The Number of Minimum k-cuts: Beating the Karger-Stein Bound

Jason Li (CMU)

Joint work with Anupam Gupta (CMU), Euiwoong Lee (NYU)

STOC 2019
6/24/2019

Introduction

minimum k-cut: delete min weight edges to cut graph into $\geq k$ connect components

Setting: exact algorithm, k constant

Q: How fast to compute min k-cut?

Q: How <u>many</u> min k-cuts are there?

Prior Work

Goldschmidt–Hochbaum 1994: $O(n^{(1/2-o(1))k^2})$ time deterministic

Karger–Stein 1994: $\tilde{O}(n^{2(k-1)})$ time randomized

Thorup 2008: $\tilde{O}(mn^{2k-2})$ time deterministic

Chekuri et al. 2018: $\tilde{O}(mn^{2k-3})$ time deterministic

This work: $O_k(n^{(1.981+o(1))k})$ time randomized

All of these algorithms can <u>enumerate</u> all min k-cuts

$\Longrightarrow$ get corresponding extremal bound

Prior Work

Goldschmidt–Hochbaum 1994: $O(n^{(1/2-o(1))k^2})$ time deterministic

Karger–Stein 1994: $\tilde{O}(n^{2(k-1)})$ time randomized

Thorup 2008: $\tilde{O}(mn^{2k-2})$ time deterministic

Chekuri et al. 2018: $\tilde{O}(mn^{2k-3})$ time deterministic

This work: $O_k(n^{(1.981+o(1))k})$ time randomized

All of these algorithms can <u>enumerate</u> all min k-cuts

$\Longrightarrow$ get corresponding extremal bound

Lower bound: as hard as min weight k-clique: $\Omega(n^{(1-o(1))k})$

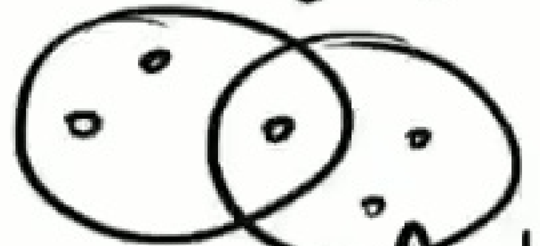
Our Approach

- Combines Karger-Stein's randomized contraction with Thorup's tree packing algorithm
- Makes new connection to extremal set theory in context of graph cut algorithms

Our Approach

- Combines Karger-Stein's randomized contraction with Thorup's tree packing algorithm
- Makes new connection to extremal set theory in context of graph cut algorithms

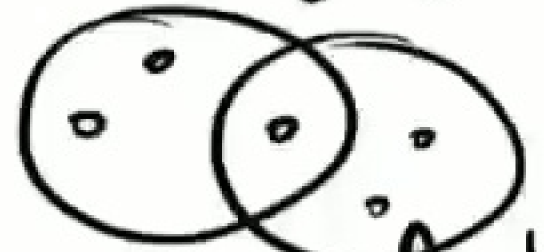
[Folklore] If $\mathcal{A}$ is a collection of distinct sets on $[n]$ (a set system), and $|\mathcal{A}| \geq 2n$, then $\exists A, B \in \mathcal{A}$ that cross:



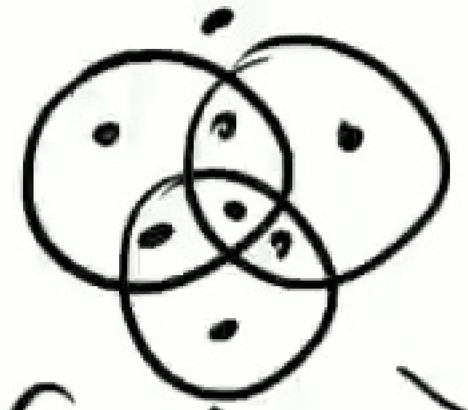
"If $|\mathcal{A}| \geq 2n$, then $\mathcal{A}$ has dual VC dimension ≥ 2"

Our Approach

- Combines Karger-Stein's randomized contraction with Thorup's tree packing algorithm
- Makes new connection to extremal set theory in context of graph cut algorithms

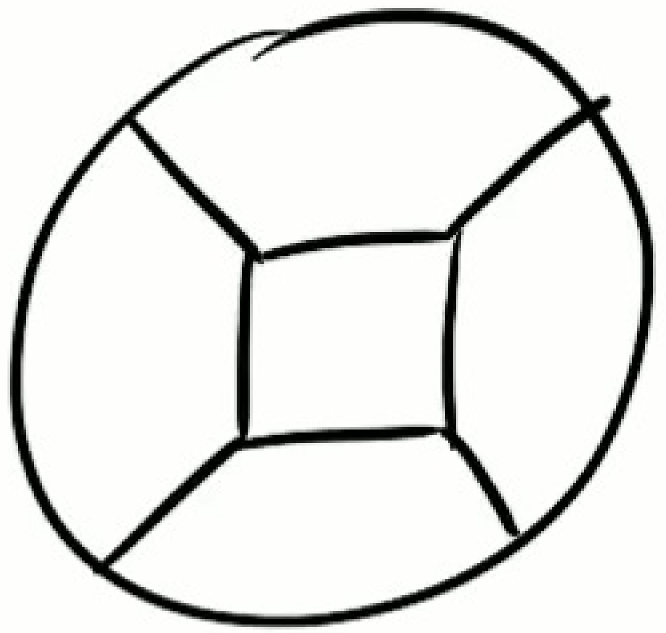
[Folklore] If $\mathcal{A}$ is a collection of distinct sets on $[n]$ (a set system), and $|\mathcal{A}| \geq 2n$, then $\exists A, B \in \mathcal{A}$ that cross: 

"If $|\mathcal{A}| \geq 2n$, then $\mathcal{A}$ has dual VC dimension ≥ 2"

[This work] If $|\mathcal{A}| \geq \Omega(n^{3.75})$, then $\exists A, B, C \in \mathcal{A}$: 

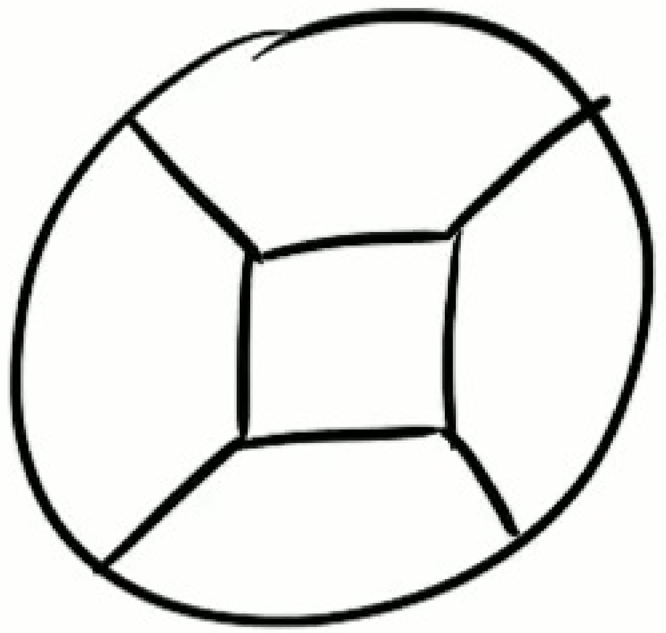
"If $|\mathcal{A}| \geq \Omega(n^{3.75})$, then $\mathcal{A}$ has dual VC dim ≥ 3"

Branch and Bound

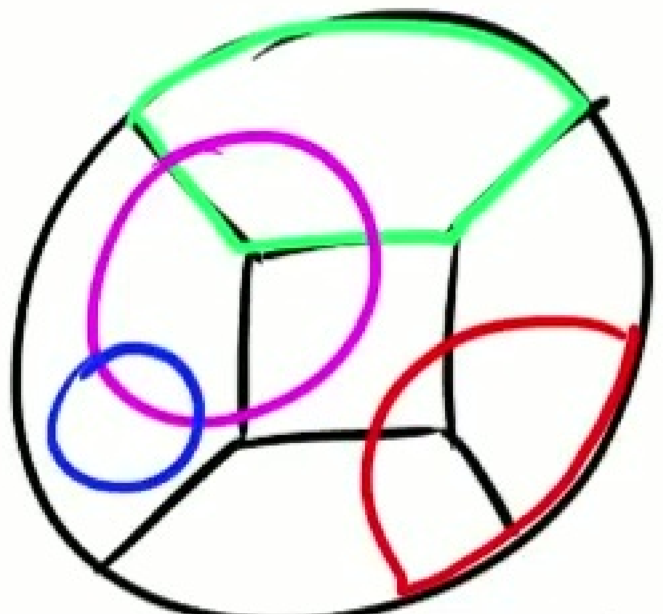


S :

Branch and Bound

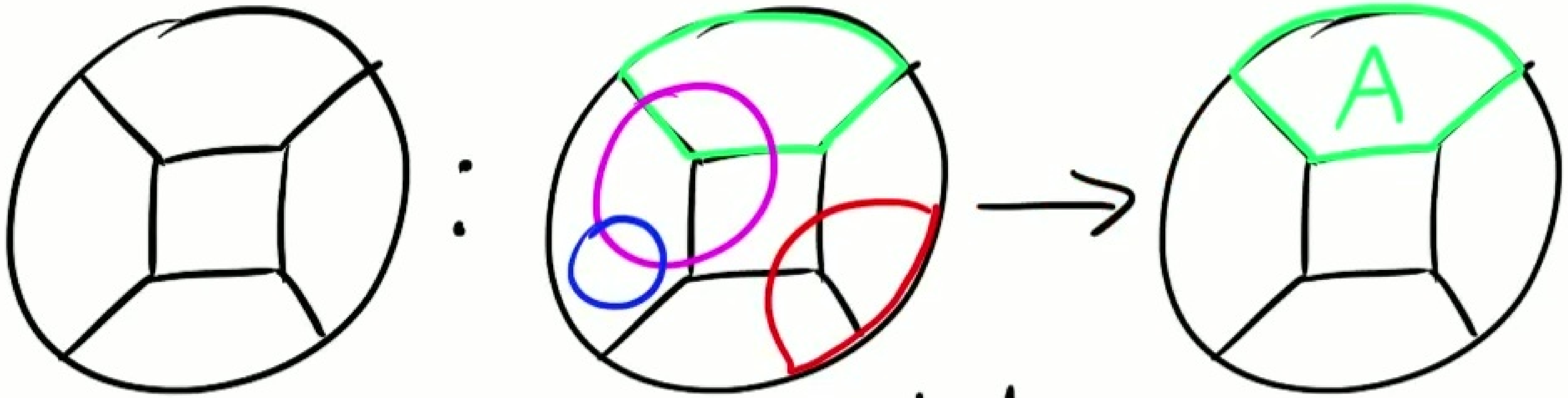


S :



Guess a set $\mathcal{A}$ of subsets of V

Branch and Bound

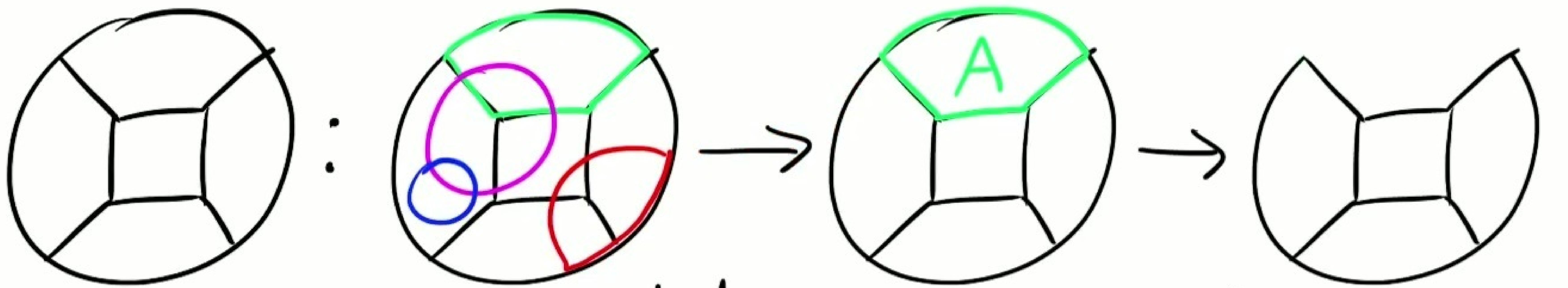


S

Guess a set $\mathcal{A}$ of subsets of V

For each set $A \in \mathcal{A}$:

Branch and Bound



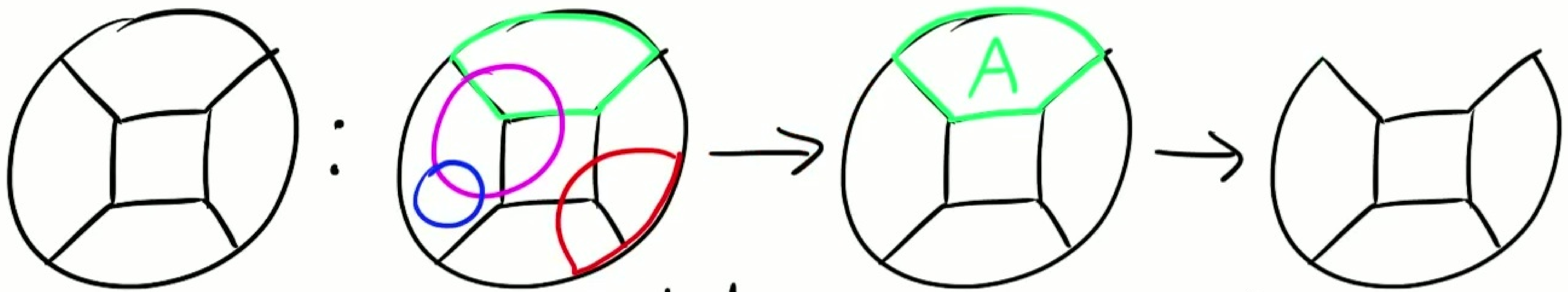
S

Guess a set $\mathcal{A}$ of subsets of V

For each set $A \in \mathcal{A}$:

A is one component; recurse by calling $(k-1)$-cut on $G-A$

Branch and Bound



S

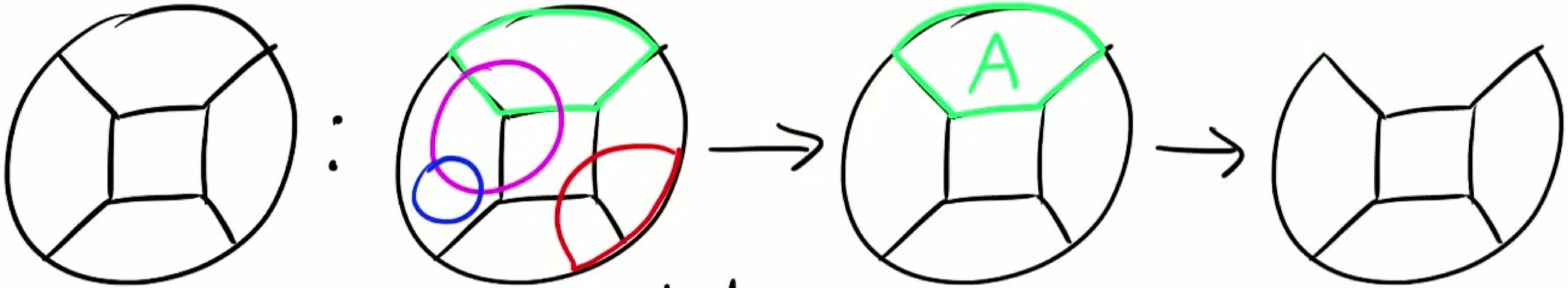
Guess a set $\mathcal{A}$ of subsets of V

For each set $A \in \mathcal{A}$:

A is one component; recurse by calling $(k-1)$-cut on $G-A$

- Works if $\exists A \in \mathcal{A}$ component of S

Branch and Bound



S:

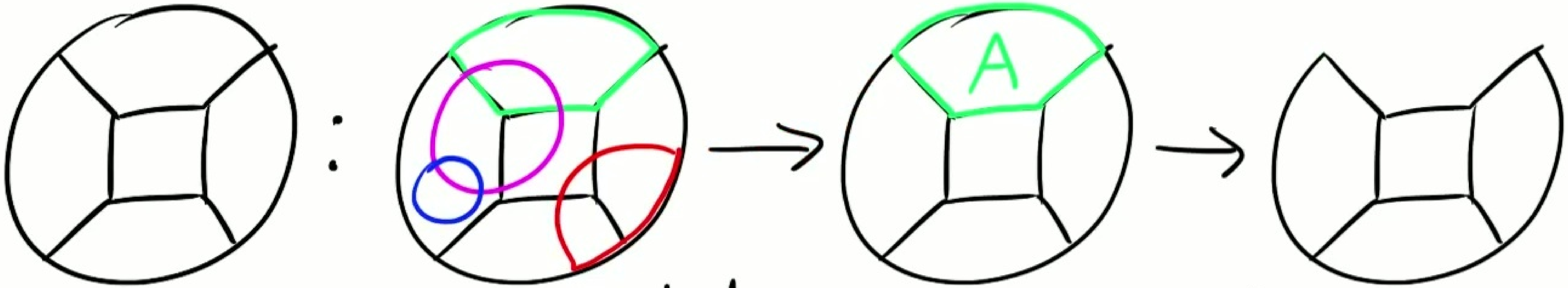
Guess a set $\mathcal{A}$ of subsets of V

For each set $A \in \mathcal{A}$:

A is one component; recurse by calling $(k-1)$-cut on $G - A$

- Works if $\exists A \in \mathcal{A}$ component of S
- Branching overhead: $|\mathcal{A}|$

Branch and Bound



S

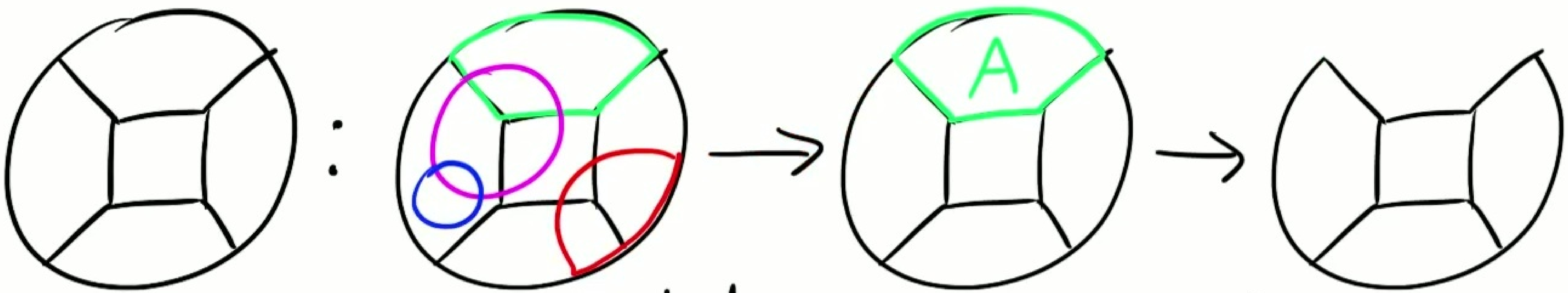
Guess a set $\mathcal{A}$ of subsets of V

For each set $A \in \mathcal{A}$:

A is one component; recurse by calling $(k-1)$-cut on $G-A$

- Works if $\exists A \in \mathcal{A}$ component of S
- Branching overhead: $|\mathcal{A}|$
- Want: small $|\mathcal{A}|$, and computable efficiently.

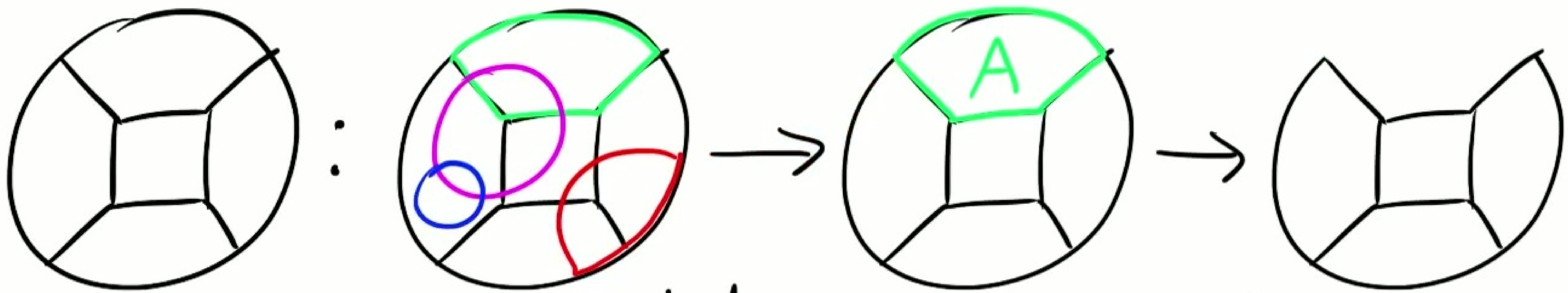
Branch and Bound



S : Guess a set $\mathcal{A}$ of subsets of V → For each set $A \in \mathcal{A}$: → A is one component; recurse by calling $(k-1)$-cut on $G-A$

- Works if $\exists A \in \mathcal{A}$ component of S
- Branching overhead: $|\mathcal{A}|$
- Want: small $|\mathcal{A}|$, and computable efficiently.
- $T(n,k) = (\text{time to compute } \mathcal{A}) + |\mathcal{A}| \cdot T(n,k-1)$

Branch and Bound



S

Guess a set $\mathcal{A}$ of subsets of V

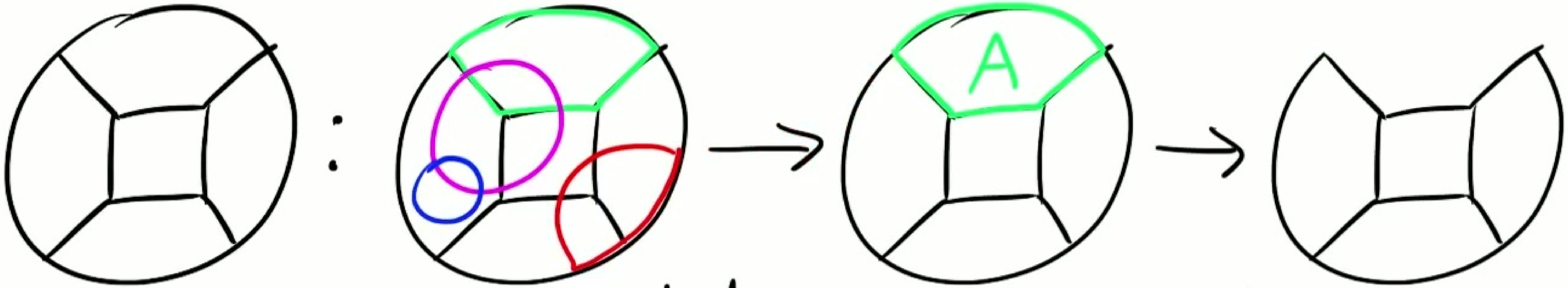
For each set $A \in \mathcal{A}$:

A is one component; recurse by calling $(k-1)$-cut on $G-A$

- Works if $\exists A \in \mathcal{A}$ component of S
- Branching overhead: $|\mathcal{A}|$
- Want: small $|\mathcal{A}|$, and computable efficiently.
- $T(n,k) = (\text{time to compute } \mathcal{A}) + |\mathcal{A}| \cdot T(n, k-1)$

Illustrative Example

Branch and Bound



S

Guess a set $\mathcal{A}$ of subsets of V

For each set $A \in \mathcal{A}$:

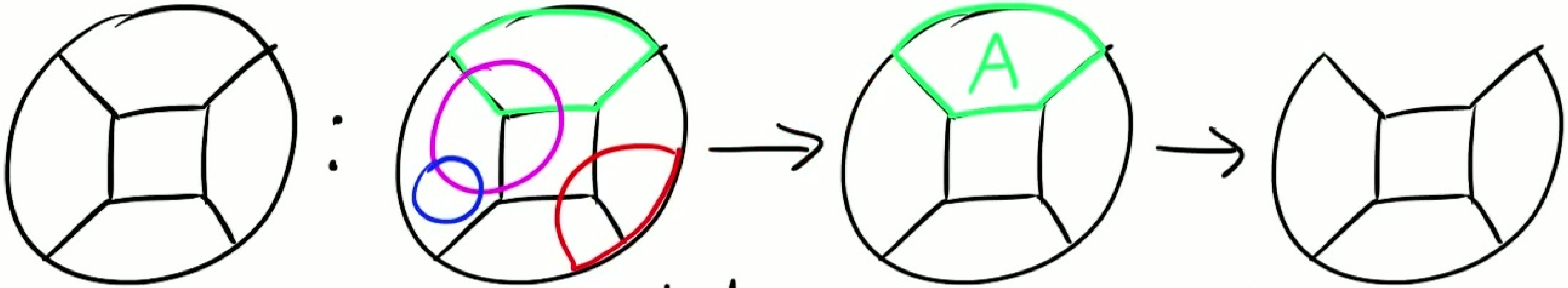
A is one component; recurse by calling $(k-1)$-cut on $G-A$

- Works if $\exists A \in \mathcal{A}$ component of S
- Branching overhead: $|\mathcal{A}|$
- Want: small $|\mathcal{A}|$, and computable efficiently.
- $T(n,k) = (\text{time to compute } \mathcal{A}) + |\mathcal{A}| \cdot T(n,k-1)$

Illustrative Example

Let $\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \leq \frac{1.49}{k} \mathrm{OPT}\}$

Branch and Bound



$\mathcal{S}$: Guess a set $\mathcal{A}$ of subsets of V → For each set $A \in \mathcal{A}$: → A is one component; recurse by calling $(k-1)$-cut on $G-A$

- Works if $\exists A \in \mathcal{A}$ component of $\mathcal{S}$
- Branching overhead: $|\mathcal{A}|$
- Want: small $|\mathcal{A}|$, and computable efficiently.
- $T(n,k) = (\text{time to compute } \mathcal{A}) + |\mathcal{A}| \cdot T(n,k-1)$

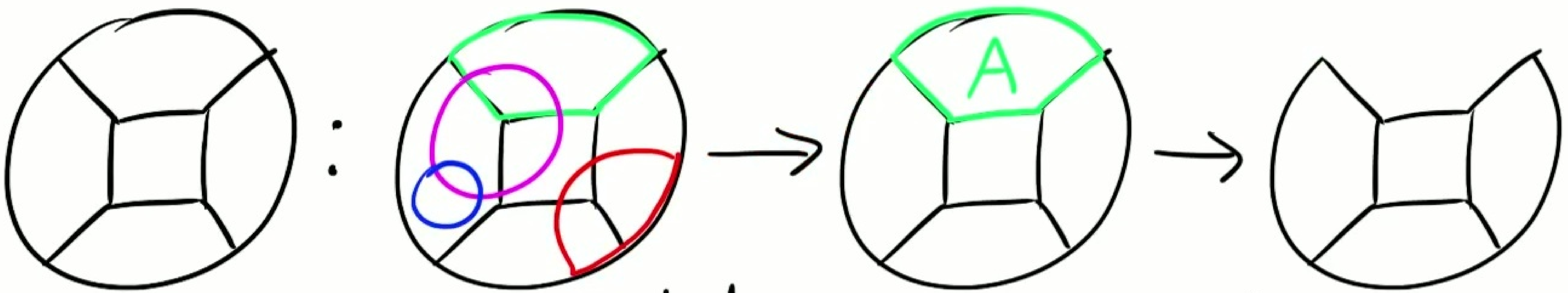
Illustrative Example

"much smaller than average"

Let $\mathcal{A}' := \left\{A \subseteq V : w(\partial_G A) \le \frac{1.49}{k} OPT\right\}$

Note: average $S \in \mathcal{S}$ is $\frac{2}{k} OPT$, since $\sum_{S \in \mathcal{S}} w(\partial_G S) = 2OPT$.

Branch and Bound



$\mathcal{S}$: Guess a set $\mathcal{A}$ of subsets of V → For each set $A \in \mathcal{A}$: → A is one component; recurse by calling $(k-1)$-cut on $G-A$

- Works if $\exists A \in \mathcal{A}$ component of $\mathcal{S}$
- Branching overhead: $|\mathcal{A}|$
- Want: small $|\mathcal{A}|$, and computable efficiently.
- $T(n,k) = (\text{time to compute } \mathcal{A}) + |\mathcal{A}| \cdot T(n,k-1)$

Illustrative Example

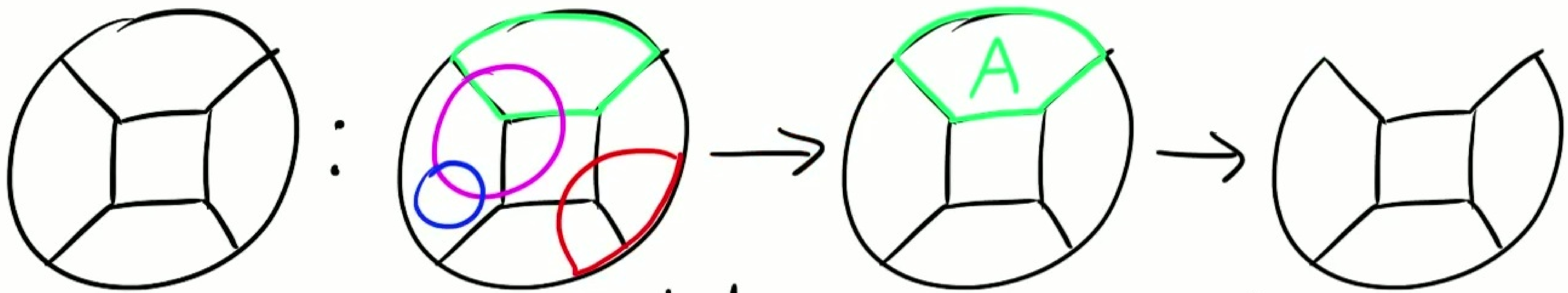
Let $\mathcal{A}' := \left\{ A \subseteq V : w(\partial_G A) \leq \frac{1.49}{k} OPT \right\}$ "much smaller than average"

Note: average $S \in \mathcal{S}$ is $\frac{2}{k} OPT$, since

$$\sum_{S \in \mathcal{S}} w(\partial_G S) = 2 OPT.$$

So, only works if G contains a component "much smaller than avg"

Branch and Bound



$\mathcal{S}$:

Guess a set $\mathcal{A}$ of subsets of V →

For each set $A \in \mathcal{A}$: →

A is one component; recurse by calling $(k-1)$-cut on $G-A$

- Works if $\exists A \in \mathcal{A}$ component of $\mathcal{S}$
- Branching overhead: $|\mathcal{A}|$
- Want: small $|\mathcal{A}|$, and computable efficiently.
- $T(n,k) = (\text{time to compute } \mathcal{A}) + |\mathcal{A}| \cdot T(n,k-1)$

Illustrative Example

"much smaller than average"

Let $\mathcal{A}' := \left\{ A \subseteq V : w(\partial_G A) \le \frac{1.49}{k} \mathrm{OPT} \right\}$

Note: average $S \in \mathcal{S}$ is $\frac{2}{k}\mathrm{OPT}$, since

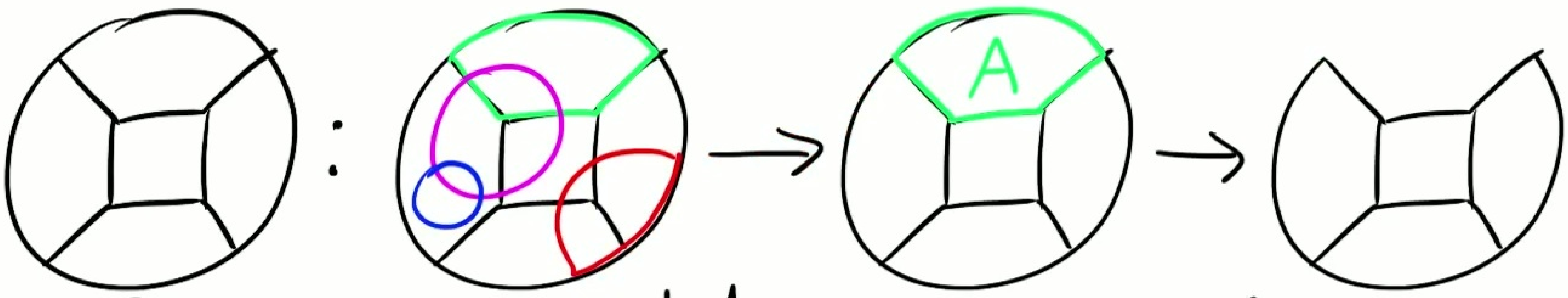
$$\sum_{S \in \mathcal{S}} w(\partial_G S) = 2\,\mathrm{OPT}.$$

So, only works if G contains a component "much smaller than avg"

Claim (bound):

$|\mathcal{A}'| = O_k(n)$.

Branch and Bound



$\mathcal{S}$

Guess a set $\mathcal{A}$ of subsets of V

For each set $A \in \mathcal{A}$:

A is one component; recurse by calling $(k-1)$-cut on $G-A$

- Works if $\exists A \in \mathcal{A}$ component of $\mathcal{S}$
- Branching overhead: $|\mathcal{A}|$
- Want: small $|\mathcal{A}|$, and computable efficiently.
- $T(n,k) = (\text{time to compute } \mathcal{A}) + |\mathcal{A}| \cdot T(n,k-1)$

Illustrative Example

"much smaller than average"

Let $\mathcal{A}' := \left\{ A \subseteq V : w(\partial_G A) \le \frac{1.49}{k}\mathrm{OPT} \right\}$

Note: average $S \in \mathcal{S}$ is $\frac{2}{k}\mathrm{OPT}$, since

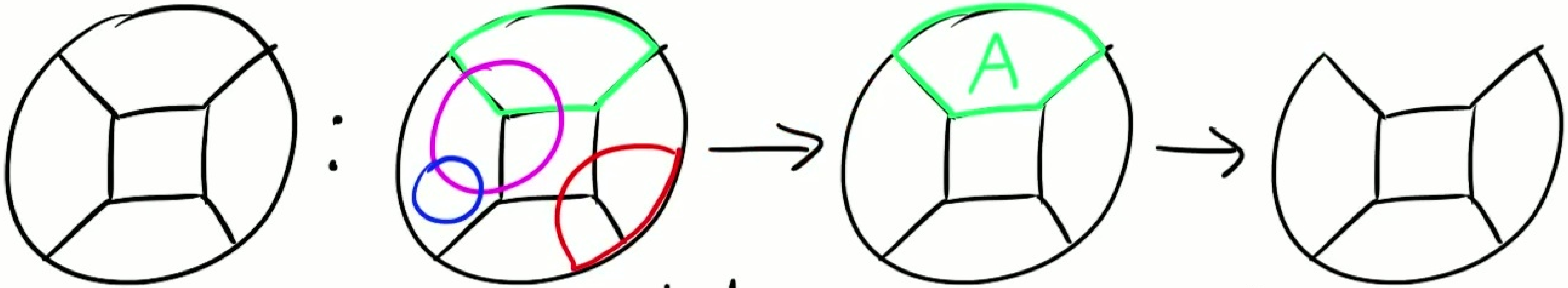
$$\sum_{S \in \mathcal{S}} w(\partial_G S) = 2\,\mathrm{OPT}.$$

So, only works if G contains a component "much smaller than avg"

Claim (bound): $|\mathcal{A}'| = O_k(n)$.

Claim (time): $\mathcal{A}'$ can be computed in poly(n).

Branch and Bound



$\mathcal{S}$: Guess a set $\mathcal{A}$ of subsets of V → For each set $A \in \mathcal{A}$: → A is one component; recurse by calling $(k-1)$-cut on $G-A$

- Works if $\exists A \in \mathcal{A}$ component of $\mathcal{S}$
- Branching overhead: $|\mathcal{A}|$
- Want: small $|\mathcal{A}|$, and computable efficiently.
- $T(n,k) = (\text{time to compute } \mathcal{A})$ poly(n)

  $+ |\mathcal{A}| \cdot T(n, k-1)$ n

  $\Rightarrow O_k(n^{k+O(1)})$!

Illustrative Example

"much smaller than average"

Let $\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \leq \frac{1.49}{k} OPT\}$

Note: average $S \in \mathcal{S}$ is $\frac{2}{k}OPT$, since $\sum_{S \in \mathcal{S}} w(\partial_G S) = 2OPT$.

So, only works if G contains a component "much smaller than avg"

Claim (bound): $|\mathcal{A}'| = O_k(n)$.

Claim (time): $\mathcal{A}'$ can be computed in poly(n).

Extremal Bound

$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \le \frac{1.49}{k} \mathrm{OPT}\}$

$\to |\mathcal{A}'| = O_k(n)$

$\mathcal{A}'$ computed in polytime

- Strategy: if $|\mathcal{A}'| \gg n$, i.e. too many cheap cuts, then can construct a k-cut solution $<$ OPT, contradiction.

Extremal Bound

$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \le \frac{1.49}{k} \mathrm{OPT}\}$

$\rightarrow |\mathcal{A}'| = O_k(n)$

$\mathcal{A}'$ computed in polytime

- Strategy: if $|\mathcal{A}'| \gg n$, i.e. too many cheap cuts, then can construct a k-cut solution $<$ OPT, contradiction.
- Fact: if a set system $|\mathcal{A}| > 2n$, then there are two sets $A, B \in \mathcal{A}$ that cross.

Extremal Bound

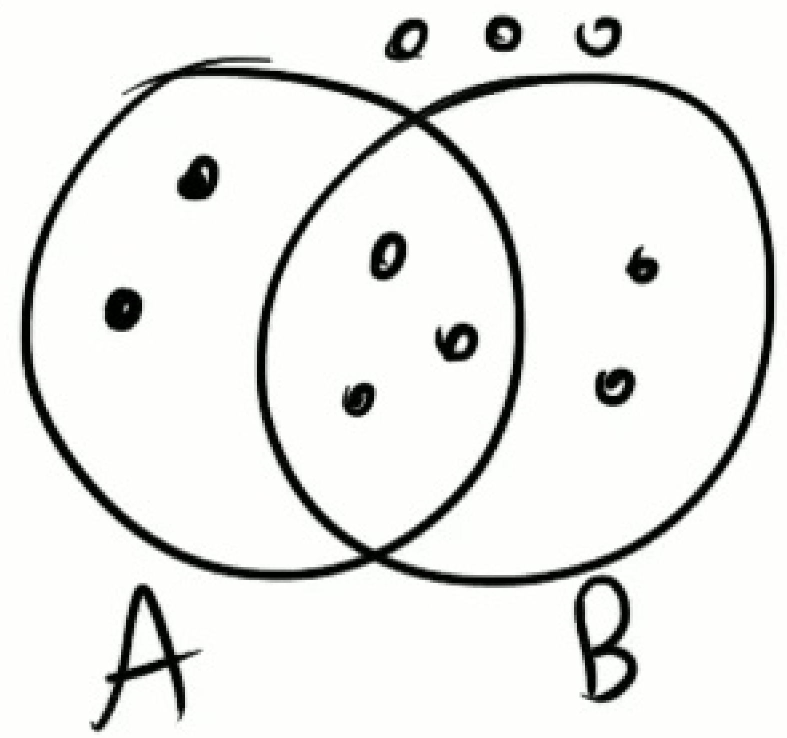
$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \le \frac{1.49}{k}\mathrm{OPT}\}$

$\rightarrow |\mathcal{A}'| = O_k(n)$

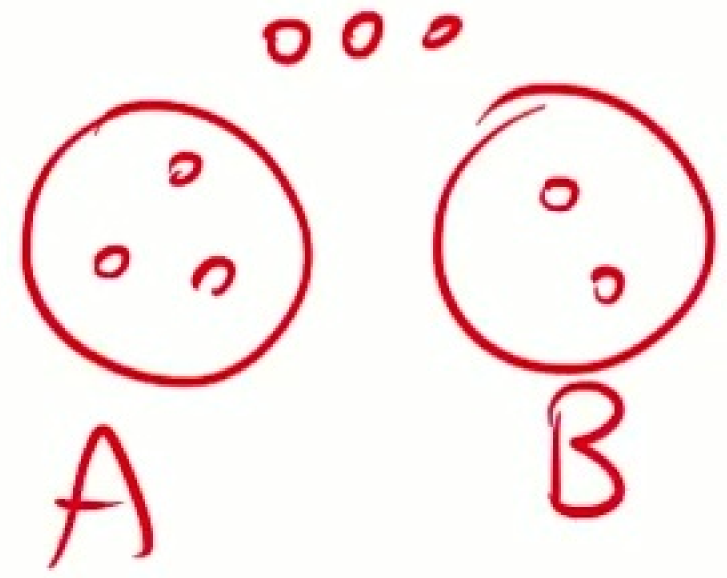
$\mathcal{A}'$ computed in polytime

- Strategy: if $|\mathcal{A}'| \gg n$, i.e. too many cheap cuts, then can construct a k-cut solution $<$ OPT, contradiction.
- Fact: if a set system $|\mathcal{A}| > 2n$, then there are two sets $A, B \in \mathcal{A}$ that cross.

cross:



don't cross:



Extremal Bound

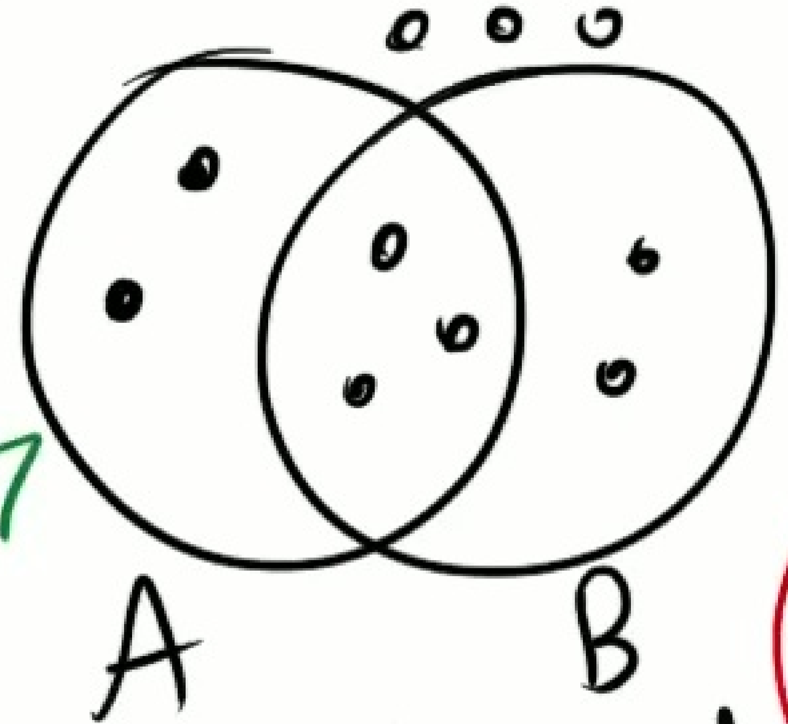
$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \le \frac{1.49}{k}\mathrm{OPT}\}$

$\rightarrow |\mathcal{A}'| = O_k(n)$

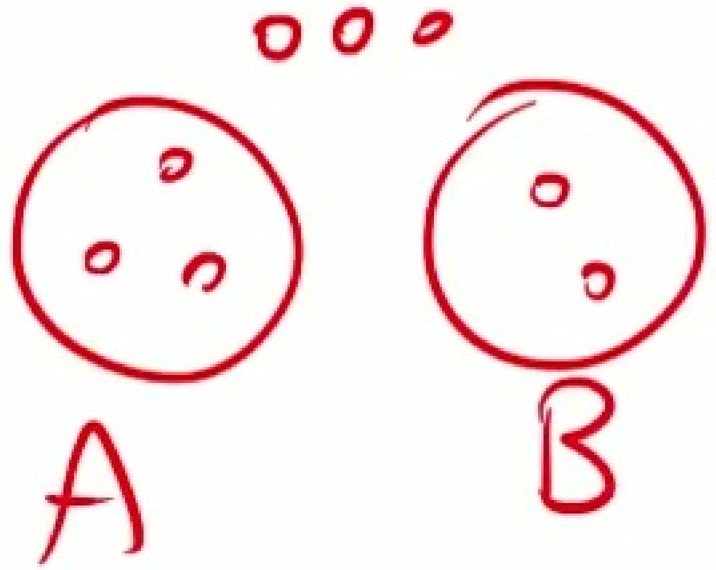
$\mathcal{A}'$ computed in polytime

- Strategy: if $|\mathcal{A}'| \gg n$, i.e. too many cheap cuts, then can construct a k-cut solution < OPT, contradiction.
- Fact: if a set system $|\mathcal{A}| > 2n$, then there are two sets $A, B \in \mathcal{A}$ that cross.

cross:



don't cross:



- Suppose we "cut out" A and B. (cut the edges $\partial A \cup \partial B$.) We get a 4-cut (not a 3-cut) since A and B cross.

Extremal Bound

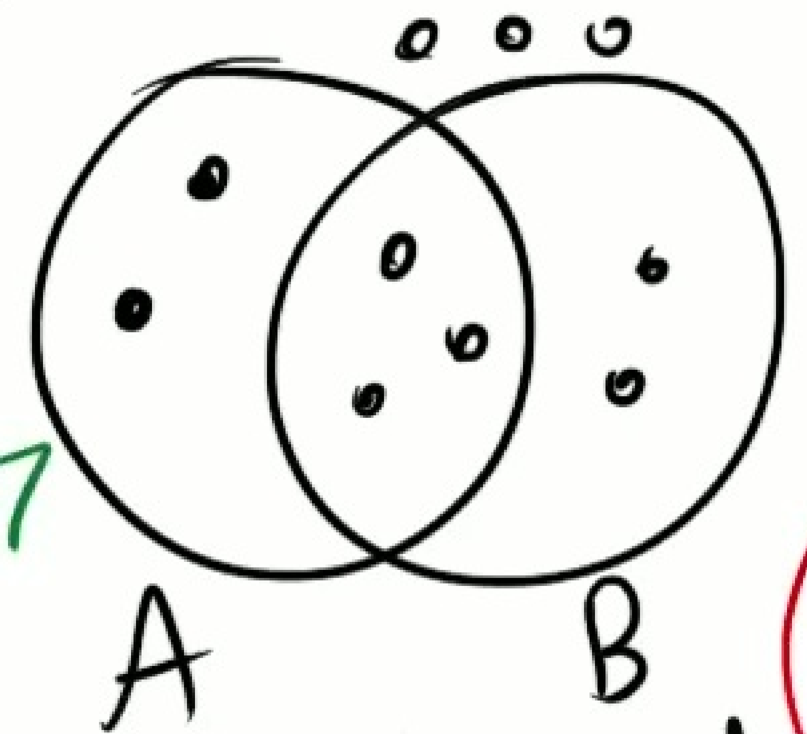
$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \le \frac{1.49}{k}\mathrm{OPT}\}$

$\rightarrow$ $|\mathcal{A}'| = O_k(n)$

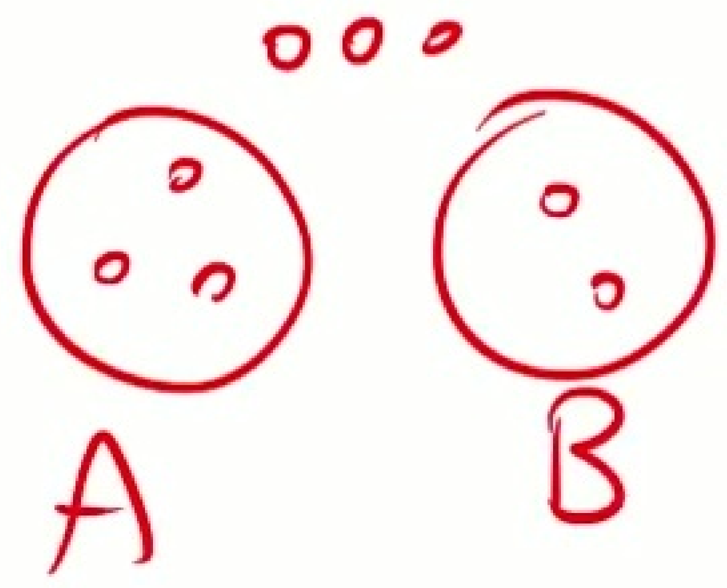
$\mathcal{A}'$ computed in polytime

- Strategy: if $|\mathcal{A}'| \gg n$, i.e. too many cheap cuts, then can construct a k-cut solution $< \mathrm{OPT}$, contradiction.
- Fact: if a set system $|\mathcal{A}| > 2n$, then there are two sets $A, B \in \mathcal{A}$ that cross.

cross:



don't cross:



- Suppose we "cut out" A and B. (cut the edges $\partial A \cup \partial B$.)
  We get a 4-cut (not a 3-cut) since A and B cross.
- Think of it as $+3$ additional components for price of $\le 2 \cdot \frac{1.49}{k}\mathrm{OPT}$
  $\Longrightarrow \frac{1-\varepsilon}{k}\mathrm{OPT}$ price per $+1$ component.

Extremal Bound

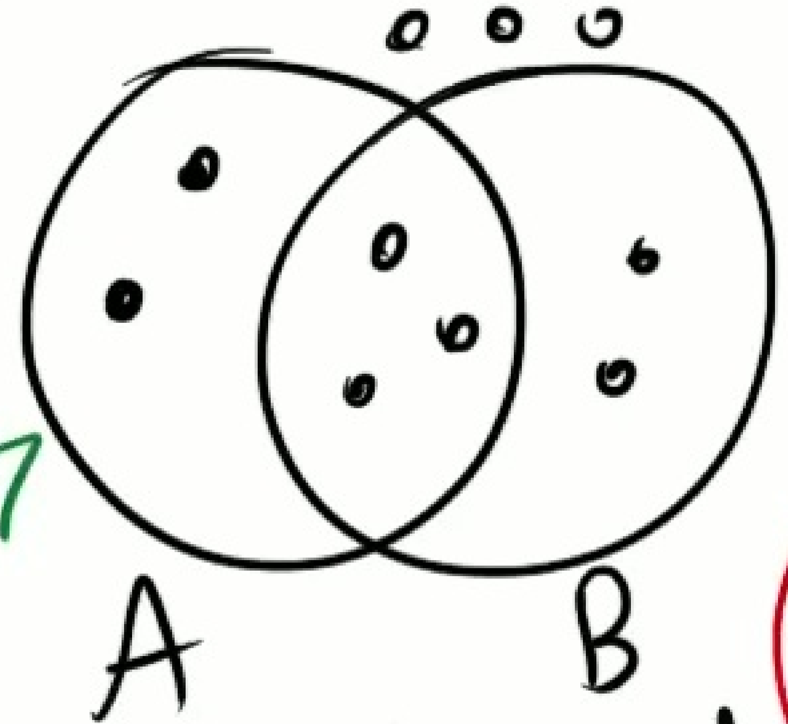
$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \le \frac{1.49}{k} OPT\}$

$\rightarrow |\mathcal{A}'| = O_k(n)$

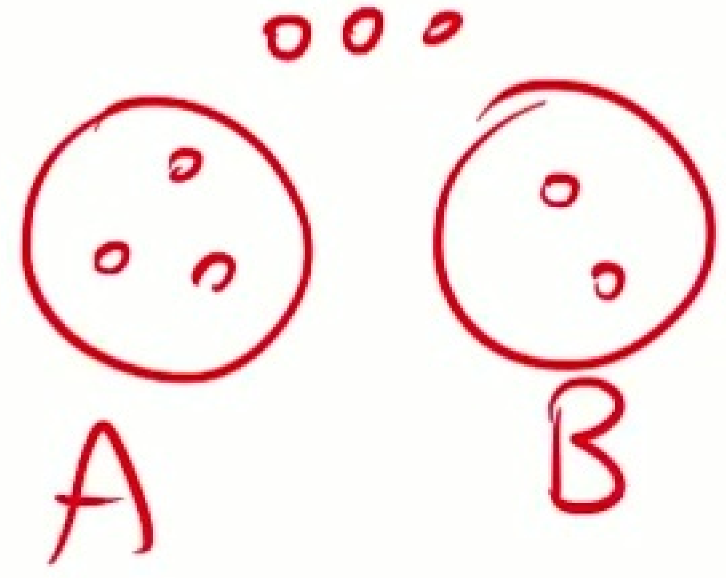
$\mathcal{A}'$ computed in polytime

- Strategy: if $|\mathcal{A}'| \gg n$, i.e. too many cheap cuts, then can construct a k-cut solution $<$ OPT, contradiction.
- Fact: if a set system $|\mathcal{A}| > 2n$, then there are two sets $A, B \in \mathcal{A}$ that cross.

cross:



don't cross:



- Suppose we "cut out" A and B. (cut the edges $\partial A \cup \partial B$.)
  We get a 4-cut (not a 3-cut) since A and B cross.
- Think of it as +3 additional components for price of $\le 2 \cdot \frac{1.49}{k} OPT$
  $\Longrightarrow \frac{1-\varepsilon}{k} OPT$ price per +1 component.
- Repeat until k comps?

Extremal Bound

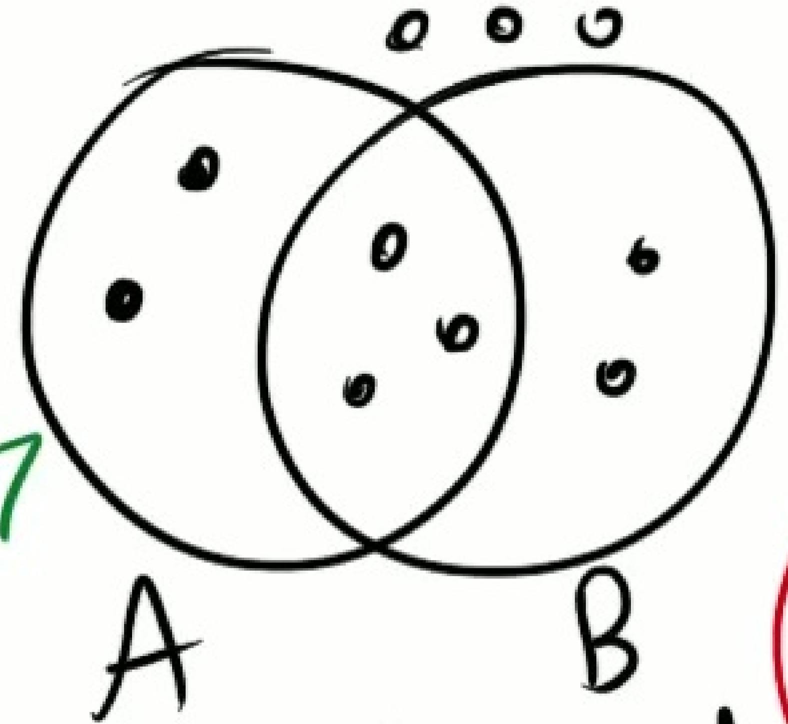
$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \le \frac{1.49}{k} OPT\}$

$\rightarrow |\mathcal{A}'| = O_k(n)$

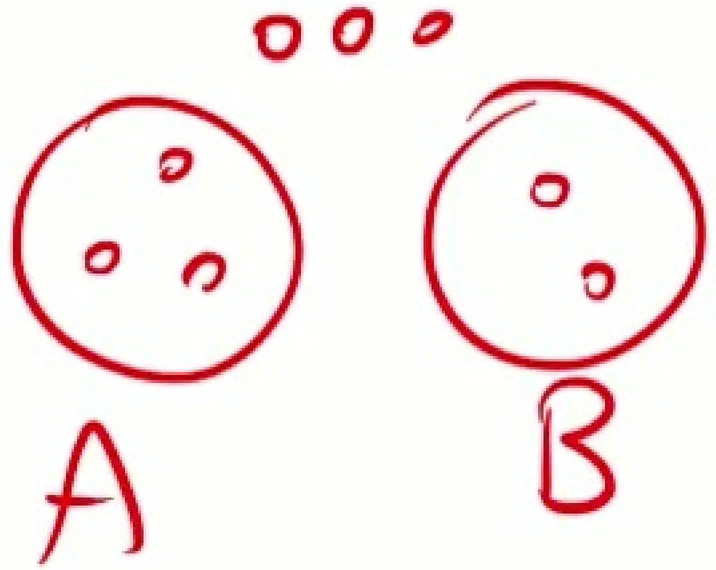
$\mathcal{A}'$ computed in polytime

- Strategy: if $|\mathcal{A}'| \gg n$, i.e. too many cheap cuts, then can construct a k-cut solution $<$ OPT, contradiction.
- Fact: if a set system $|\mathcal{A}| > 2n$, then there are two sets $A, B \in \mathcal{A}$ that <u>cross</u>.

cross:



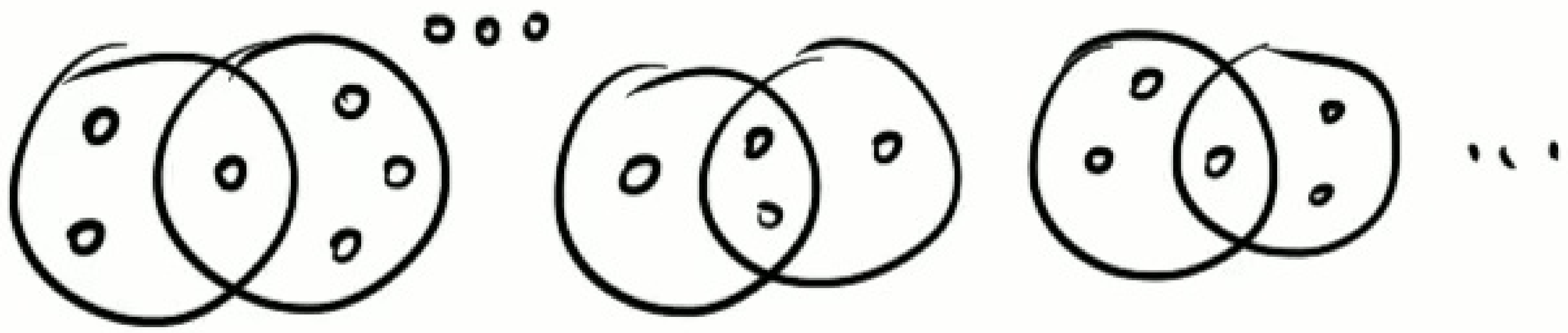
don't cross:



- Suppose we "cut out" A and B. (cut the edges $\partial A \cup \partial B$.) We get a <u>4-cut</u> (<u>not a 3-cut</u>) since A and B cross.
- Think of it as $+3$ additional components for price of $\le 2 \cdot \frac{1.49}{k}$ OPT

$$\Longrightarrow \frac{1-\varepsilon}{k} OPT \text{ price per } +1 \text{ component.}$$

- Repeat until k comps?



Extremal Bound

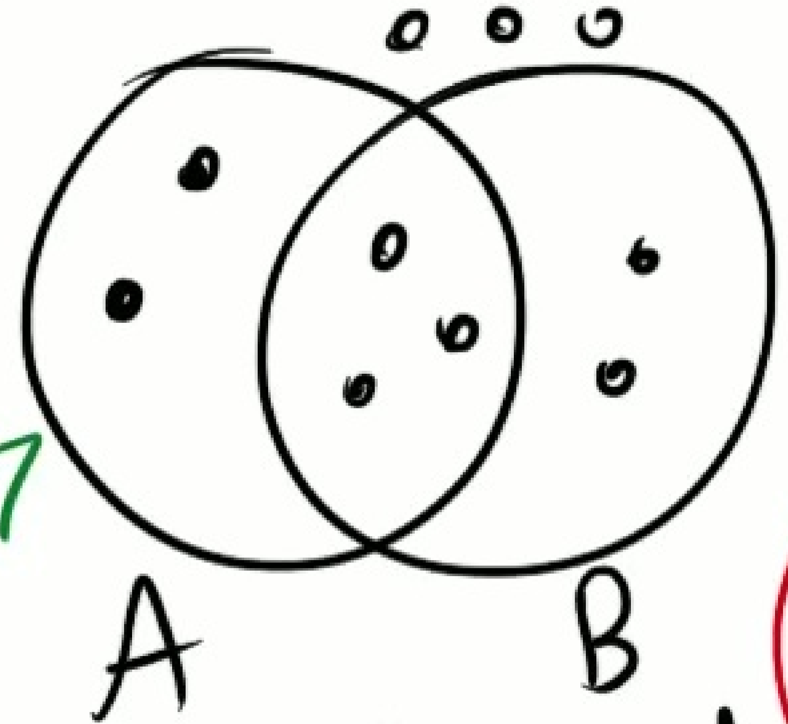
$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \le \frac{1.49}{k} OPT\}$

$\rightarrow |\mathcal{A}'| = O_k(n)$

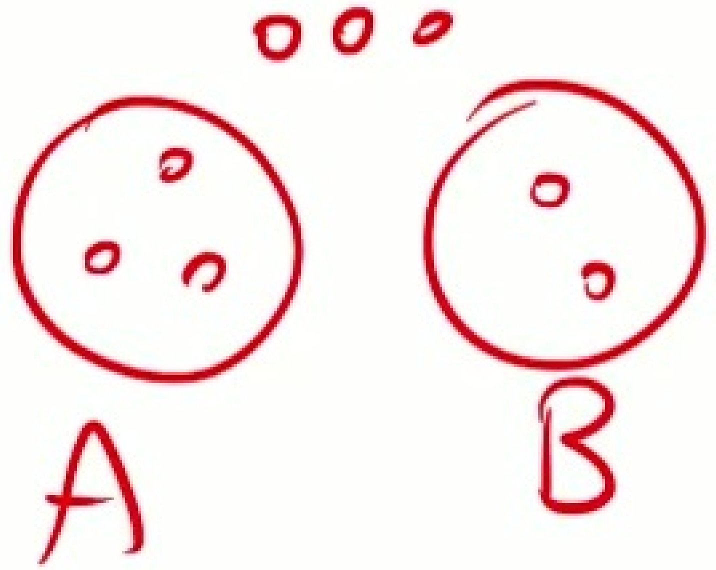
$\mathcal{A}'$ computed in polytime

- Strategy: if $|\mathcal{A}'| \gg n$, i.e. too many cheap cuts, then can construct a k-cut solution $< OPT$, contradiction.
- Fact: if a set system $|\mathcal{A}| > 2n$, then there are two sets $A, B \in \mathcal{A}$ that cross.

cross:

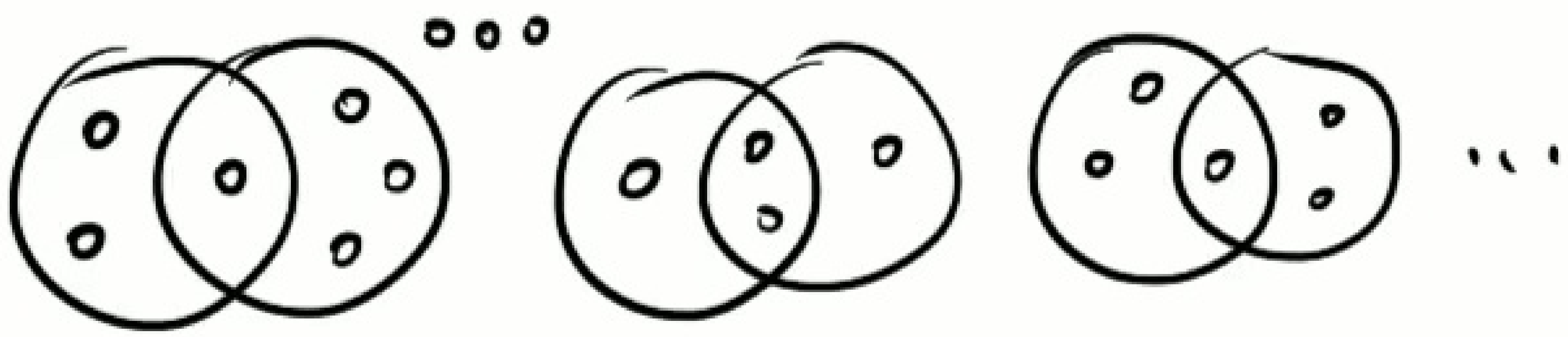


don't cross:



- Suppose we "cut out" A and B. (cut the edges $\partial A \cup \partial B$.)
  We get a 4-cut (not a 3-cut) since A and B cross.
- Think of it as +3 additional components for price of $\le 2 \cdot \frac{1.49}{k} OPT$

  $\Longrightarrow \frac{1-\varepsilon}{k} OPT$ price per +1 component.
- Repeat until k comps?



pay $\lceil \frac{k}{3} \rceil \cdot 2 \cdot \frac{1.49}{k} OPT < OPT$

for k large enough, contradiction.

Computing $\mathcal{A}'$

- Idea: modified Karger-Stein algorithm.

$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \leq \frac{1.49}{k} OPT\}$

$|\mathcal{A}'| = O_k(n)$

$\rightarrow \mathcal{A}'$ computed in polytime

Computing $\mathcal{A}'$

- Idea: modified Karger-Stein algorithm.

  While G has $>k$ vertices left:
  - Contract random edge (proportional to weight)
  - Merge parallel edges

$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \leq \frac{1.49}{k} OPT\}$

$|\mathcal{A}'| = O_k(n)$

$\rightarrow$ $\mathcal{A}'$ computed in polytime

$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \le \frac{1.49}{k} OPT\}$

$|\mathcal{A}'| = O_k(n)$

$\rightarrow$ $\mathcal{A}'$ computed in polytime

Computing $\mathcal{A}'$

- Idea: modified Karger-Stein algorithm.

  While G has $>k$ vertices left:

  - Contract random edge (proportional to weight)
  - Merge parallel edges

  For each of the 2^k subsets, output the set formed by uncontracting it.

Computing $\mathcal{A}'$

$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \le \frac{1.49}{k} \mathrm{OPT}\}$

$|\mathcal{A}'| = O_k(n)$

$\rightarrow$ $\mathcal{A}'$ computed in polytime

- Idea: modified Karger-Stein algorithm.

  While G has $>k$ vertices left:
  - Contract random edge (proportional to weight)
  - Merge parallel edges

  For each of the 2^k subsets, output the set formed by uncontracting it.
- Repeat poly(n) times.

Computing $\mathcal{A}'$

$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \leq \frac{1.49}{k} OPT\}$

$|\mathcal{A}'| = O_k(n)$

$\rightarrow$ $\mathcal{A}'$ computed in polytime

- Idea: modified Karger-Stein algorithm.

  While G has $>k$ vertices left:
  - Contract random edge (proportional to weight)
  - Merge parallel edges

  For each of the 2^k subsets, output the set formed by uncontracting it.
- Repeat poly(n) times.

K-S analysis: fix cut $A \in \mathcal{A}'$. Pr[output A]? Must not contract edge in ∂A, ever.

Computing $\mathcal{A}'$

$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \le \frac{1.49}{k} OPT\}$

$|\mathcal{A}'| = O_k(n)$

$\rightarrow \mathcal{A}'$ computed in polytime

- Idea: modified Karger-Stein algorithm.

  While G has $>k$ vertices left:
  - Contract random edge (proportional to weight)
  - Merge parallel edges

  For each of the 2^k subsets, output the set formed by uncontracting it.
- Repeat poly(n) times.

K–S analysis: fix cut $A \in \mathcal{A}'$. Pr[output A]? Must not contract edge in ∂A, ever.

Pr[contract edge in ∂A when r vertices left]?

Computing $\mathcal{A}'$

$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \le \frac{1.49}{k} OPT\}$

$|\mathcal{A}'| = O_k(n)$

$\rightarrow$ $\mathcal{A}'$ computed in polytime

- Idea: modified Karger-Stein algorithm.

  While G has $>k$ vertices left:
  - Contract random edge (proportional to weight)
  - Merge parallel edges

  For each of the 2^k subsets, output the set formed by uncontracting it.
- Repeat poly(n) times.

K–S analysis: fix cut $A \in \mathcal{A}'$. Pr[output A]? Must not contract edge in ∂A, ever.

Pr[contract edge in ∂A when r vertices left]?

$$(\text{min } k\text{-cut}) \le (k-1)\cdot(\text{avg degree}) = (k-1)\cdot\frac{2\sum_e w_e}{r}.$$

Computing $\mathcal{A}'$

$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \le \frac{1.49}{k} OPT\}$

$|\mathcal{A}'| = O_k(n)$

$\rightarrow$ $\mathcal{A}'$ computed in polytime

- Idea: modified Karger-Stein algorithm.

  While G has $>k$ vertices left:
  - Contract random edge (proportional to weight)
  - Merge parallel edges

  For each of the 2^k subsets, output the set formed by uncontracting it.
- Repeat poly(n) times.

K–S analysis: fix cut $A \in \mathcal{A}'$. Pr[output A]? Must not contract edge in ∂A, ever.

Pr[contract edge in ∂A when r vertices left]?

$(\min k\text{-cut}) \le (k-1)\cdot(\text{avg degree}) = (k-1)\cdot \frac{2\sum_e w_e}{r}$.

$\Pr[\text{failure: contract edge}] = \frac{w(\partial_G A)}{\sum_e w_e} \le \frac{\frac{1.49}{k} OPT}{\sum_e w_e}$

Computing $\mathcal{A}'$

$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \le \frac{1.49}{k} OPT\}$

$|\mathcal{A}'| = O_k(n)$

$\rightarrow$ $\mathcal{A}'$ computed in polytime

- Idea: modified Karger-Stein algorithm.

  While G has $>k$ vertices left:
  - Contract random edge (proportional to weight)
  - Merge parallel edges

  For each of the 2^k subsets, output the set formed by uncontracting it.
- Repeat poly(n) times.

K–S analysis: fix cut $A \in \mathcal{A}'$. Pr[output A]? Must not contract edge in ∂A, ever.

Pr[contract edge in ∂A when r vertices left]?

$$\text{(min } k\text{-cut)} \le (k-1)\cdot(\text{avg degree}) = \cancel{(k-1)}\cdot \frac{2\cancel{\sum_e w_e}}{r}.$$

$$\Pr[\text{failure: contract edge}] = \frac{w(\partial_G A)}{\sum_e w_e} \le \frac{\frac{1.49}{\cancel{k}} OPT}{\cancel{\sum_e w_e}} \le \frac{2.98}{r}$$

Computing $\mathcal{A}'$

$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \le \frac{1.49}{k} OPT\}$

$|\mathcal{A}'| = O_k(n)$

$\rightarrow$ $\mathcal{A}'$ computed in polytime

- Idea: modified Karger-Stein algorithm.

  While G has $>k$ vertices left:
  - Contract random edge (proportional to weight)
  - Merge parallel edges

  For each of the 2^k subsets, output the set formed by uncontracting it.
- Repeat poly(n) times.

K–S analysis: fix cut $A \in \mathcal{A}'$. Pr[output A]? Must not contract edge in ∂A, ever.

Pr[contract edge in ∂A when r vertices left]?

$$(\text{min } k\text{-cut}) \le (k-1)\cdot(\text{avg degree}) = \cancel{(k-1)}\cdot \frac{2 \cancel{\sum_e w_e}}{r}.$$

$$\Pr[\text{failure: contract edge}] = \frac{w(\partial_G A)}{\sum_e w_e} \le \frac{\frac{1.49}{\cancel{k}} OPT}{\cancel{\sum_e w_e}} \le \frac{2.98}{r}$$

$$\Longrightarrow \Pr[\text{success}] = \left(1 - \frac{2.98}{n}\right)\left(1 - \frac{2.98}{n-1}\right)\cdots \gtrsim \frac{1}{n^{2.98}}$$

- Repeat $\Theta(n^{2.98} \log n)$ times. Output the $\Theta(2^k n)$ sets with smallest boundaries found.

What if $\mathcal{S}$ "balanced"?

$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \leq \frac{1.49}{k} OPT\}$

$|\mathcal{A}'| = O_k(n)$

$\mathcal{A}'$ computed in polytime

Example: $\mathcal{S}$ is "perfectly balanced": $w(\partial_G S_i) = \frac{2}{k} OPT \quad \forall i$

What if $\mathcal{S}$ "balanced"?

Example: $\mathcal{S}$ is "perfectly balanced": $w(\partial_G S_i) = \frac{2}{k}\mathrm{OPT} \;\; \forall i$

- Could try taking $\mathcal{A} = \{A \subseteq V : w(\partial_G A) \leq \frac{2}{k}\mathrm{OPT}\}$

$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \leq \frac{1.49}{k}\mathrm{OPT}\}$

$|\mathcal{A}'| = O_k(n)$

$\mathcal{A}'$ computed in polytime

What if $\mathcal{S}$ "balanced"?

Example: $\mathcal{S}$ is "perfectly balanced": $w(\partial_G S_i) = \frac{2}{k}\mathrm{OPT} \;\; \forall i$

- Could try taking $\mathcal{A} = \{A \subseteq V : w(\partial_G A) \leq \frac{2}{k}\mathrm{OPT}\}$
- Problem: can have $|\mathcal{A}| \geq \Omega(n^2)$.

  $\Rightarrow$ recursion is $T(n,k) \geq \Omega(n^2) \cdot T(n, k-1)$

  solves to n^{2k}, no good!

$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \leq \frac{1.49}{k}\mathrm{OPT}\}$

$|\mathcal{A}'| = O_k(n)$

$\mathcal{A}'$ computed in polytime

$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \le \frac{1.49}{k} OPT\}$

$|\mathcal{A}'| = O_k(n)$

$\mathcal{A}'$ computed in polytime

What if $\mathcal{S}$ "balanced"?

Example: $\mathcal{S}$ is "perfectly balanced": $w(\partial_G S_i) = \frac{2}{k} OPT \ \forall i$

- Could try taking $\mathcal{A} = \{A \subseteq V : w(\partial_G A) \le \frac{2}{k} OPT\}$
- Problem: can have $|\mathcal{A}| \ge \Omega(n^2)$.

  $\Longrightarrow$ recursion is $T(n,k) \ge \Omega(n^2) \cdot T(n,k-1)$

  solves to n^{2k}, no good!
- Message: cannot restart from scratch each time. Keep global measure of progress.

What if $\mathcal{S}$ "balanced"?

$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \leq \frac{1.49}{k} OPT\}$

$|\mathcal{A}'| = O_k(n)$

$\mathcal{A}'$ computed in polytime

Example: $\mathcal{S}$ is "perfectly balanced": $w(\partial_G S_i) = \frac{2}{k} OPT \ \forall i$

- Could try taking $\mathcal{A} = \{A \subseteq V : w(\partial_G A) \leq \frac{2}{k} OPT\}$
- Problem: can have $|\mathcal{A}| \geq \Omega(n^2)$.
  $\Rightarrow$ recursion is $T(n,k) \geq \Omega(n^2) \cdot T(n,k-1)$
  solves to n^{2k}, no good!
- Message: cannot restart from scratch each time. Keep global measure of progress.
- Every time we branch on a large $\mathcal{A}$ (e.g. $|\mathcal{A}| \geq \Omega(n^2)$), make sure we make enough progress.

What if $\mathcal{S}$ "balanced"?

$\mathcal{A}' := \{A \subseteq V : w(\partial_G A) \le \frac{1.49}{k} OPT\}$

$|\mathcal{A}'| = O_k(n)$

$\mathcal{A}'$ computed in polytime

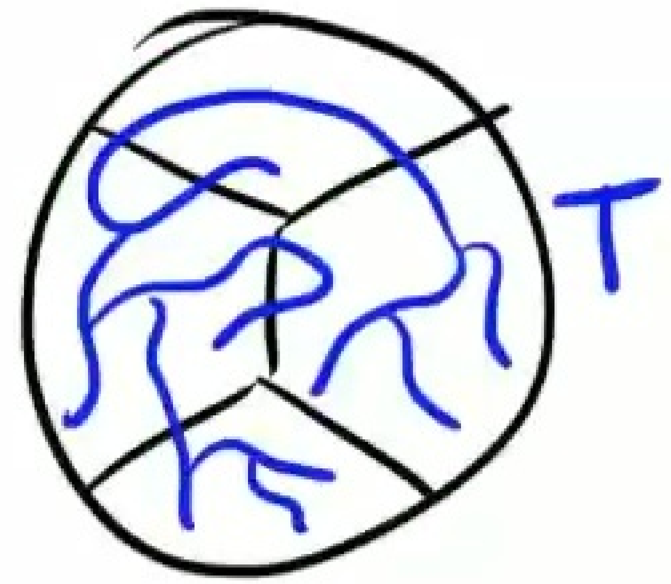
Example: $\mathcal{S}$ is "perfectly balanced": $w(\partial_G S_i) = \frac{2}{k} OPT \;\; \forall i$

- Could try taking $\mathcal{A} = \{A \subseteq V : w(\partial_G A) \le \frac{2}{k} OPT\}$
- Problem: can have $|\mathcal{A}| \ge \Omega(n^2)$.
  $\Longrightarrow$ recursion is $T(n,k) \ge \Omega(n^2) \cdot T(n, k-1)$
  solves to n^{2k}, no good!
- Message: cannot restart from scratch each time. Keep global measure of progress.
- Every time we branch on a large $\mathcal{A}$ (e.g. $|\mathcal{A}| \ge \Omega(n^2)$), make sure we make enough progress.
- Progress is based on tree packing [Thorup 2008].

Thorup's Tree Packing

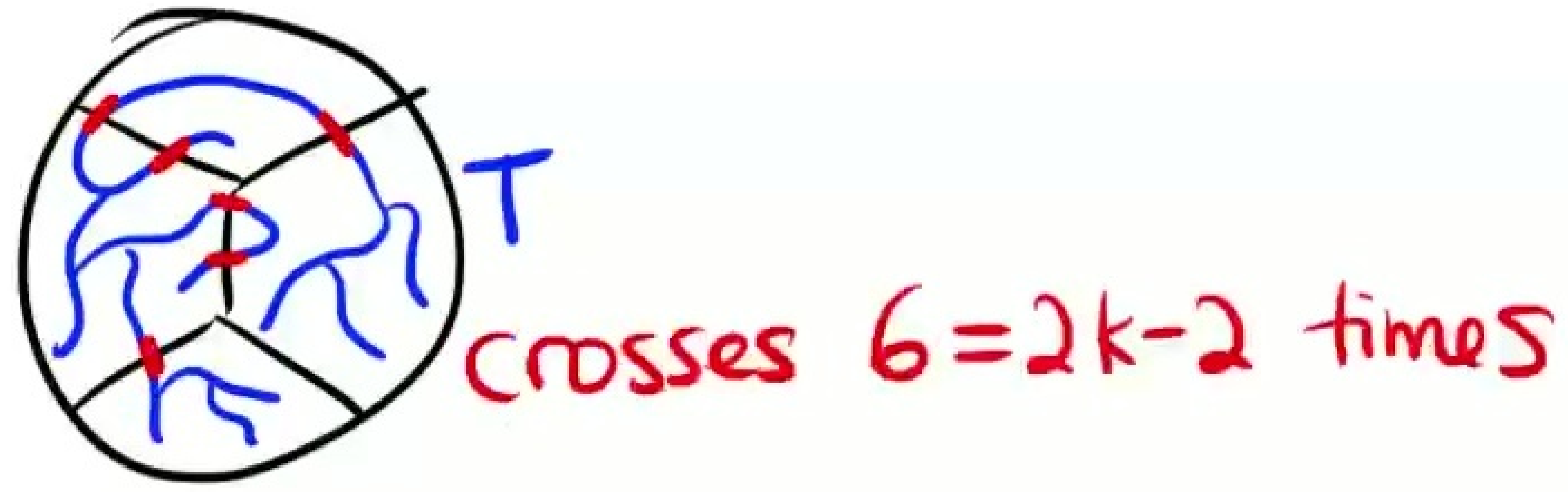
Thm [Thorup '08] Can compute in poly(n) time a set $\mathcal{T}$ of poly(n) spanning trees of G, s.t. for any min k-cut S, there exists $T \in \mathcal{T}$ that crosses S $\le 2k-2$ times.

$k=4$:



Thorup's Tree Packing

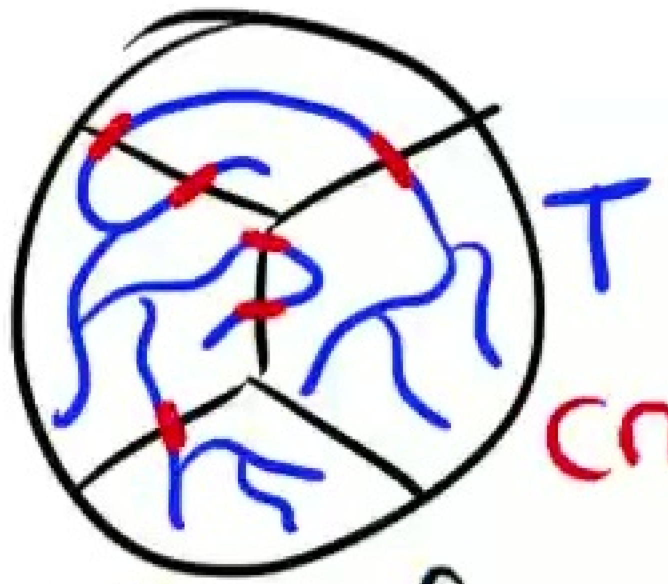
Thm [Thorup '08] Can compute in poly(n) time a set $\mathcal{T}$ of poly(n) spanning trees of G, s.t. for any min k-cut S, there exists $T \in \mathcal{T}$ that crosses S $\leq 2k-2$ times.

k=4: 

Thorup's Tree Packing

Thm [Thorup '08] Can compute in poly(n) time a set $\mathcal{T}$ of poly(n) spanning trees of G, s.t. for any min k-cut S, there exists $T \in \mathcal{T}$ that crosses S $\leq 2k-2$ times.

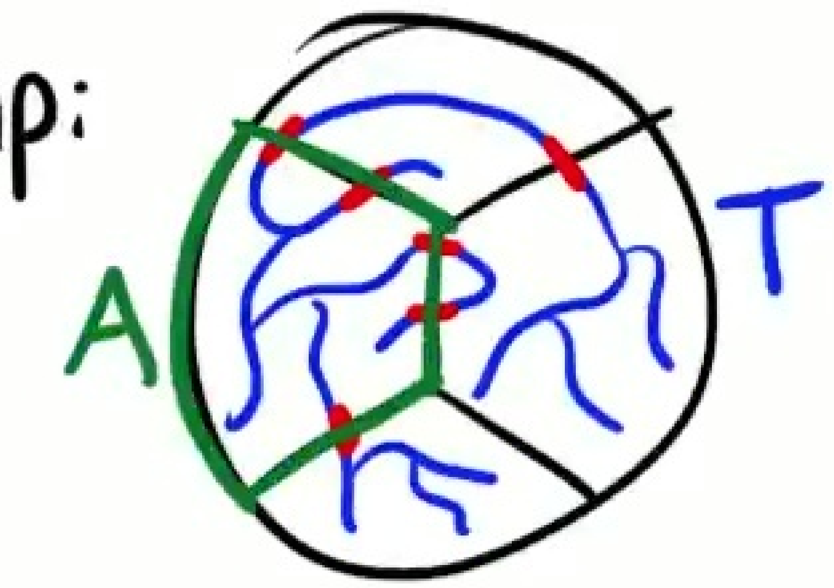
$k=4$:



Thorup's min k-cut algo: for each $T \in \mathcal{T}$, for every way to delete $\leq 2k-2$ edges of T and merge the components into k components, compute the min k-cut value.

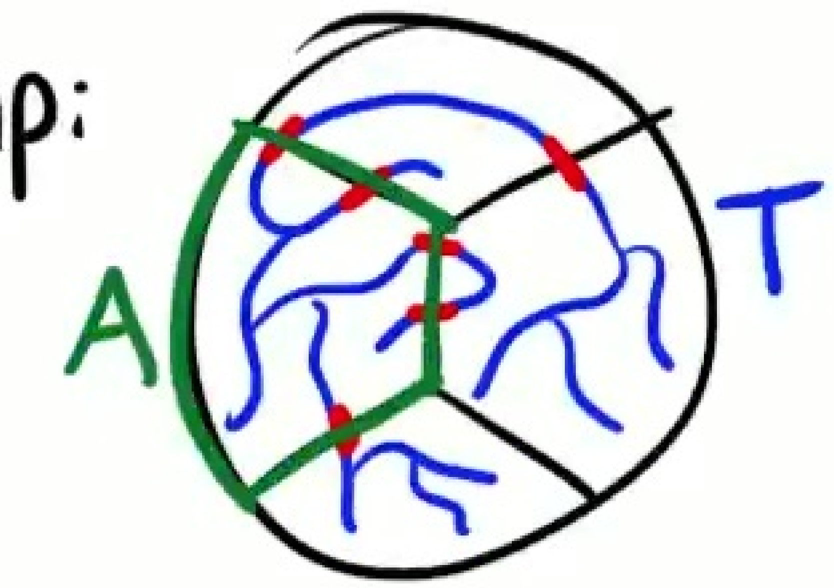
Tree as a Measure of Progress

- Suppose $\mathcal{A}$ contains this comp:



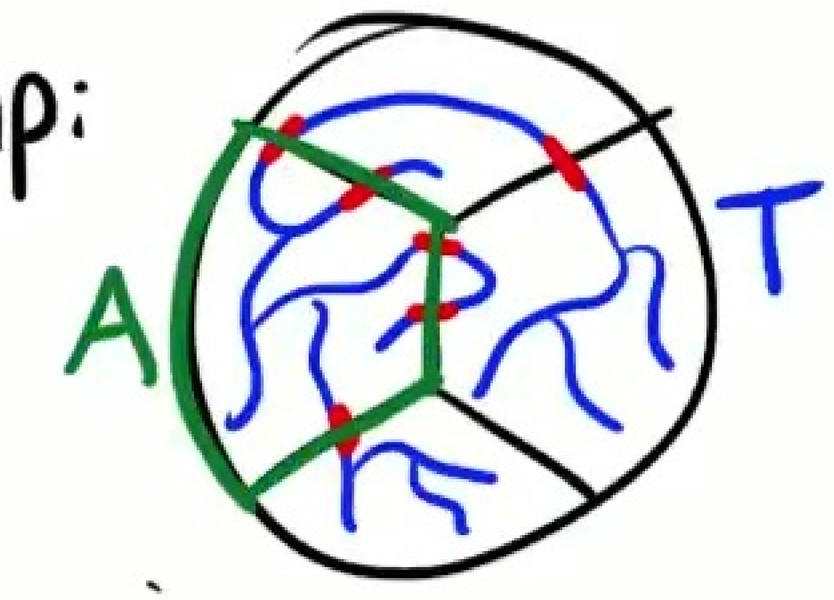
Tree as a Measure of Progress

- Suppose $\mathcal{A}$ contains this comp:
- If we cut out A, we also delete many edges of T.



Tree as a Measure of Progress

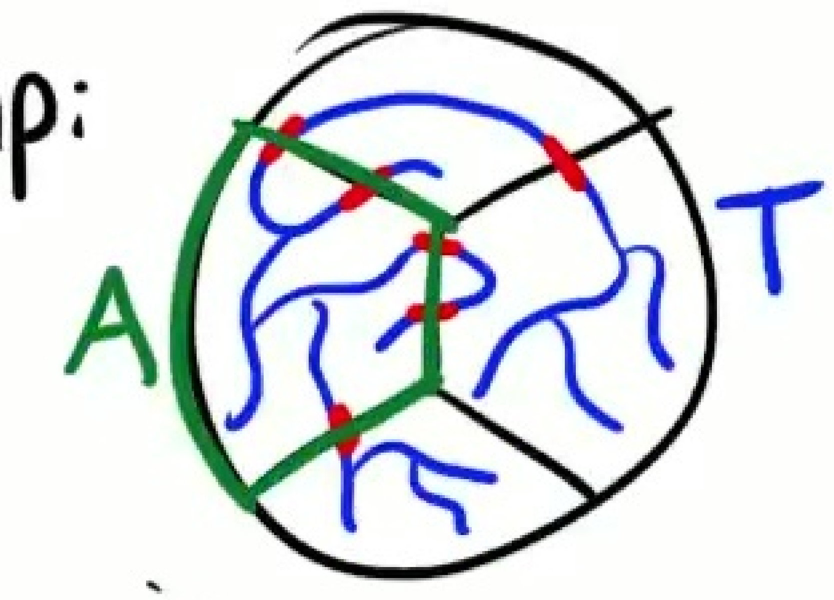
- Suppose $\mathcal{A}$ contains this comp:



- If we cut out A, we also delete many edges of T.
- Guessing $A \in \mathcal{A}$ is "worth" guessing 5 edges of T to delete (in Thorup's brute-force algo)

Tree as a Measure of Progress

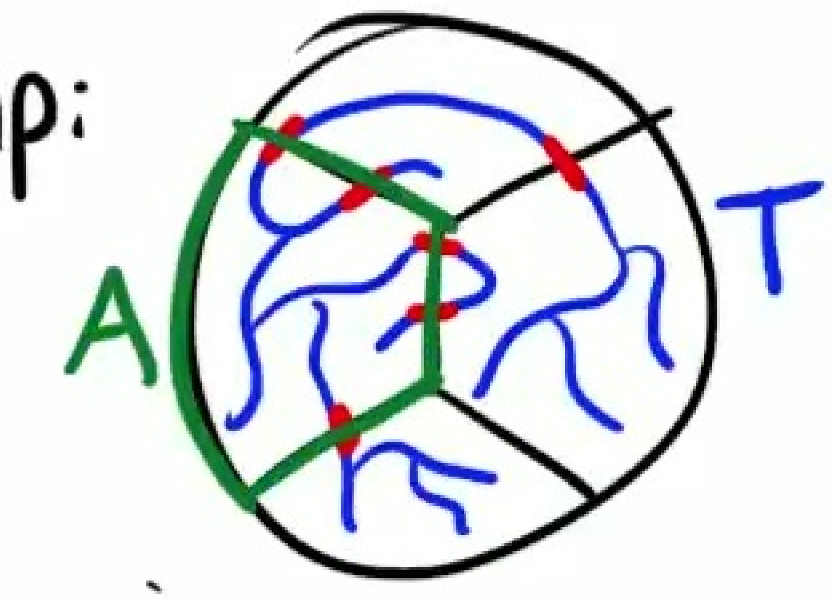
- Suppose $\mathcal{A}$ contains this comp:



- If we cut out A, we also delete many edges of T.
- Guessing $A \in \mathcal{A}$ is "worth" guessing 5 edges of T to delete (in Thorup's brute-force algo)
- As long as $|\mathcal{A}| \ll n^5$, we can guess 5 edges in $\ll n^5$ time

Tree as a Measure of Progress

- Suppose $\mathcal{A}$ contains this comp:

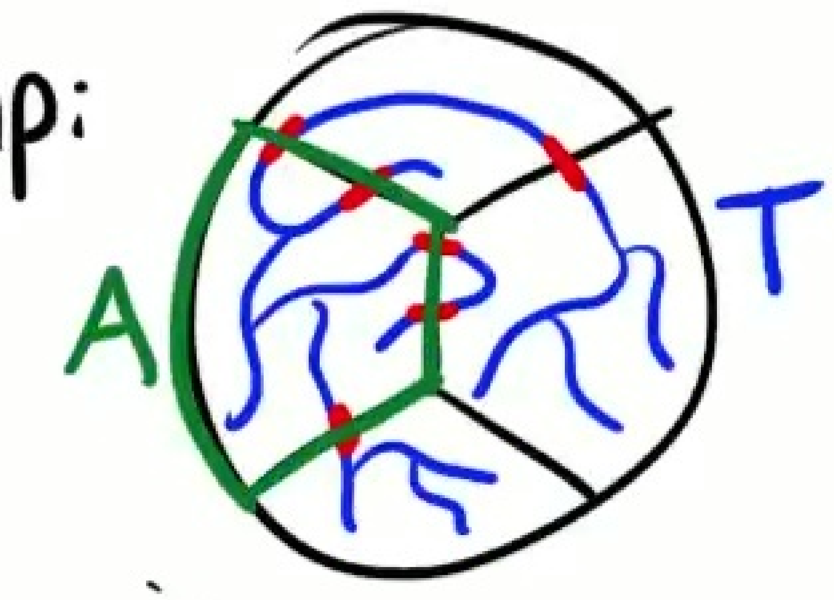


- If we cut out A, we also delete many edges of T.
- Guessing $A \in \mathcal{A}$ is "worth" guessing 5 edges of T to delete (in Thorup's brute-force algo)
- As long as $|\mathcal{A}| \ll n^5$, we can guess 5 edges in $\ll n^5$ time

$\Rightarrow$ can "beat brute force"

Tree as a Measure of Progress

- Suppose $\mathcal{A}$ contains this comp:



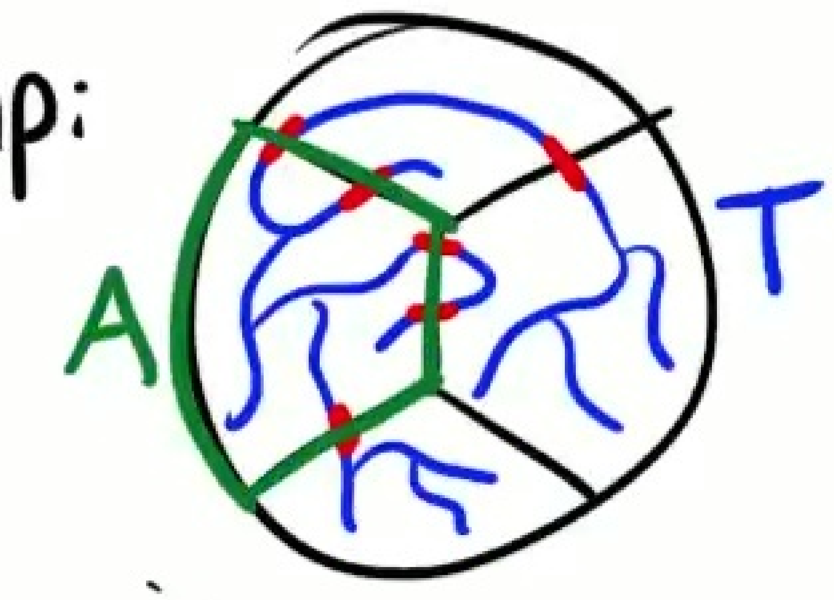
- If we cut out A, we also delete many edges of T.
- Guessing $A \in \mathcal{A}$ is "worth" guessing 5 edges of T to delete (in Thorup's brute-force algo)
- As long as $|\mathcal{A}| \ll n^5$, we can guess 5 edges in $\ll n^5$ time

$\Longrightarrow$ can "beat brute force"

Measure of progress: how many edges of T have we deleted?

Tree as a Measure of Progress

- Suppose $\mathcal{A}$ contains this comp:

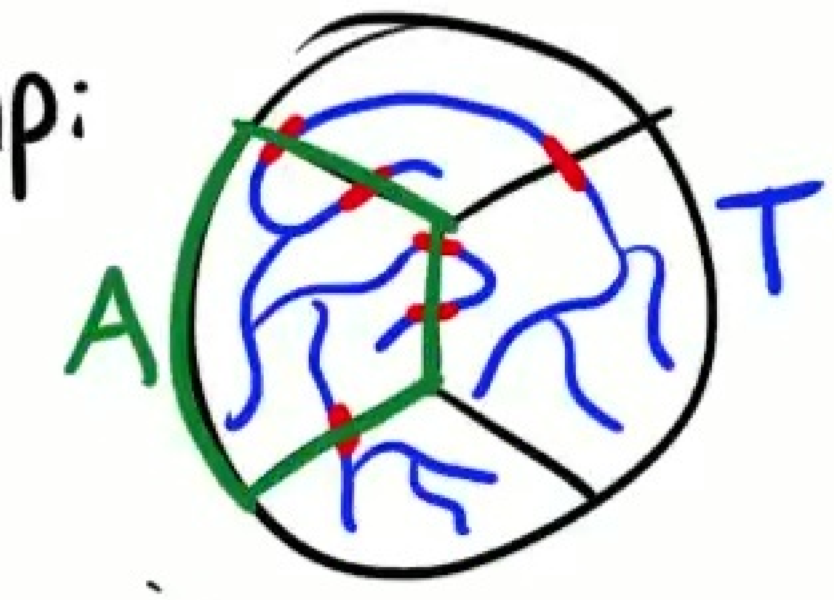


- If we cut out A, we also delete many edges of T.
- Guessing $A \in \mathcal{A}$ is "worth" guessing 5 edges of T to delete (in Thorup's brute-force algo)
- As long as $|\mathcal{A}| \ll n^5$, we can guess 5 edges in $\ll n^5$ time

$\Longrightarrow$ can "beat brute force"

Measure of progress: how many edges of T have we deleted?
If deleted s so far, only need to guess remaining $(2k-2)-s$ edges (n^{2k-2-s} time).

Tree as a Measure of Progress



- Suppose $\mathcal{A}$ contains this comp:
- If we cut out A, we also delete many edges of T.
- Guessing $A \in \mathcal{A}$ is "worth" guessing 5 edges of T to delete (in Thorup's brute-force algo)
- As long as $|\mathcal{A}| \ll n^5$, we can guess 5 edges in $\ll n^5$ time

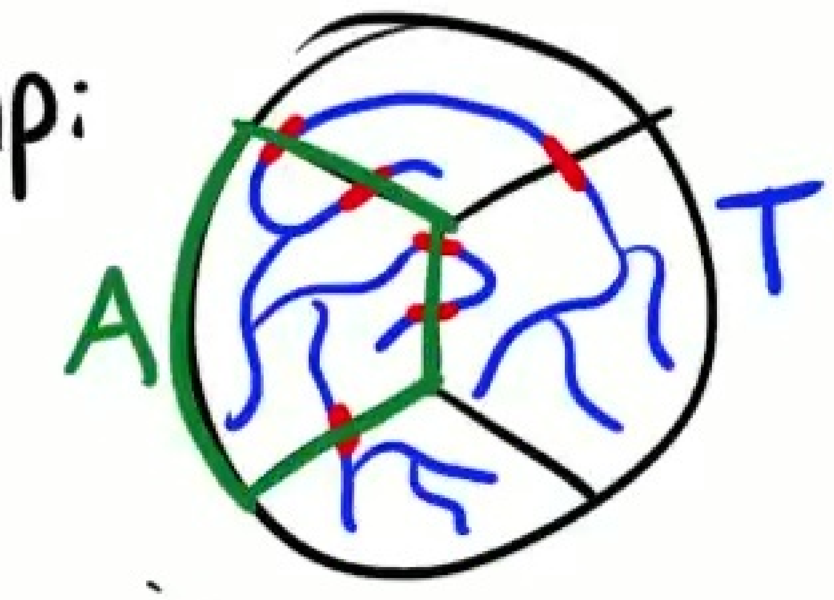
$\Longrightarrow$ can "beat brute force"

Measure of progress: how many edges of T have we deleted?
If deleted s so far, only need to guess remaining $(2k-2)-s$ edges (n^{2k-2-s} time).

If $|\mathcal{A}| = n^\beta$ and $\exists A \in \mathcal{A}$ cutting $> \beta$ edges of T, then we make progress.

Tree as a Measure of Progress

- Suppose $\mathcal{A}$ contains this comp:
- If we cut out A, we also delete many edges of T.



- Guessing $A \in \mathcal{A}$ is "worth" guessing 5 edges of T to delete (in Thorup's brute-force algo)
- As long as $|\mathcal{A}| \ll n^5$, we can guess 5 edges in $\ll n^5$ time

$\Longrightarrow$ can "beat brute force"

Measure of progress: how many edges of T have we deleted?
If deleted s so far, only need to guess remaining $(2k-2)-s$ edges (n^{2k-2-s} time).

If $|\mathcal{A}| = n^\beta$ and $\exists A \in \mathcal{A}$ cutting $>\beta$ edges of T, then we make progress.

E.g. $|\mathcal{A}'| = O_k(n)$, so if $\exists S_i \in \mathcal{A}'$ cutting ≥ 2 edges of T, then branch on $\mathcal{A}$.

Balanced Example

- Suppose G is "balanced": $w(\partial_G S_i) = \frac{2}{k}\mathrm{OPT}$ for all S_i.

Balanced Example

- Suppose G is "balanced": $w(\partial_G S_i) = \frac{2}{k}\mathrm{OPT}$ for all S_i.
- To make progress, find $\mathcal{A}$, $|\mathcal{A}| = O_k(n^{4-\varepsilon})$, s.t. $\exists\, S_i \in \mathcal{A}$ cutting ≥ 4 edges of T: $|\partial_T S_i| \geq 4$

Balanced Example

- Suppose G is "balanced": $w(\partial_G S_i) = \frac{2}{k} OPT$ for all S_i.
- To make progress, find $\mathcal{A}$, $|\mathcal{A}| = O_k(n^{4-\varepsilon})$, s.t. $\exists\, S_i \in \mathcal{A}$ cutting ≥ 4 edges of T: $|\partial_T S_i| \geq 4$
- Such S_i exists because $\sum |\partial_T S_i| = 2(2k-2) \Rightarrow$ avg $|\partial_T S_i|$ is $\frac{4k-4}{k} \approx 4$

Balanced Example

- Suppose G is "balanced": $w(\partial_G S_i) = \frac{2}{k}\mathrm{OPT}$ for all S_i.
- To make progress, find $\mathcal{A}$, $|\mathcal{A}| = O_k(n^{4-\varepsilon})$, s.t. $\exists\, S_i \in \mathcal{A}$ cutting ≥ 4 edges of T: $|\partial_T S_i| \geq 4$
- Such S_i exists because $\sum |\partial_T S_i| = 2(2k-2) \Rightarrow$ avg $|\partial_T S_i|$ is $\frac{4k-4}{k} \approx 4$

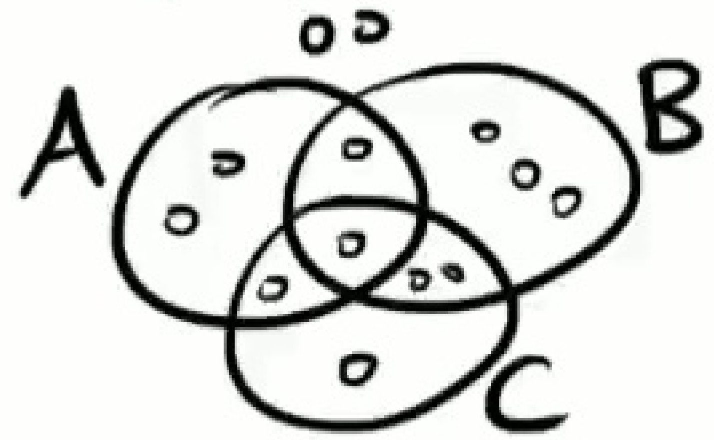
Thm: Let $\mathcal{A} := \{ A \subseteq V : w(\partial_G S_i) \leq \frac{2.3}{k}\mathrm{OPT} \}$. Then, $|\mathcal{A}| \leq n^{3.75}$

Balanced Example

- Suppose G is "balanced": $w(\partial_G S_i) = \frac{2}{k}\mathrm{OPT}$ for all S_i.
- To make progress, find $\mathcal{A}$, $|\mathcal{A}| = O_k(n^{4-\varepsilon})$, s.t. $\exists\, S_i \in \mathcal{A}$ cutting ≥ 4 edges of T: $|\partial_T S_i| \geq 4$
- Such S_i exists because $\sum |\partial_T S_i| = 2(2k-2) \Rightarrow$ avg $|\partial_T S_i|$ is $\frac{4k-4}{k} \approx 4$

Thm: Let $\mathcal{A} := \{ A \subseteq V : w(\partial_G S_i) \leq \frac{2.3}{k}\mathrm{OPT} \}$. Then, $|\mathcal{A}| \leq n^{3.75}$

Pf: Use extremal result: if $|\mathcal{A}| \geq \Omega(n^{3.75})$, then $\exists$ sets A, B, C:

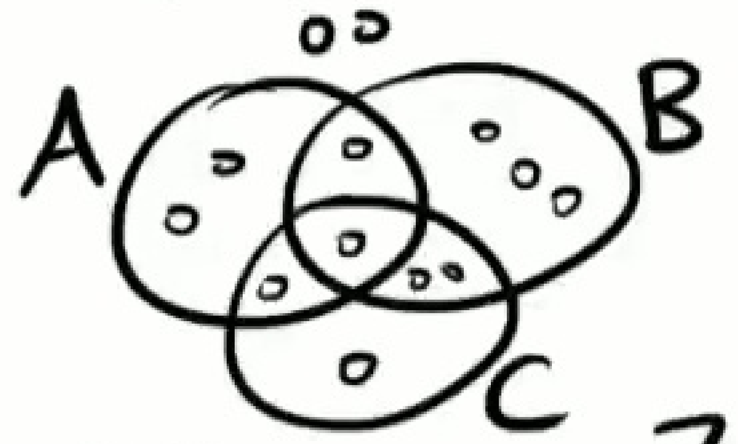


Balanced Example

- Suppose G is "balanced": $w(\partial_G S_i) = \frac{2}{k}\mathrm{OPT}$ for all S_i.
- To make progress, find $\mathcal{A}$, $|\mathcal{A}| = O_k(n^{4-\varepsilon})$, s.t. $\exists\, S_i \in \mathcal{A}$ cutting ≥ 4 edges of T: $|\partial_T S_i| \geq 4$
- Such S_i exists because $\sum |\partial_T S_i| = 2(2k-2) \Rightarrow$ avg $|\partial_T S_i|$ is $\frac{4k-4}{k} \approx 4$

Thm: Let $\mathcal{A} := \{ A \subseteq V : w(\partial_G S_i) \leq \frac{2.3}{k}\mathrm{OPT} \}$. Then, $|\mathcal{A}| \leq n^{3.75}$

Pf: Use extremal result: if $|\mathcal{A}| \geq \Omega(n^{3.75})$, then $\exists$ sets A, B, C:



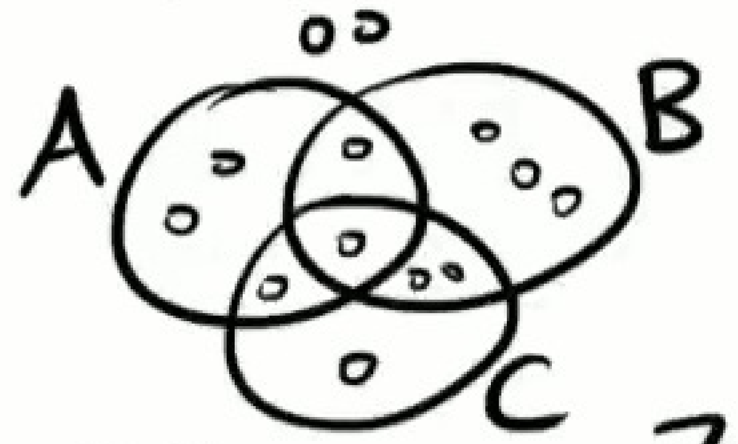
Cut out A, B, C: pay $3 \cdot \frac{2.3}{k}\mathrm{OPT} = \frac{7-\varepsilon}{k}\mathrm{OPT}$ for $+7$ components.

Balanced Example

- Suppose G is "balanced": $w(\partial_G S_i) = \frac{2}{k}\mathrm{OPT}$ for all S_i.
- To make progress, find $\mathcal{A}$, $|\mathcal{A}| = O_k(n^{4-\varepsilon})$, s.t. $\exists\, S_i \in \mathcal{A}$ cutting ≥ 4 edges of T: $|\partial_T S_i| \geq 4$
- Such S_i exists because $\sum |\partial_T S_i| = 2(2k-2) \Rightarrow$ avg $|\partial_T S_i|$ is $\frac{4k-4}{k} \approx 4$

Thm: Let $\mathcal{A} := \{A \subseteq V : w(\partial_G S_i) \leq \frac{2.3}{k}\mathrm{OPT}\}$. Then, $|\mathcal{A}| \leq n^{3.75}$

Pf: Use extremal result: if $|\mathcal{A}| \geq \Omega(n^{3.75})$, then $\exists$ sets A, B, C:



Cut out A, B, C: pay $3 \cdot \frac{2.3}{k}\mathrm{OPT} = \frac{7-\varepsilon}{k}\mathrm{OPT}$ for $+7$ components.

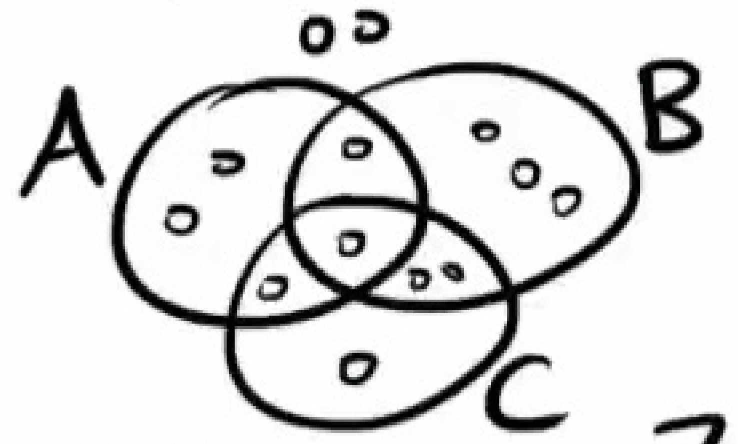
Construct cut iteratively, cost $< \mathrm{OPT}$, contradiction.

Balanced Example

- Suppose G is "balanced": $w(\partial_G S_i) = \frac{2}{k}\mathrm{OPT}$ for all S_i.
- To make progress, find $\mathcal{A}$, $|\mathcal{A}| = O_k(n^{4-\varepsilon})$, s.t. $\exists\, S_i \in \mathcal{A}$ cutting ≥ 4 edges of T: $|\partial_T S_i| \geq 4$
- Such S_i exists because $\sum |\partial_T S_i| = 2(2k-2) \Rightarrow$ avg $|\partial_T S_i|$ is $\frac{4k-4}{k} \approx 4$

Thm: Let $\mathcal{A} := \{ A \subseteq V : w(\partial_G S_i) \leq \frac{2.3}{k}\mathrm{OPT} \}$. Then, $|\mathcal{A}| \leq n^{3.75}$

Pf: Use extremal result: if $|\mathcal{A}| \geq \Omega(n^{3.75})$, then $\exists$ sets A, B, C:



Cut out A, B, C: pay $3 \cdot \frac{2.3}{k}\mathrm{OPT} = \frac{7-\varepsilon}{k}\mathrm{OPT}$ for $+7$ components.

Construct cut iteratively, cost $<$ OPT, contradiction.

To find $\mathcal{A}$: run modified Karger-Stein again, output smallest $\Theta(n^{3.75})$ cuts.

Balanced Example

- Suppose G is "balanced": $w(\partial_G S_i) = \frac{2}{k}\mathrm{OPT}$ for all S_i.
- To make progress, find $\mathcal{A}$, $|\mathcal{A}| = O_k(n^{4-\varepsilon})$, s.t.
  $\exists\, S_i \in \mathcal{A}$ cutting ≥ 4 edges of T: $|\partial_T S_i| \geq 4$
- Such S_i exists because $\sum |\partial_T S_i| = 2(2k-2) \Rightarrow$ avg $|\partial_T S_i|$ is $\frac{4k-4}{k} \approx 4$

Thm: 2.3 3.75 A, B, C:

Pf:

Recursion?

ponents.

To find $\mathcal{A}$: run modified Karger-Stein again, output smallest $\Theta(n^{3.75})$ cuts.

Balanced Example

- Suppose G is "balanced": $w(\partial_G S_i) = \frac{2}{k}\mathrm{OPT}$ for all S_i.
- To make progress, find $\mathcal{A}$, $|\mathcal{A}| = O_k(n^{4-\varepsilon})$, s.t. $\exists\, S_i \in \mathcal{A}$ cutting ≥ 4 edges of T: $|\partial_T S_i| \geq 4$
- Such S_i exists because $\sum |\partial_T S_i| = 2(2k-2) \Rightarrow$ avg $|\partial_T S_i|$ is $\frac{4k-4}{k} \approx 4$

Thm: … 2.3 … 3.75

Pf: … A, B, C:

Recursion?
If can do this $\Omega(k)$ times, then save $n^{\Omega(k)}$ runtime
$\Rightarrow n^{(2-\varepsilon)k}$ time.

… ponents.

To find $\mathcal{A}$: run modified Karger-Stein again, output smallest $\Theta(n^{3.75})$ cuts.

Balanced Example $(k' < k)$

- Suppose G is "balanced": $w(\partial_G S_i) = \frac{2}{k'}\mathrm{OPT}'$ for all S_i.
- To make progress, find $\mathcal{A}$, $|\mathcal{A}| = O_k(n^{4-\varepsilon})$, s.t. $\exists S_i \in \mathcal{A}$ cutting ≥ 4 edges of T: $|\partial_T S_i| \geq \cancel{4}\ >3k$
- Such S_i exists because $\sum |\partial_T S_i| = \cancel{2(2k-2)}\ >3k \Rightarrow$ avg $|\partial_T S_i|$ is $\cancel{\frac{4k-4}{k'}}\ \frac{>3k}{k'} > 3$

Thm: $\ldots\ 2.3\ \ldots\ 3.75$

A, B, C:

Pf:

Recursion?

If can do this $\Omega(k)$ times, then save $n^{\Omega(k)}$ runtime

$\Rightarrow n^{(2-\varepsilon)k}$ time.

Suppose $>3k$ edges left.

…ponents.

To find $\mathcal{A}$: run modified Karger-Stein again, output smallest $\Theta(n^{3.75})$ cuts.

Balanced Example $(k' < k)$

- Suppose G is "balanced": $w(\partial_G S_i) = \frac{2}{k'}\mathrm{OPT}'$ for all S_i.
- To make progress, find $\mathcal{A}$, $|\mathcal{A}| = O_k(n^{4-\varepsilon})$, s.t. $\exists\, S_i \in \mathcal{A}$ cutting ≥ 4 edges of T: $|\partial_T S_i| \geq 4$
- Such S_i exists because $\sum |\partial_T S_i| = \cancel{2(2k-2)}\ {>3k} \Rightarrow$ avg $|\partial_T S_i|$ is $\frac{\cancel{4k-4}\ >3k}{k'} > 3$

Thm: ... 2.3 ... 3.75

Pf: ...

A, B, C:

...ponents.

> **Recursion?**
>
> If can do this $\Omega(k)$ times, then save $n^{\Omega(k)}$ runtime $\Rightarrow n^{(2-\varepsilon)k}$ time.
>
> Suppose $>3k$ edges left.
> Then, can still pay $n^{3.75}$ to cut 4 edges of T.

To find $\mathcal{A}$: run modified Karger-Stein again, output smallest $\Theta(n^{3.75})$ cuts.

General Case

- Morally, have solved two extreme cases

General Case

- Morally, have solved two extreme cases
  1. Unbalanced: $\exists\, S_i$ with $w(\partial_G S_i) < \frac{1.49}{k} OPT$
  2. Balanced: all S_i satisfy $w(\partial_G S_i) = \frac{2}{k} OPT$

General Case

- Morally, have solved two extreme cases
  1. Unbalanced: $\exists\, S_i$ with $w(\partial_G S_i) < \frac{1.49}{k} OPT$
  2. Balanced: all S_i satisfy $w(\partial_G S_i) = \frac{2}{k} OPT$
- "Interpolate" between the two cases
  (technical; is where we get 1.98 factor)

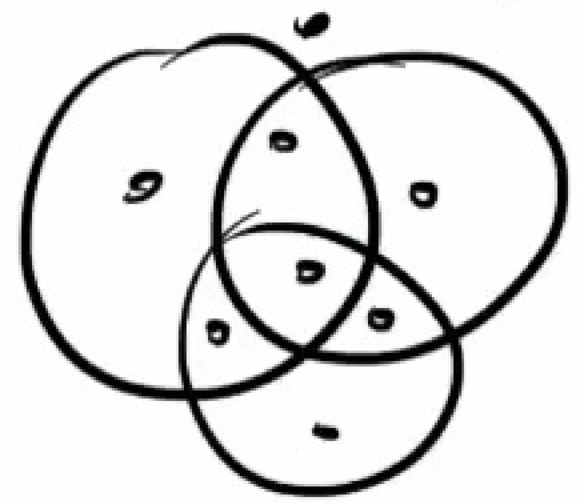
Open Questions

WIP: Beating factor 1.98

Open Questions

WIP: Beating factor 1.98

Conjecture: If $|\mathcal{A}| \geq \Theta(n^3)$, then $\exists A, B, C$:

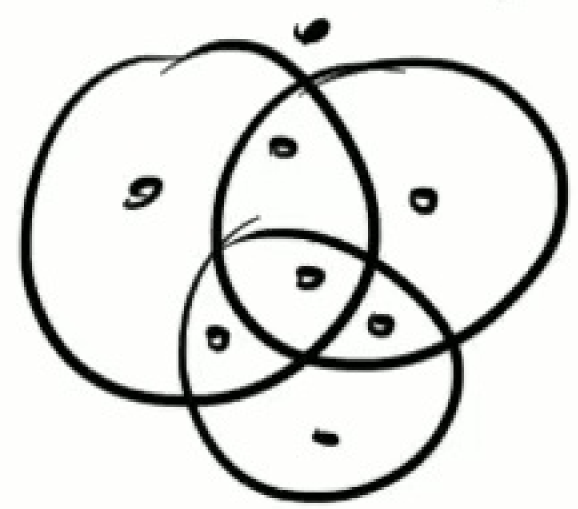


(easy lower bound of $\Omega(n^3)$)

Open Questions

WIP: Beating factor 1.98

Conjecture: If $|\mathcal{A}| \geq \Theta(n^3)$, then $\exists A, B, C$:



(easy lower bound of $\Omega(n^3)$)

k-cut for hypergraphs?