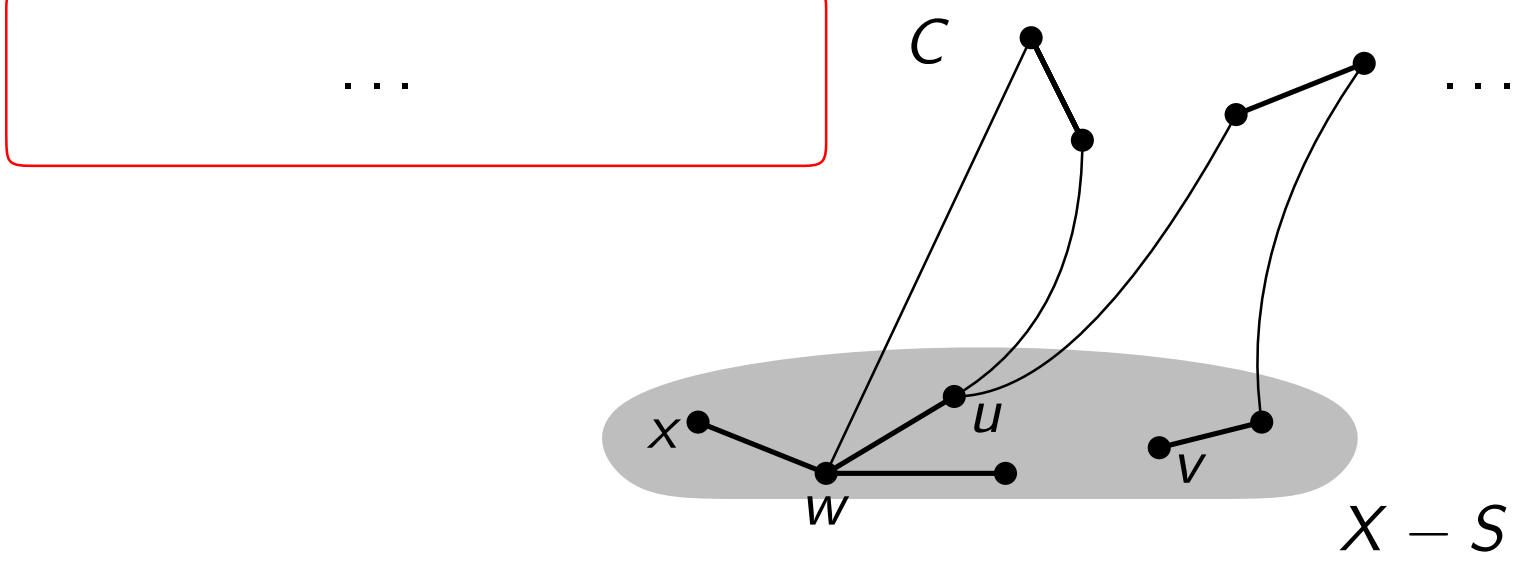
Removing components of $G - X$: using classification

Let $L = \{u \rightsquigarrow x, u \rightsquigarrow w, x \rightsquigarrow w\}$

- Suppose no optimal FVS in C breaks all connections in L
  - but any proper subset of L can be broken
- If there are many other such components
  - Remove C, decrease budget by opt(C).

$G - (S \cup S_C)$

components where S does not break L



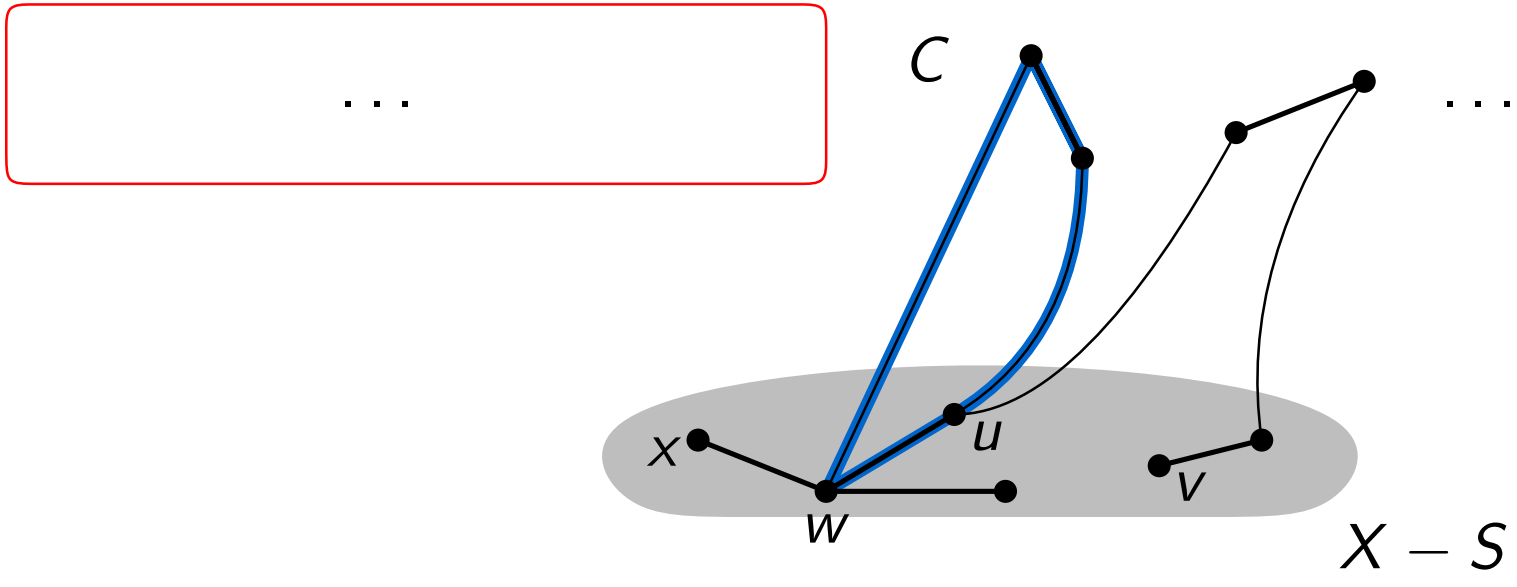
Removing components of $G - X$: using classification

Let $L = \{u \rightsquigarrow x, u \rightsquigarrow w, x \rightsquigarrow w\}$

- Suppose no optimal FVS in C breaks all connections in L
  - but any proper subset of L can be broken
- If there are many other such components
  - Remove C, decrease budget by opt(C).

$G - (S \cup S_C)$

components where S does not break L

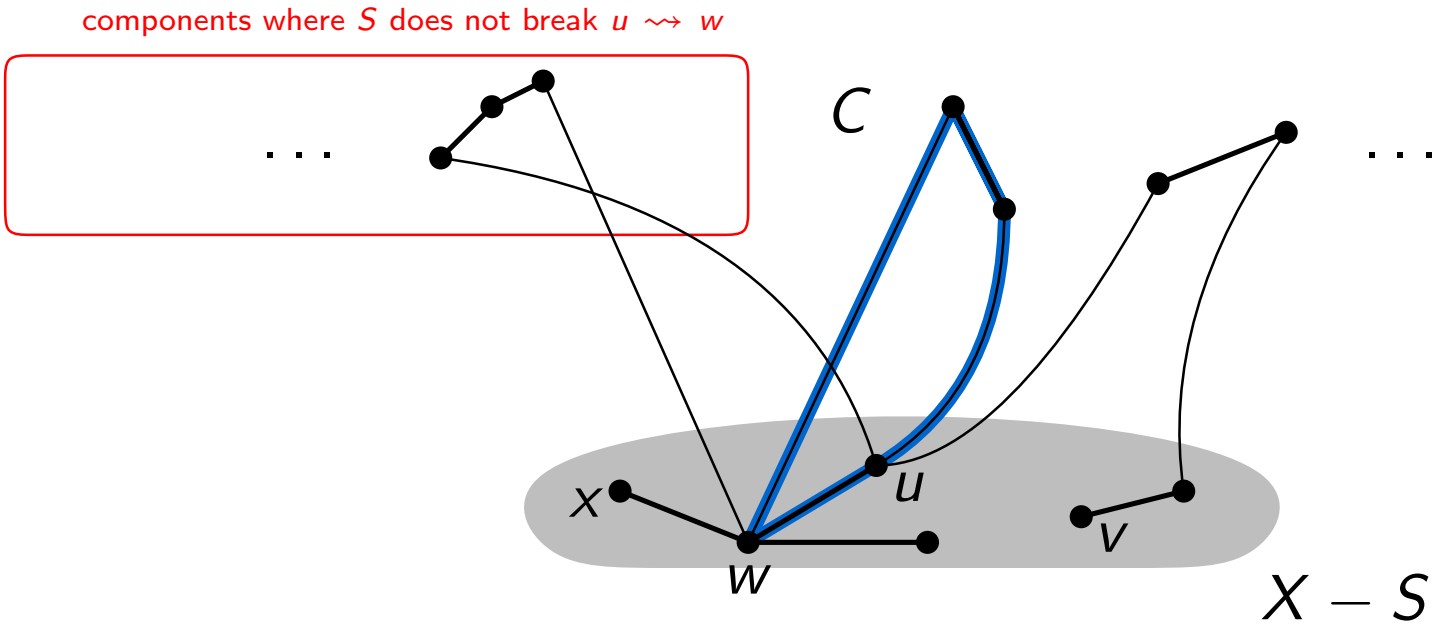


Removing components of $G - X$: using classification

Let $L = \{u \rightsquigarrow x, u \rightsquigarrow w, x \rightsquigarrow w\}$

- Suppose no optimal FVS in C breaks all connections in L
  - but any proper subset of L can be broken
- If there are many other such components
  - Remove C, decrease budget by $\mathrm{opt}(C)$.

$G - (S \cup S_C)$

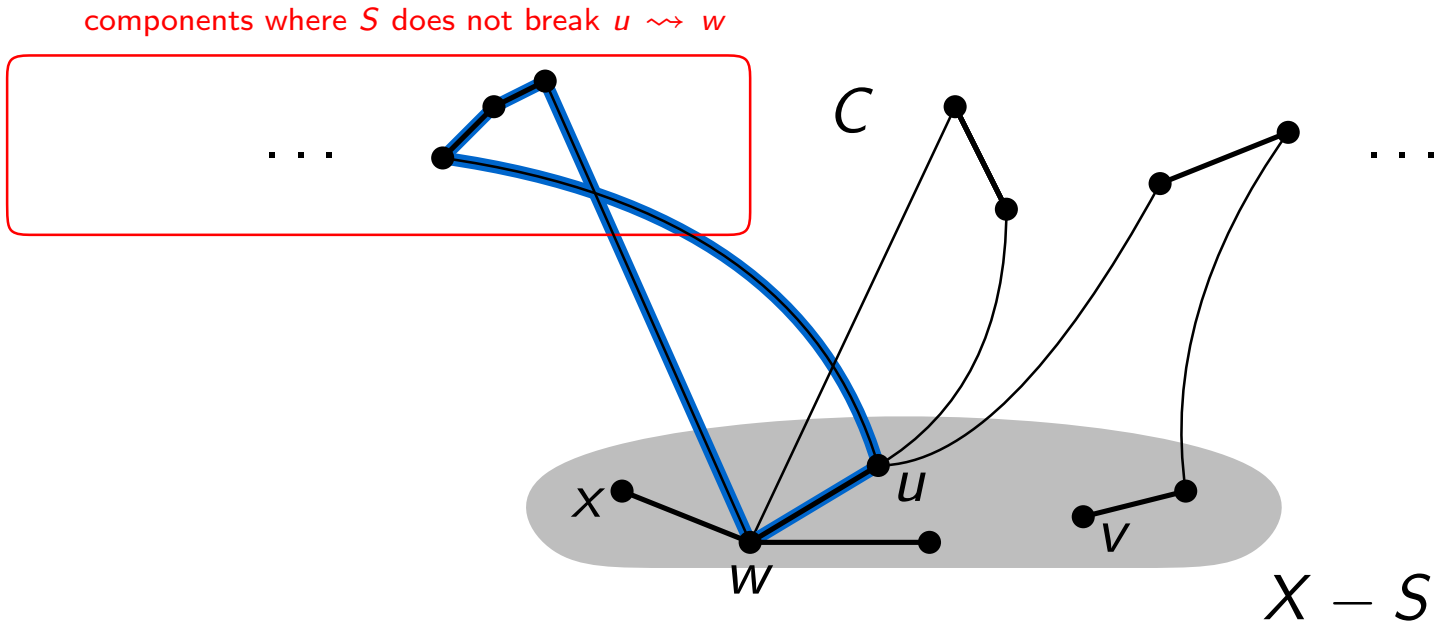


Removing components of $G - X$: using classification

Let $L = \{u \rightsquigarrow x, u \rightsquigarrow w, x \rightsquigarrow w\}$

- Suppose no optimal FVS in C breaks all connections in L
  - but any proper subset of L can be broken
- If there are many other such components
  - Remove C, decrease budget by opt(C).

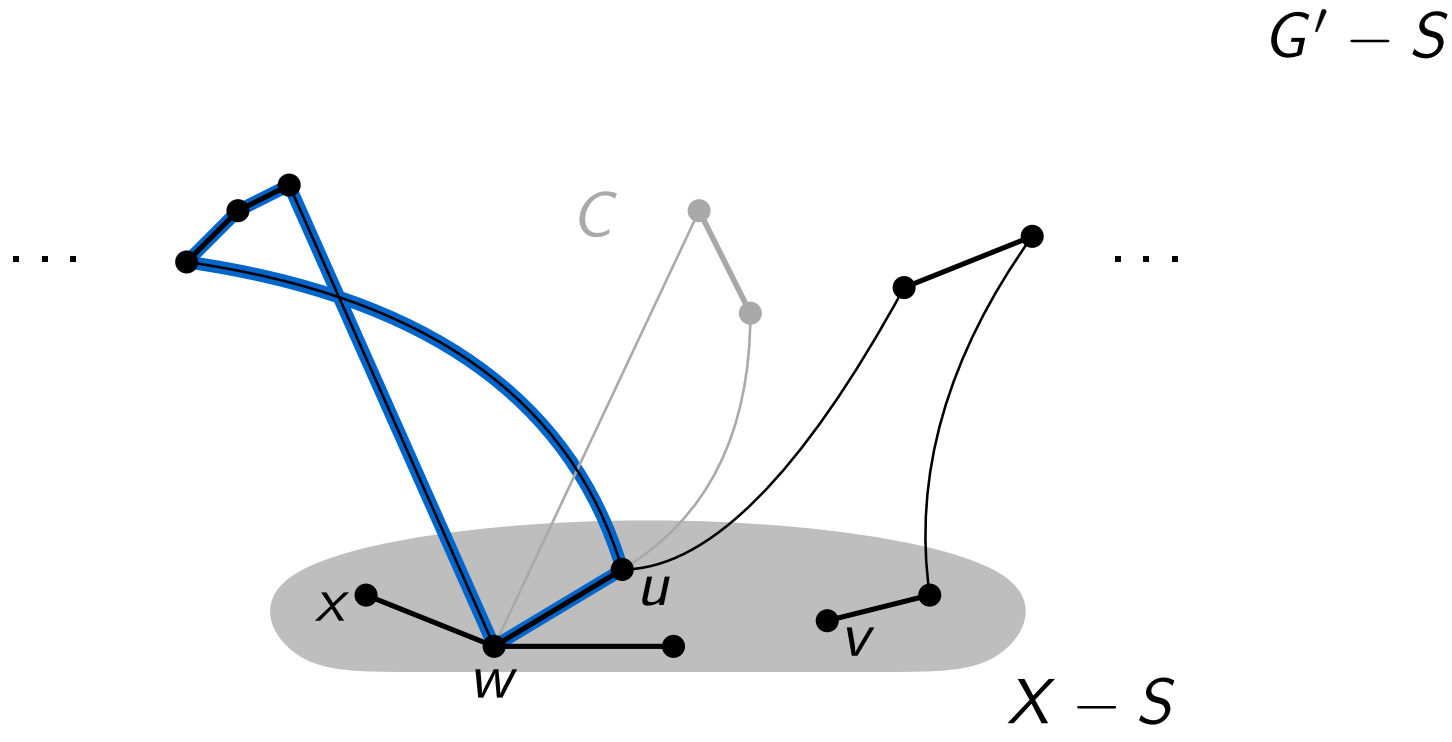
$G - (S \cup S_C)$



Removing components of $G - X$: using classification

Let $L = \{u \rightsquigarrow x, u \rightsquigarrow w, x \rightsquigarrow w\}$

- Suppose no optimal FVS in C breaks all connections in L
  - but any proper subset of L can be broken
- If there are many other such components
  - Remove C, decrease budget by $\mathrm{opt}(C)$.



Towards a reduction rule

Behaviour of components described by which sets L can be broken

- Use previous rule
- For each L, mark $\text{poly}(|X|)$ representative components
- Remove unmarked components

But there are many sets to consider

- Subsets of $u \rightsquigarrow v$ connections
- Exponentially many such sets

Ideally: only need to consider sets of constant size γ

- Dependent on $\mathcal{F}$ and η

Towards a reduction rule

Behaviour of components described by which sets L can be broken

- Use previous rule
- For each L, mark poly($|X|$) representative components
- Remove unmarked components

But there are many sets to consider

- Subsets of $u \rightsquigarrow v$ connections
- Exponentially many such sets

Ideally: only need to consider sets of constant size γ

- Dependent on $\mathcal{F}$ and η

Towards a reduction rule

Behaviour of components described by which sets L can be broken

- Use previous rule
- For each L, mark poly($|X|$) representative components
- Remove unmarked components

But there are many sets to consider

- Subsets of $u \rightsquigarrow v$ connections
- Exponentially many such sets

Ideally: only need to consider sets of constant size γ

- Dependent on $\mathcal{F}$ and η

Towards a reduction rule

Do we need large sets?

- What if a FVS cannot break very large set L
- but can break $L \setminus \{u \rightsquigarrow v\}$ for all u, v

Main effort in our paper: Showing that this does not happen

Main Lemma (sketch)

Let L be a set of $u \rightsquigarrow v$ connections, let C be a graph of constant treedepth. If no optimal FVS in C breaks all connections in L, then there exists $L' \subseteq L$ such that

- $|L'| \leq \gamma$
- No optimal FVS of C breaks all connections in L'

Towards a reduction rule

Do we need large sets?

- What if a FVS cannot break very large set L
- but can break $L \setminus \{u \rightsquigarrow v\}$ for all u, v

Main effort in our paper: Showing that this does not happen

Main Lemma (sketch)

Let L be a set of $u \rightsquigarrow v$ connections, let C be a graph of constant treedepth. If no optimal FVS in C breaks all connections in L, then there exists $L' \subseteq L$ such that

- $|L'| \leq \gamma$
- No optimal FVS of C breaks all connections in L'

Removing components of $G - X$

Reduction rule (sketch)

For each set L with $|L| \leq \gamma$ of $u \rightsquigarrow v$-connections, mark $\text{poly}(|X|)$ connected components C of $G - X$ s.t.

- There is no optimal FVS in C that breaks all connections in L

Remove all unmarked components

This leaves polynomially many components in $G - X$.

Removing components of $G - X$

Reduction rule (sketch)

For each set L with $|L| \leq \gamma$ of $u \rightsquigarrow v$-connections, mark $\text{poly}(|X|)$ connected components C of $G - X$ s.t.

- There is no optimal FVS in C that breaks all connections in L

Remove all unmarked components

This leaves polynomially many components in $G - X$.

Conclusion

$\mathcal{F}$-Minor-Free Deletion parameterized by a treedepth-η modulator has a polynomial kernel

- Graphs in $\mathcal{F}$ must be connected

Future work

Find the most general graph class $\mathcal{G}$ such that

- Vertex Cover parameterized by a modulator to $\mathcal{G}$ has a polynomial kernel