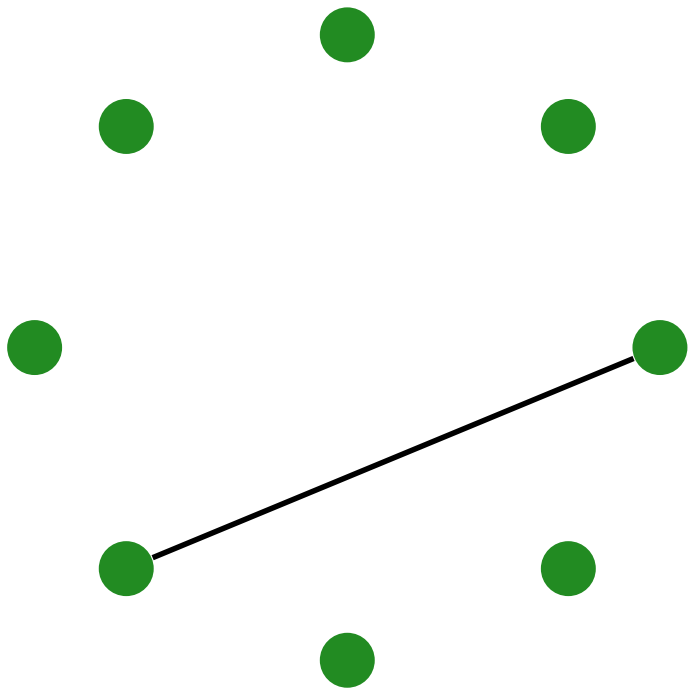
Graph Sketching

Michael Kapralov
EPFL

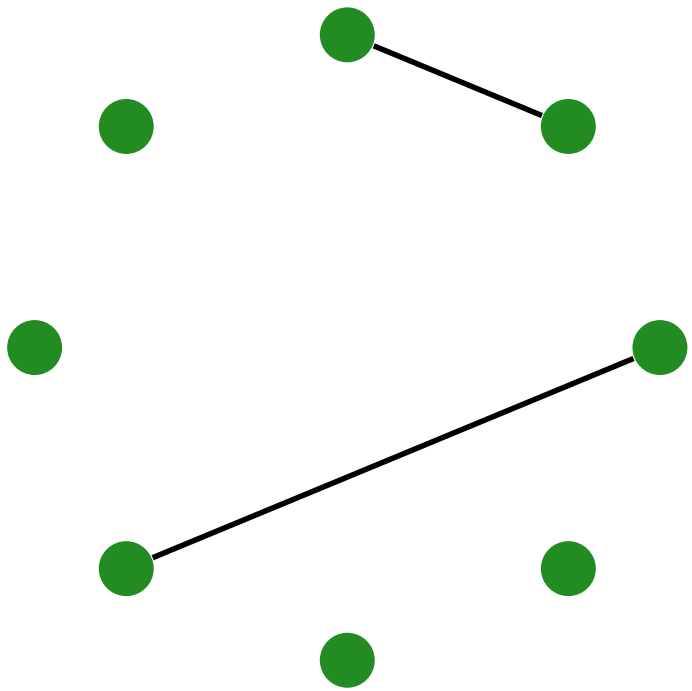
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\tilde{O}(n)$ space
- several passes over the stream



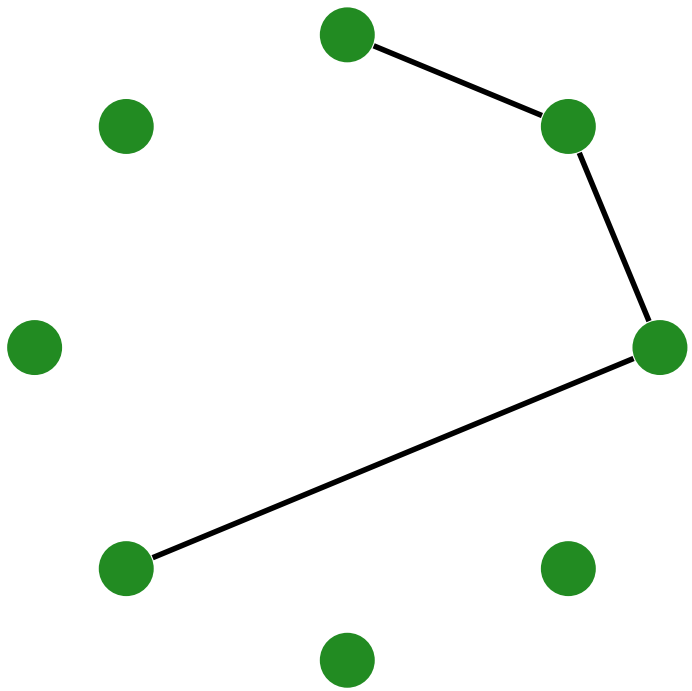
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\tilde{O}(n)$ space
- several passes over the stream



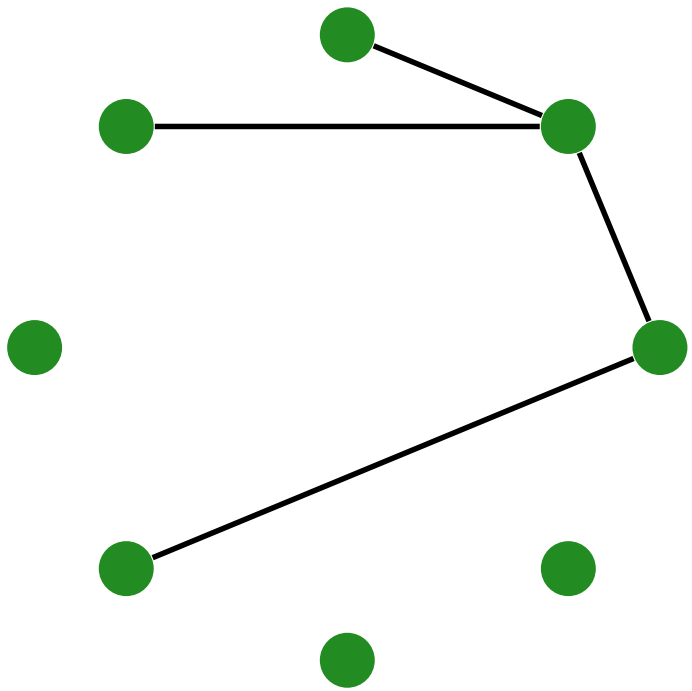
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\tilde{O}(n)$ space
- several passes over the stream



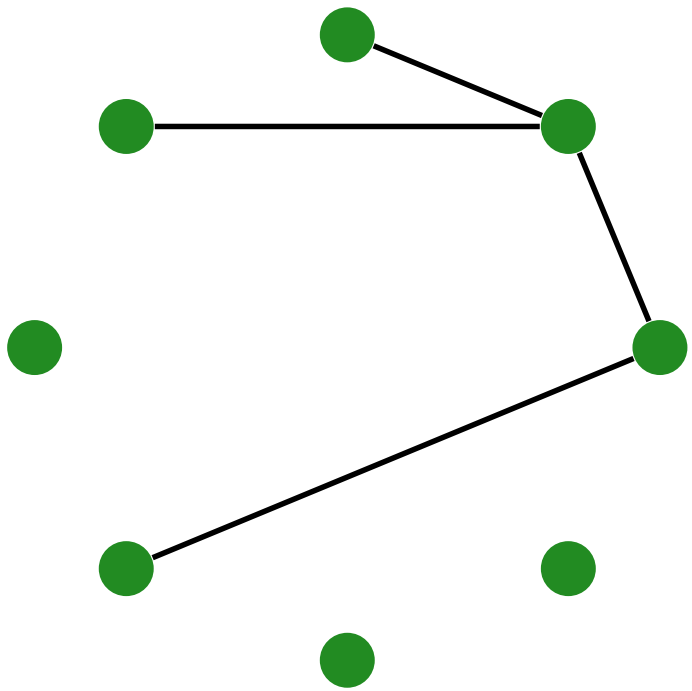
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\tilde{O}(n)$ space
- several passes over the stream



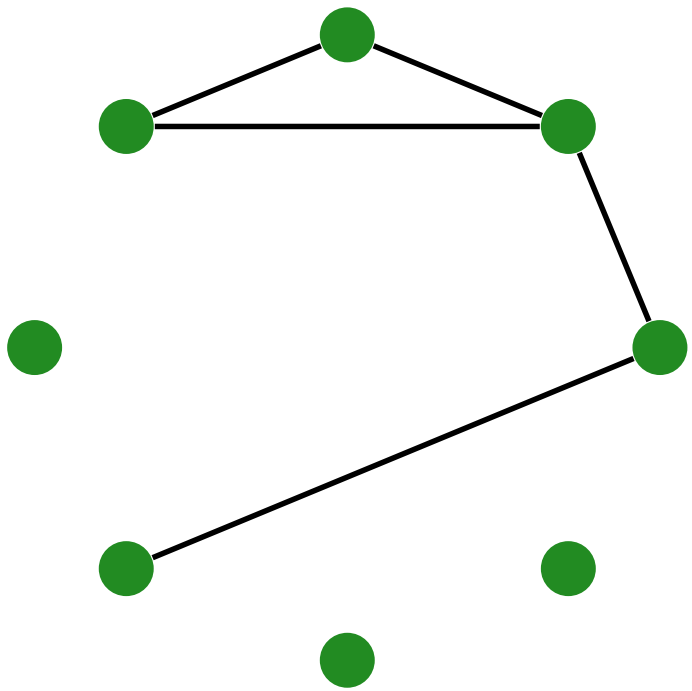
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\tilde{O}(n)$ space
- several passes over the stream



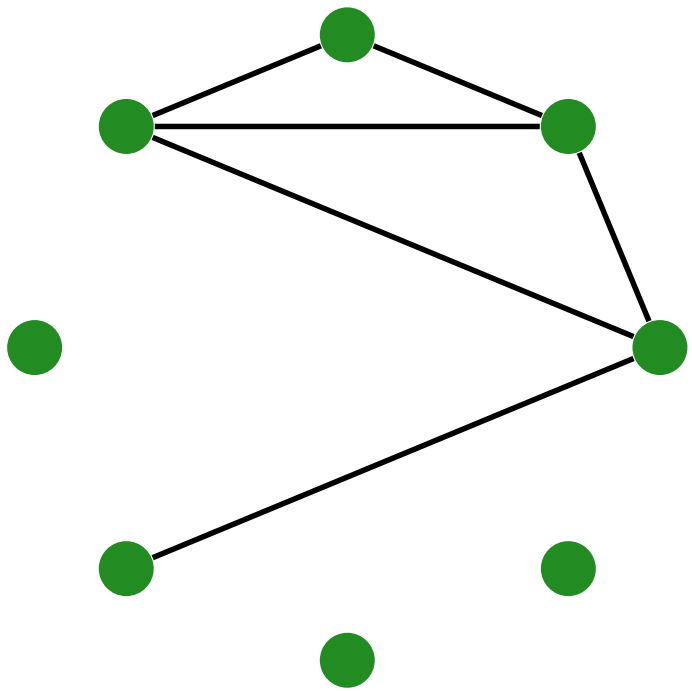
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\tilde{O}(n)$ space
- several passes over the stream



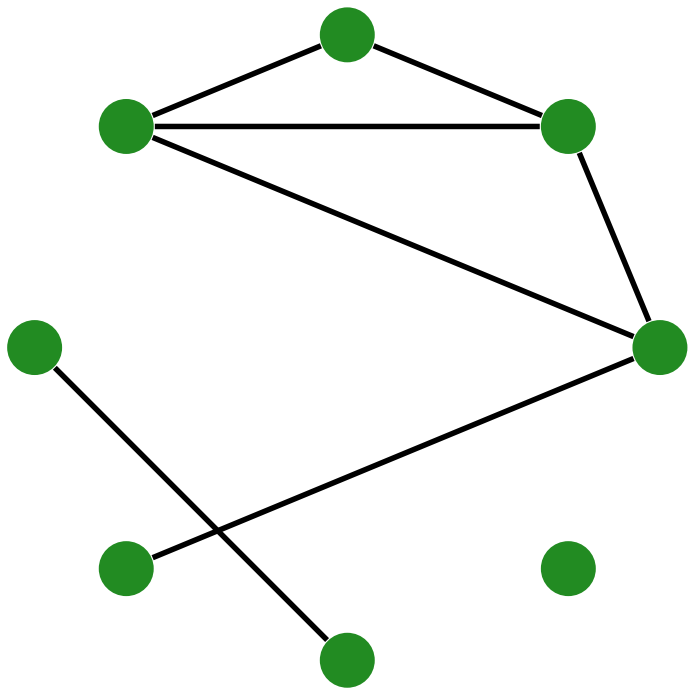
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\widetilde{O}(n)$ space
- several passes over the stream



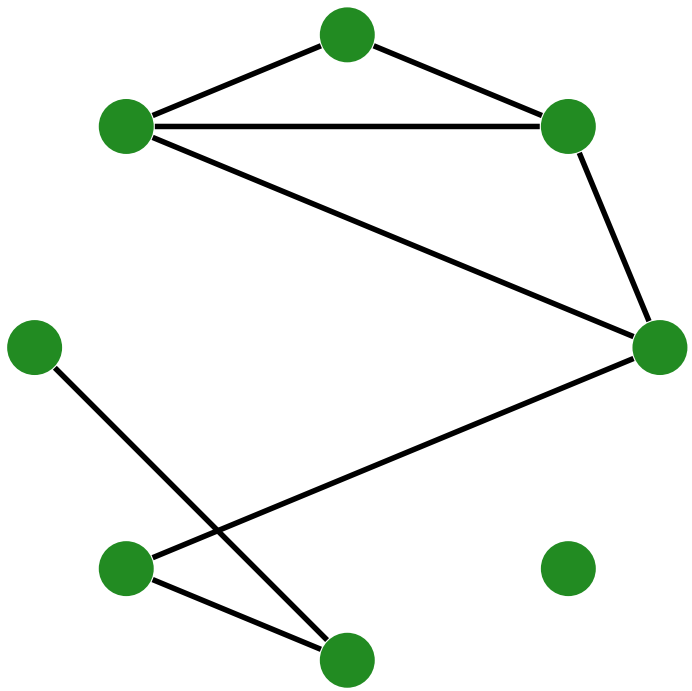
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\tilde{O}(n)$ space
- several passes over the stream



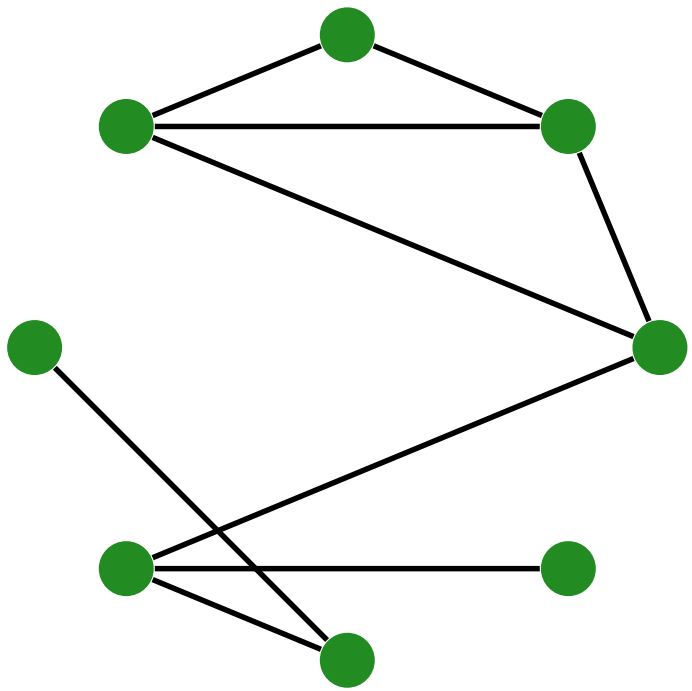
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\widetilde{O}(n)$ space
- several passes over the stream



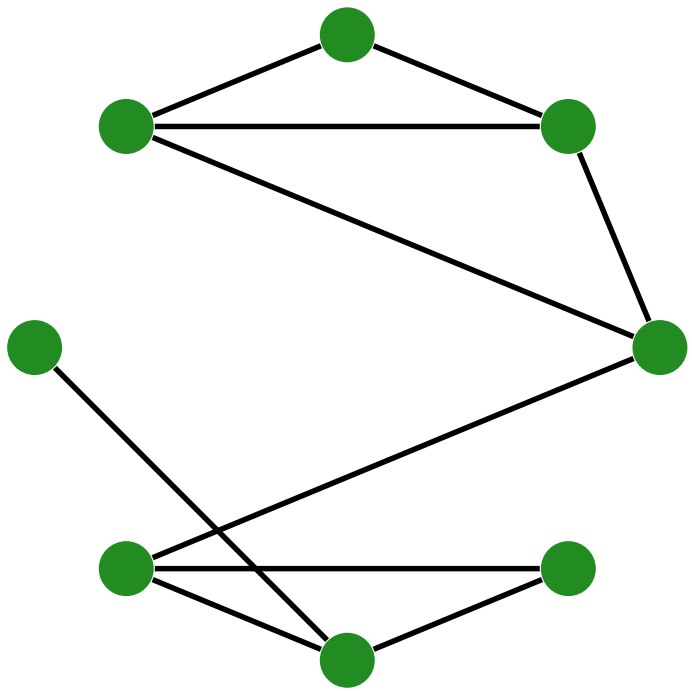
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\widetilde{O}(n)$ space
- several passes over the stream



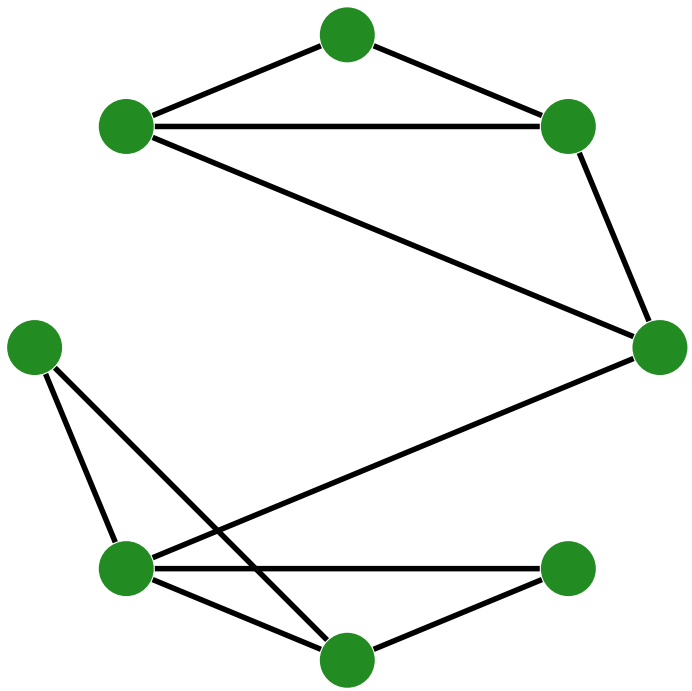
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\tilde{O}(n)$ space
- several passes over the stream



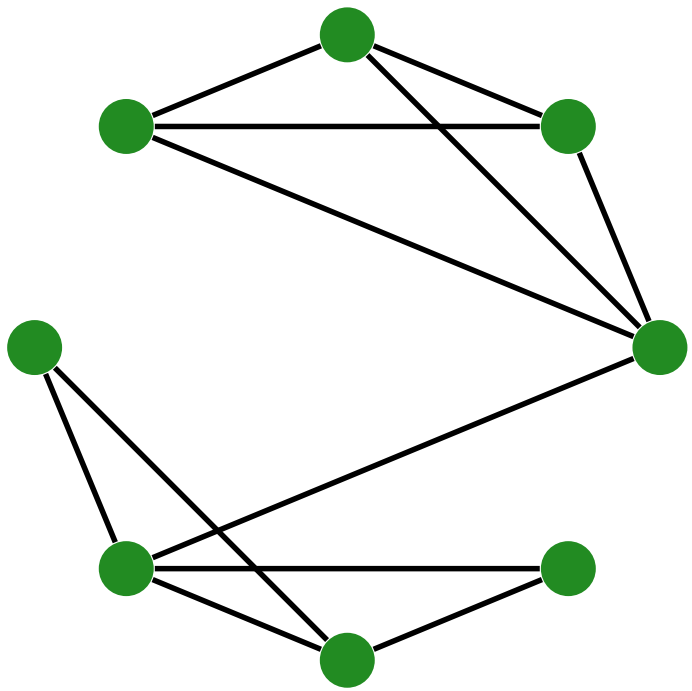
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\widetilde{O}(n)$ space
- several passes over the stream



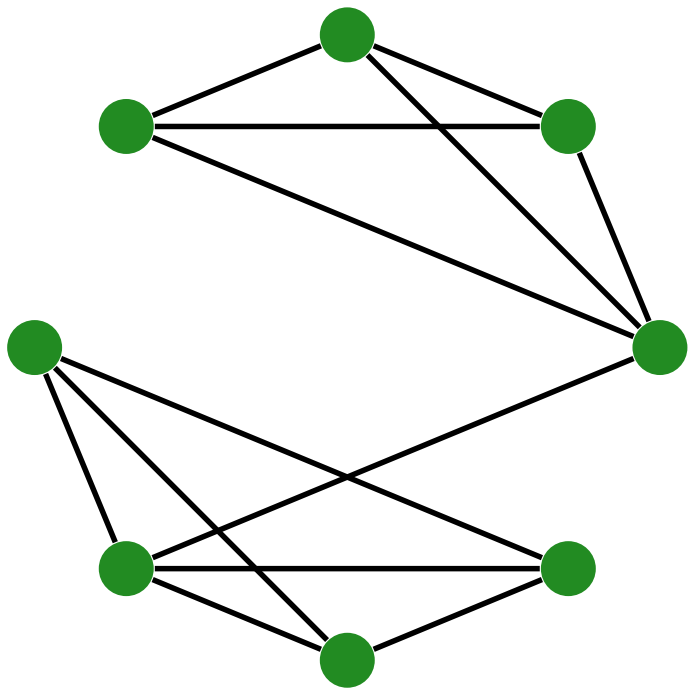
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\tilde{O}(n)$ space
- several passes over the stream



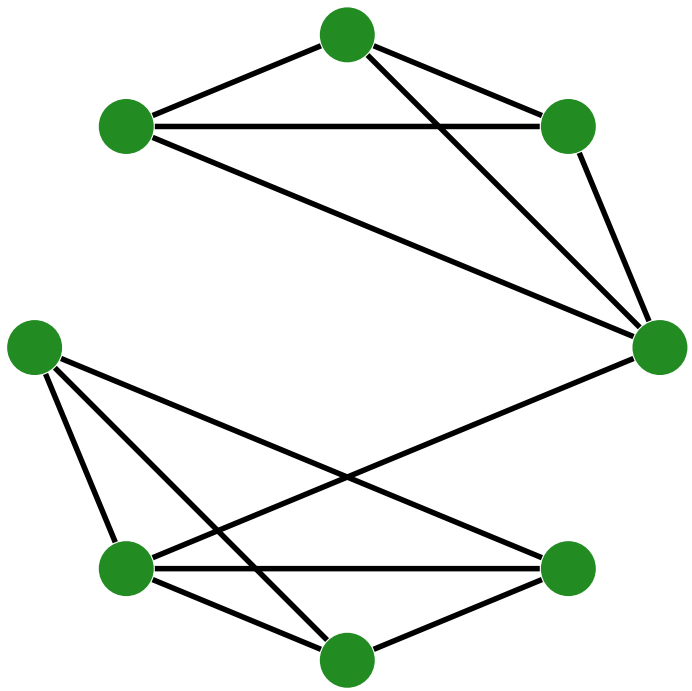
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\tilde{O}(n)$ space
- several passes over the stream



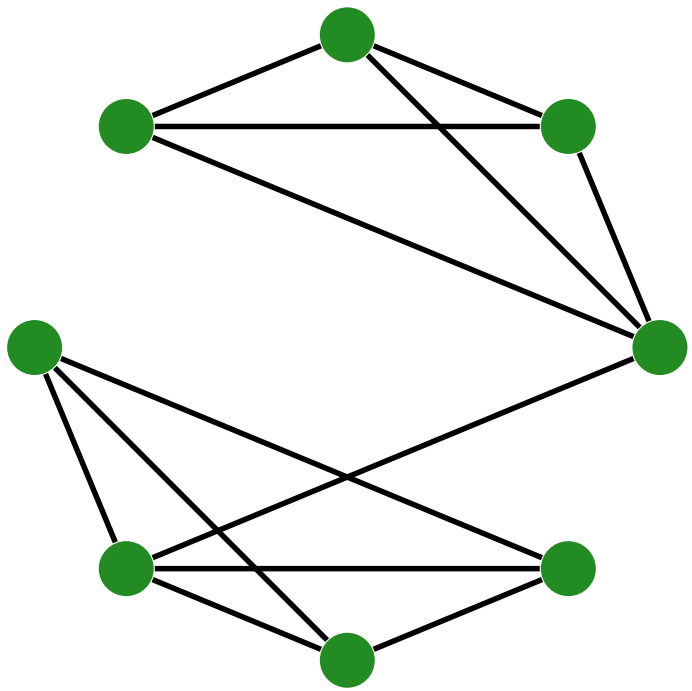
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\tilde{O}(n)$ space
- several passes over the stream



Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\widetilde{O}(n)$ space
- several passes over the stream **(ideally one pass)**



Insertion-only stream

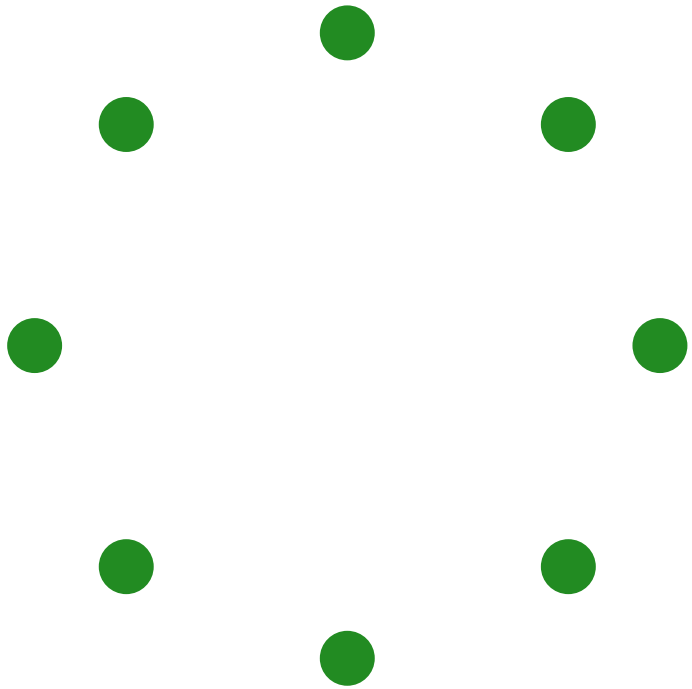
Construct a spanning tree of the graph G in a single pass?

Construct a spanning tree of the graph G in a single pass?

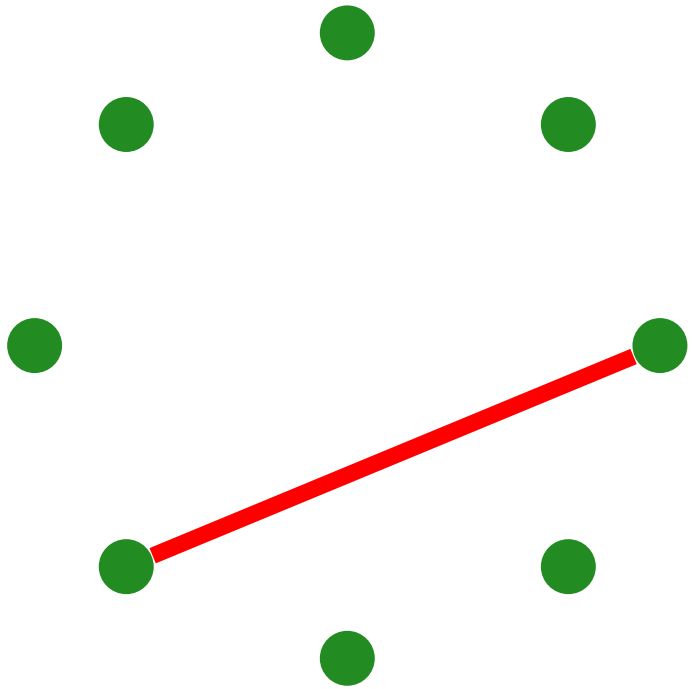
Very easy in insertion only streams:

- maintain a spanning forest
- add incoming edge if it connects two components, discard otherwise

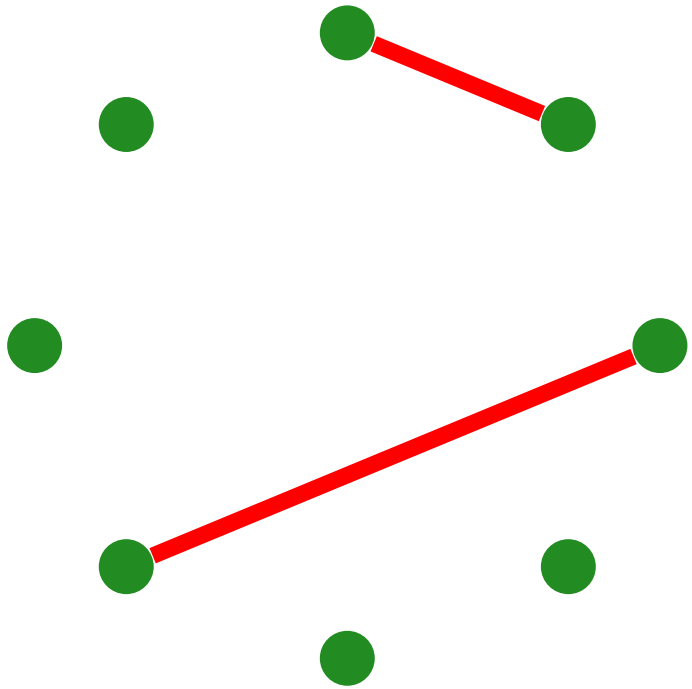
Spanning trees in insertion-only streams



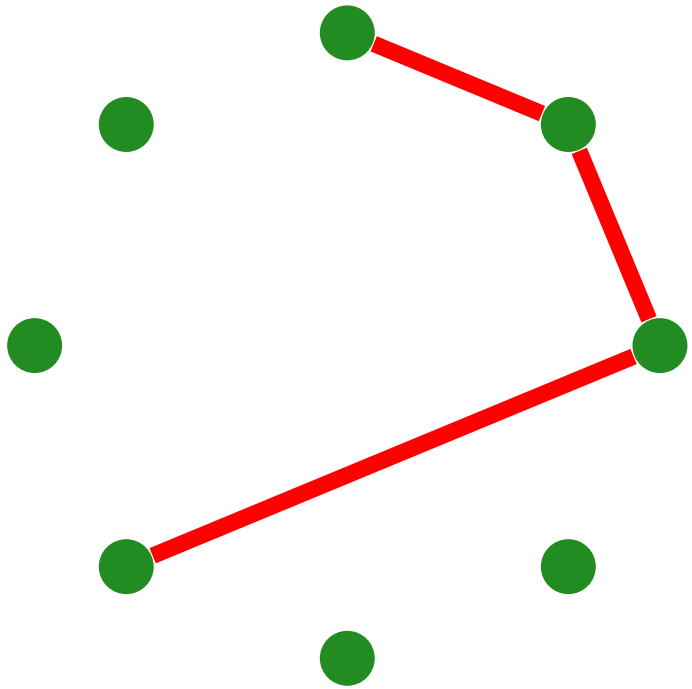
Spanning trees in insertion-only streams



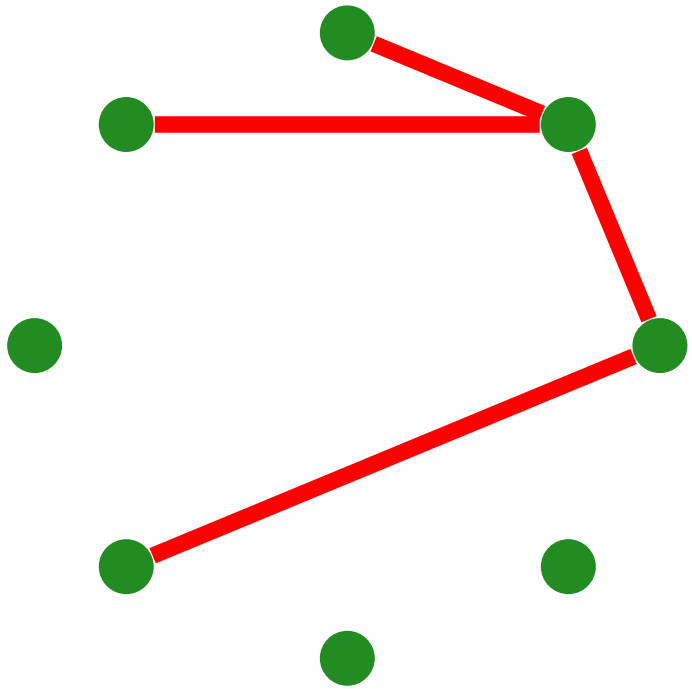
Spanning trees in insertion-only streams



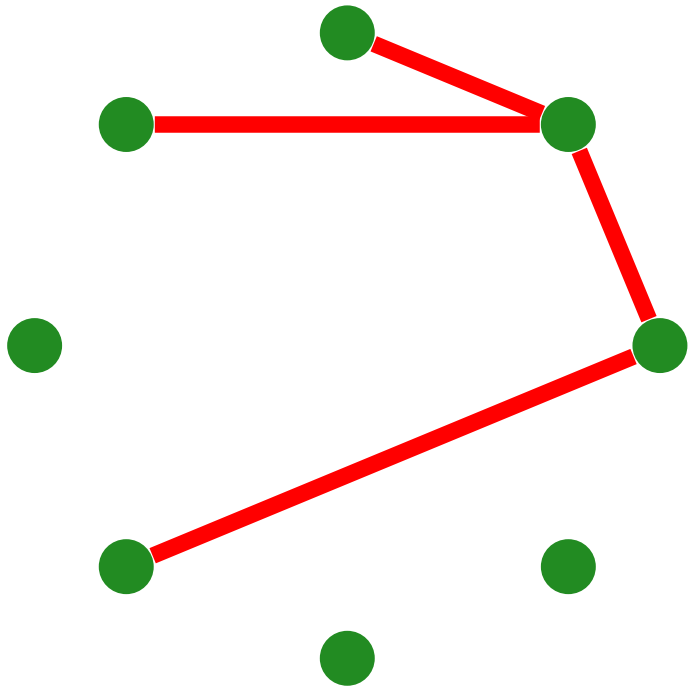
Spanning trees in insertion-only streams



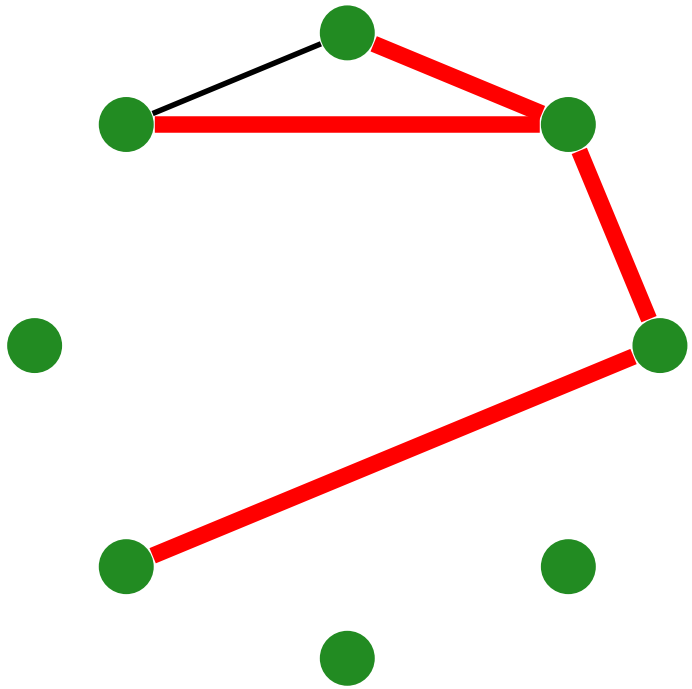
Spanning trees in insertion-only streams



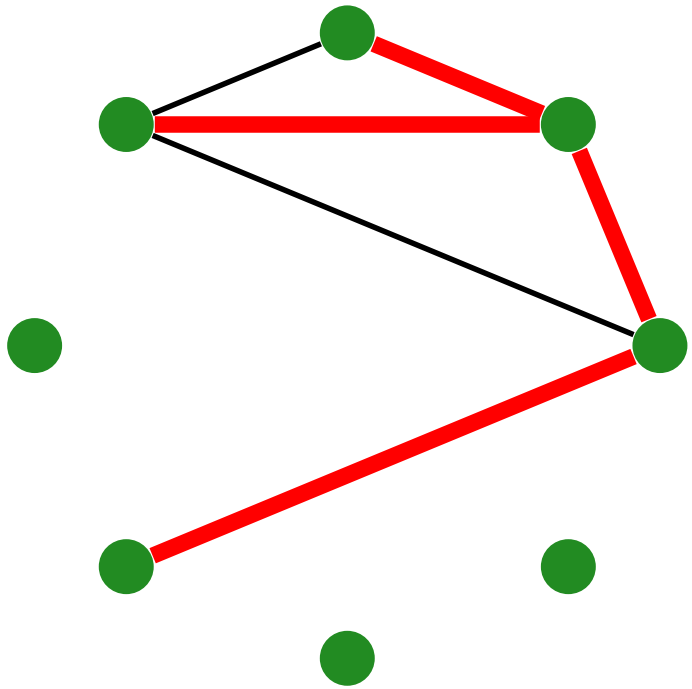
Spanning trees in insertion-only streams



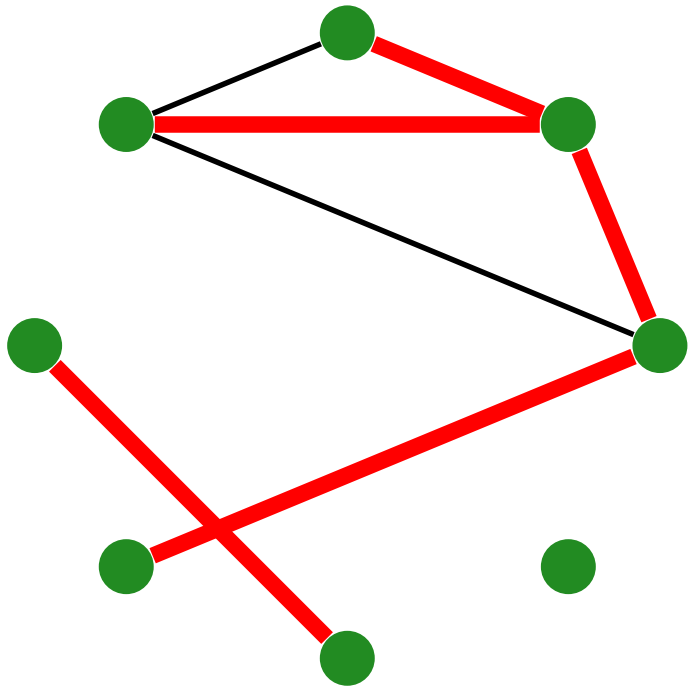
Spanning trees in insertion-only streams



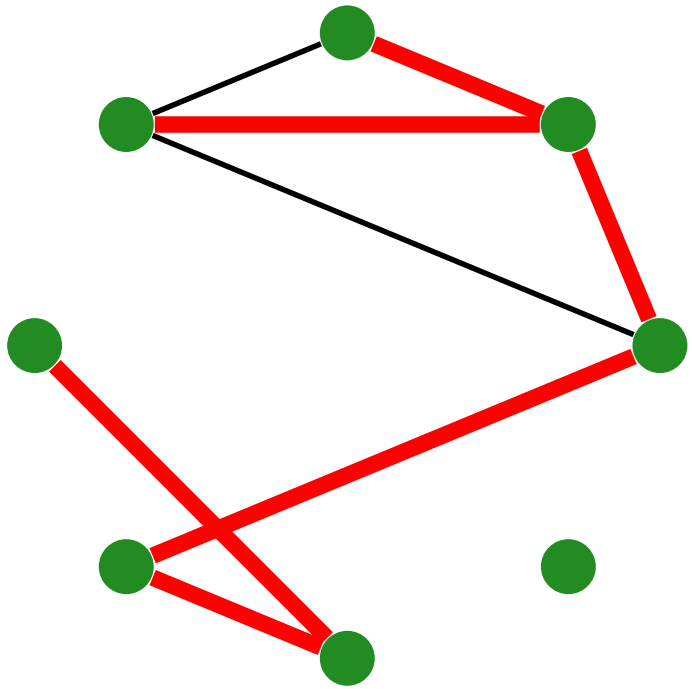
Spanning trees in insertion-only streams



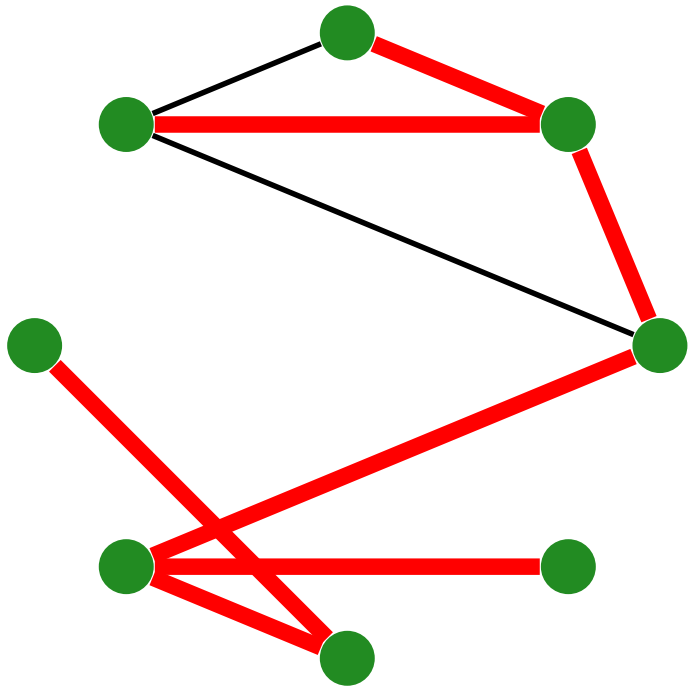
Spanning trees in insertion-only streams



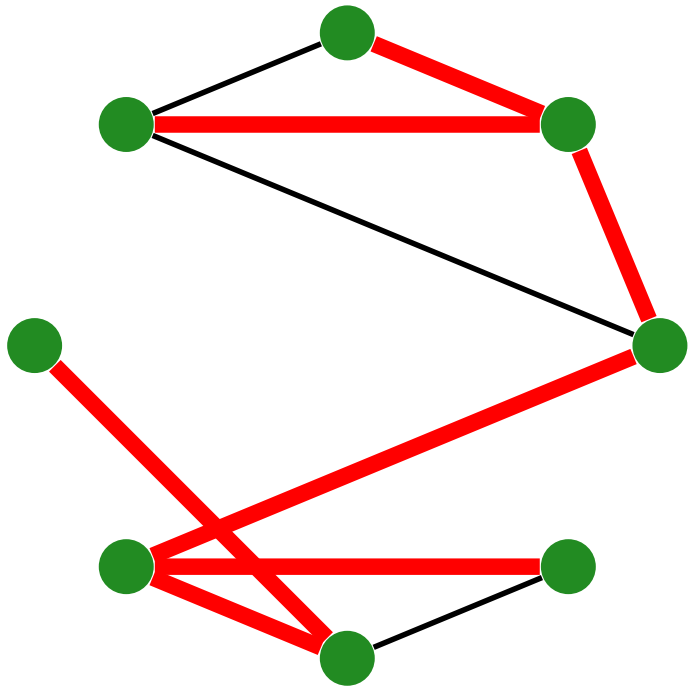
Spanning trees in insertion-only streams



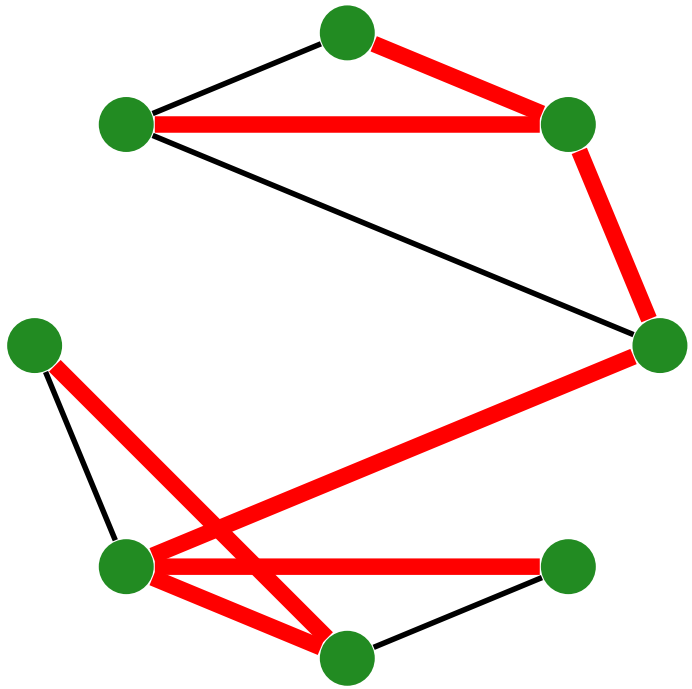
Spanning trees in insertion-only streams



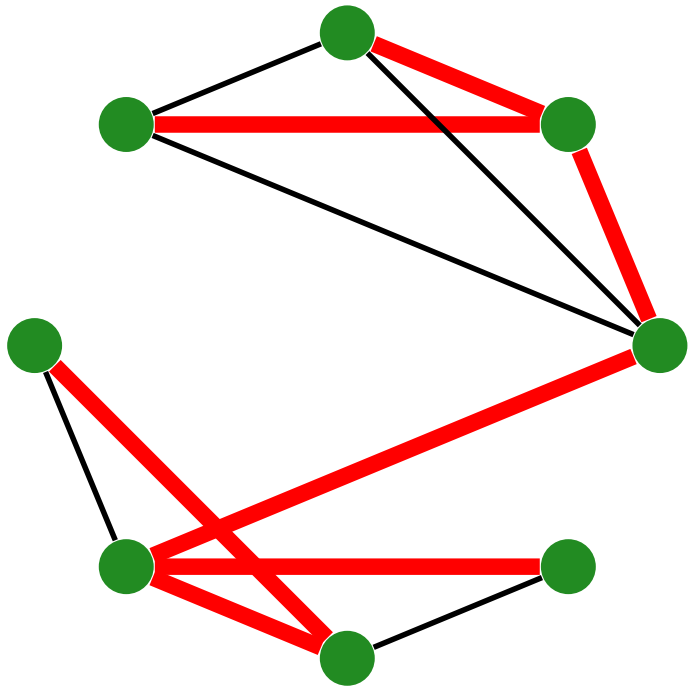
Spanning trees in insertion-only streams



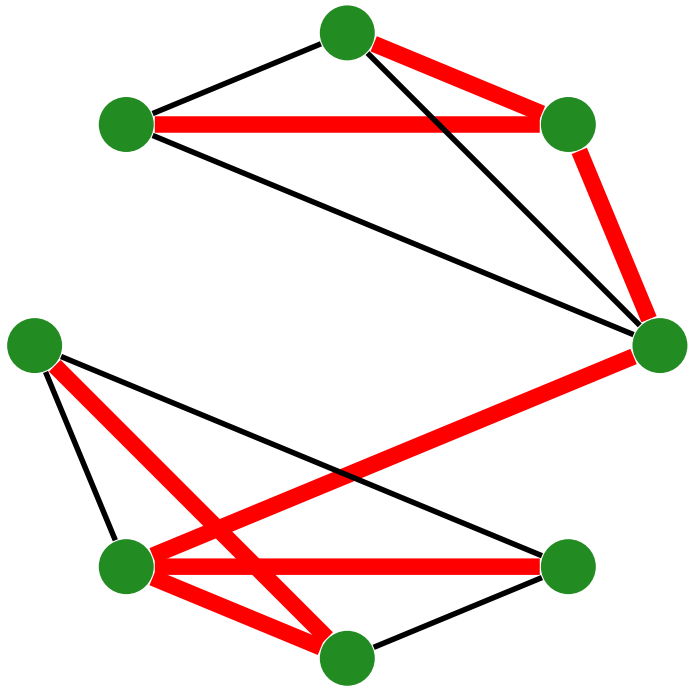
Spanning trees in insertion-only streams



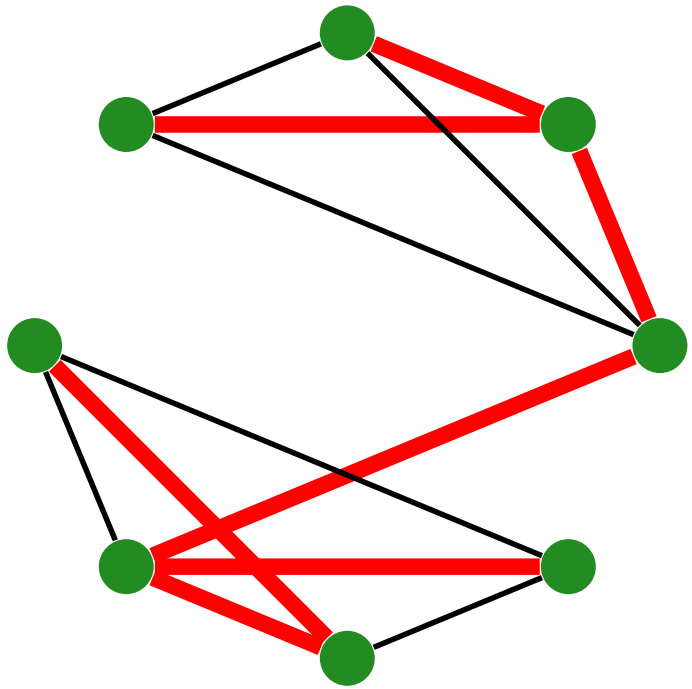
Spanning trees in insertion-only streams



Spanning trees in insertion-only streams



Spanning trees in insertion-only streams



Construct a spanning tree of the graph G in a single pass?

Very easy in insertion only streams:

- maintain a spanning forest
- add incoming edge if it connects two components, discard otherwise

Construct a spanning tree of the graph G in a single pass?

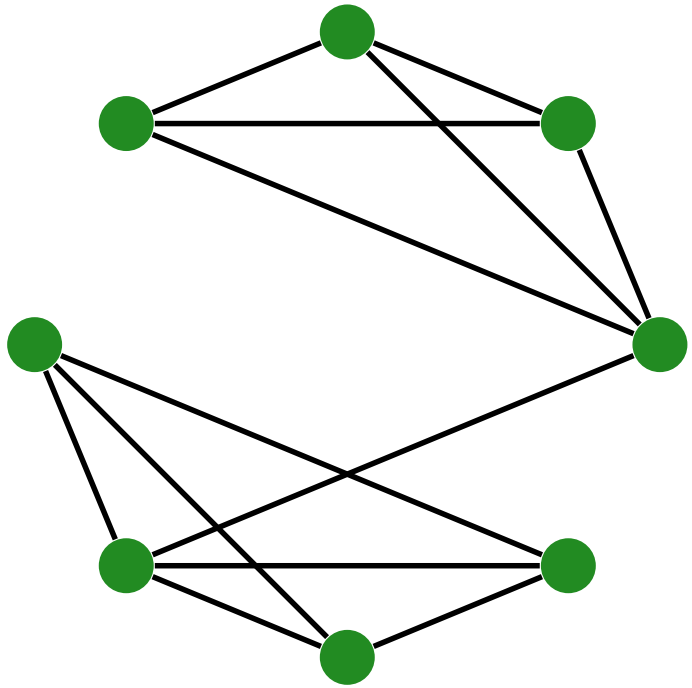
Very easy in insertion only streams:

- maintain a spanning forest
- add incoming edge if it connects two components, discard otherwise

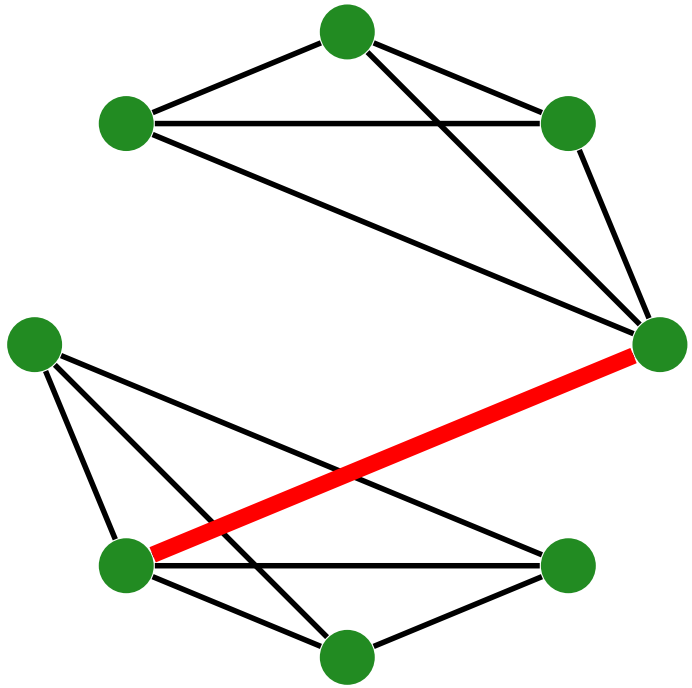
Many modern networks evolve over time, edges both inserted and deleted

Construct spanning trees in dynamic streams in small space?

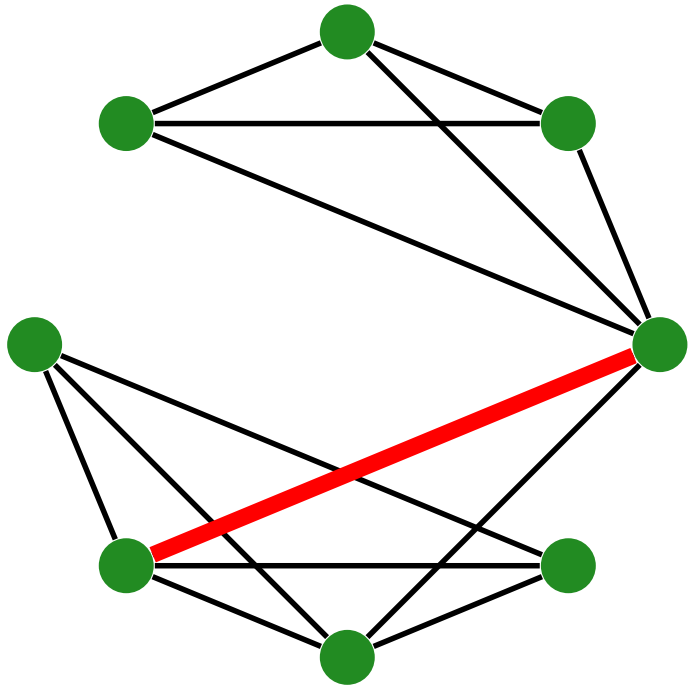
What if we have deletions?



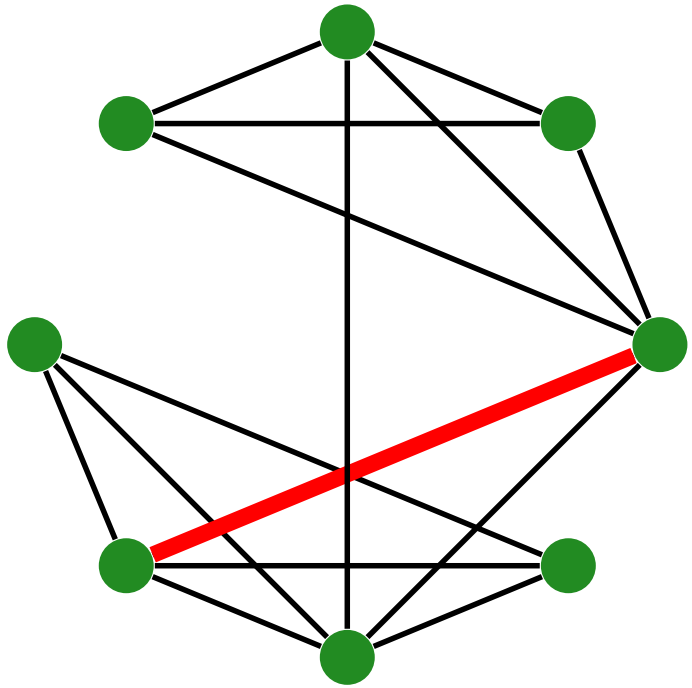
What if we have deletions?



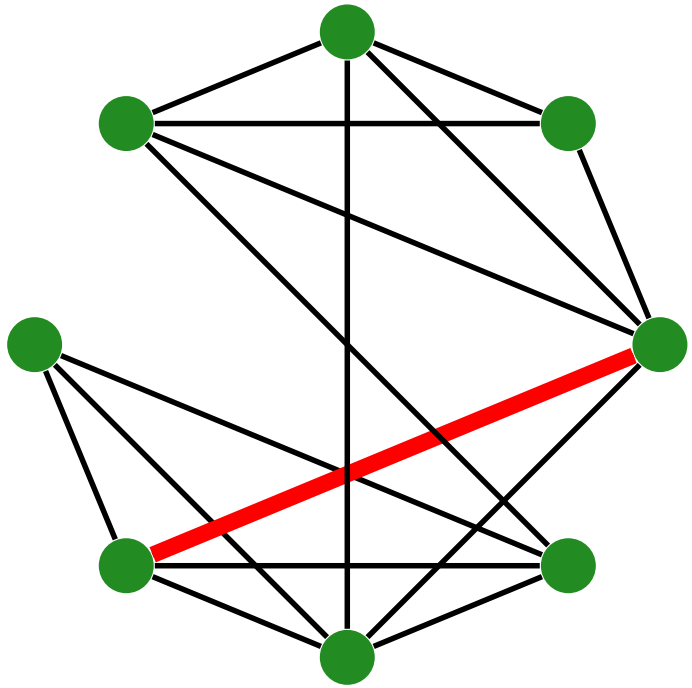
What if we have deletions?



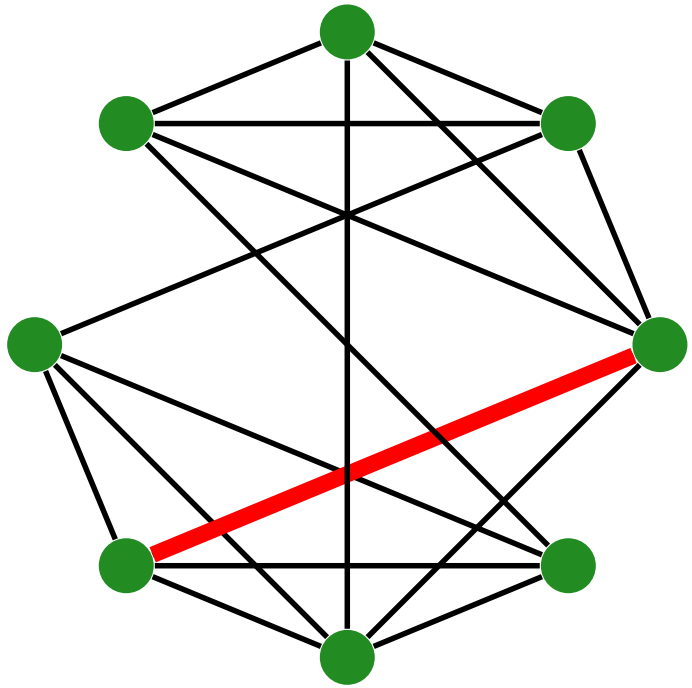
What if we have deletions?



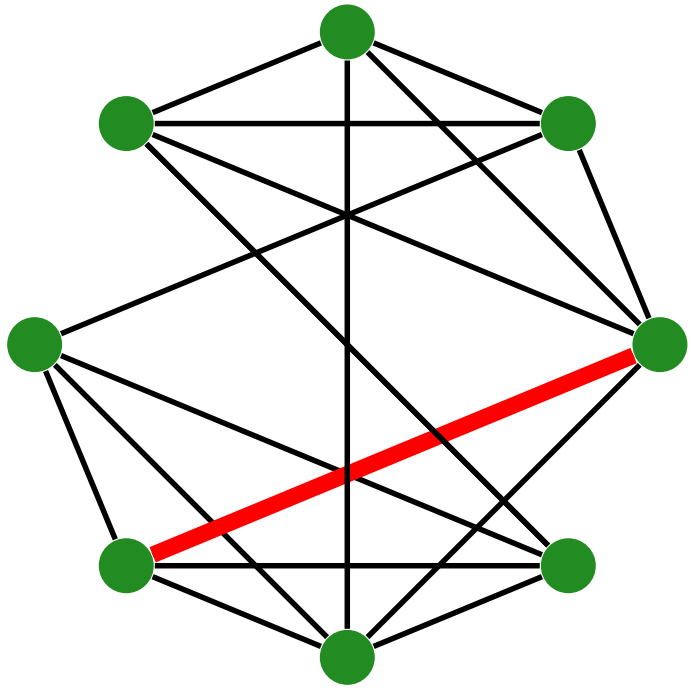
What if we have deletions?



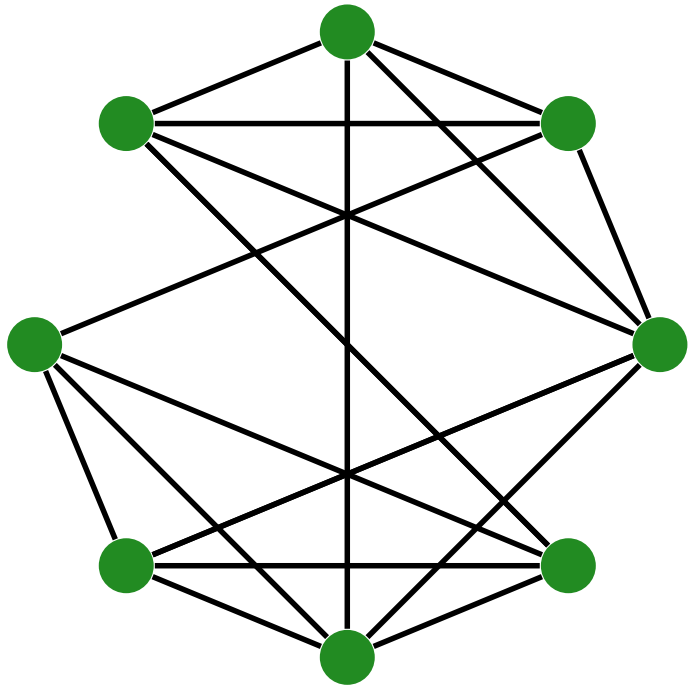
What if we have deletions?



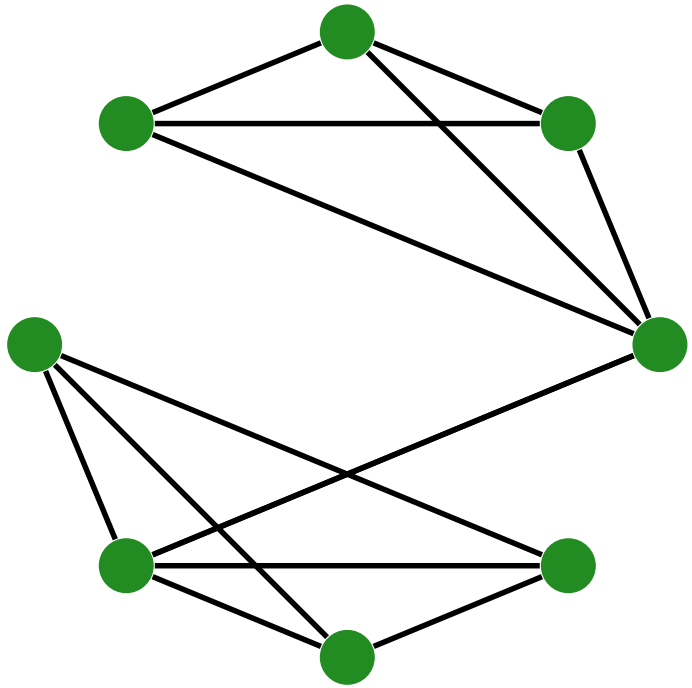
What if we have deletions?



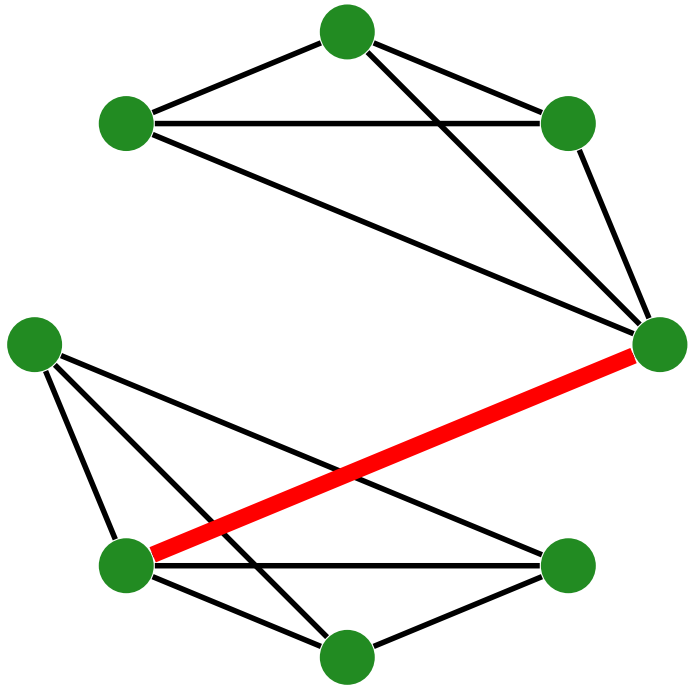
What if we have deletions?



What if we have deletions?

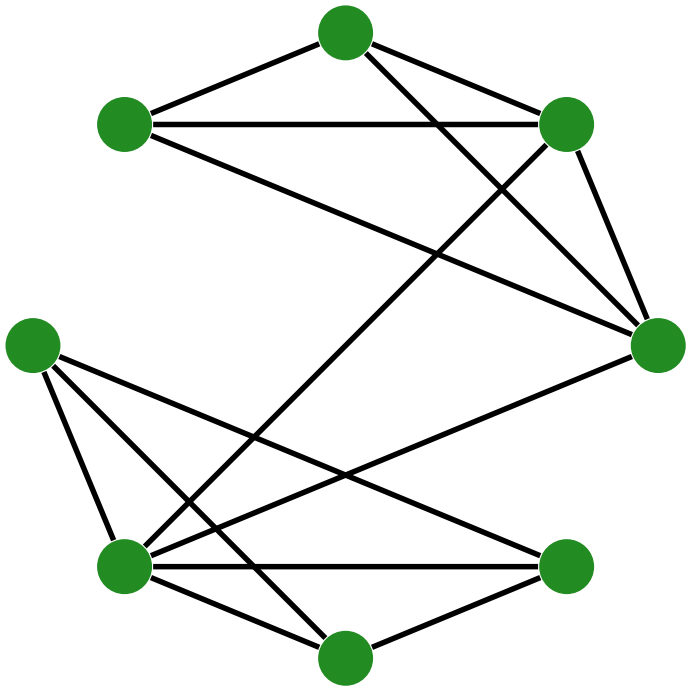


What if we have deletions?



Very different algorithms are needed...

Main idea: sketch the edge-vertex incidence matrix of G!



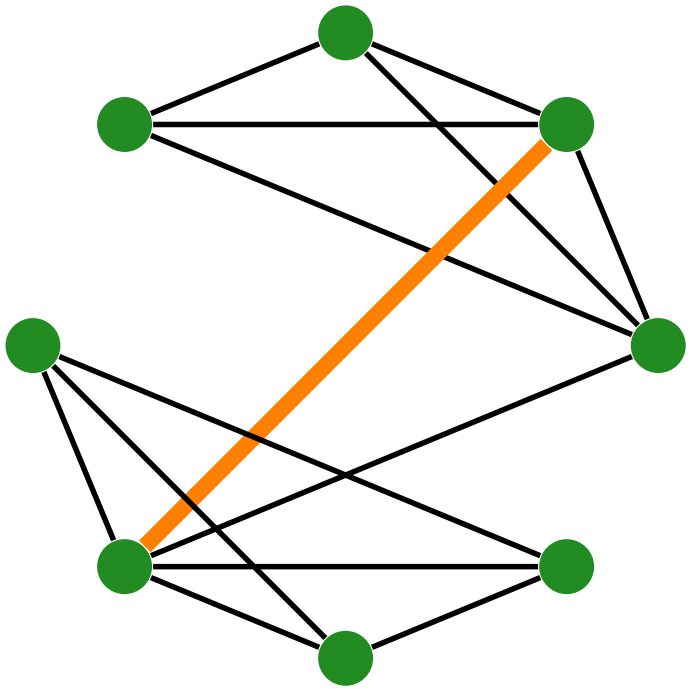
$$B = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & & & & & \vdots \end{bmatrix}$$

$\binom{n}{2} \times n$

Each row of B=potential edge in G.

If $e = (u, v) \in E$, then $b_e = \chi_u - \chi_v$, otherwise $b_e = 0$

Main idea: sketch the edge-vertex incidence matrix of G!



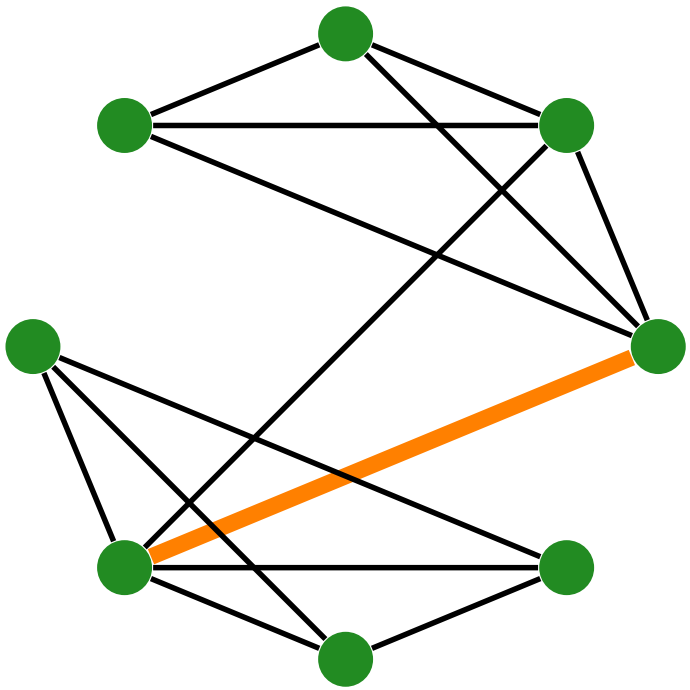
$$B = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \mathbf{-1} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & & & & & \vdots \end{bmatrix}$$

$\binom{n}{2} \times n$

Each row of B=potential edge in G.

If $e = (u, v) \in E$, then $b_e = \chi_u - \chi_v$, otherwise $b_e = 0$

Main idea: sketch the edge-vertex incidence matrix of G!



$$B = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 \\ \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{-1} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & & & & & \vdots \end{bmatrix}$$

$\binom{n}{2} \times n$

Each row of B=potential edge in G.

If $e = (u, v) \in E$, then $b_e = \chi_u - \chi_v$, otherwise $b_e = 0$

For every $S \subseteq V$ let

$$\delta(C) := E \cap (C \times (V \setminus C))$$

denote the set of edges crossing the cut and let

$$x = \mathbf{1}_C \qquad \text{(indicator of } C\text{)}$$

For every $S \subseteq V$ let

$$\delta(C) := E \cap (C \times (V \setminus C))$$

denote the set of edges crossing the cut and let

$$x = \mathbf{1}_C \qquad \text{(indicator of } C\text{)}$$

Key claim: Bx is the (signed) indicator of $\delta(C)$

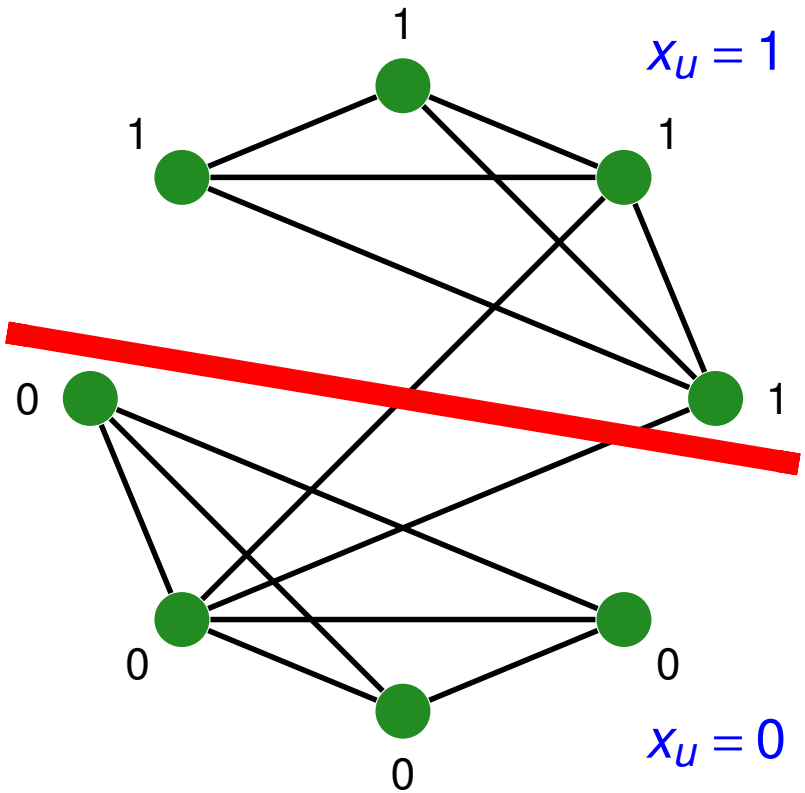
For every $S \subseteq V$ let

$$\delta(C) := E \cap (C \times (V \setminus C))$$

denote the set of edges crossing the cut and let

$$x = \mathbf{1}_C \qquad \text{(indicator of } C\text{)}$$

Key claim: Bx is the (signed) indicator of $\delta(C)$



Key claim: Bx is the (signed) indicator of $\delta(S)$

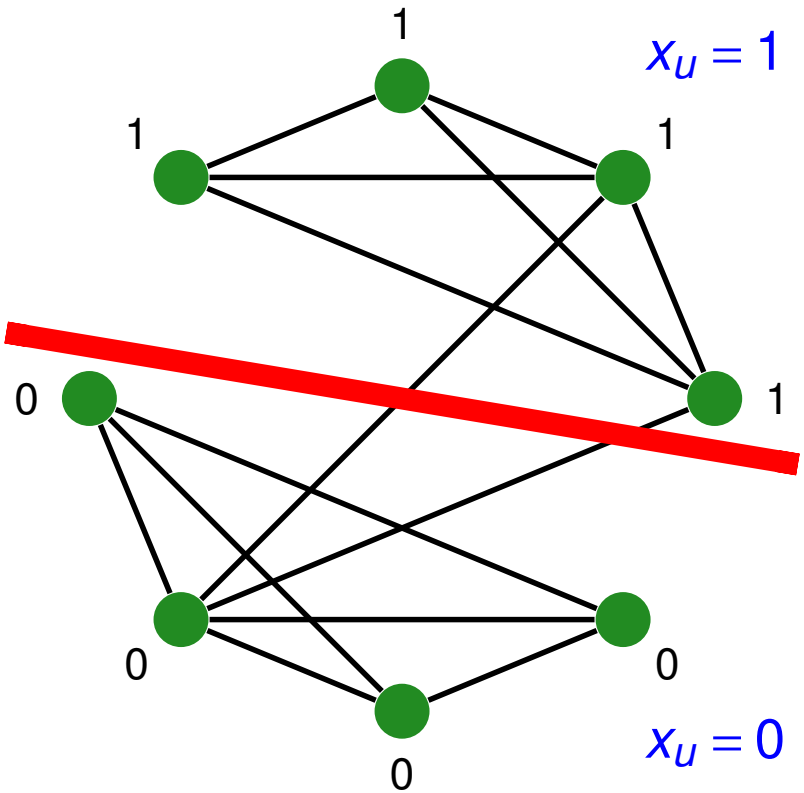
Proof

Key claim: Bx is the (signed) indicator of $\delta(S)$

Proof by picture.

Key claim: Bx is the (signed) indicator of $\delta(S)$

Proof by picture.



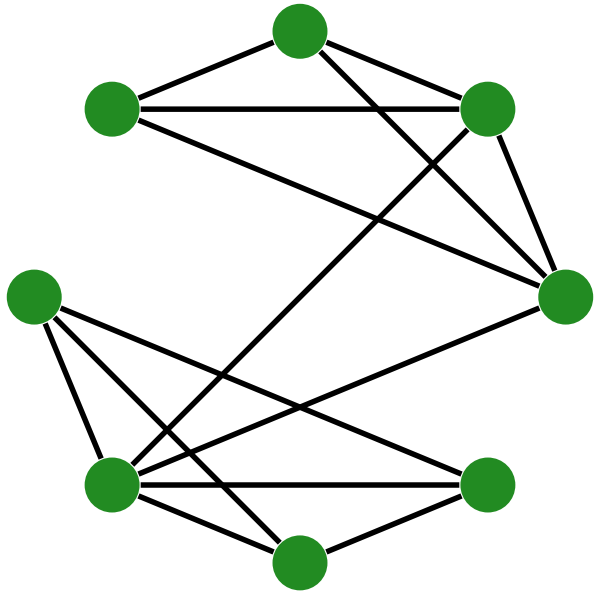
$$B = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & & & & & \vdots \end{bmatrix}$$

$\binom{n}{2} \times n$

- Graph streaming
- Connectivity via sketching

- Graph streaming
- **Connectivity via sketching**

A simple algorithm for connectivity



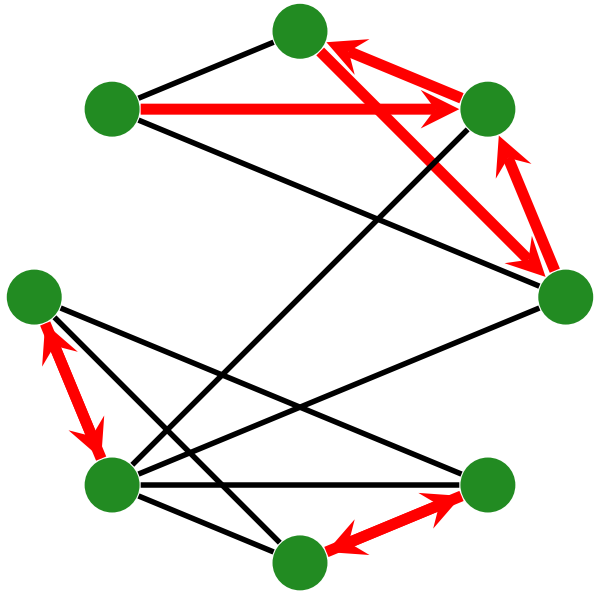
$F_0 \leftarrow \emptyset$
$\mathscr{C}_0 \leftarrow \bigcup_{u \in V} \{u\}$ (current components)
for $t = 0$ **to** T

  for each $C \in \mathscr{C}_t$
    choose an edge in $\delta(C)$ (nonzero of $B \cdot \mathbf{1}_C$)

  $F_{t+1} \leftarrow F_t \cup \{\text{spanning forest on selected edges}\}$
  $\mathscr{C}_{t+1} \leftarrow \{\text{new connected components}\}$

return F_{T+1}

A simple algorithm for connectivity



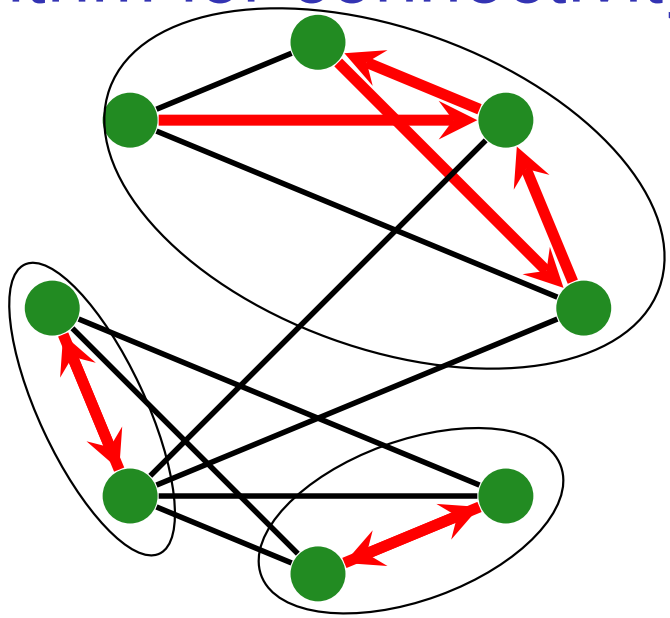
$F_0 \leftarrow \emptyset$
$\mathscr{C}_0 \leftarrow \bigcup_{u \in V} \{u\}$ (current components)
for $t = 0$ **to** T

  for each $C \in \mathscr{C}_t$
    choose an edge in $\delta(C)$ (nonzero of $B \cdot \mathbf{1}_C$)

  $F_{t+1} \leftarrow F_t \cup \{\text{spanning forest on selected edges}\}$
  $\mathscr{C}_{t+1} \leftarrow \{\text{new connected components}\}$

return F_{T+1}

A simple algorithm for connectivity



$F_0 \leftarrow \emptyset$

$\mathscr{C}_0 \leftarrow \bigcup_{u \in V} \{u\}$ (current components)

for $t = 0$ **to** T

  for each $C \in \mathscr{C}_t$

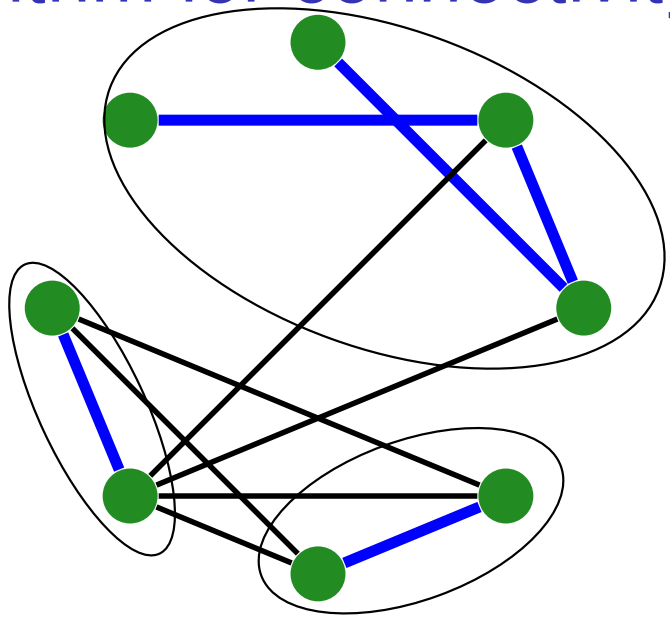
    choose an edge in $\delta(C)$ (nonzero of $B \cdot \mathbf{1}_C$)

  $F_{t+1} \leftarrow F_t \cup \{\text{spanning forest on selected edges}\}$

  $\mathscr{C}_{t+1} \leftarrow \{\text{new connected components}\}$

return F_{T+1}

A simple algorithm for connectivity



$F_0 \leftarrow \emptyset$

$\mathscr{C}_0 \leftarrow \bigcup_{u \in V} \{u\}$ (current components)

for $t = 0$ **to** T

  for each $C \in \mathscr{C}_t$

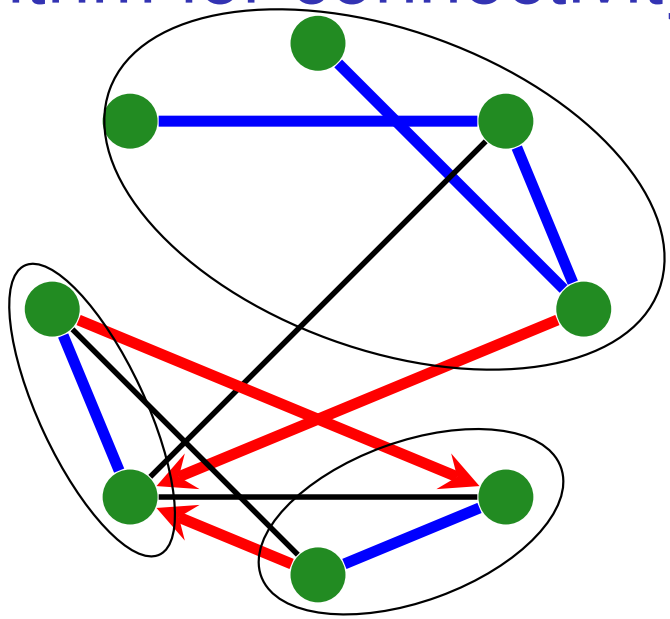
    choose an edge in $\delta(C)$ (nonzero of $B \cdot \mathbf{1}_C$)

  $F_{t+1} \leftarrow F_t \cup \{\text{spanning forest on selected edges}\}$

  $\mathscr{C}_{t+1} \leftarrow \{\text{new connected components}\}$

return F_{T+1}

A simple algorithm for connectivity



$F_0 \leftarrow \emptyset$

$\mathscr{C}_0 \leftarrow \bigcup_{u \in V} \{u\}$ (current components)

for $t = 0$ **to** T

  for each $C \in \mathscr{C}_t$

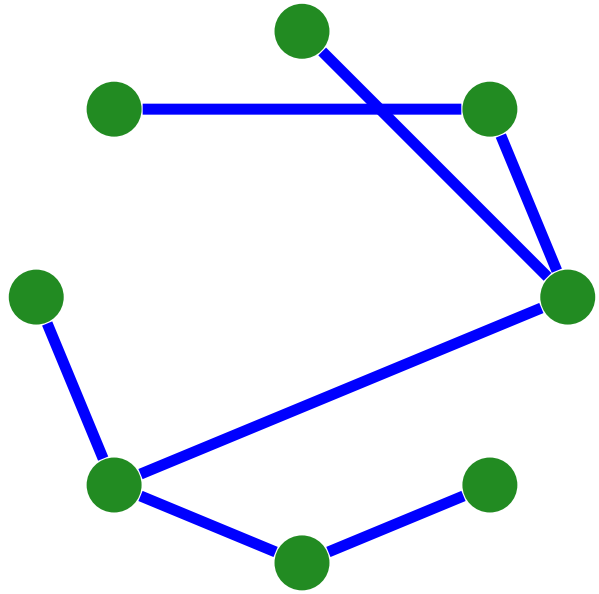
    choose an edge in $\delta(C)$ (nonzero of $B \cdot \mathbf{1}_C$)

  $F_{t+1} \leftarrow F_t \cup \{\text{spanning forest on selected edges}\}$

  $\mathscr{C}_{t+1} \leftarrow \{\text{new connected components}\}$

return F_{T+1}

A simple algorithm for connectivity



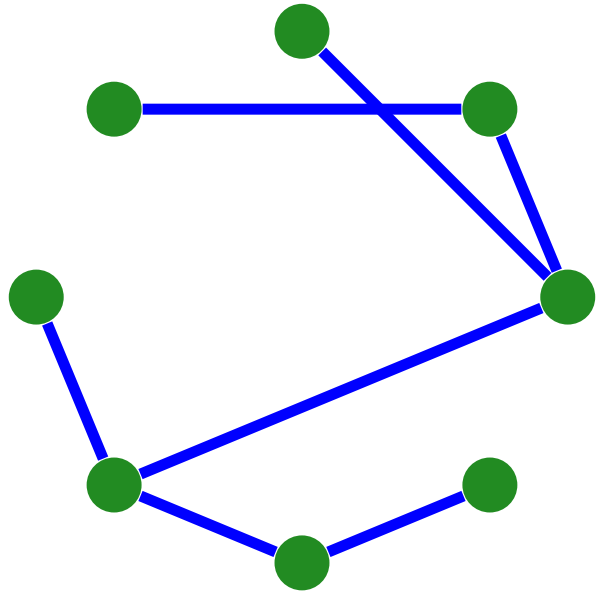
$F_0 \leftarrow \emptyset$
$\mathscr{C}_0 \leftarrow \bigcup_{u \in V} \{u\}$ (current components)
for $t = 0$ **to** T

  for each $C \in \mathscr{C}_t$
    choose an edge in $\delta(C)$ (nonzero of $B \cdot \mathbf{1}_C$)

  $F_{t+1} \leftarrow F_t \cup \{\text{spanning forest on selected edges}\}$
  $\mathscr{C}_{t+1} \leftarrow \{\text{new connected components}\}$

return F_{T+1}

A simple algorithm for connectivity



$F_0 \leftarrow \emptyset$
$\mathscr{C}_0 \leftarrow \bigcup_{u \in V} \{u\}$ (current components)
for $t = 0$ **to** T

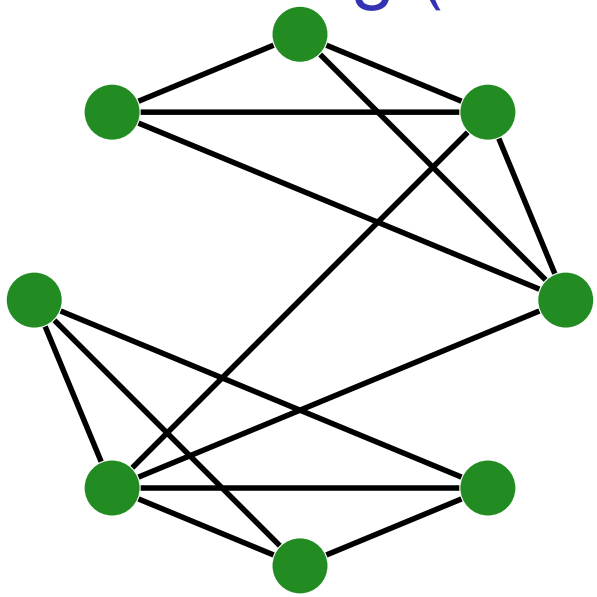
  for each $C \in \mathscr{C}_t$
    choose an edge in $\delta(C)$

(?? nonzero of $B \cdot \mathbf{1}_C$??)

  $F_{t+1} \leftarrow F_t \cup \{\text{spanning forest on selected edges}\}$
  $\mathscr{C}_{t+1} \leftarrow \{\text{new connected components}\}$

return F_{T+1}

Connectivity via sketching (Ahn-Guha-McGregor'12)



$F_0 \leftarrow \emptyset$
$\mathscr{C}_0 \leftarrow \bigcup_{u \in V} \{u\}$
for $t = 0$ **to** T

  for each $C \in \mathscr{C}_t$
    choose an edge in $\delta(C)$

  $F_{t+1} \leftarrow F_t \cup \{\text{spanning forest on selected edges}\}$
  $\mathscr{C}_{t+1} \leftarrow \{\text{new connected components}\}$

return F_{T+1}

$S^0, \ldots, S^T \leftarrow \ell_0$-samplers
Maintain $S^0 B, \ldots, S^T B$

Run $Dec(S^t B \cdot \mathbf{1}_C)$

Some remarks

The result of Ahn, Guha, McGregor'12 independently and simultaneously obtained by Kapron, King'12

Some remarks

The result of Ahn, Guha, McGregor'12 independently and simultaneously obtained by Kapron, King'12

Tightening parameters, get a sketch with bit complexity $O(n \log^3 n)$

Some remarks

The result of Ahn, Guha, McGregor'12 independently and simultaneously obtained by Kapron, King'12

Tightening parameters, get a sketch with bit complexity $O(n \log^3 n)$

Q: why did we need several sketches $S^0, \ldots, S^T$?

Some remarks

The result of Ahn, Guha, McGregor'12 independently and simultaneously obtained by Kapron, King'12

Tightening parameters, get a sketch with bit complexity $O(n \log^3 n)$

Q: why did we need several sketches $S^0, \ldots, S^T$?

A: adaptivity

Some remarks

The result of Ahn, Guha, McGregor'12 independently and simultaneously obtained by Kapron, King'12

Tightening parameters, get a sketch with bit complexity $O(n \log^3 n)$

Q: why did we need several sketches $S^0, \ldots, S^T$?

A: adaptivity

Very surprising: decoding is adaptive ($T = O(\log n)$ rounds), but sketch is not

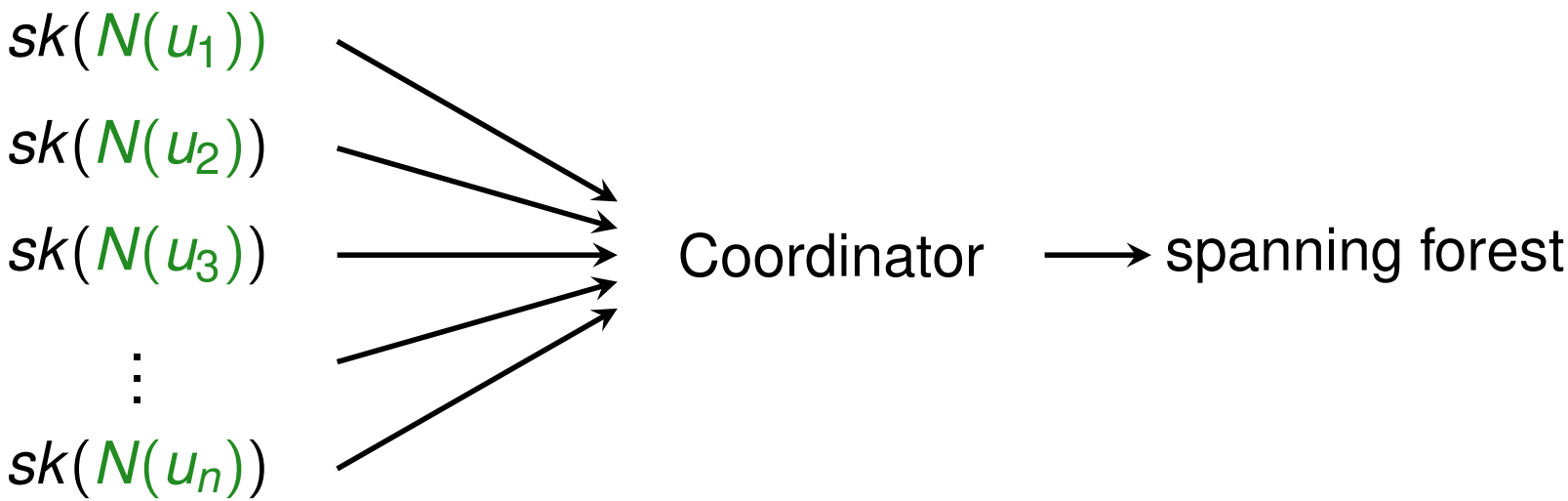
Nelson, Yu'19, Yu'21 – $\Omega(n \log^3 n)$ bits of space are needed

Input: Each $u \in V$ holds list of its neighbors $N(u)$

Each $u \in V$ sends a sketch of $N(u)$ to coordinator

(assume shared randomness)

Output: Spanning forest of G



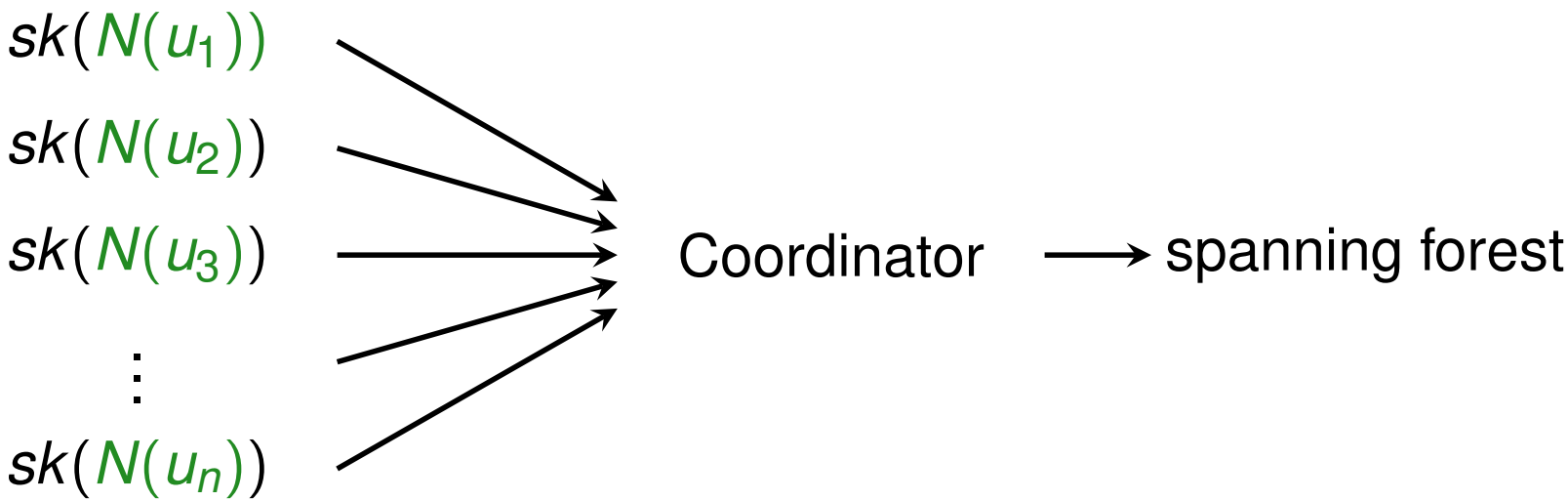
Every party $u \in V$ sends a function of their neighborhood to coordinator

Input: Each $u \in V$ holds list of its neighbors $N(u)$

Each $u \in V$ sends a sketch of $N(u)$ to coordinator

(assume shared randomness)

Output: Spanning forest of G



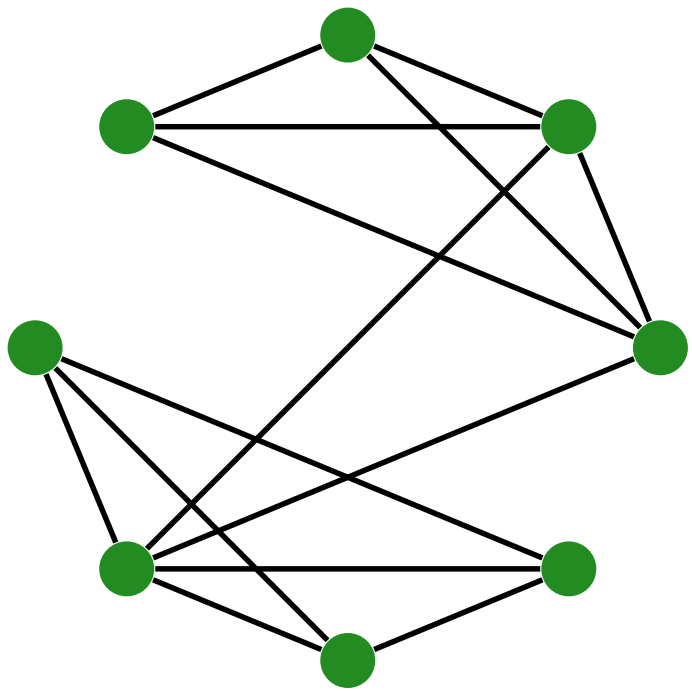
Every party $u \in V$ sends a function of their neighborhood to coordinator

Nelson, Yu'19, Yu'21 – $\Omega(n \log^3 n)$ bits of communication are needed

Spectral sparsification of graphs

Sparsification

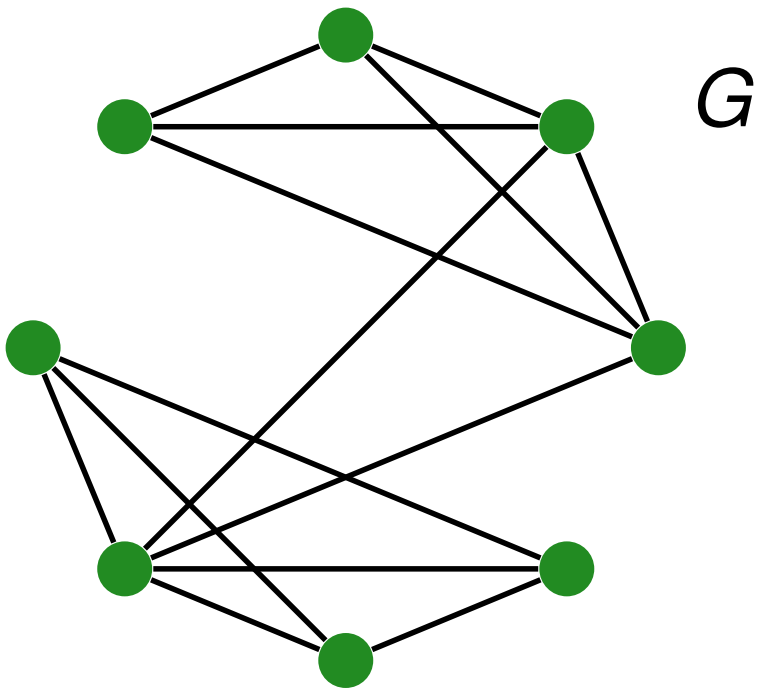
- $G = (V, E)$ an undirected graph, $|V| = n, |E| = m$
- Find sparse subgraph G' of G that approximates G



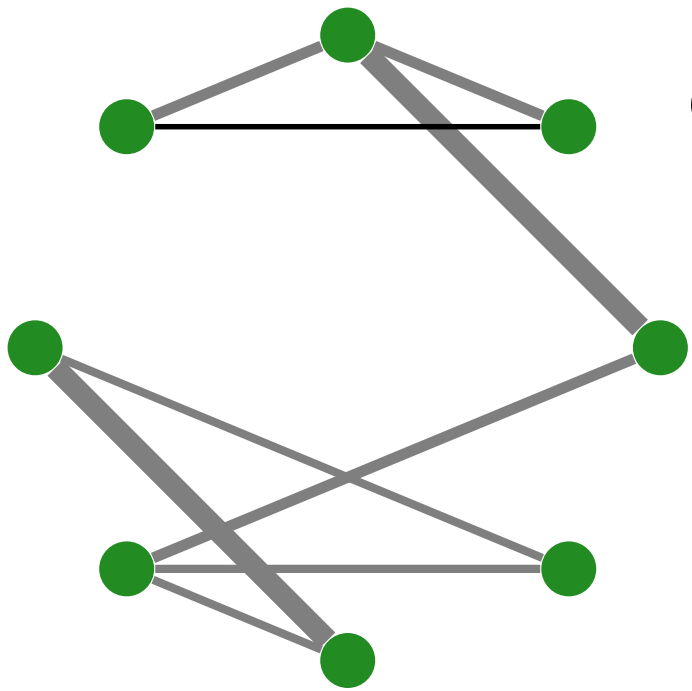
G

Sparsification

- $G = (V, E)$ an undirected graph, $|V| = n, |E| = m$
- Find sparse subgraph G' of G that approximates G



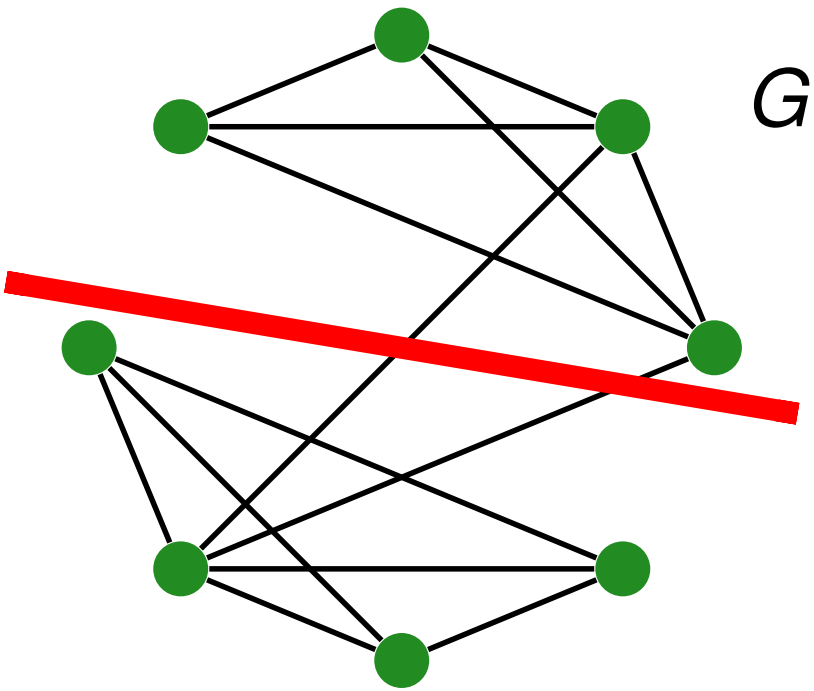
G

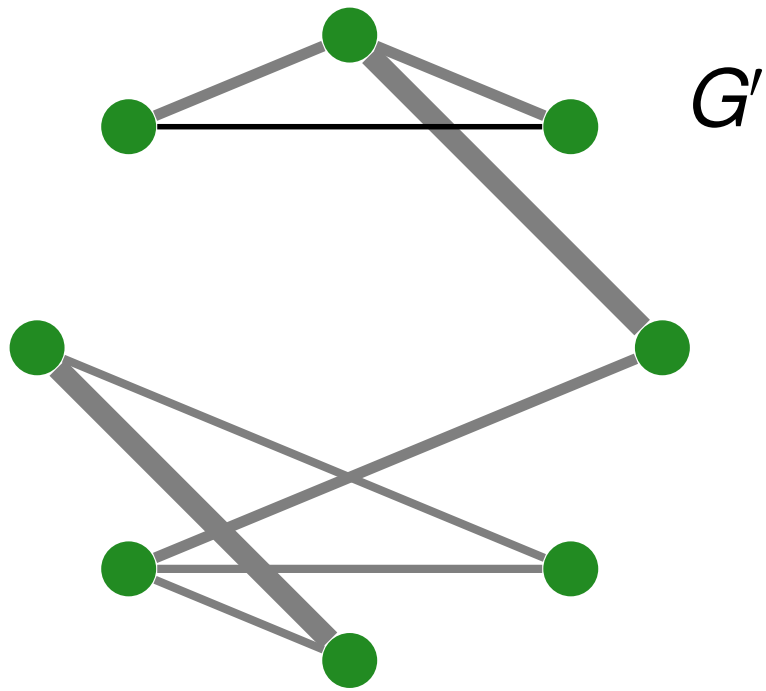


G'

Sparsification

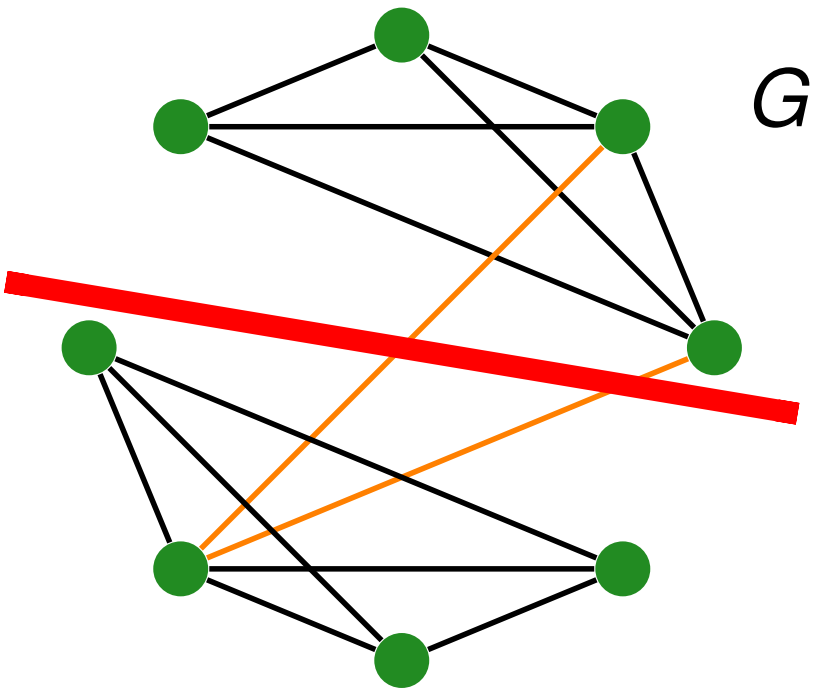
- $G = (V, E)$ an undirected graph, $|V| = n, |E| = m$
- Find sparse subgraph G' of G that approximates G

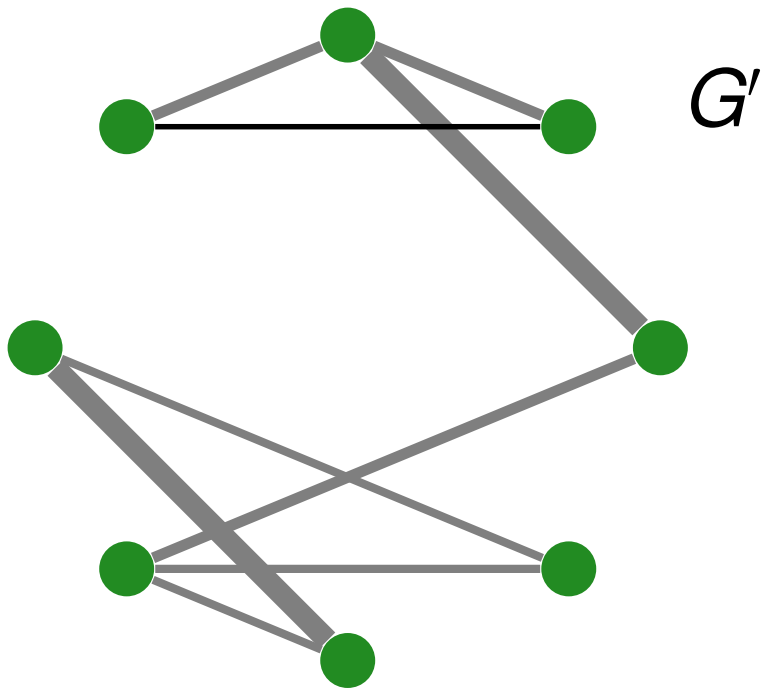




Sparsification

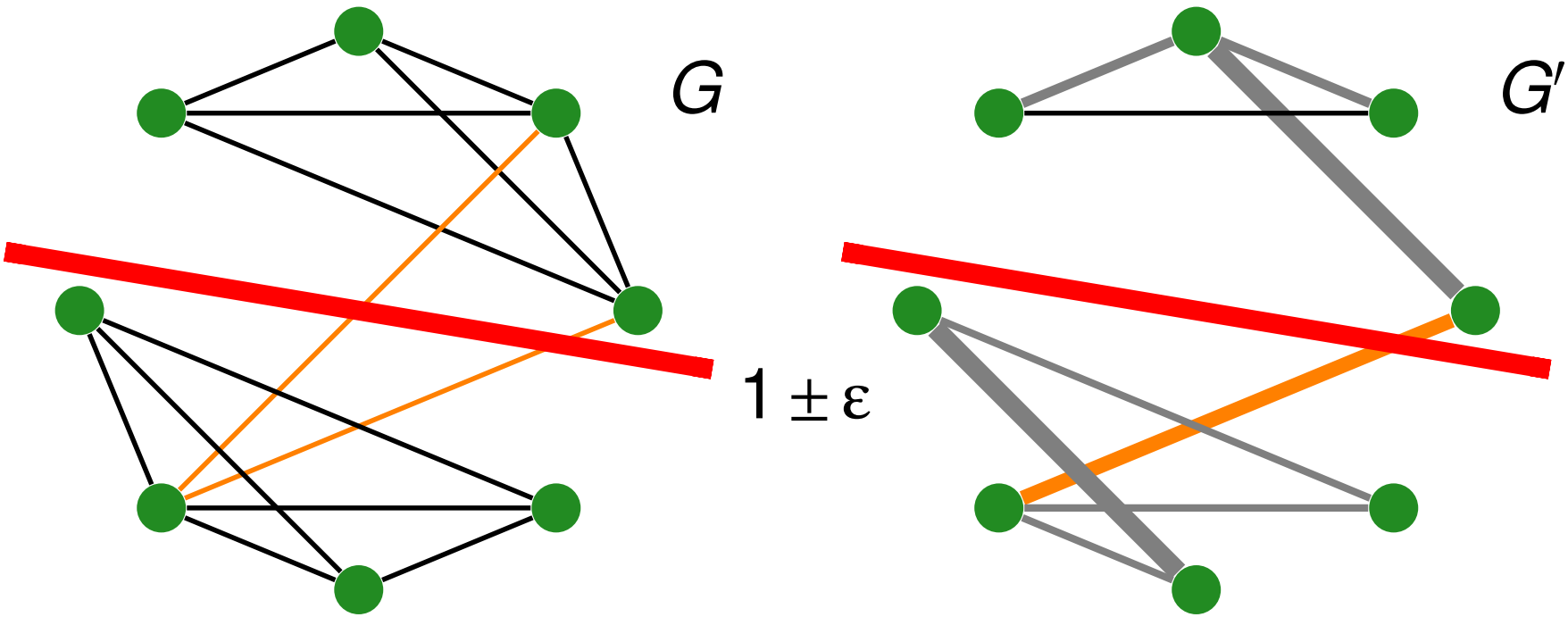
- $G = (V, E)$ an undirected graph, $|V| = n, |E| = m$
- Find sparse subgraph G' of G that approximates G





Sparsification

- $G = (V, E)$ an undirected graph, $|V| = n, |E| = m$
- Find sparse subgraph G' of G that approximates G

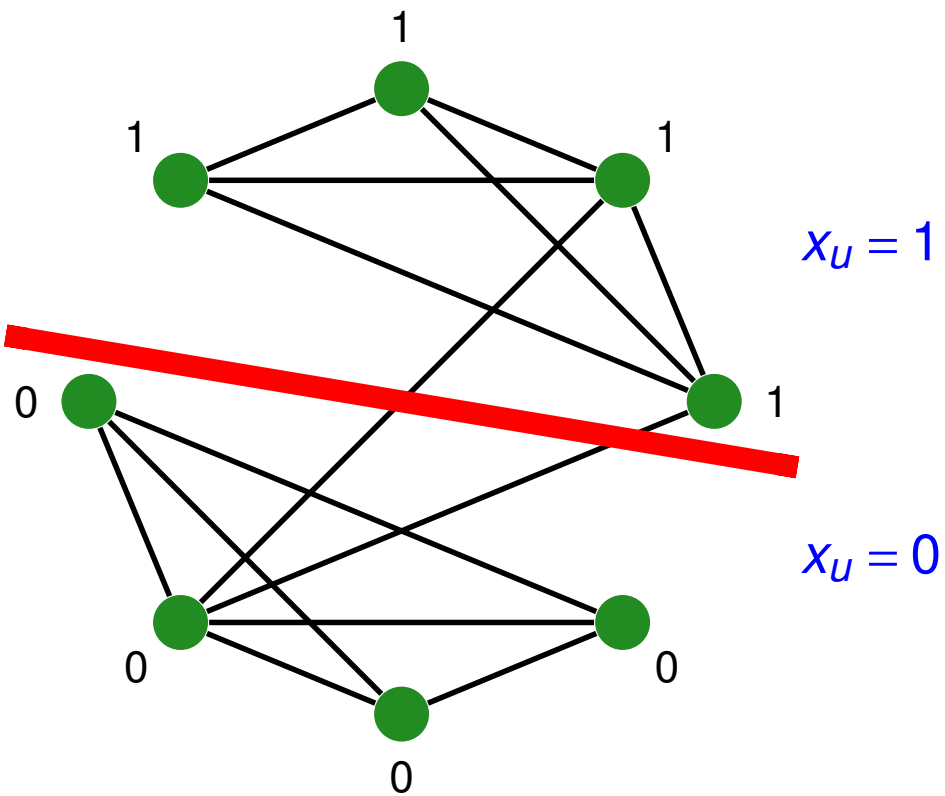


1. Spectral sparsification
2. Dynamic streaming and linear sketching
3. Spectral sparsification using small sketches
4. Fast decoding

1. **Spectral sparsification**
2. Dynamic streaming and linear sketching
3. Spectral sparsification using small sketches
4. Fast decoding

Sparsification

$$\text{value of cut} = \sum_{e=(u,v)\in E} (x_u - x_v)^2$$

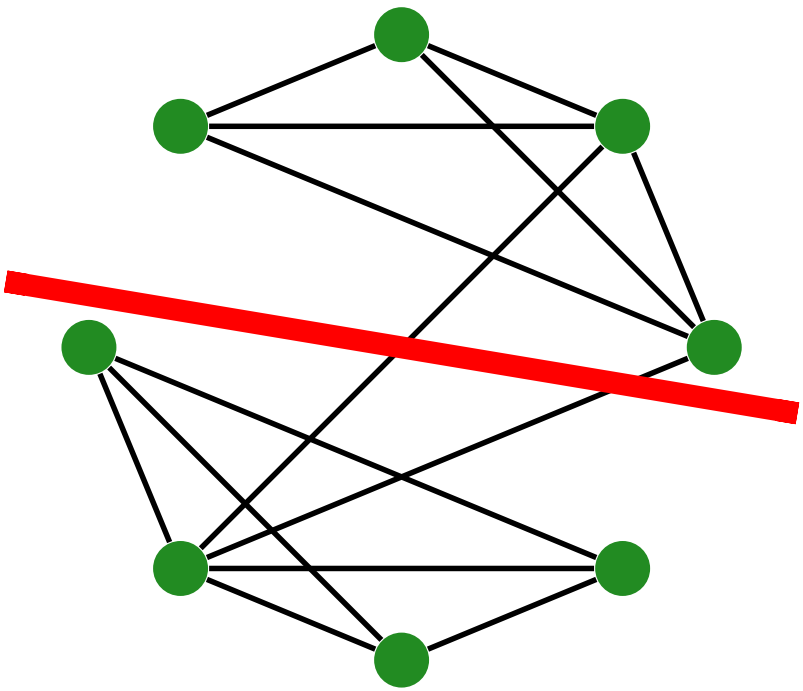


$x_u = 1$

$x_u = 0$

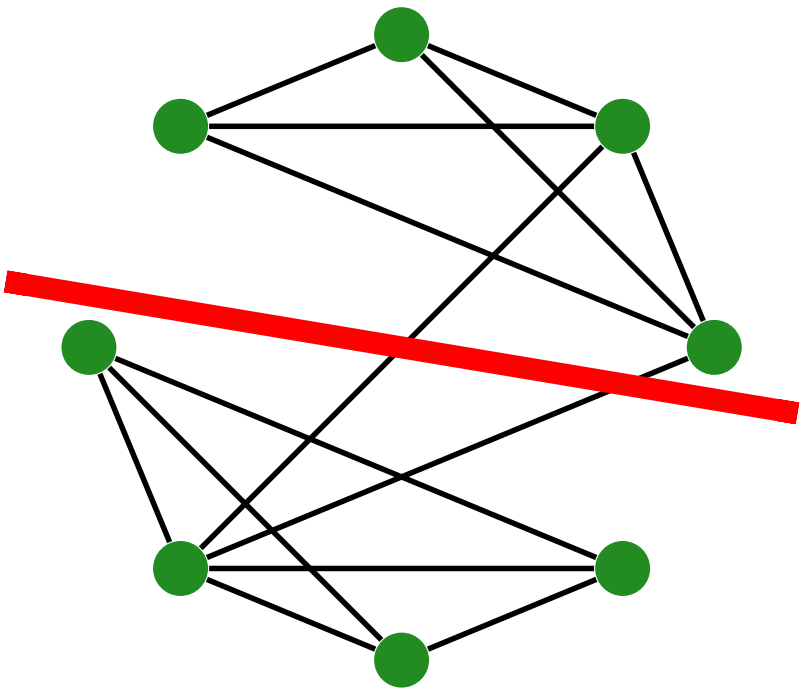
Sparsification

$$\text{value of cut} = \sum_{e=(u,v)\in E} (x_u - x_v)^2$$



Sparsification

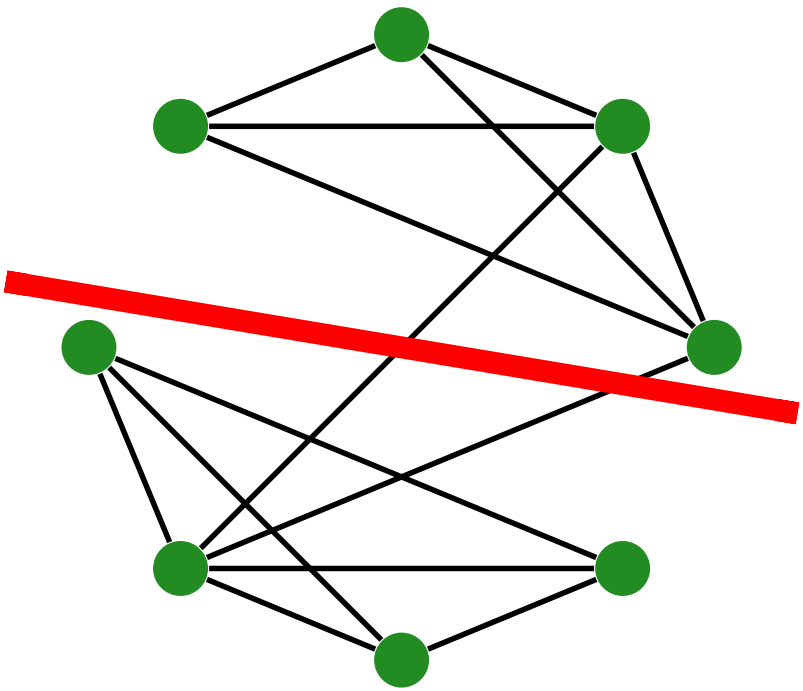
$$\text{value of cut} = \sum_{e=(u,v)\in E} (x_u - x_v)^2 = ||z||^2$$



$$z = \begin{bmatrix} 0 \\ x_1 - x_2 \\ 0 \\ x_4 - x_3 \\ 0 \\ x_4 - x_5 \\ \vdots \end{bmatrix}$$

Sparsification

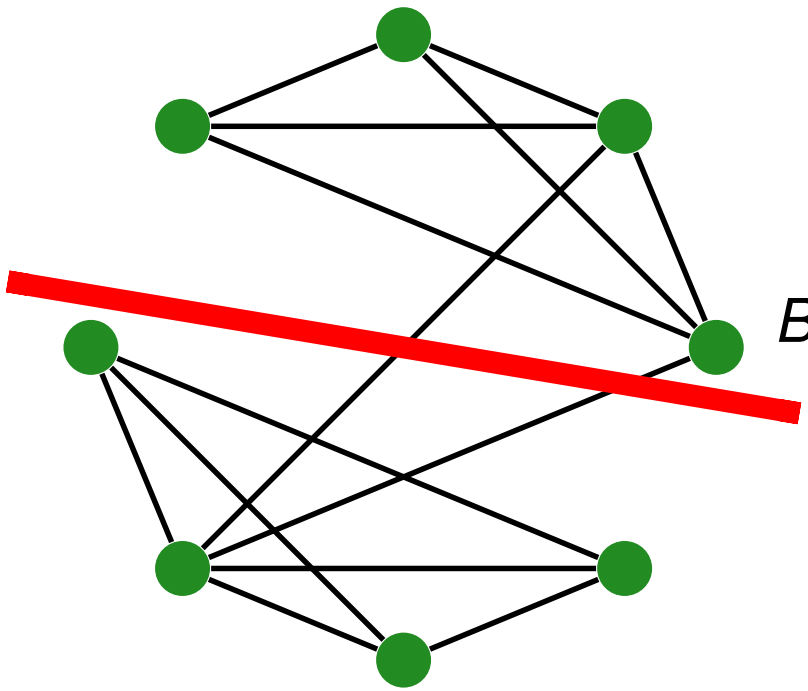
$$\text{value of cut} = \sum_{e=(u,v)\in E} (x_u - x_v)^2 = ||z||^2$$



$$z = \begin{bmatrix} 0 \\ x_1 - x_2 \\ 0 \\ x_4 - x_3 \\ 0 \\ x_4 - x_5 \\ \vdots \end{bmatrix} = Bx$$

Sparsification

$$\text{value of cut} = \sum_{e=(u,v)\in E} (x_u - x_v)^2 = ||Bx||^2$$

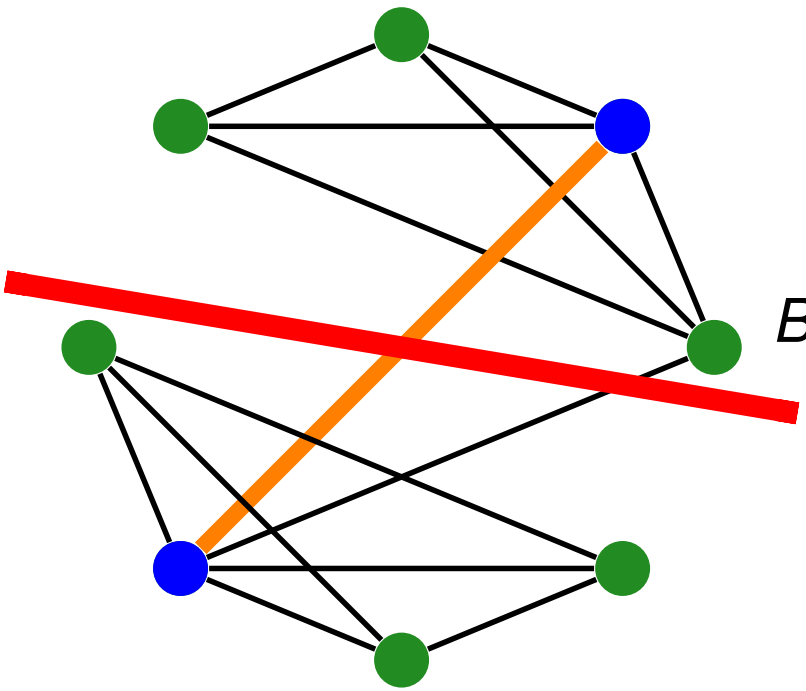


$$Bx = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & & & & & \vdots \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

$\binom{n}{2} \times n$

Sparsification

$$\text{value of cut} = \sum_{e=(u,v)\in E} (x_u - x_v)^2 = ||Bx||^2$$

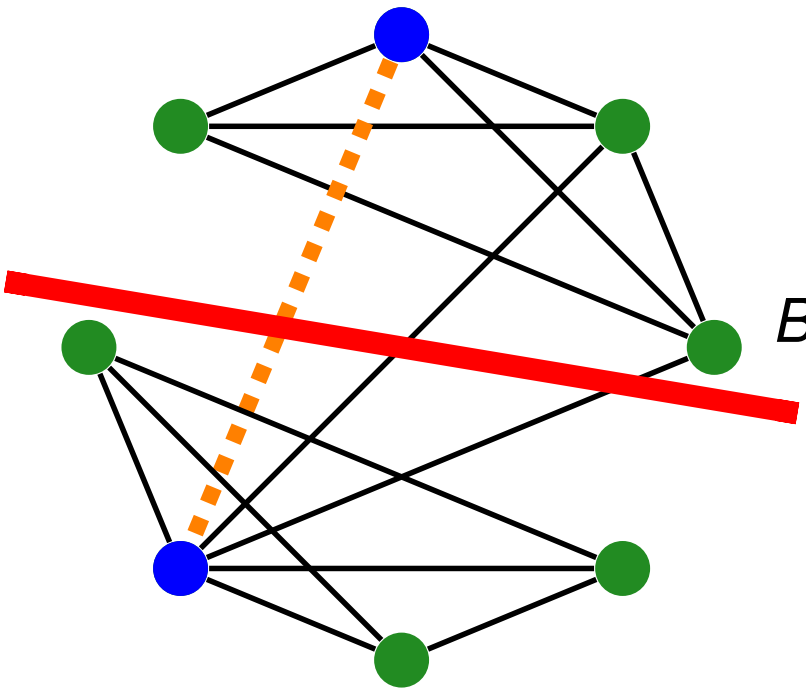


$$Bx = \begin{bmatrix} \mathbf{1} & \mathbf{-1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & & & & & \vdots \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

$\binom{n}{2} \times n$

Sparsification

$$\text{value of cut} = \sum_{e=(u,v)\in E} (x_u - x_v)^2 = ||Bx||^2$$



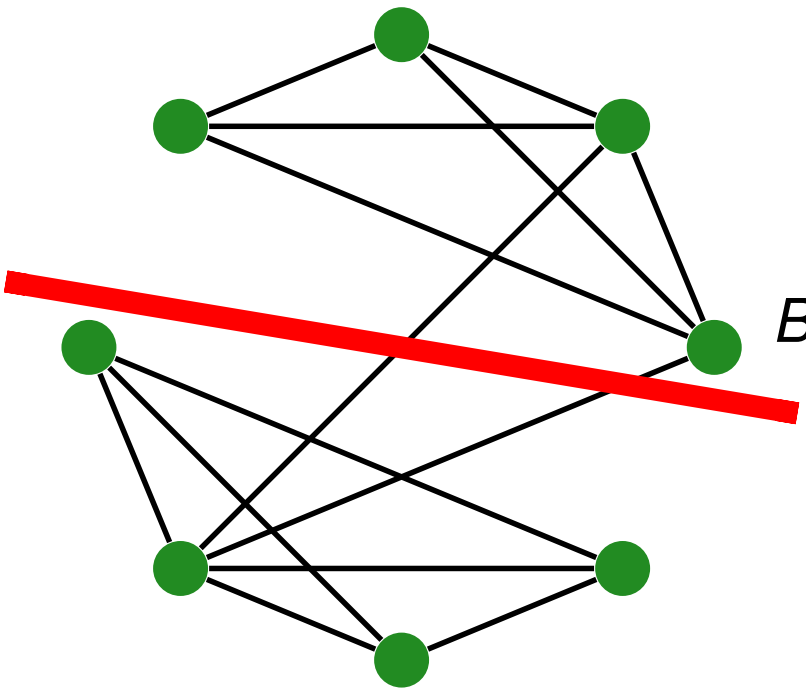
$$Bx = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ -1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & & & & & \vdots \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

$\binom{n}{2} \times n$

Sparsification

$$\text{value of cut} = \sum_{e=(u,v)\in E} (x_u - x_v)^2 = ||Bx||^2 = x^T B^T Bx$$

$L = \mathbf{B}^T\mathbf{B}$ is the Laplacian of G



$$Bx = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & & & & & \vdots \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

$\binom{n}{2} \times n$

Definition (Karger'94, Cut sparsifiers)

G' is an ε-cut sparsifier of G if

$$(1-\varepsilon)x^T Lx \leq x^T L'x \leq (1+\varepsilon)x^T Lx$$

for all $x \in \{0,1\}^V$ (all cuts).

Theorem (Karger'94, Benczur-Karger'96)

For any G there exists an ε-cut sparsifier G' with $O(\frac{1}{\varepsilon^2} n \log n)$ edges, and it can be constructed in $\tilde{O}(m)$ time.

Definition (Spielman-Teng'04, Spectral sparsifiers)

G' is an ε-spectral sparsifier of G if

$$(1-\varepsilon)x^T Lx \leq x^T L'x \leq (1+\varepsilon)x^T Lx$$

for all ~~$x \in \{0,1\}^V$ (all cuts).~~ all $x \in \mathbb{R}^V$.

Equivalently, $(1-\varepsilon)L \prec L' \prec (1+\varepsilon)L$

Theorem (Spielman-Teng'04, Spielman-Srivastava'09)

For any G there exists an ε-spectral sparsifier G' with $O(\frac{1}{\varepsilon^2} n \log n)$ edges, and it can be constructed in $\widetilde{O}(m)$ time.

Definition (Spielman-Teng'04, Spectral sparsifiers)

G' is an ε-spectral sparsifier of G if

$$(1-\varepsilon)x^T Lx \leq x^T L'x \leq (1+\varepsilon)x^T Lx$$

for all ~~$x \in \{0,1\}^V$ (all cuts).~~ all $x \in \mathbb{R}^V$.

Equivalently, $(1-\varepsilon)L \prec L' \prec (1+\varepsilon)L$

Theorem (Spielman-Teng'04, Spielman-Srivastava'09)

For any G there exists an ε-spectral sparsifier G' with $O(\frac{1}{\varepsilon^2} n \log n)$ edges, and it can be constructed in $\tilde{O}(m)$ time.

Central role in numerical linear algebra, combinatorial optimization etc

Constructing spectral sparsifiers

Theorem (Spielman-Srivastava'09)

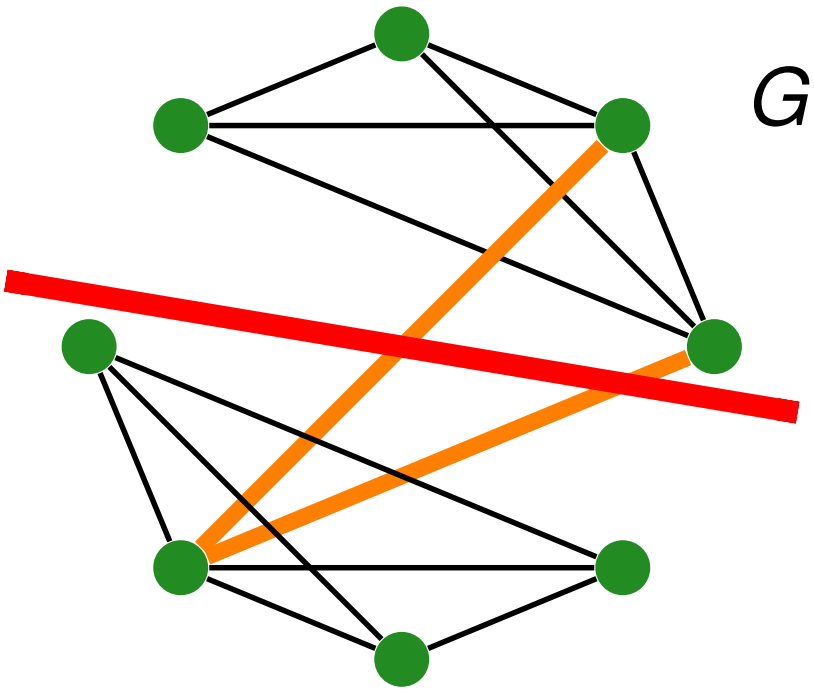
Let $G = (V, E)$ be an undirected graph. Let G' be obtained by including every edge $e \in E$ independently with probability proportional to its effective resistance:

$$p_e \geq \min\left\{1, \frac{C \log n}{\varepsilon^2} R_{uv}\right\}.$$

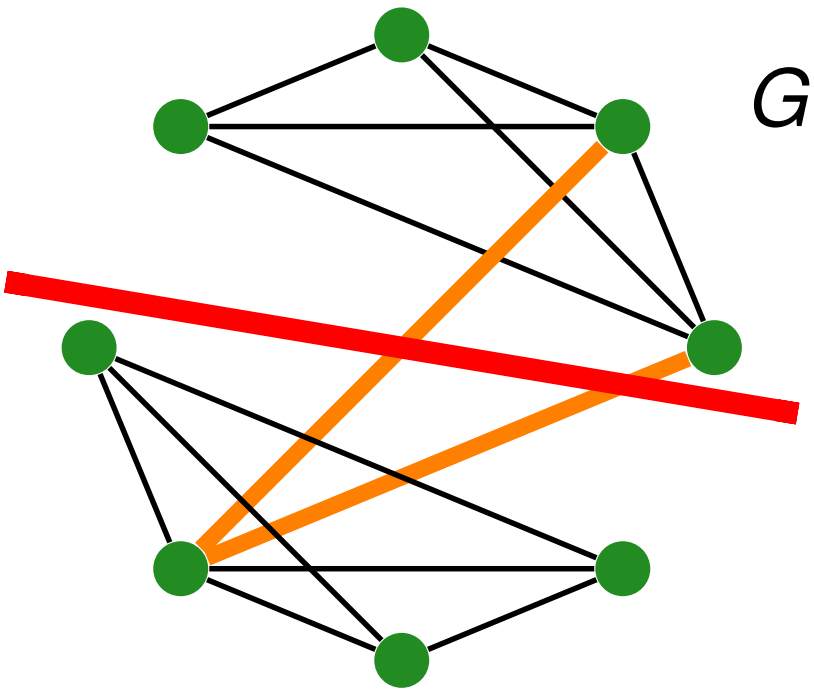
Assigning weight $1/p_e$ if sampled. Then $(1-\varepsilon)L \prec L' \prec (1+\varepsilon)L$ whp.

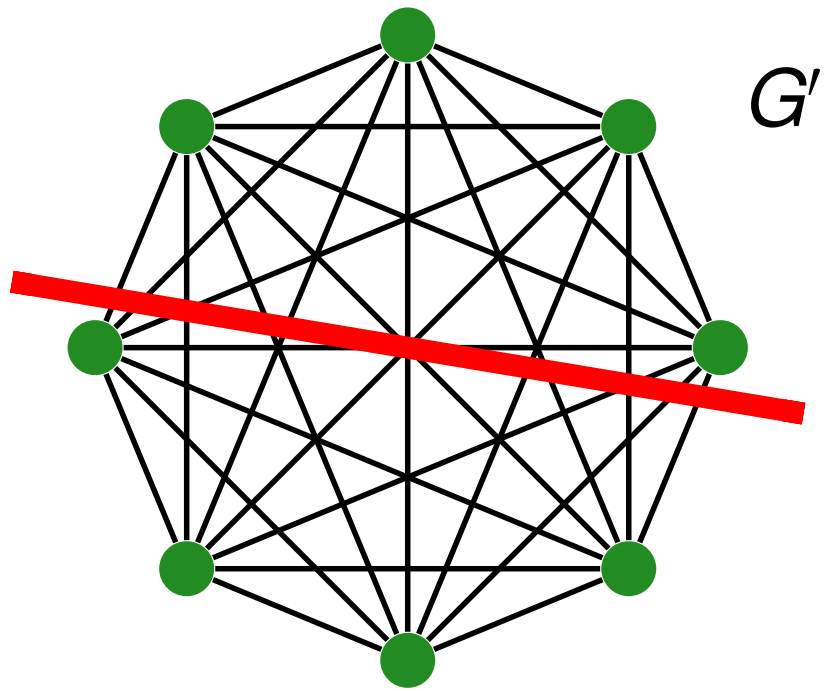
Sample edges according to a measure of importance, assign weights to make estimate unbiased

Sparsification



Sparsification



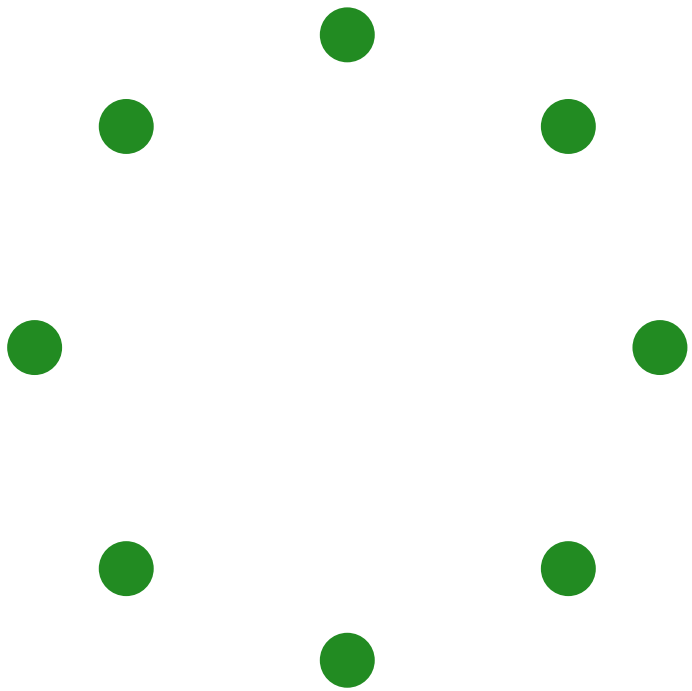


1. Spectral sparsification
2. Dynamic streaming and linear sketching
3. Spectral sparsification using small sketches
4. Fast decoding

1. Spectral sparsification
2. **Dynamic streaming and linear sketching**
3. Spectral sparsification using small sketches
4. Fast decoding

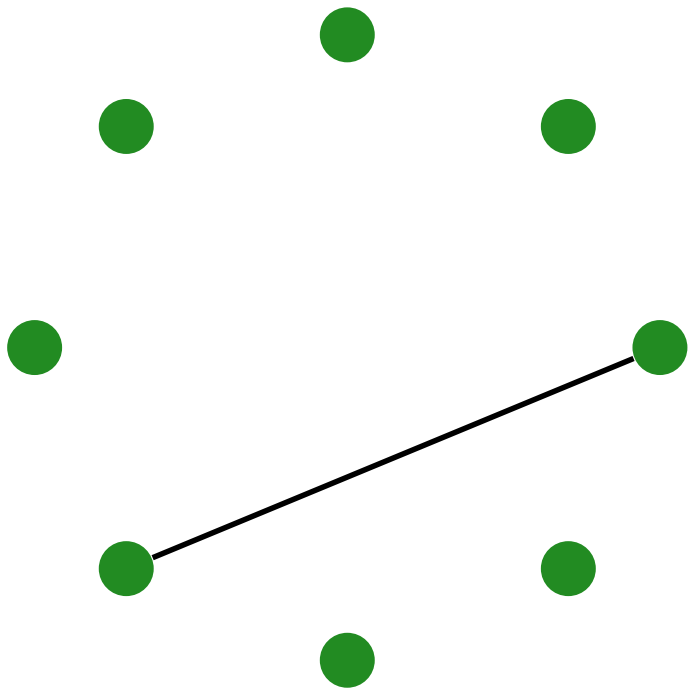
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\tilde{O}(n)$ space
- several passes over the stream



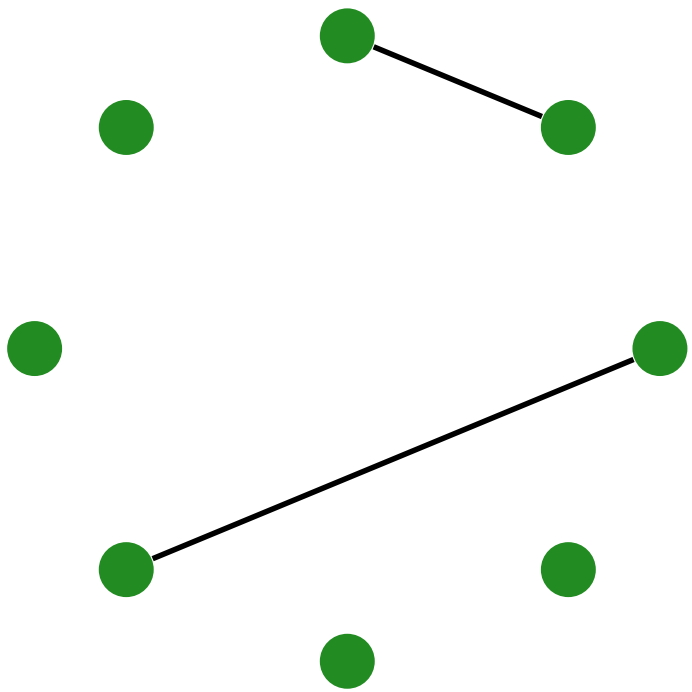
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\tilde{O}(n)$ space
- several passes over the stream



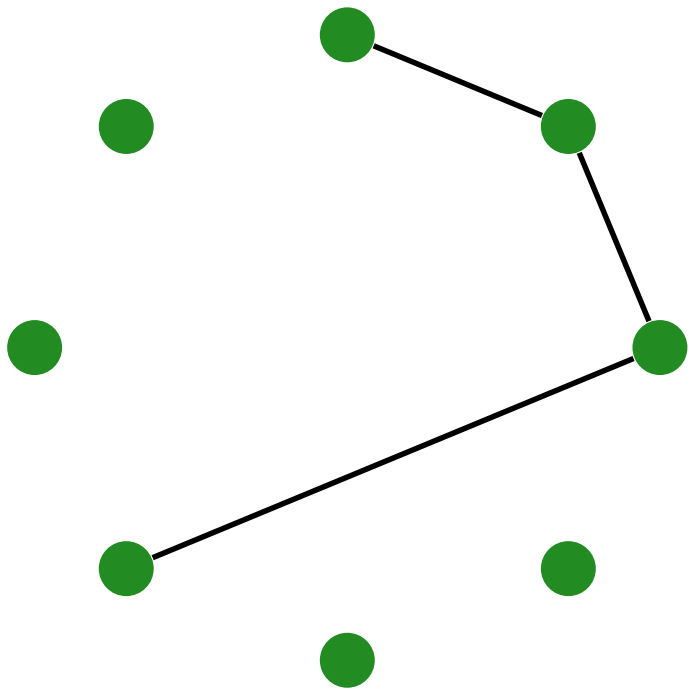
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\tilde{O}(n)$ space
- several passes over the stream



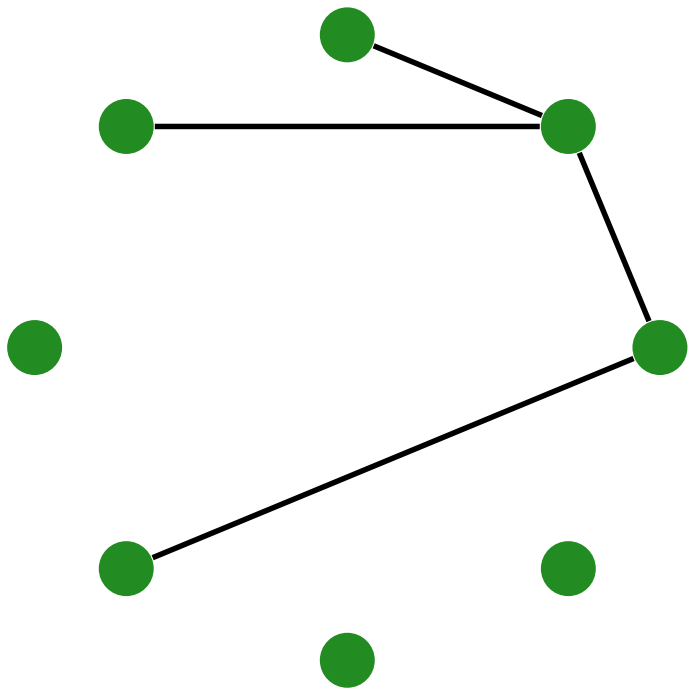
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\tilde{O}(n)$ space
- several passes over the stream



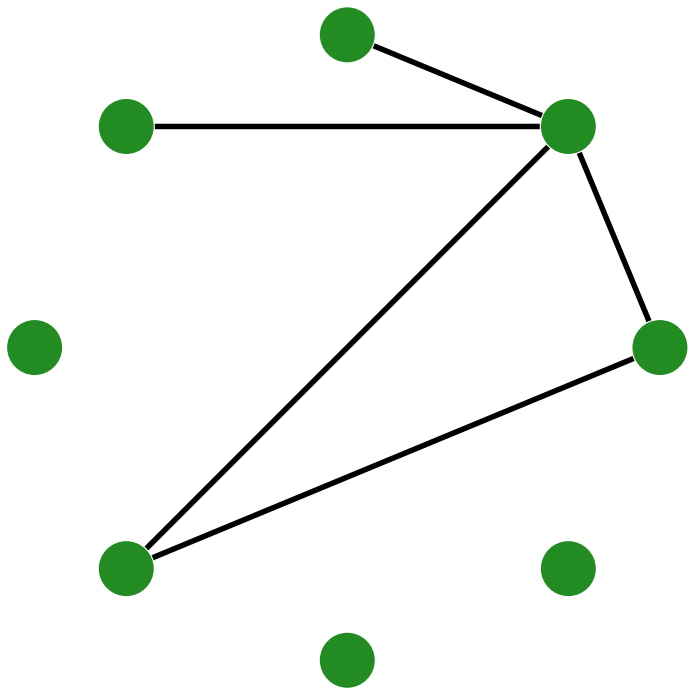
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\tilde{O}(n)$ space
- several passes over the stream



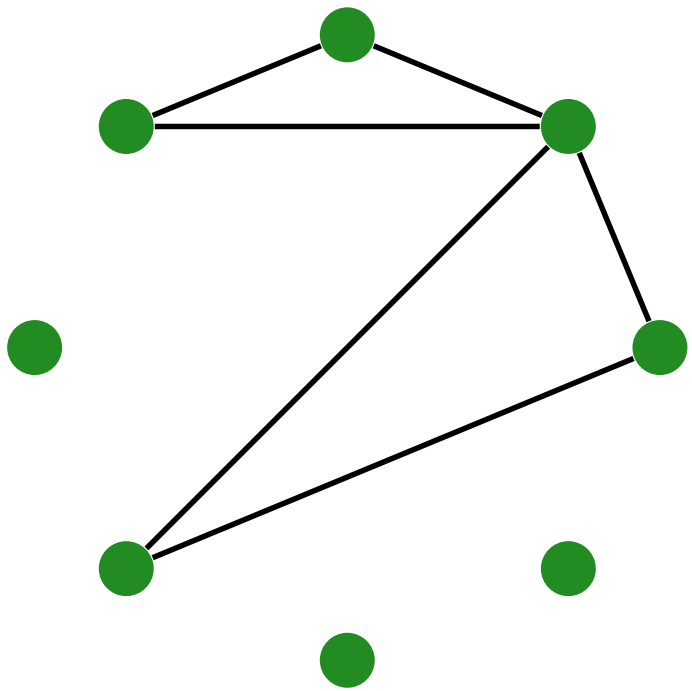
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\tilde{O}(n)$ space
- several passes over the stream



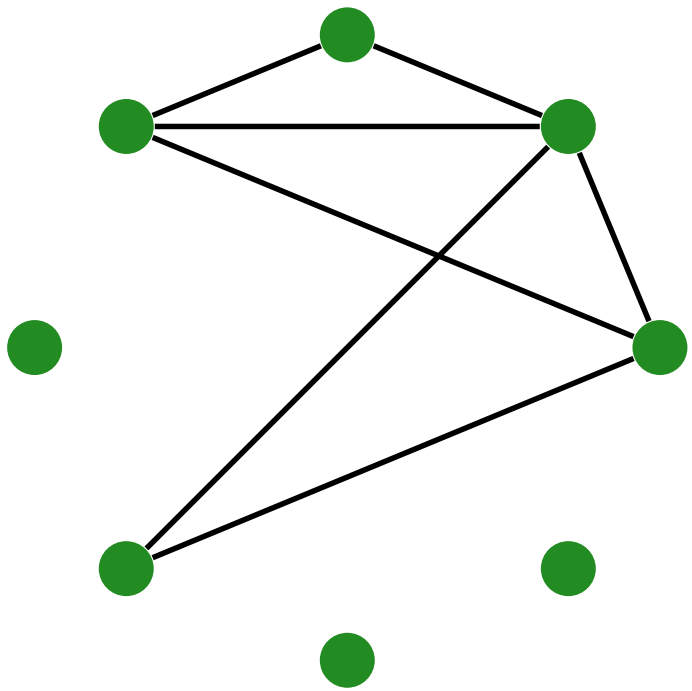
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\widetilde{O}(n)$ space
- several passes over the stream



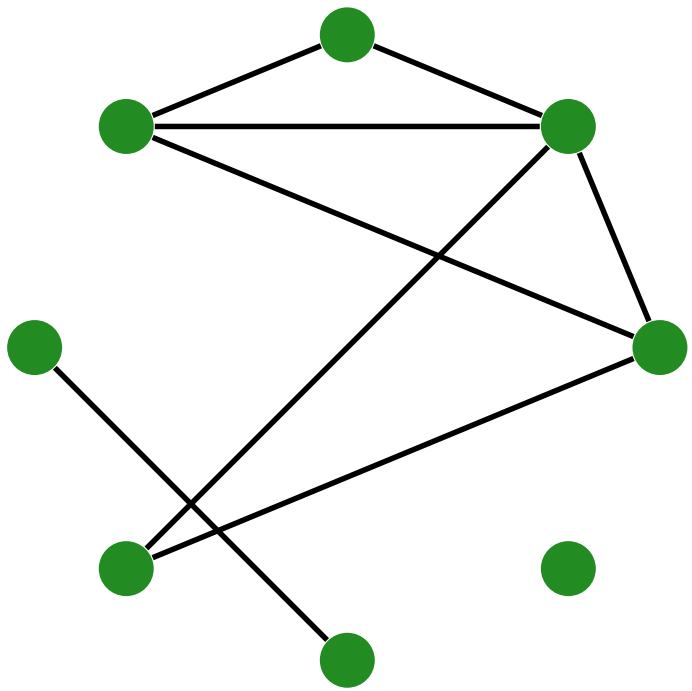
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\tilde{O}(n)$ space
- several passes over the stream



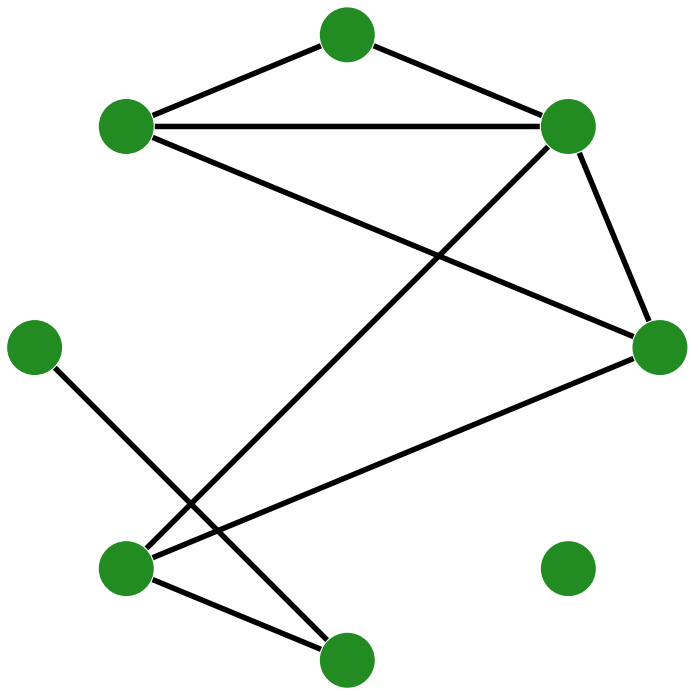
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\widetilde{O}(n)$ space
- several passes over the stream



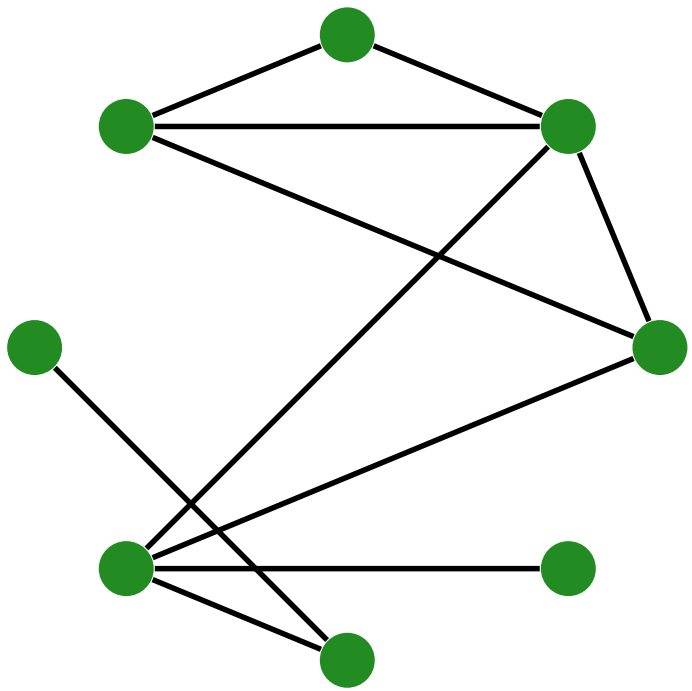
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\widetilde{O}(n)$ space
- several passes over the stream



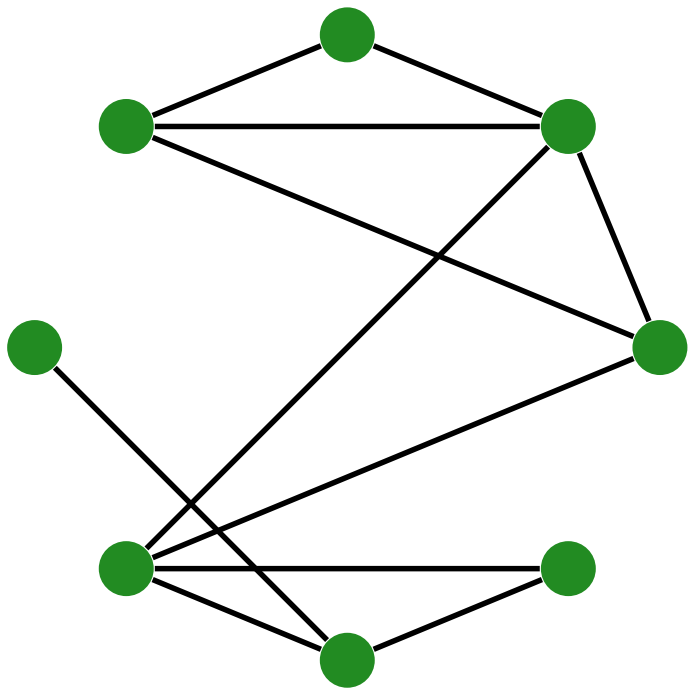
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\widetilde{O}(n)$ space
- several passes over the stream



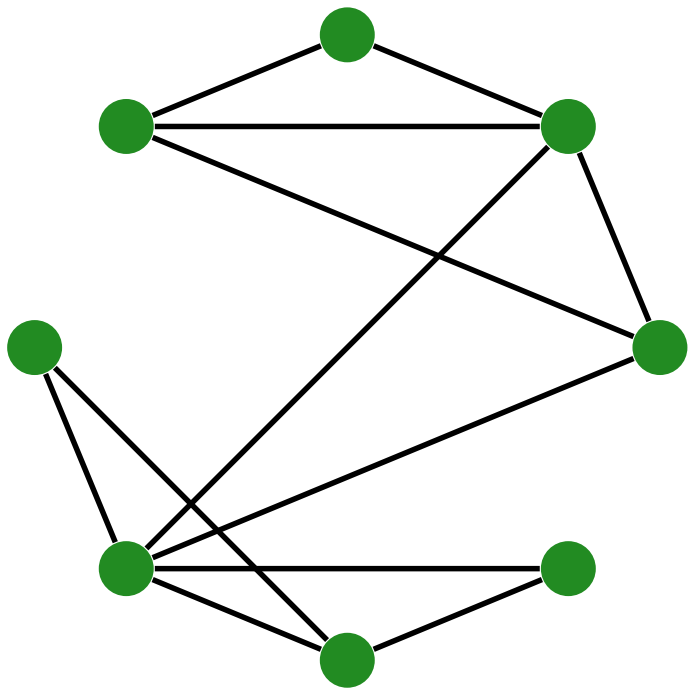
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\widetilde{O}(n)$ space
- several passes over the stream



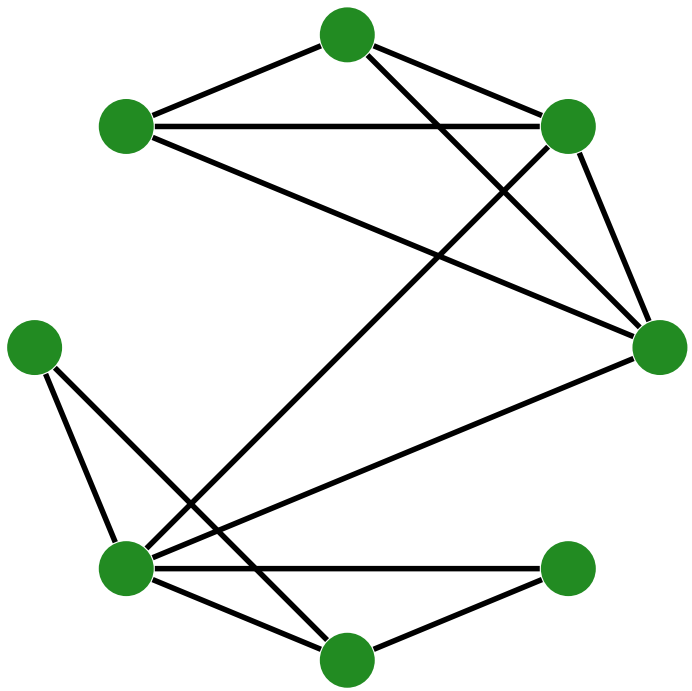
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\widetilde{O}(n)$ space
- several passes over the stream



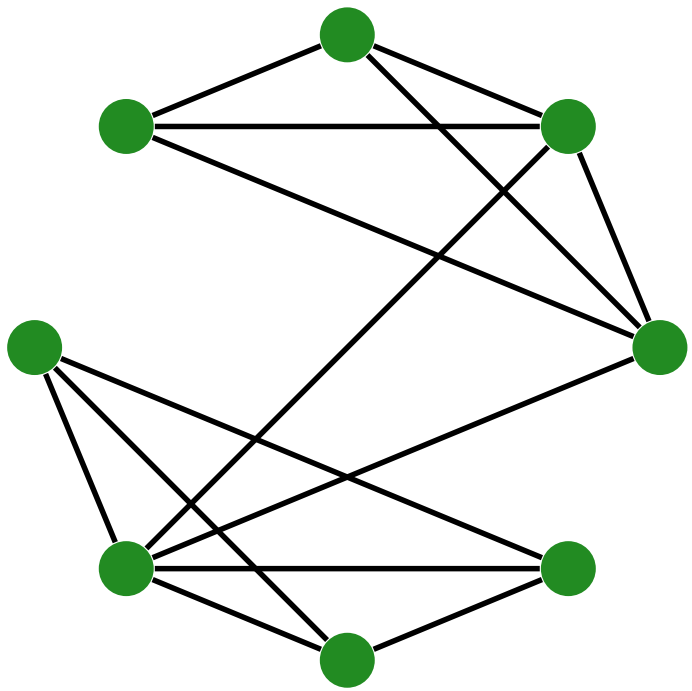
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\widetilde{O}(n)$ space
- several passes over the stream



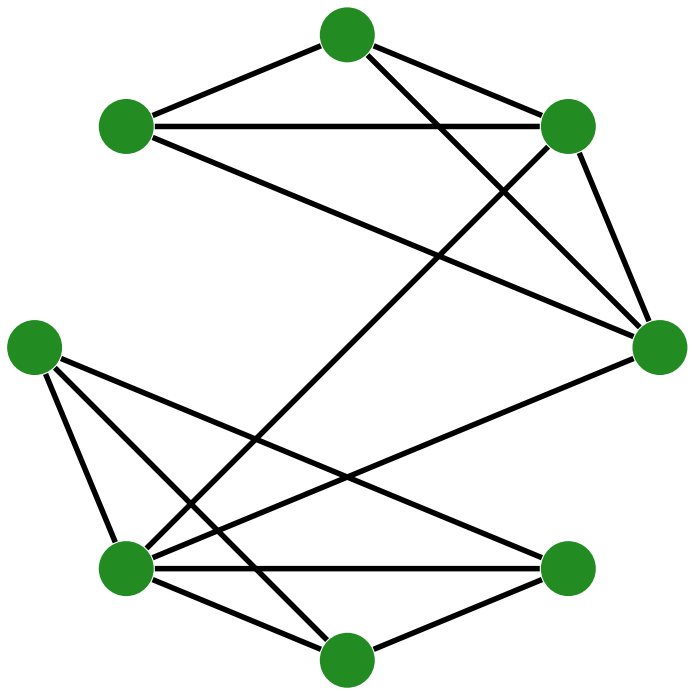
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\widetilde{O}(n)$ space
- several passes over the stream



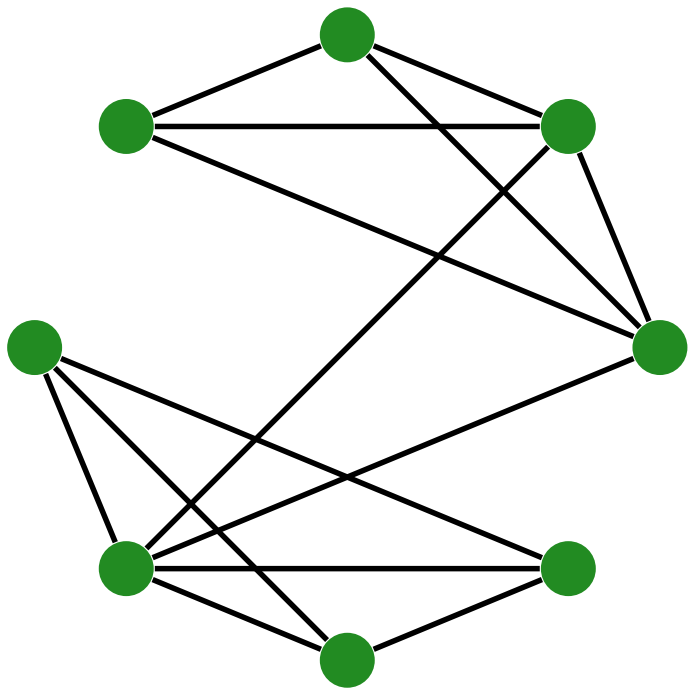
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\widetilde{O}(n)$ space
- several passes over the stream



Streaming model

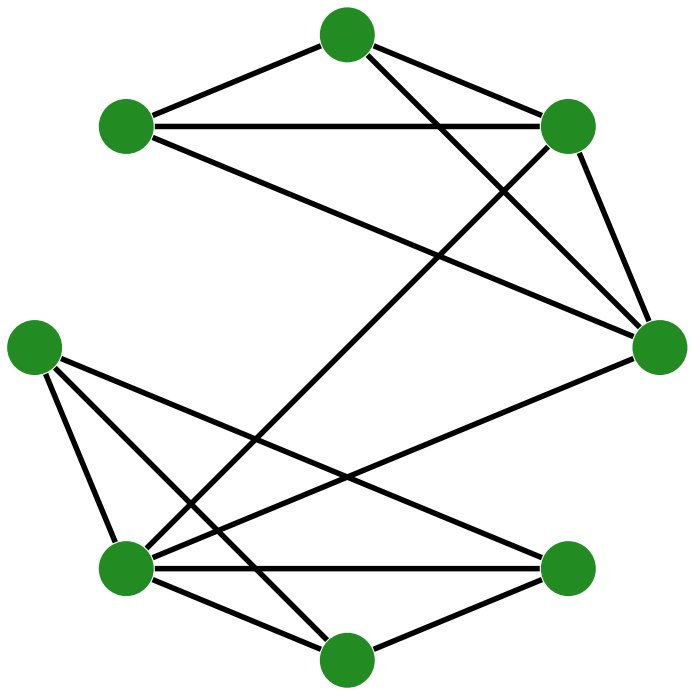
- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\widetilde{O}(n)$ space
- several passes over the stream **(ideally one pass)**



Insertion-only stream

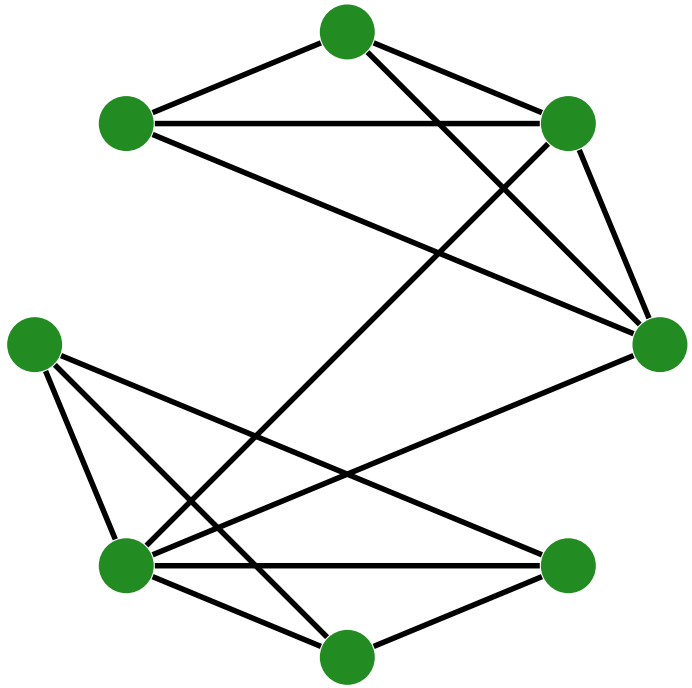
Streaming model

- edges of G arrive in an arbitrary order in a stream
- algorithm can only use $\widetilde{O}(n)$ space
- several passes over the stream **(ideally one pass)**

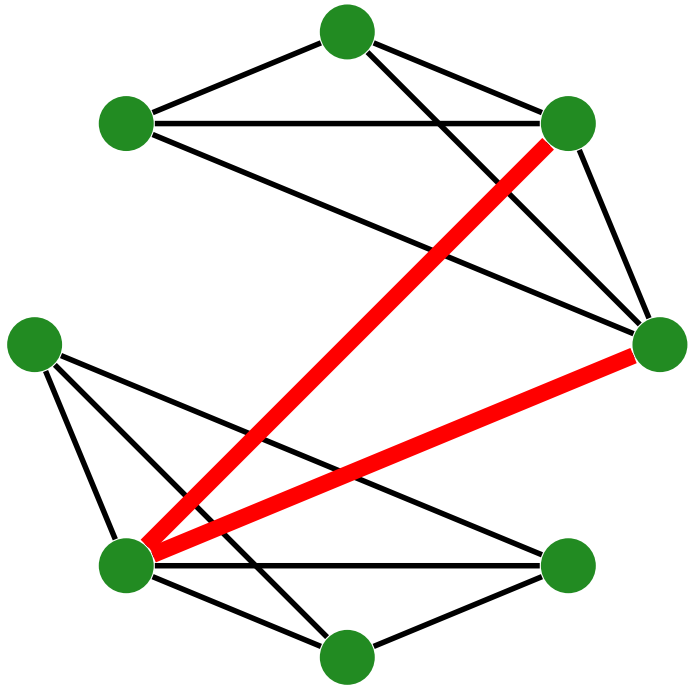


~~Insertion-only~~ stream

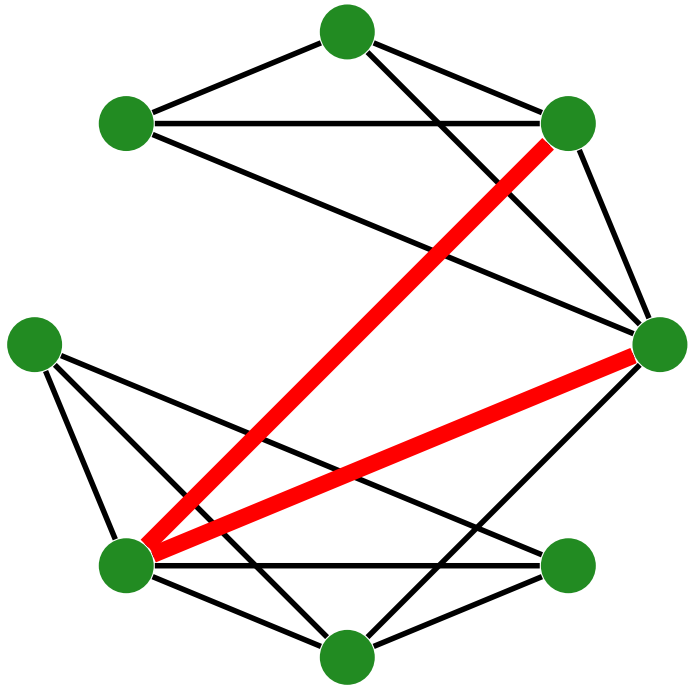
What if we have deletions?



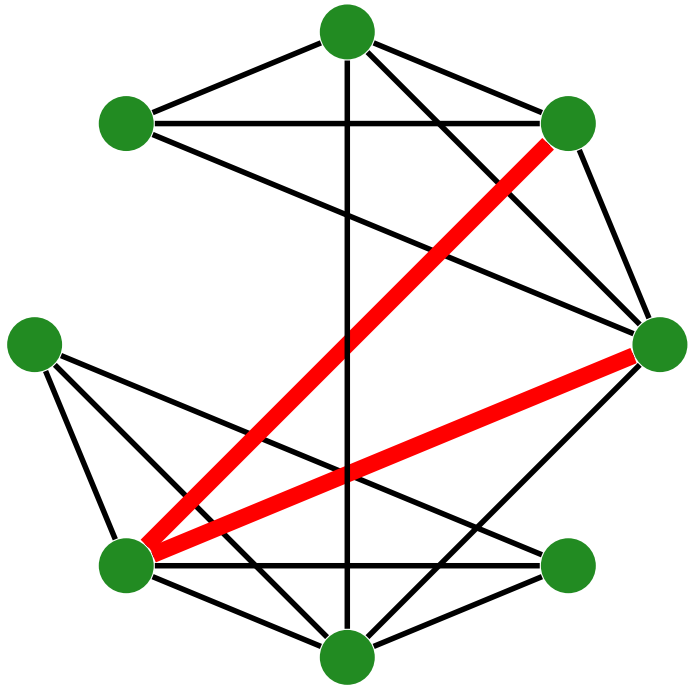
What if we have deletions?



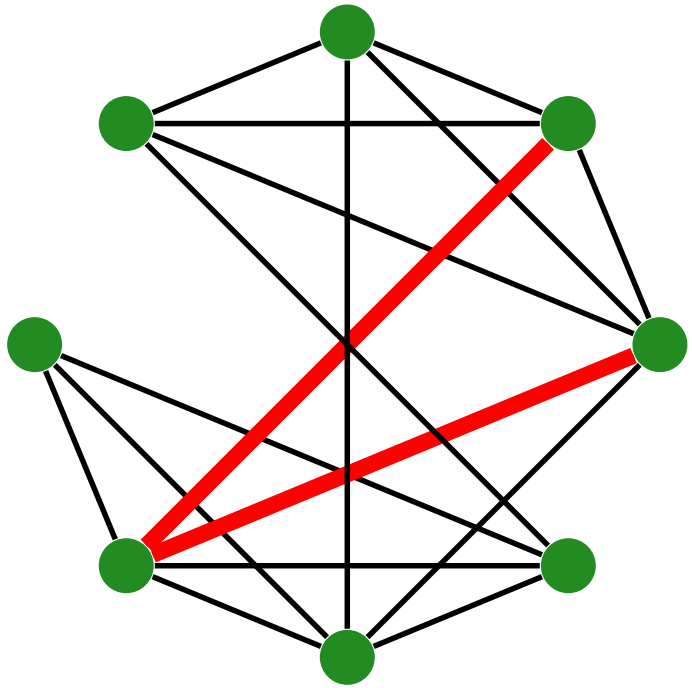
What if we have deletions?



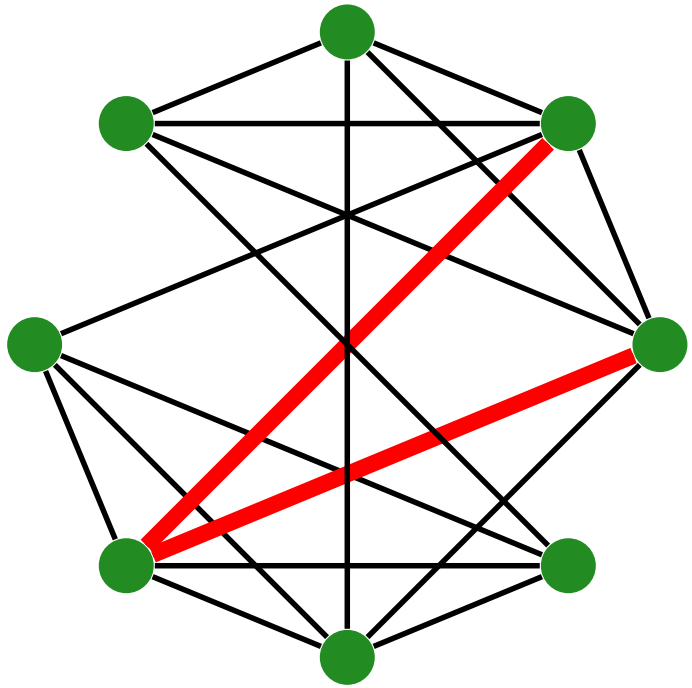
What if we have deletions?



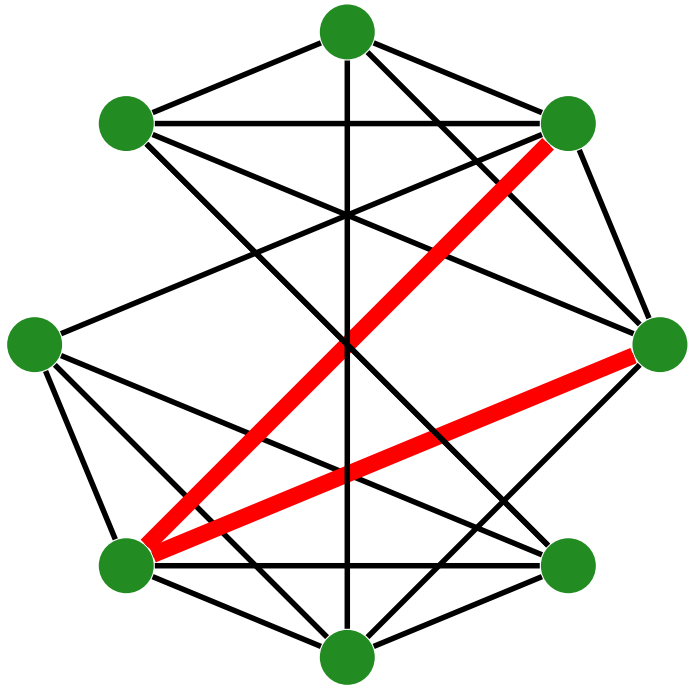
What if we have deletions?



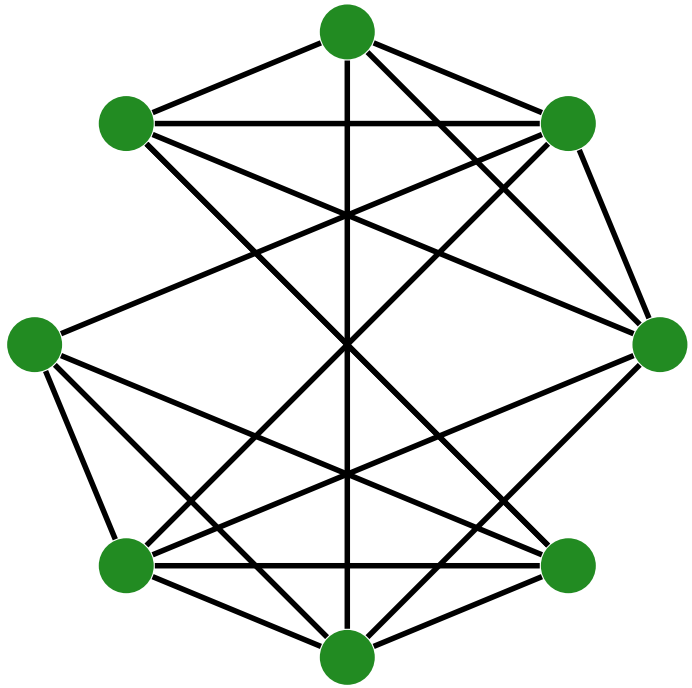
What if we have deletions?



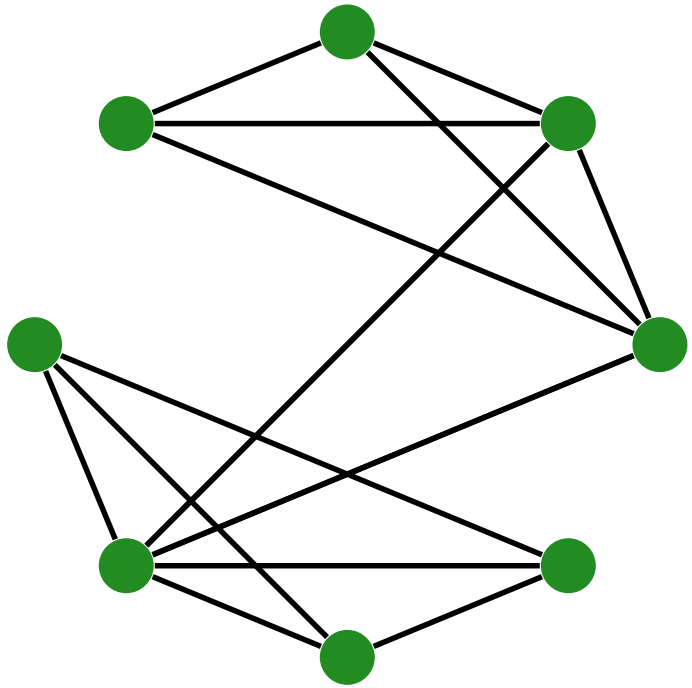
What if we have deletions?



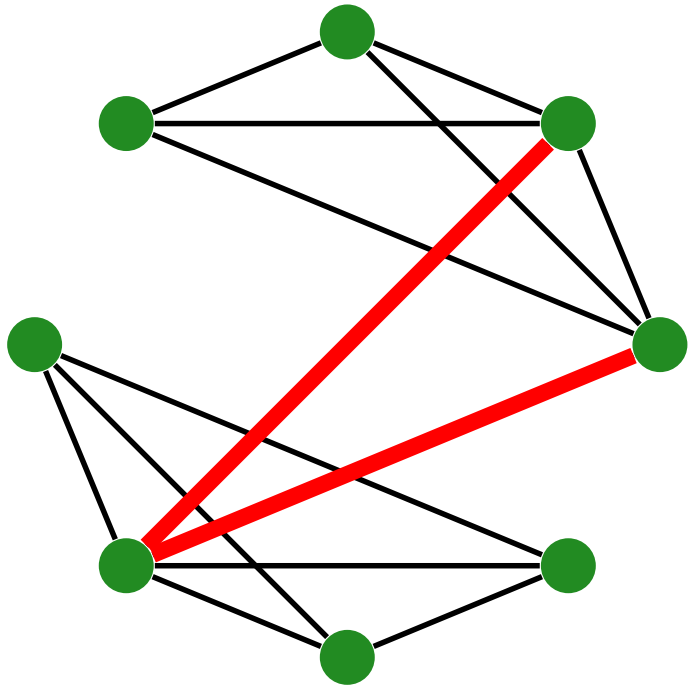
What if we have deletions?



What if we have deletions?



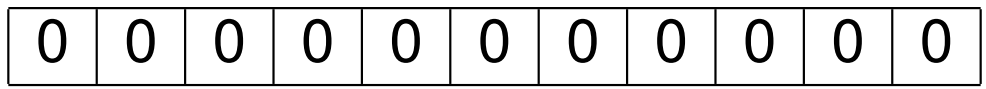
What if we have deletions?



Very different algorithms are needed...

Linear sketching

Classical data stream application: approximating frequency moments.



Goal: approximate $||x||_2^2 = \sum_i x_i^2$ using $\ll n$ space

Linear sketching

Classical data stream application: approximating frequency moments.

0	0	**1**	0	0	0	0	0	0	0	0

Goal: approximate $||x||_2^2 = \sum_i x_i^2$ using $\ll n$ space

Linear sketching

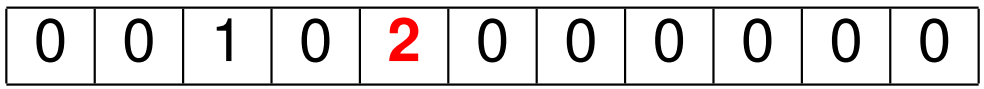
Classical data stream application: approximating frequency moments.

0	0	1	0	**1**	0	0	0	0	0	0

Goal: approximate $||x||_2^2 = \sum_i x_i^2$ using $\ll n$ space

Linear sketching

Classical data stream application: approximating frequency moments.



Goal: approximate $||x||_2^2 = \sum_i x_i^2$ using $\ll n$ space

Linear sketching

Classical data stream application: approximating frequency moments.

0	0	1	0	**2**	0	0	0	0	0	0

Goal: approximate $||x||_2^2 = \sum_i x_i^2$ using $\ll n$ space

Maintain $x^T v_i = 1, \ldots, O(1/\varepsilon^2)$ for random Gaussians $v_i \in \mathbb{R}^n$.
Output average of $(x^T v_i)^2$.

Linear sketching

Classical data stream application: approximating frequency moments.

0	0	1	0	**2**	0	0	0	0	0	0

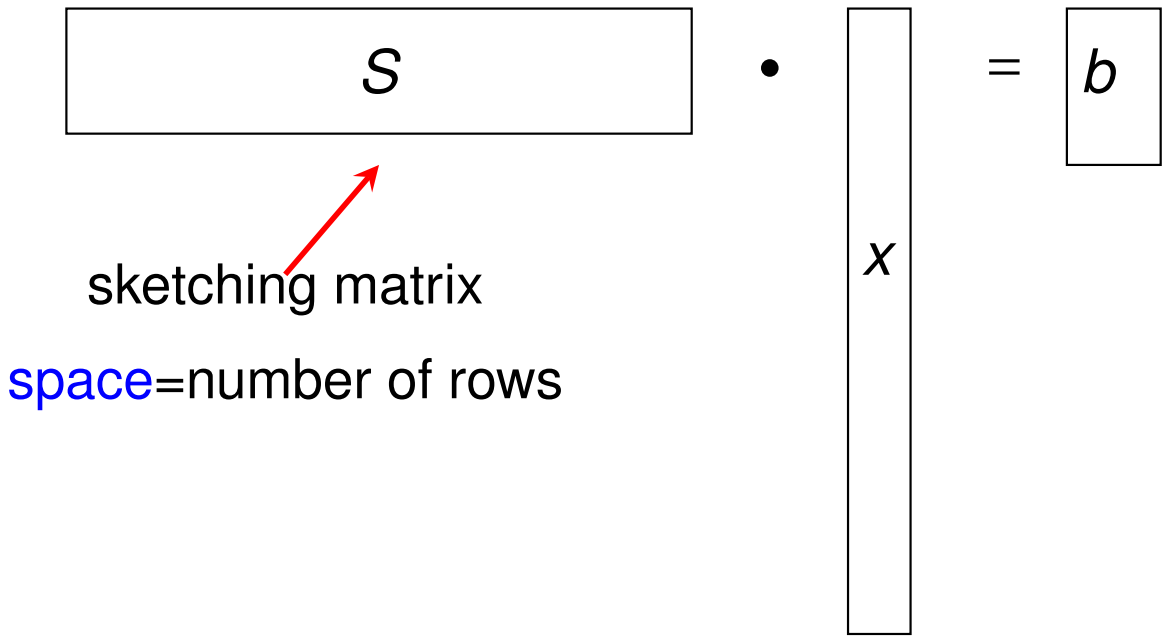
Goal: approximate $||x||_2^2 = \sum_i x_i^2$ using $\ll n$ space

Maintain $x^T v_i = 1, \ldots, O(1/\varepsilon^2)$ for random Gaussians $v_i \in \mathbb{R}^n$.
Output average of $(x^T v_i)^2$.

$(1 \pm \varepsilon)$-approximation with $O(\frac{1}{\varepsilon^2} \log n)$ space

Linear sketching

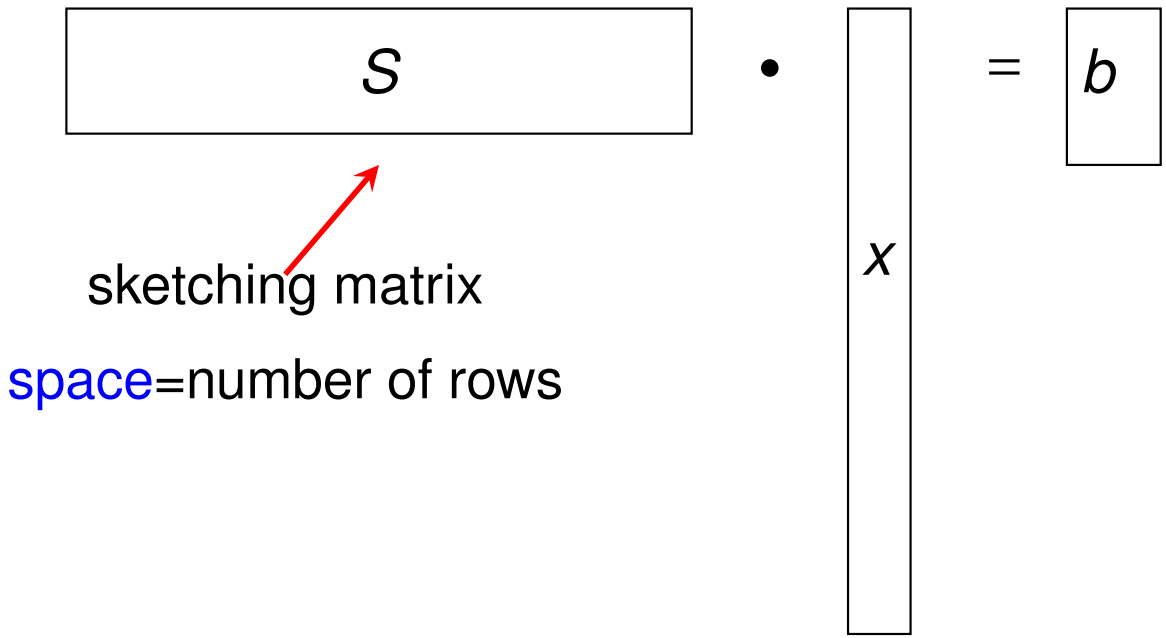
Take (randomized) linear measurements of the input



space=number of rows

Linear sketching

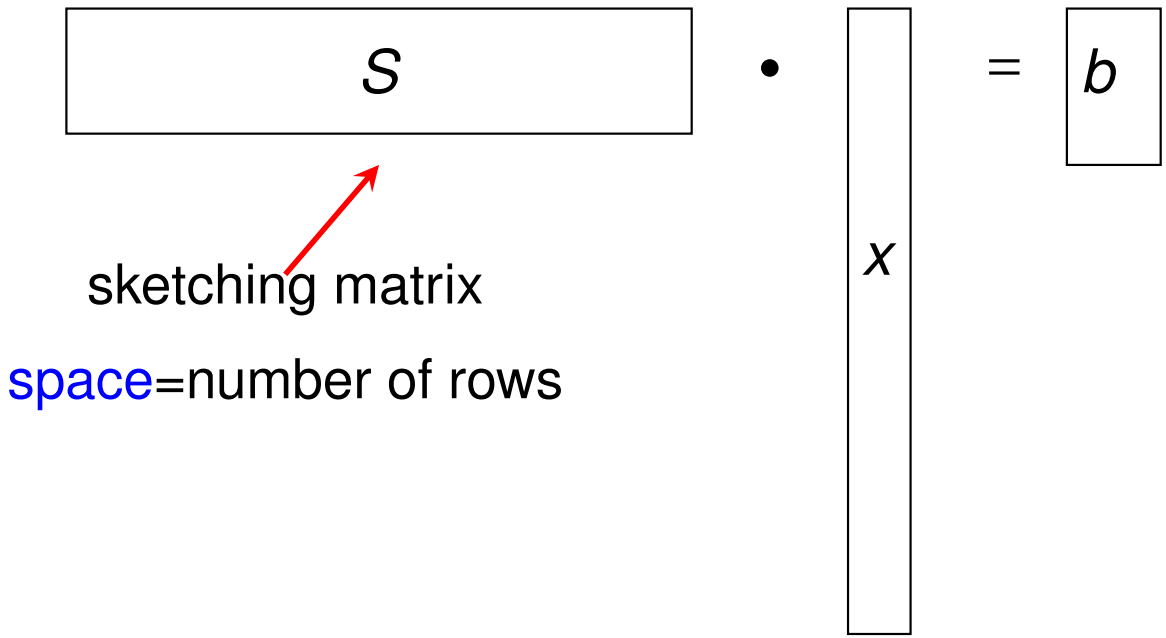
Take (randomized) linear measurements of the input



Can get constant approximation using $\log^{O(1)} n$ rows

Linear sketching

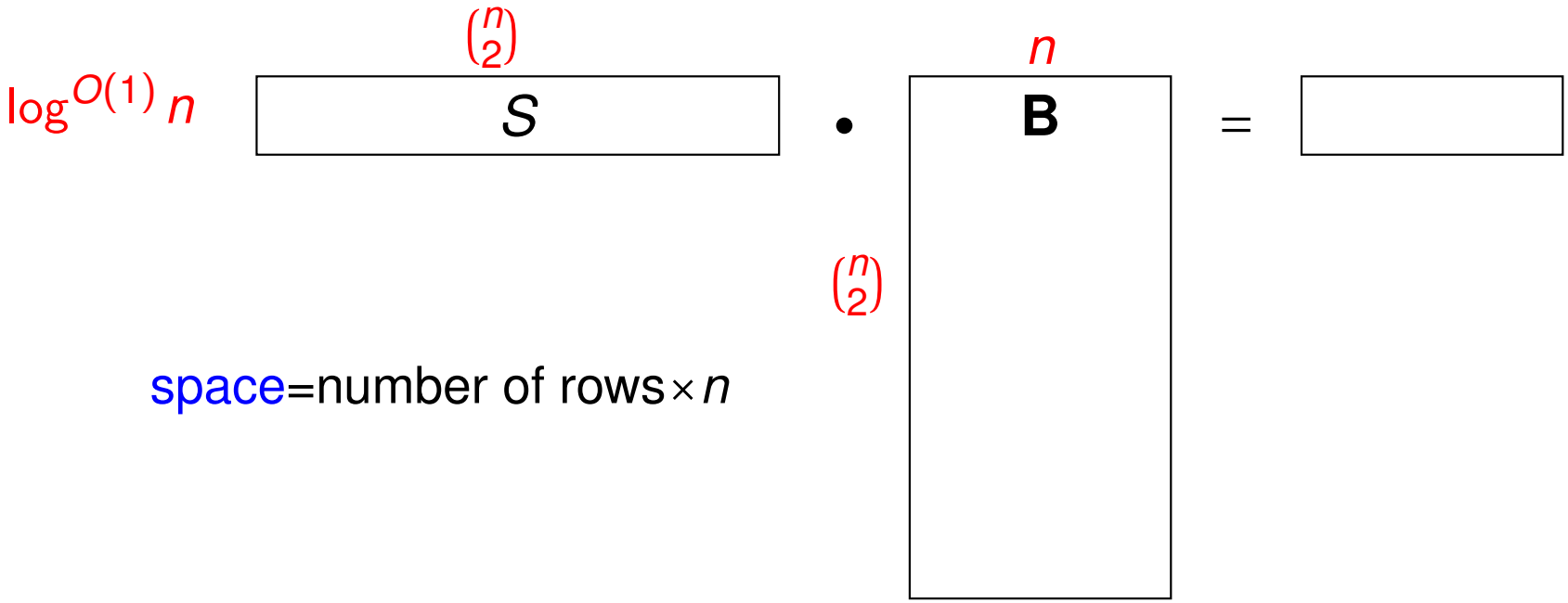
Take (randomized) linear measurements of the input



Can get constant approximation using $\log^{O(1)} n$ rows

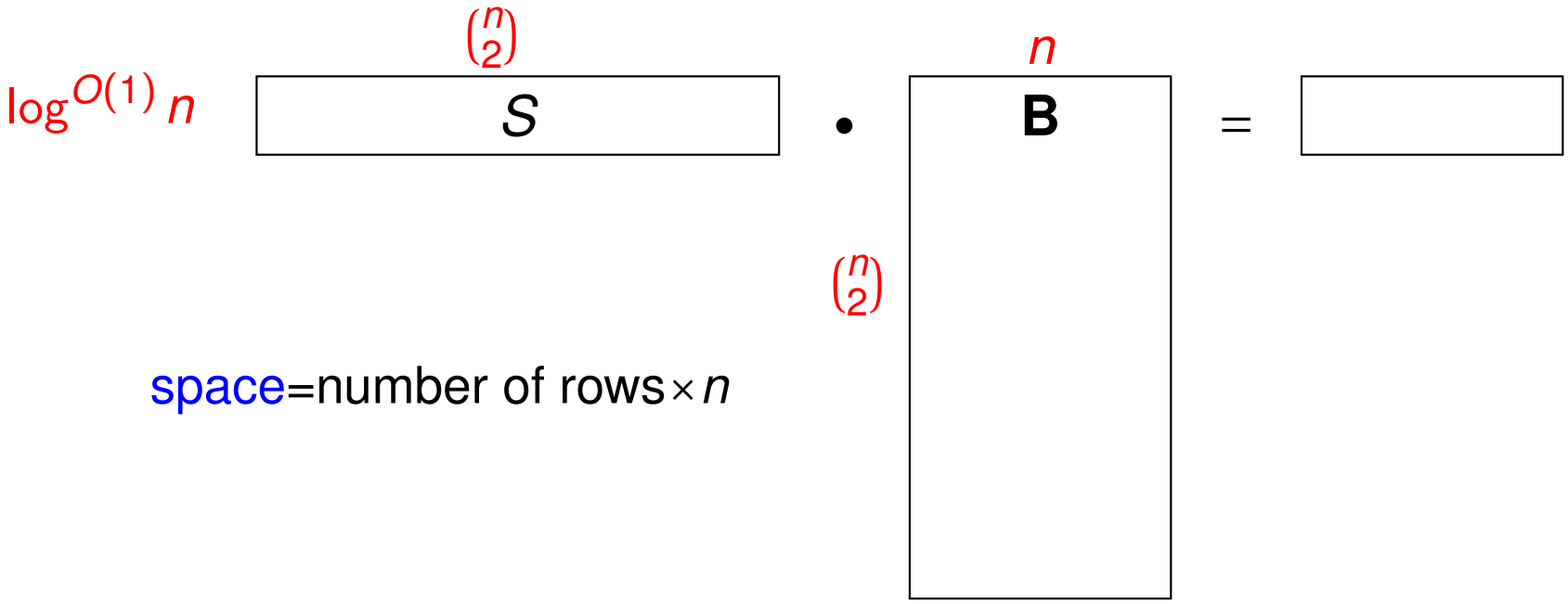
Easy to maintain linear sketches in the (dynamic) streaming model

Ahn-Guha-McGregor'SODA12 – connectivity in $O(n \cdot \log^3 n)$ space.



space=number of rows$\times n$

Ahn-Guha-McGregor'SODA12 – connectivity in $O(n \cdot \log^3 n)$ space.



Every vertex sketches its own neighborhood independently – single round distributed algorithms

Reconstruct edges of a sparsifier quickly from a small sketch?

Reconstruct edges of a sparsifier quickly from a small sketch?

Theorem (K.-Nouri-Sidford-Tardos'19)

Can recover a spectral sparsifier from an oblivious sketch of size $\tilde{O}(n)$ space in time $\tilde{O}(n)$.

Known before:

$\tilde{O}(n)$ space and $\Omega(n^2)$ recovery time
K.-Lee-Sidford-Musco-Musco'FOCS'14

$\tilde{O}(n^{5/3})$ space and time Ahn-Guha-McGregor'APPROX'14

1. Spectral sparsification
2. Dynamic streaming and linear sketching
3. Spectral sparsification using small sketches
4. Fast decoding

1. Spectral sparsification
2. Dynamic streaming and linear sketching
3. **Spectral sparsification using small sketches**
4. Fast decoding

Constructing spectral sparsifiers

Theorem (Spielman-Srivastava'09)

Let $G = (V, E)$ be an undirected graph. Let G' be obtained by including every edge $e \in E$ independently with probability proportional to its effective resistance:

$$p_e \geq \min\left\{1, \frac{C \log n}{\varepsilon^2} R_{uv}\right\}.$$

Assign weight $1/p_e$ if sampled. Then $(1-\varepsilon)G \prec G' \prec (1+\varepsilon)G$ whp.

Sample edges according to a measure of importance,
assign weights to make estimate unbiased

Note: edges e with resistance $R_{uv} = \Omega(1/\log n)$ included w.p. 1

Core primitive: recover edges of large resistance

Goal: design a sketch S that allows recovery of high resistance ($\geq 1/\log n$) edges of G given

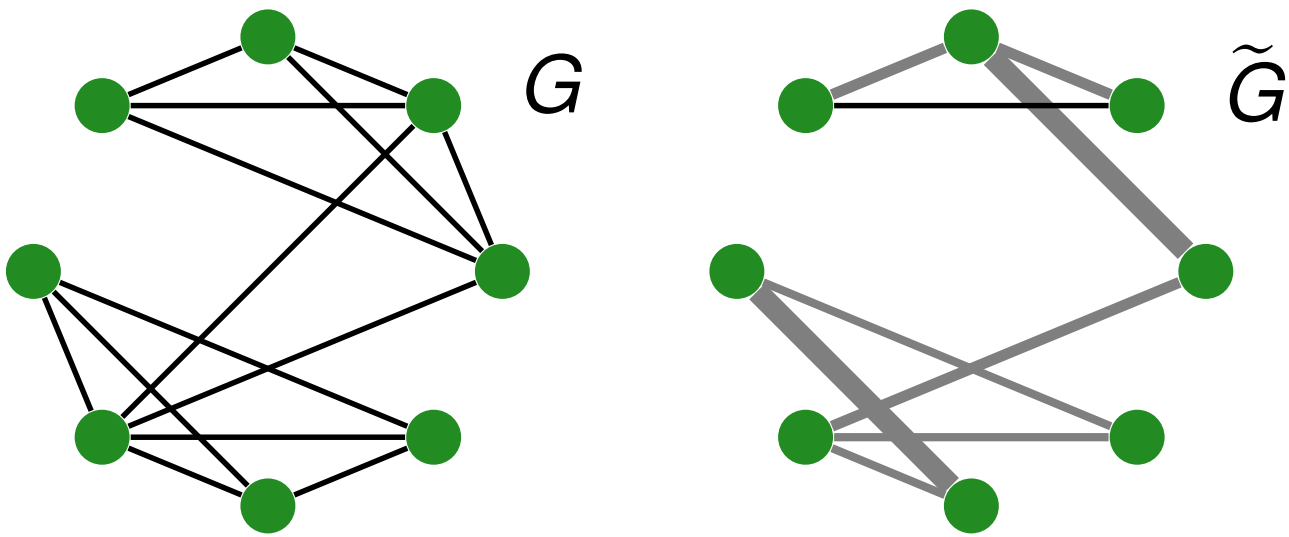
- $S \cdot \mathbf{B}$ (sketch of edge incidence matrix)

Core primitive: recover edges of large resistance

Goal: design a sketch S that allows recovery of high resistance ($\geq 1/\log n$) edges of G given

- $S \cdot \mathbf{B}$ (sketch of edge incidence matrix)
- crude constant factor spectral sparsifier $\tilde{G}$

$$\frac{1}{C} \cdot L \prec \tilde{L} \prec L$$

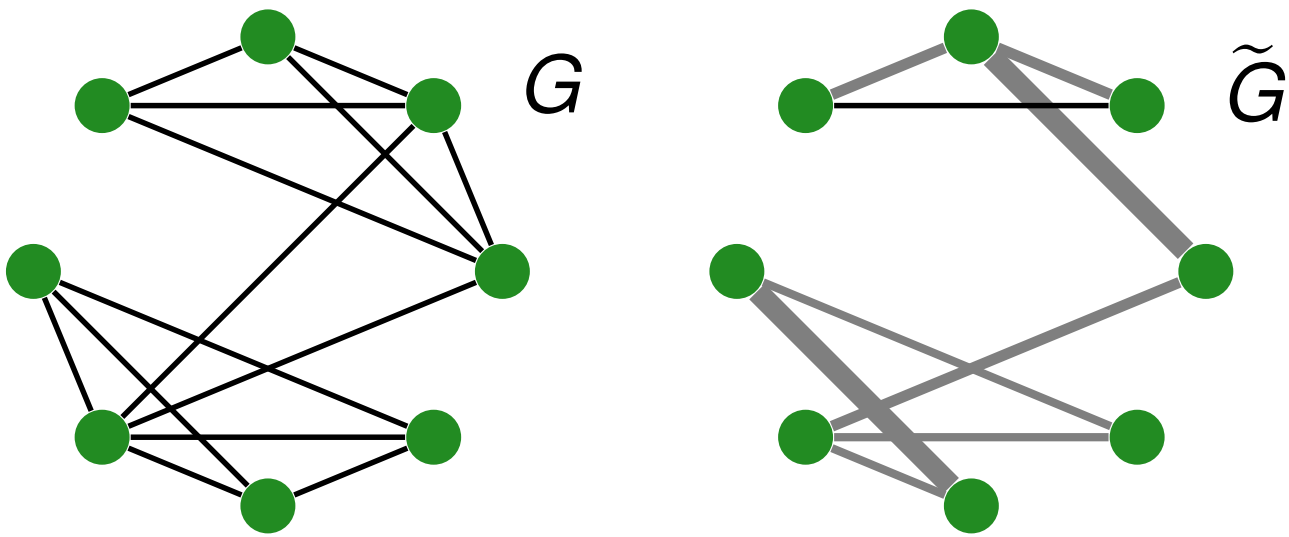


Core primitive: recover edges of large resistance

Goal: design a sketch S that allows recovery of high resistance ($\geq 1/\log n$) edges of G given

- $S \cdot \mathbf{B}$ (sketch of edge incidence matrix)
- crude constant factor spectral sparsifier $\tilde{G}$

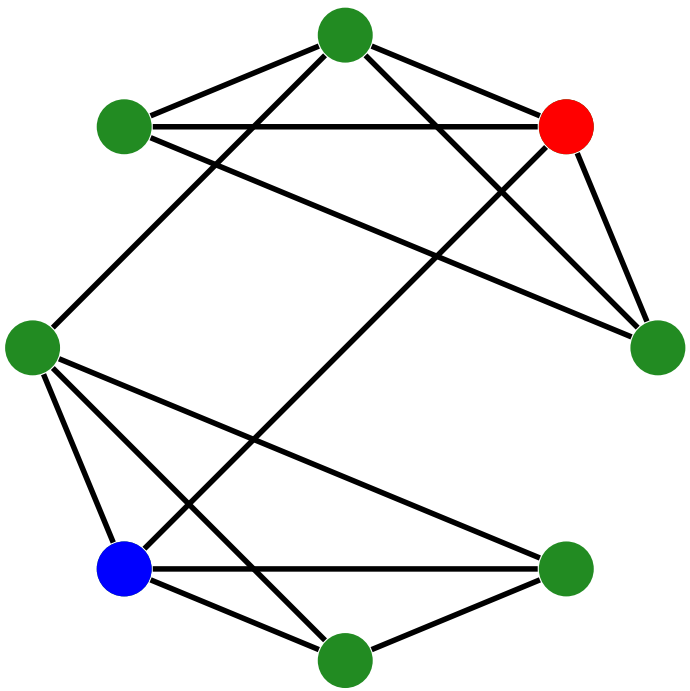
$$\frac{1}{C} \cdot L \prec \tilde{L} \prec L$$



Effective resistance

$$R_{uv} = \mathbf{b}_{uv}^T L^+ \mathbf{b}_{uv}$$

Note: defined for any pair (u, v).

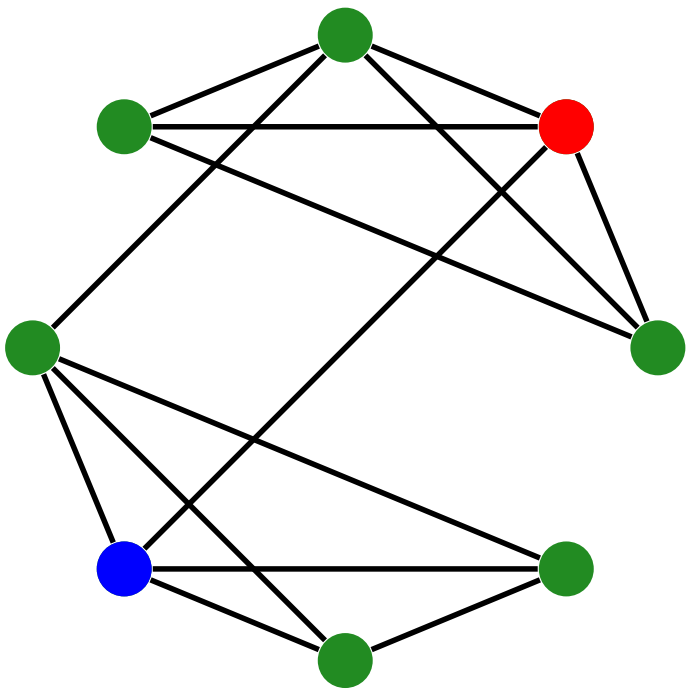


Inject current at u, take out at v.

Effective resistance

$$R_{uv} = \mathbf{b}_{uv}^T L^+ \mathbf{b}_{uv}$$

Note: defined for any pair (u, v).



Inject current at u, take out at v.

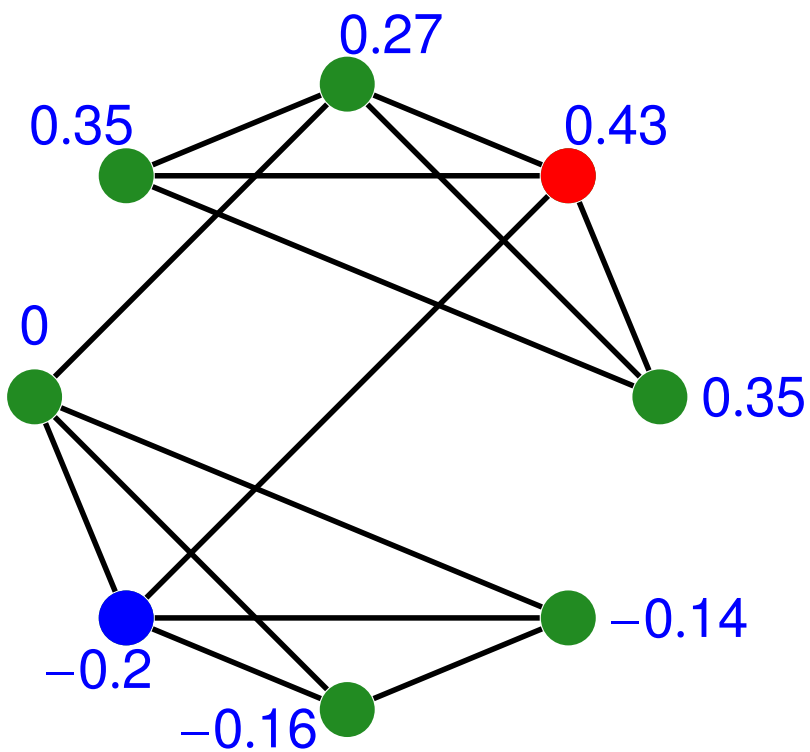
$\phi = L^+ \mathbf{b}_{uv}$=vertex potentials

$$\mathbf{b}_{uv} = \begin{bmatrix} 0 \\ \mathbf{1} \\ 0 \\ \mathbf{-1} \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Effective resistance

$$R_{uv} = \mathbf{b}_{uv}^T L^+ \mathbf{b}_{uv}$$

Note: defined for any pair (u, v).



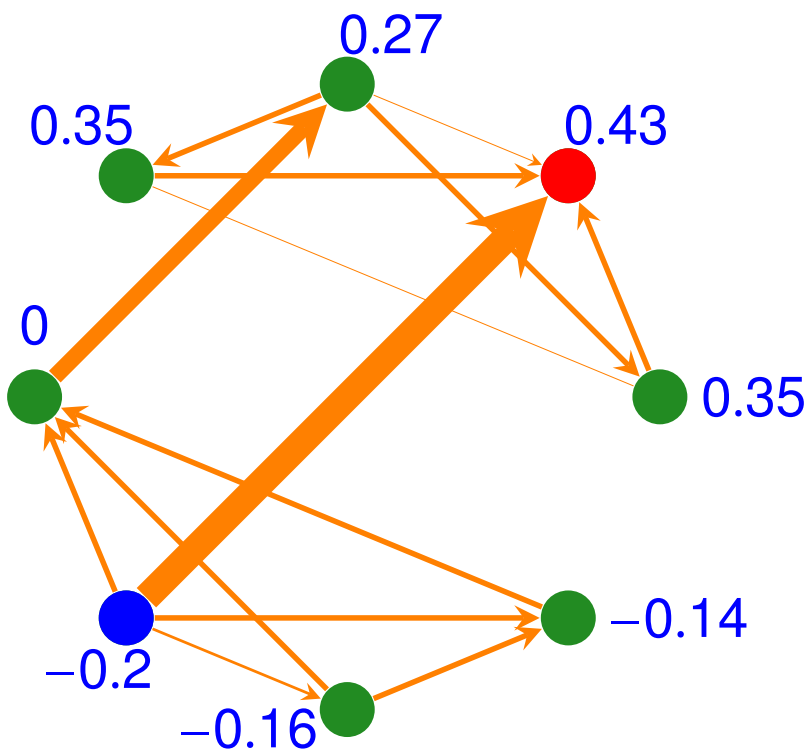
Inject current at u, take out at v.

$\phi = L^+ \mathbf{b}_{uv}$=vertex potentials

Effective resistance

$$R_{uv} = \mathbf{b}_{uv}^T L^+ \mathbf{b}_{uv}$$

Note: defined for any pair (u, v).



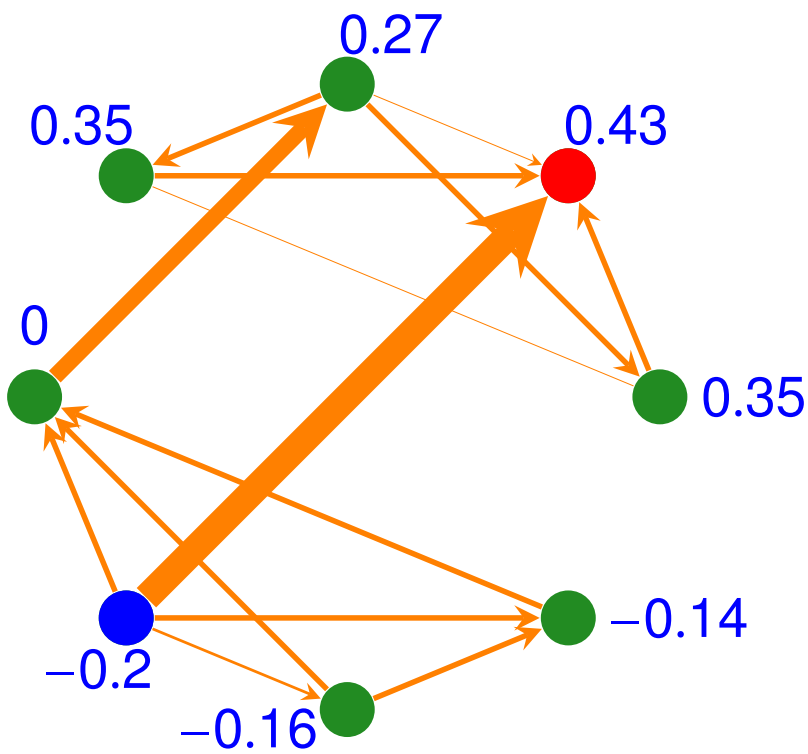
Inject current at u, take out at v.

$\phi = L^+ \mathbf{b}_{uv}$=vertex potentials

Effective resistance

$$R_{uv} = \mathbf{b}_{uv}^T L^+ \mathbf{b}_{uv}$$

Note: defined for any pair (u, v).



Inject current at u, take out at v.

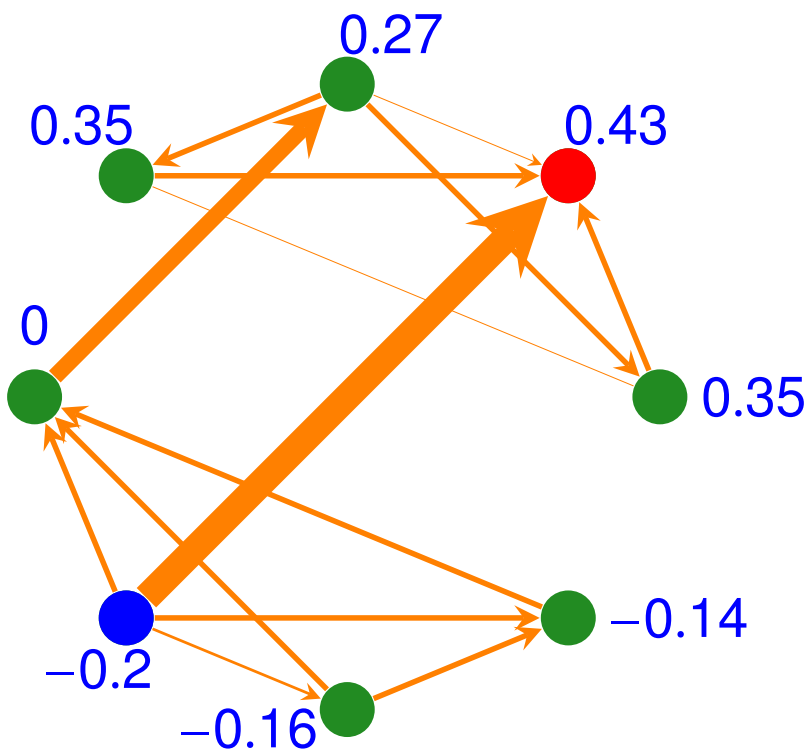
$\phi = L^+ \mathbf{b}_{uv}$=vertex potentials

$f_{xy} = \phi_y - \phi_x = \mathbf{b}_{xy}^T L^+ \mathbf{b}_{uv}$

Effective resistance

$$R_{uv} = \mathbf{b}_{uv}^T L^+ \mathbf{b}_{uv}$$

Note: defined for any pair (u, v).



Inject current at u, take out at v.

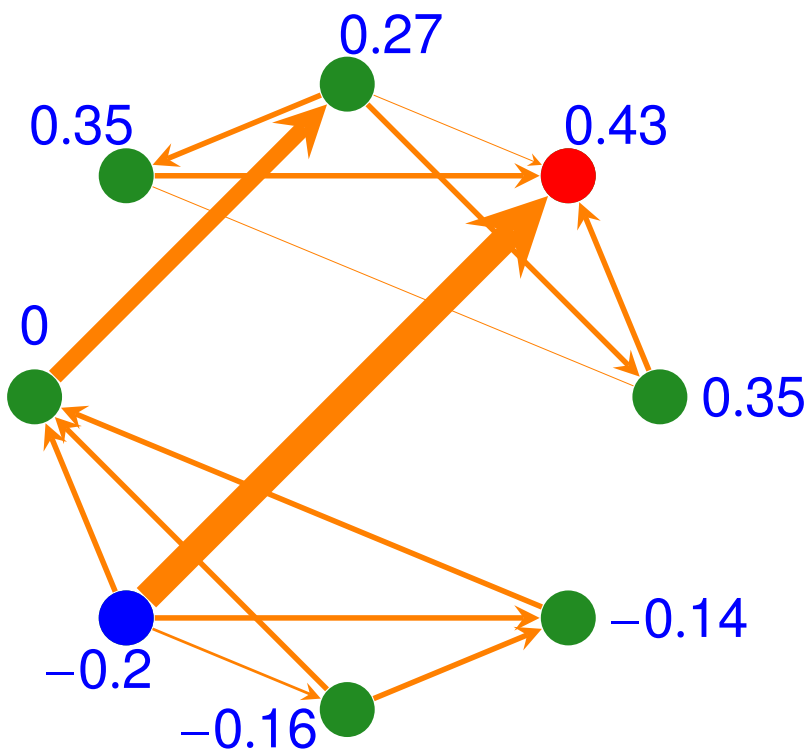
$\phi = L^+ \mathbf{b}_{uv}$=vertex potentials

$f = \mathbf{B}\phi$=currents on edges

Effective resistance

$$R_{uv} = \mathbf{b}_{uv}^T L^+ \mathbf{b}_{uv}$$

Note: defined for any pair (u, v).



Inject current at u, take out at v.

$\phi = L^+ \mathbf{b}_{uv}$=vertex potentials

$f = \mathbf{B}\phi$=currents on edges

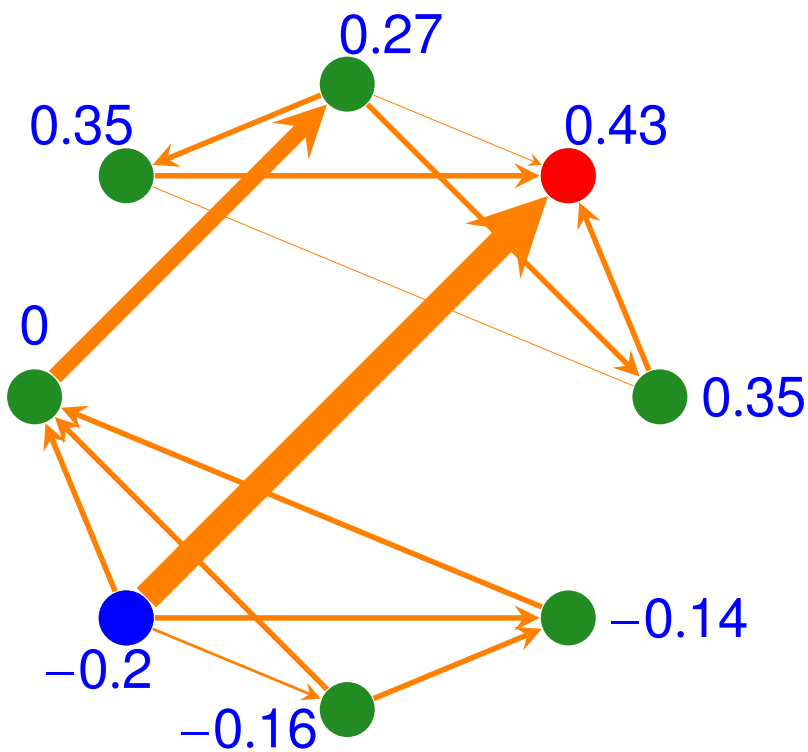
We have

$$R_{uv} = \frac{f_{uv}^2}{||f||^2}.$$

Effective resistance

$$R_{uv} = \mathbf{b}_{uv}^T L^+ \mathbf{b}_{uv}$$

Note: defined for any pair (u, v).



Inject current at u, take out at v.

$\phi = L^+ \mathbf{b}_{uv}$=vertex potentials

$f = \mathbf{B}\phi$=currents on edges

We have

$$R_{uv} = \frac{f_{uv}^2}{||f||^2}.$$

R_{uv}= fraction of $||f||_2^2$ contributed by $e = (u, v)$

Given:

- a sketch $S \cdot \mathbf{B}$ of G
- crude sparsifier $\widetilde{G}$
- pair $(u, v) \in V \times V$

Need:

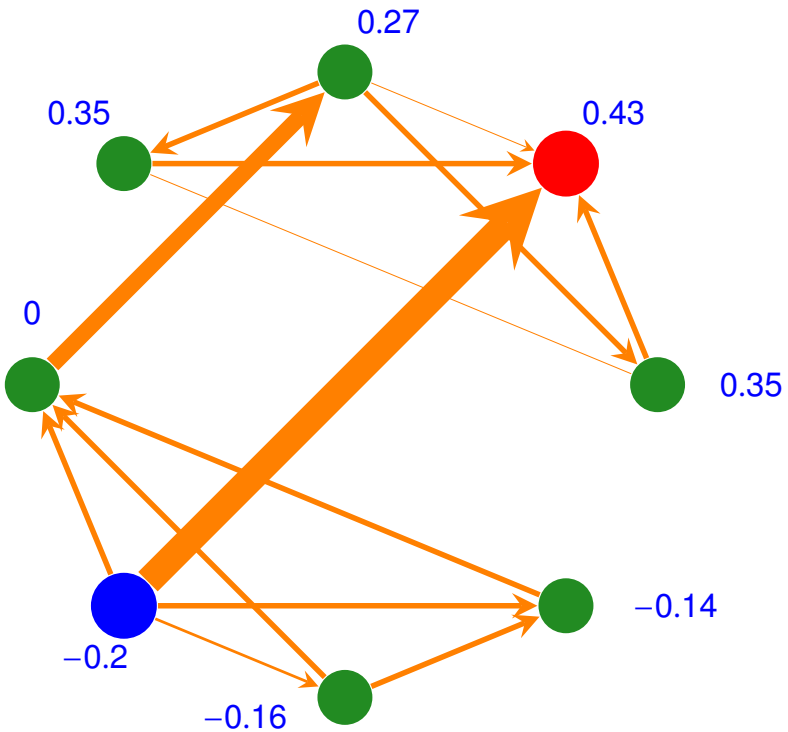
- is (u, v) an edge in G of resistance $\Omega(1/\log n)$?

Given:

- a sketch $S \cdot \mathbf{B}$ of G
- crude sparsifier $\widetilde{G}$
- pair $(u, v) \in V \times V$

Need:

- is (u, v) an edge in G of resistance $\Omega(1/\log n)$?



Compute $\phi = L^{+}\mathbf{b}_{uv}$ – vertex potentials

Compute $f = \mathbf{B}\phi$=currents on edges

Check if

$$R_{uv} = \frac{f_{uv}^2}{||f||^2} = \Omega(1/\log n).$$

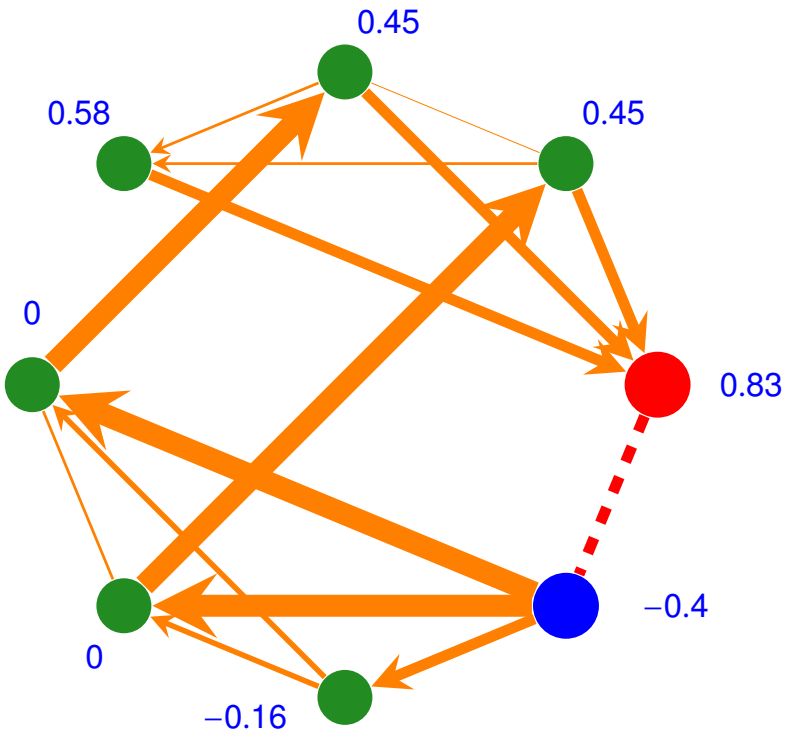
0	0	0.01	0	**200**	1	0	2	0	0	0

Given:

- a sketch $S \cdot \mathbf{B}$ of G
- crude sparsifier $\widetilde{G}$
- pair $(u, v) \in V \times V$

Need:

- is (u, v) an edge in G of resistance $\Omega(1/\log n)$?



Compute $\phi = L^{+}\mathbf{b}_{uv}$ – vertex potentials

Compute $f = \mathbf{B}\phi$=currents on edges

Check if

$$R_{uv} = \frac{f_{uv}^2}{||f||^2} = \Omega(1/\log n).$$

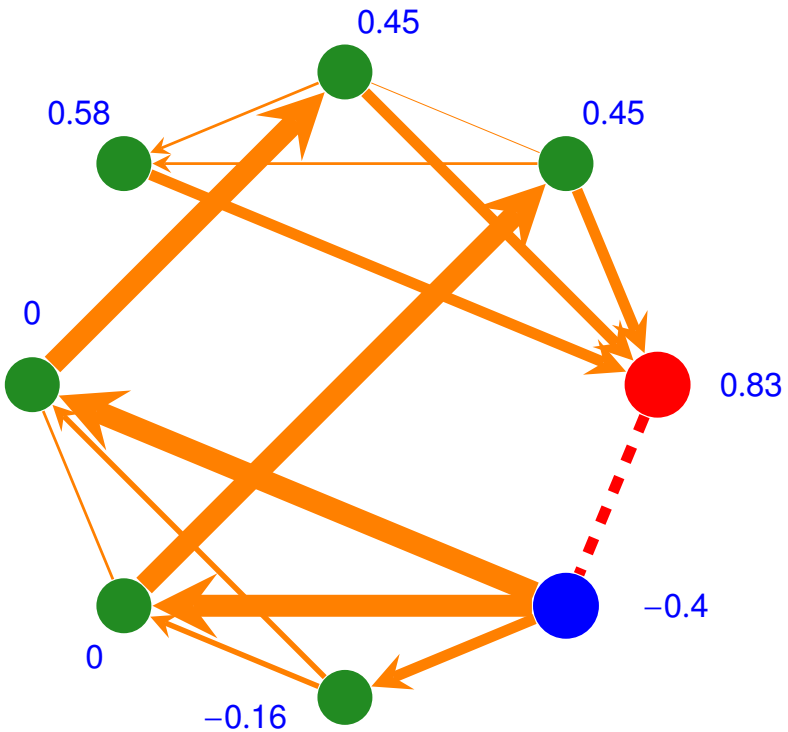
0	0	0.01	0	200	1	0	2	0	**0**	0

Given:

- a sketch $S \cdot \mathbf{B}$ of G
- crude sparsifier $\widetilde{G}$
- pair $(u, v) \in V \times V$

Need:

- is (u, v) an edge in G of resistance $\Omega(1/\log n)$?



Compute $\phi = \mathbf{L}^{+}\mathbf{b}_{uv}$ – vertex potentials

Compute $f = \mathbf{B}\phi$=currents on edges

Check if

$$R_{uv} = \frac{f_{uv}^2}{||f||^2} = \Omega(1/\log n).$$

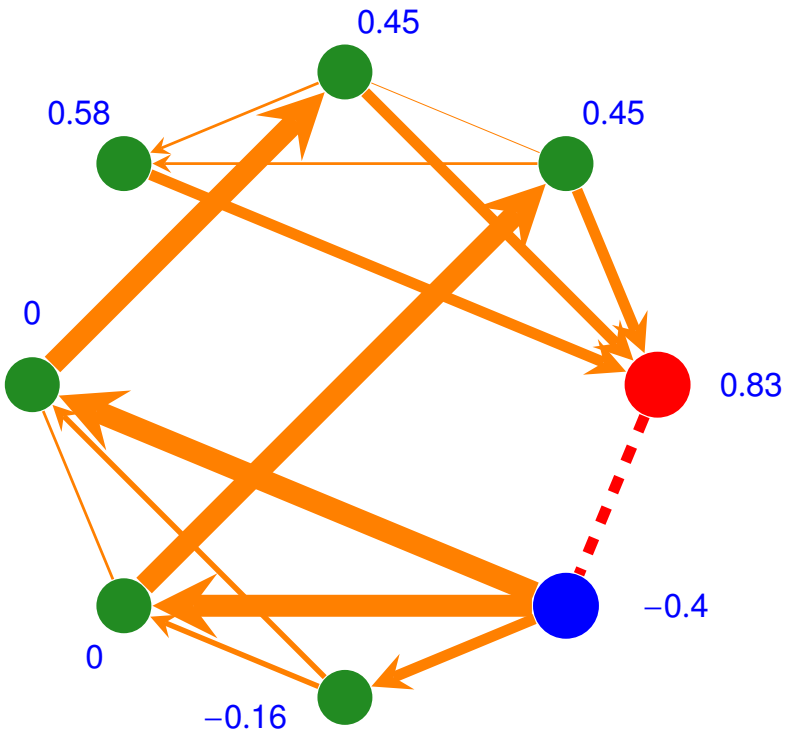
0	0	0.01	0	200	1	0	2	0	**0**	0

Given:

- a sketch $S \cdot \mathbf{B}$ of G
- crude sparsifier $\widetilde{G}$
- pair $(u, v) \in V \times V$

Need:

- is (u, v) an edge in G of resistance $\Omega(1/\log n)$?



Compute $\phi = \widetilde{\mathbf{L}}^{+}\mathbf{b}_{uv}$ – vertex potentials

Compute $\Delta = \mathbf{B}\phi \approx$ currents on edges

Check if

$$R_{uv} = \frac{\Delta_{uv}^2}{||\Delta||^2} = \Omega(1/\log n).$$

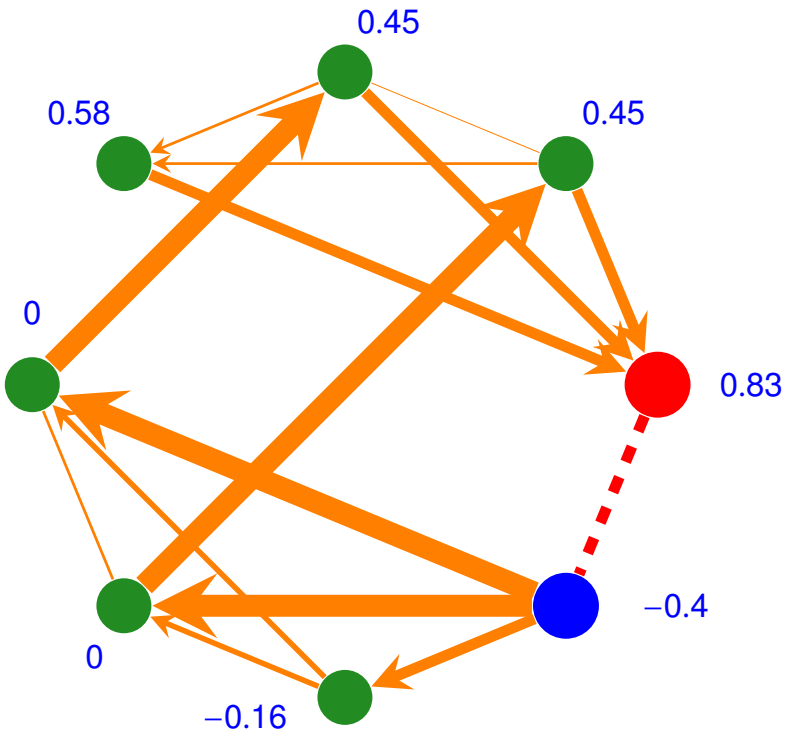
0	0	0.01	0	200	1	0	2	0	**0**	0

Given:

- a sketch $S \cdot \mathbf{B}$ of G
- crude sparsifier $\widetilde{G}$
- pair $(u, v) \in V \times V$

Need:

- is (u, v) an edge in G of resistance $\Omega(1/\log n)$?



Compute $\phi = L^{+}\mathbf{b}_{uv}$ – vertex potentials

Compute $f = \mathbf{B}\phi$=currents on edges

Check if

$$R_{uv} = \frac{f_{uv}^2}{||f||^2} = \Omega(1/\log n).$$

0	0	0.01	0	200	1	0	2	0	**0**	0

Linear sketching and sparse recovery

0	0	0.01	0	**200**	1	0	2	0	0	0

Let y be a vector of reals. Then $i \in [n]$ is an ℓ_2-heavy hitter if

$$y_i^2 \geq \eta ||y||_2^2.$$

Lemma (ℓ_2-heavy hitters; CountSketch)

For any $\eta > 0$ there exists a (randomized) sketch in dimension $\frac{1}{\eta} \log^{O(1)} n$ from which one reconstruct all η-heavy hitters. The recovery works in time $\frac{1}{\eta} \cdot \log^{O(1)} n$.

Linear sketching and sparse recovery

0	0	**0.01**	0	**200**	1	0	2	0	0	0

Let y be a vector of reals. Then $i \in [n]$ is an ℓ_2-heavy hitter if

$$y_i^2 \geq \eta ||y||_2^2.$$

Lemma (ℓ_2-heavy hitters; CountSketch)

For any $\eta > 0$ there exists a (randomized) sketch in dimension $\frac{1}{\eta} \log^{O(1)} n$ from which one reconstruct all η-heavy hitters. The recovery works in time $\frac{1}{\eta} \cdot \log^{O(1)} n$.

Sketching the edge incidence matrix

$$S \cdot \mathbf{B} = S \cdot \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & & & & & \vdots \end{bmatrix}$$

Apply ℓ_2-heavy hitters sketch S to every column $\mathbf{b}_u, u \in V$ of $\mathbf{B}$

Store the n sketches, $n \cdot \log^{O(1)} n$ space.

Sketching the edge incidence matrix

$$S \cdot \mathbf{B} = S \cdot \begin{bmatrix} \mathbf{1} & -1 & 0 & 0 & 0 & 0 \\ \mathbf{0} & 1 & -1 & 0 & 0 & 0 \\ \mathbf{0} & 0 & 0 & 0 & 0 & 0 \\ \mathbf{-1} & 0 & 1 & 0 & 0 & 0 \\ \mathbf{0} & 0 & 1 & 0 & 0 & -1 \\ \mathbf{0} & 0 & 0 & 0 & 0 & 0 \\ \vdots & & & & & \vdots \end{bmatrix}$$

Apply ℓ_2-heavy hitters sketch S to every column $\mathbf{b}_u, u \in V$ of $\mathbf{B}$

Store the n sketches, $n \cdot \log^{O(1)} n$ space.

Sketching the edge incidence matrix

$$S \cdot \mathbf{B} = S \cdot \begin{bmatrix} 1 & \mathbf{-1} & 0 & 0 & 0 & 0 \\ 0 & \mathbf{1} & -1 & 0 & 0 & 0 \\ 0 & \mathbf{0} & 0 & 0 & 0 & 0 \\ -1 & \mathbf{0} & 1 & 0 & 0 & 0 \\ 0 & \mathbf{0} & 1 & 0 & 0 & -1 \\ 0 & \mathbf{0} & 0 & 0 & 0 & 0 \\ \vdots & & & & & \vdots \end{bmatrix}$$

Apply ℓ_2-heavy hitters sketch S to every column $\mathbf{b}_u, u \in V$ of $\mathbf{B}$

Store the n sketches, $n \cdot \log^{O(1)} n$ space.

Sketching the edge incidence matrix

$$S \cdot \mathbf{B} = S \cdot \begin{bmatrix} 1 & -1 & \mathbf{0} & 0 & 0 & 0 \\ 0 & 1 & \mathbf{-1} & 0 & 0 & 0 \\ 0 & 0 & \mathbf{0} & 0 & 0 & 0 \\ -1 & 0 & \mathbf{1} & 0 & 0 & 0 \\ 0 & 0 & \mathbf{1} & 0 & 0 & -1 \\ 0 & 0 & \mathbf{0} & 0 & 0 & 0 \\ \vdots & & & & & \vdots \end{bmatrix}$$

Apply ℓ_2-heavy hitters sketch S to every column $\mathbf{b}_u, u \in V$ of $\mathbf{B}$

Store the n sketches, $n \cdot \log^{O(1)} n$ space.

Sketching the edge incidence matrix

$$S \cdot \mathbf{B} = S \cdot \begin{bmatrix} 1 & -1 & 0 & \mathbf{0} & 0 & 0 \\ 0 & 1 & -1 & \mathbf{0} & 0 & 0 \\ 0 & 0 & 0 & \mathbf{0} & 0 & 0 \\ -1 & 0 & 1 & \mathbf{0} & 0 & 0 \\ 0 & 0 & 1 & \mathbf{0} & 0 & -1 \\ 0 & 0 & 0 & \mathbf{0} & 0 & 0 \\ \vdots & & & & & \vdots \end{bmatrix}$$

Apply ℓ_2-heavy hitters sketch S to every column $\mathbf{b}_u, u \in V$ of $\mathbf{B}$

Store the n sketches, $n \cdot \log^{O(1)} n$ space.

Given:

- a sketch $S \cdot \mathbf{B}$ of G
- crude sparsifier $\widetilde{G}$
- pair $(u, v) \in V \times V$

Need:

- is (u, v) an edge in G of resistance $\Omega(1/\log n)$?

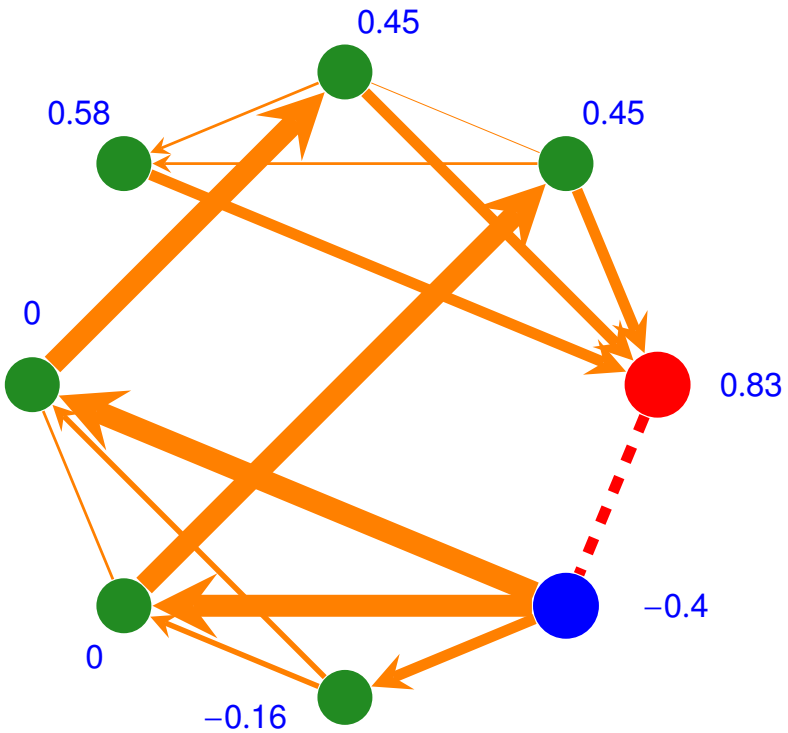
Given:

- a sketch $S \cdot \mathbf{B}$ of G
- crude sparsifier $\widetilde{G}$
- pair $(u, v) \in V \times V$

Need:

- is (u, v) an edge in G of resistance $\Omega(1/\log n)$?

Compute $\phi = \widetilde{\mathbf{L}}^{+}\mathbf{b}_{uv}$ – vertex potentials

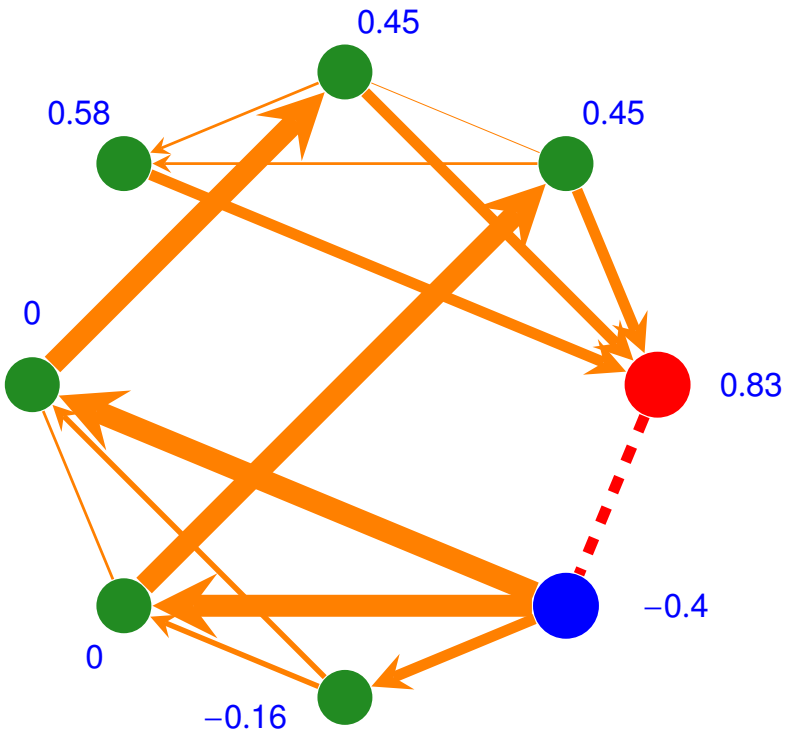


Given:

- a sketch $S \cdot \mathbf{B}$ of G
- crude sparsifier $\widetilde{G}$
- pair $(u, v) \in V \times V$

Need:

- is (u, v) an edge in G of resistance $\Omega(1/\log n)$?



Compute $\phi = \widetilde{\mathbf{L}}^{+}\mathbf{b}_{uv}$ – vertex potentials

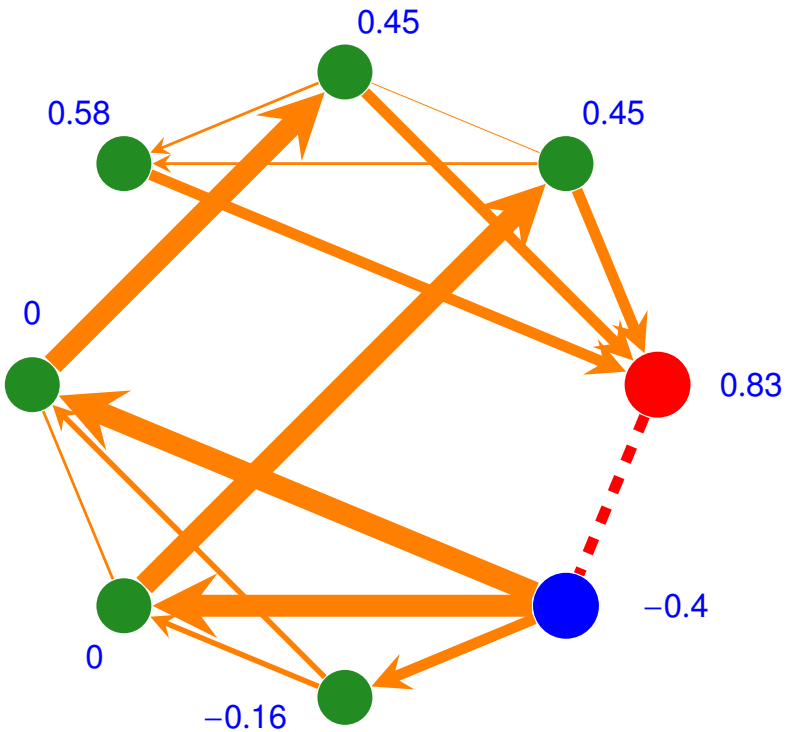
Compute $S \cdot \Delta = \mathbf{S} \cdot \mathbf{B}\phi$=potential differences

Given:

- a sketch $S \cdot \mathbf{B}$ of G
- crude sparsifier $\widetilde{G}$
- pair $(u, v) \in V \times V$

Need:

- is (u, v) an edge in G of resistance $\Omega(1/\log n)$?



Compute $\phi = \widetilde{\mathbf{L}}^{+}\mathbf{b}_{uv}$ – vertex potentials

Compute $S \cdot \Delta = \mathbf{S} \cdot \mathbf{B}\phi$=potential differences

Check if

$$\widetilde{R}_e = \frac{\Delta_e^2}{||\Delta||^2} = \Omega(1/\log n)$$

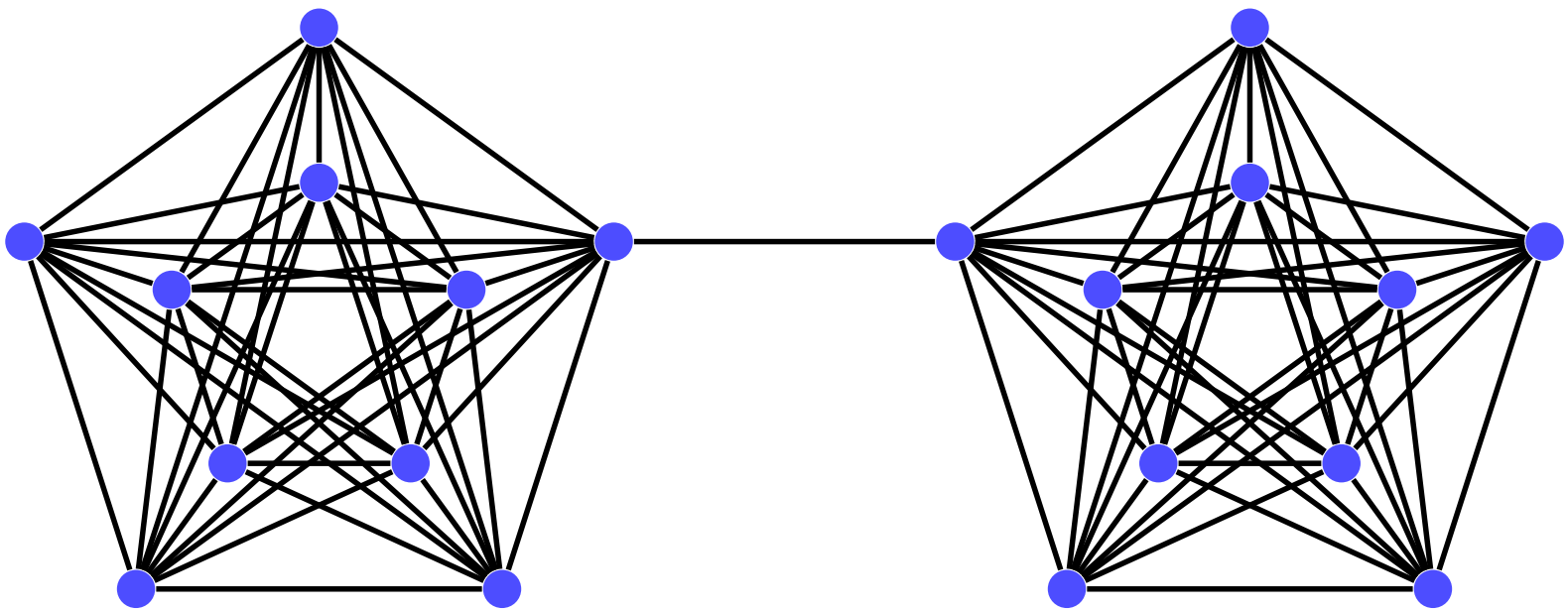
using heavy-hitters sketch S

0	0	0.01	0	200	1	0	2	0	**0**	0

Testing a pair of vertices – summary

Connect a battery to a pair of vertices $u, v \in V$

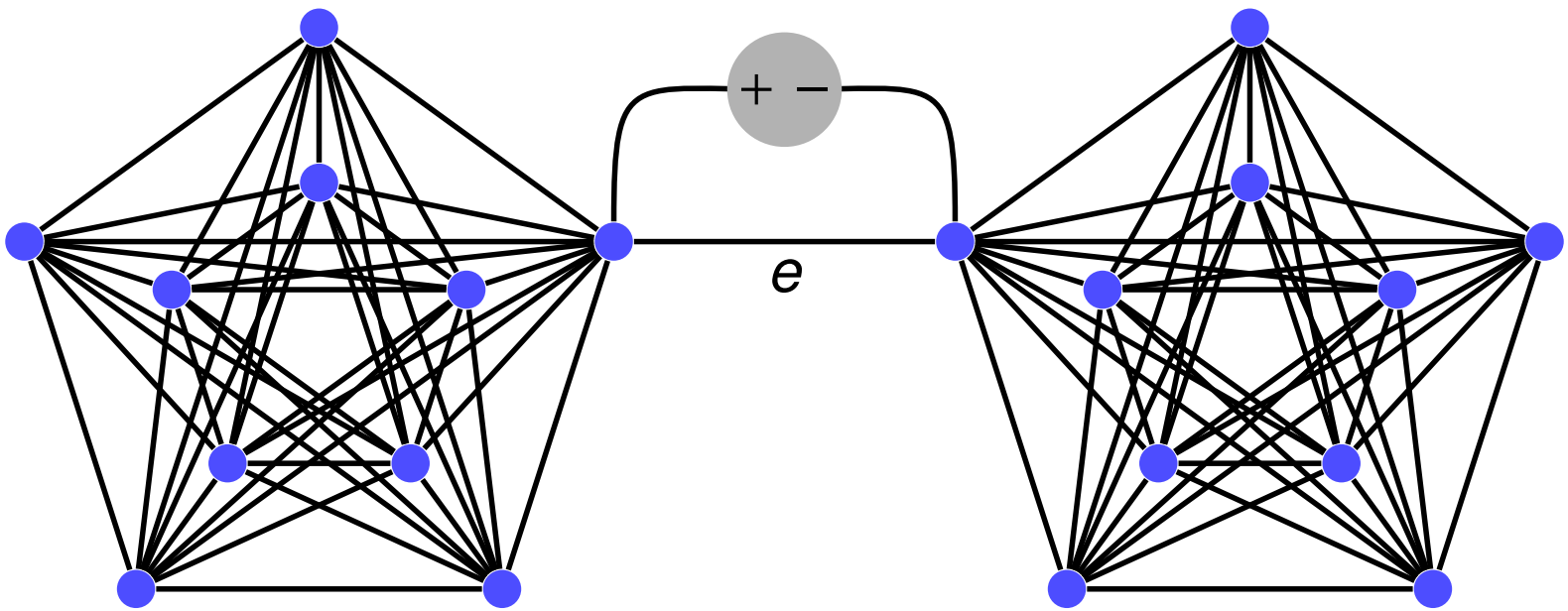
Edge e is high effective resistance iff the heat created by this edge in this setting is a non-trivial fraction of the total heat.



Testing a pair of vertices – summary

Connect a battery to a pair of vertices $u, v \in V$

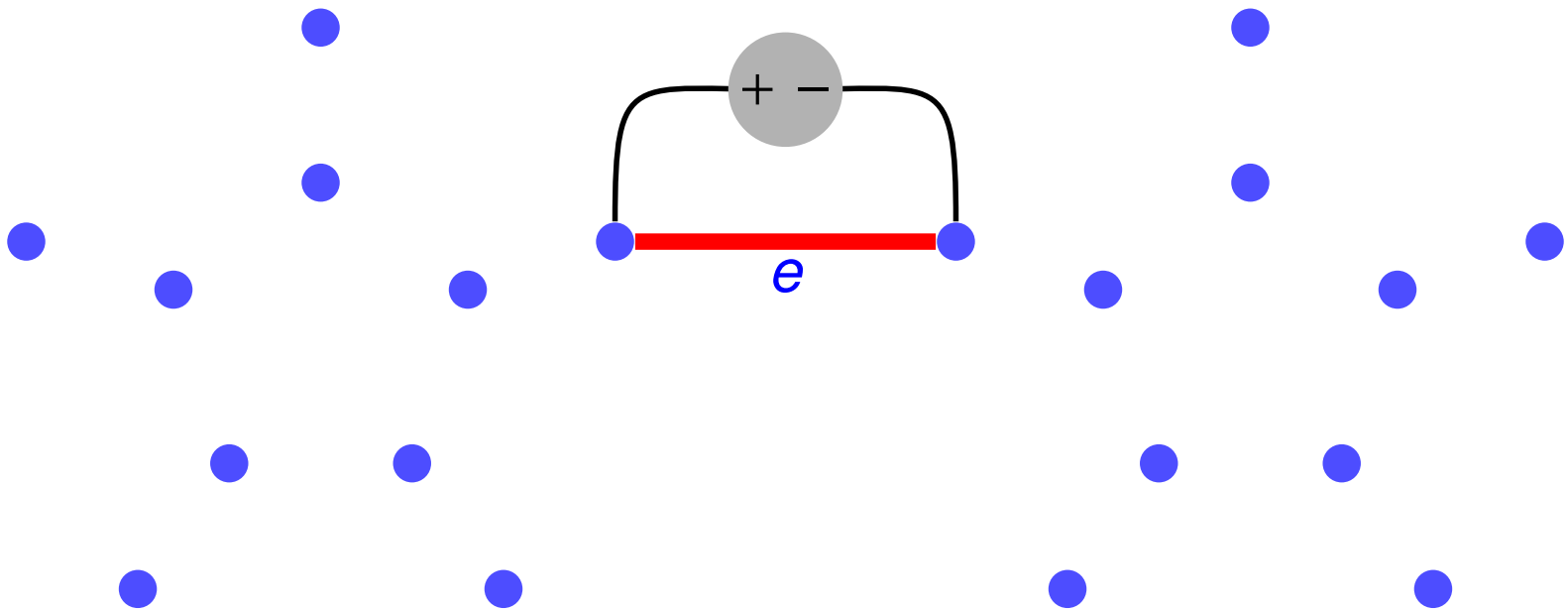
Edge e is high effective resistance iff the heat created by this edge in this setting is a non-trivial fraction of the total heat.



Testing a pair of vertices – summary

Connect a battery to a pair of vertices $u, v \in V$

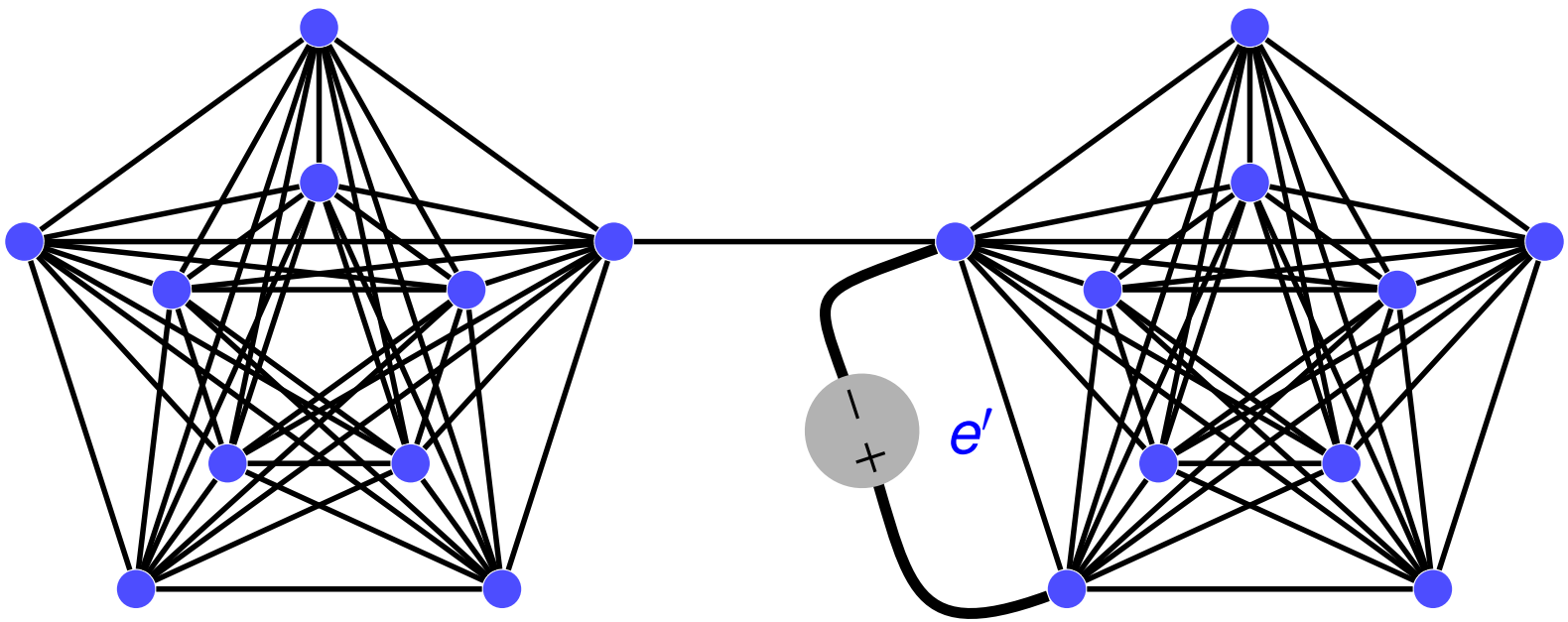
Edge e is high effective resistance iff the heat created by this edge in this setting is a non-trivial fraction of the total heat.



Testing a pair of vertices – summary

Connect a battery to a pair of vertices $u, v \in V$

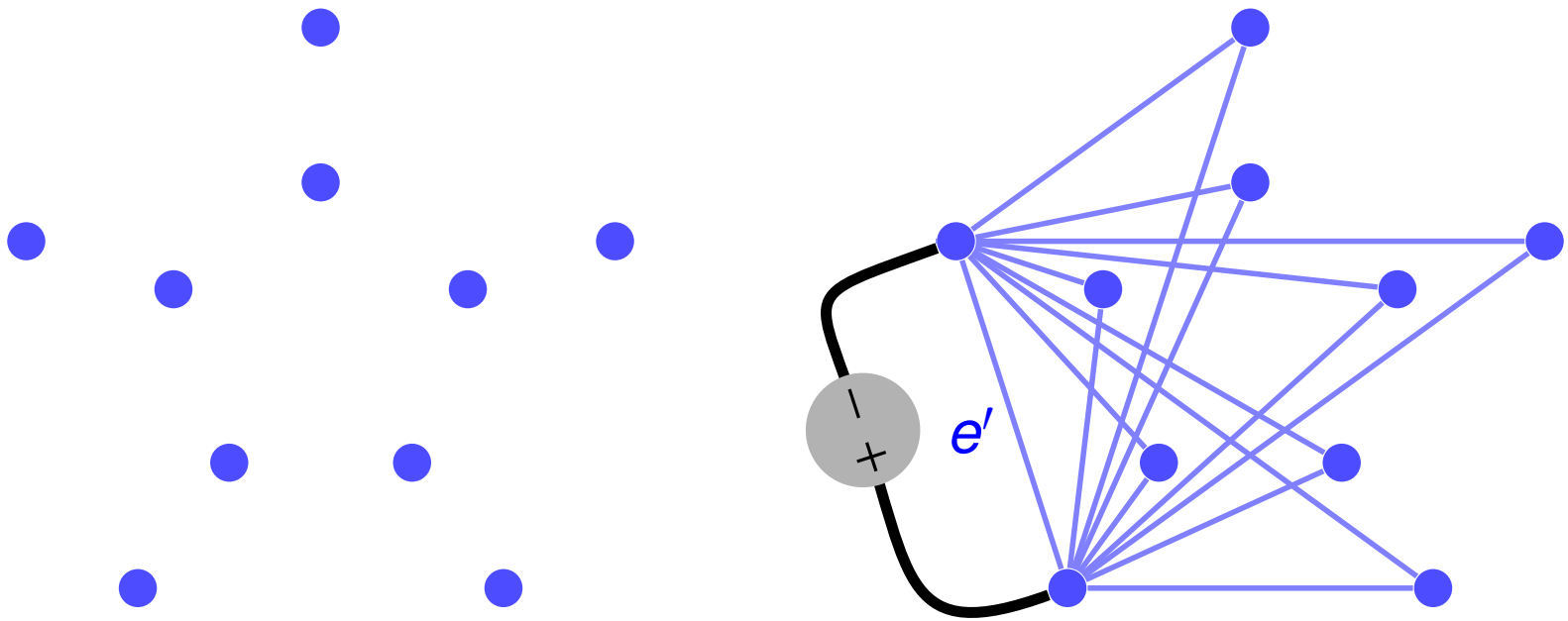
Edge e is high effective resistance iff the heat created by this edge in this setting is a non-trivial fraction of the total heat.



Testing a pair of vertices – summary

Connect a battery to a pair of vertices $u, v \in V$

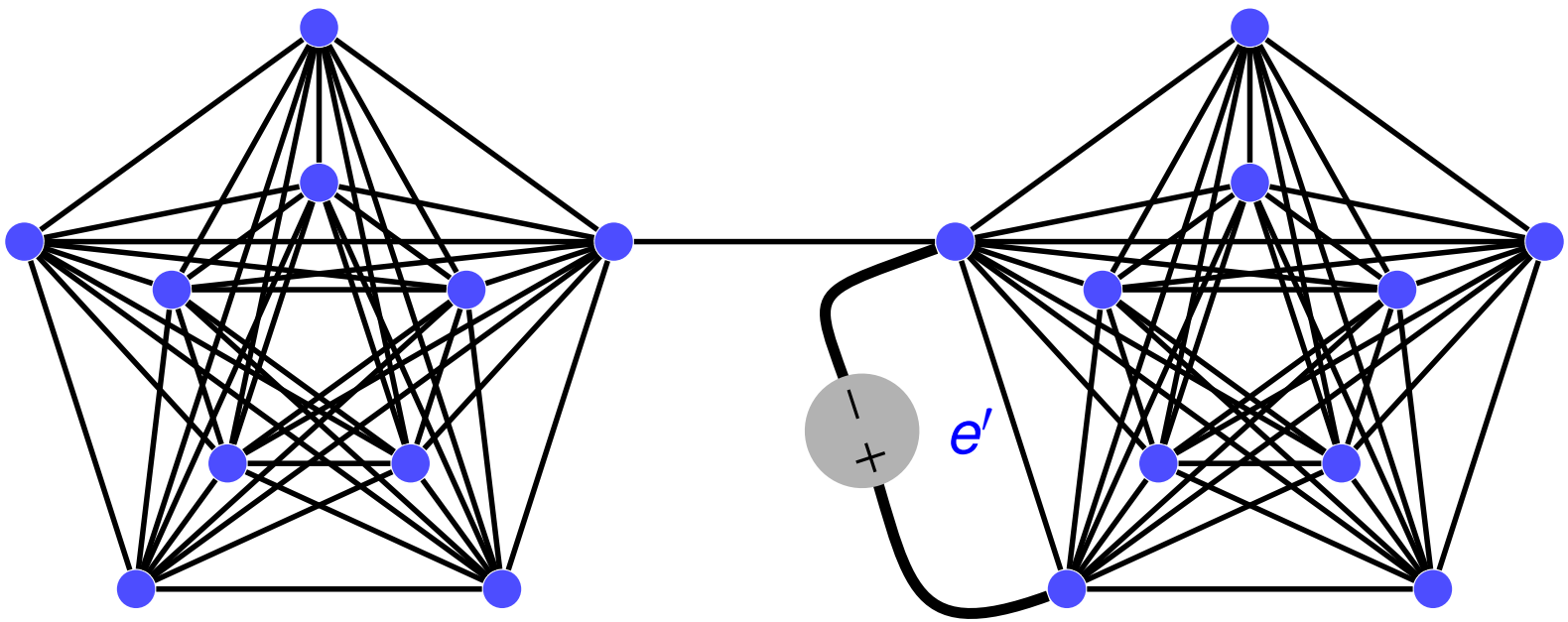
Edge e is high effective resistance iff the heat created by this edge in this setting is a non-trivial fraction of the total heat.



Testing a pair of vertices – summary

Connect a battery to a pair of vertices $u, v \in V$

Edge e is high effective resistance iff the heat created by this edge in this setting is a non-trivial fraction of the total heat.



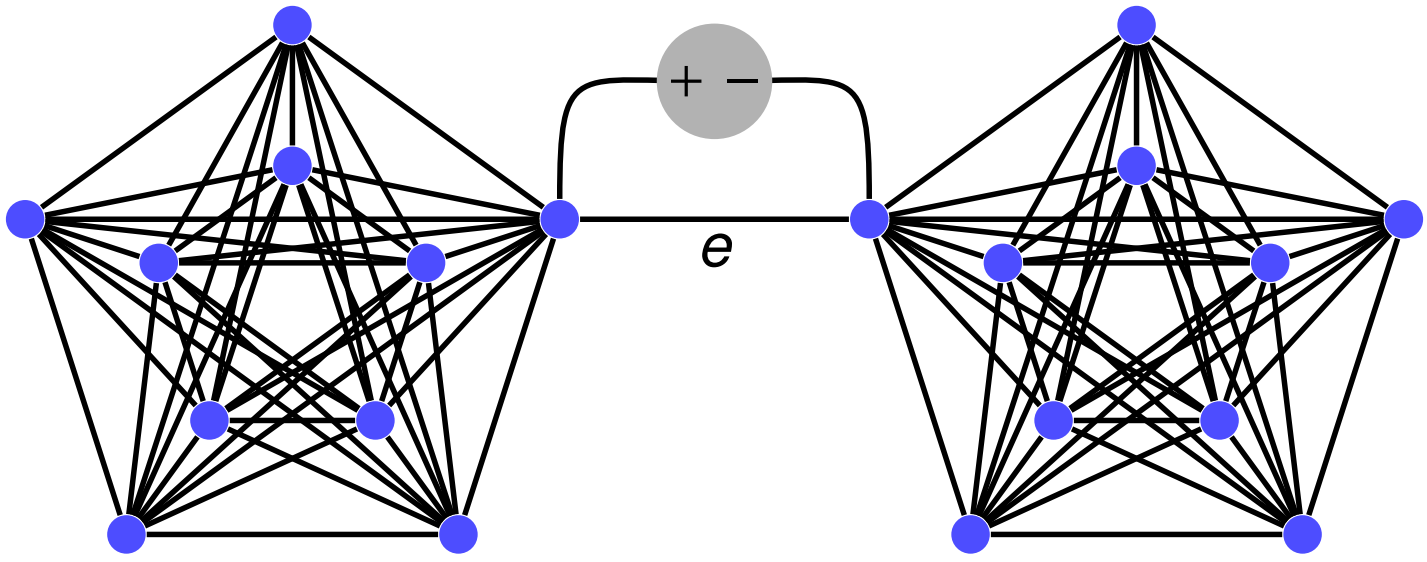
Find all heavy edges with $O(n)$ experiments?

Find shortcuts in wiring

Find shortcuts in wiring

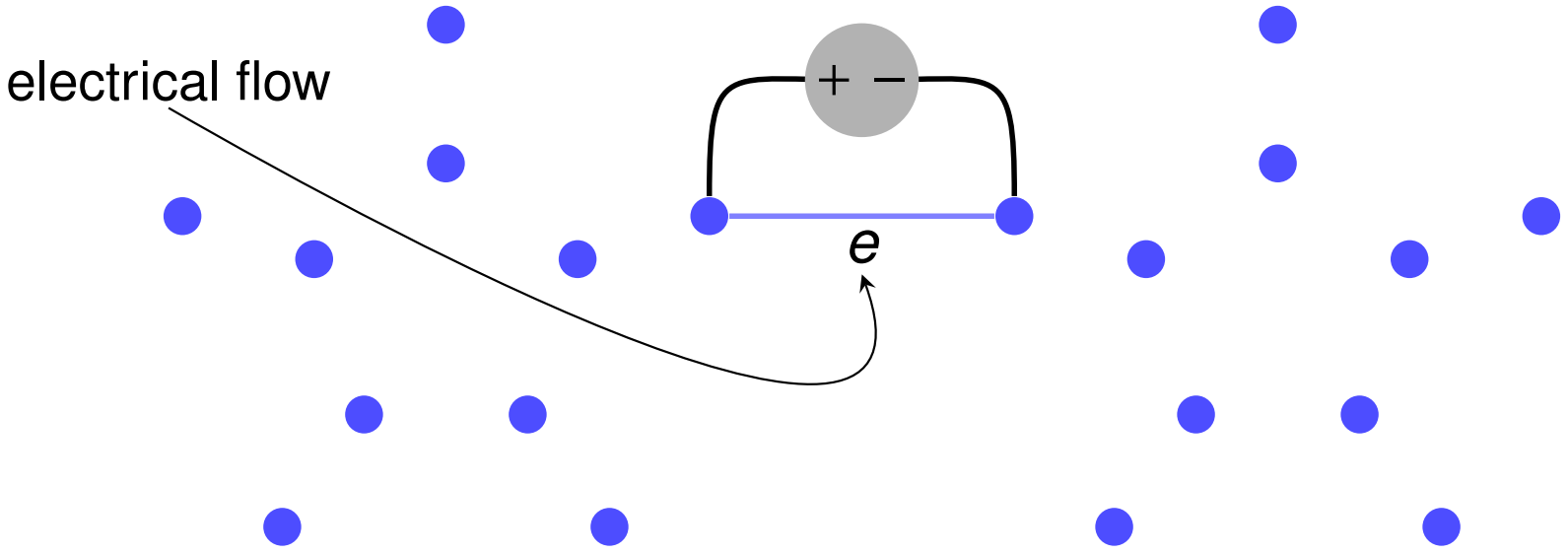
Find wires with high effective resistance

Edges with high effective resistance



Edge *e* has high effective resistance iff the heat created by this edge is a non-trivial fraction of the total heat.

Edges with high effective resistance

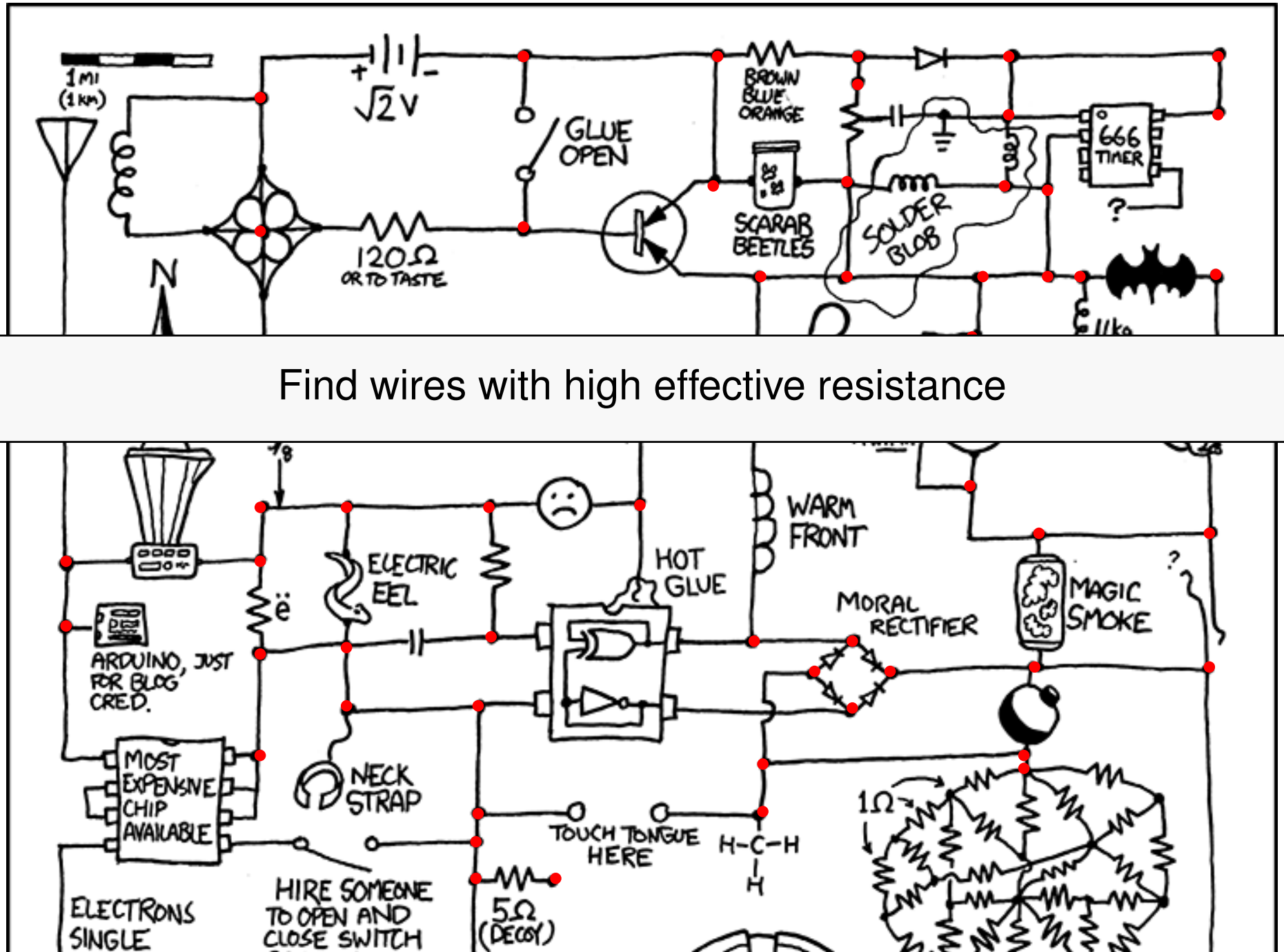


Edge *e* has high effective resistance iff the heat created by this edge is a non-trivial fraction of the total heat.

Find shortcuts in wiring

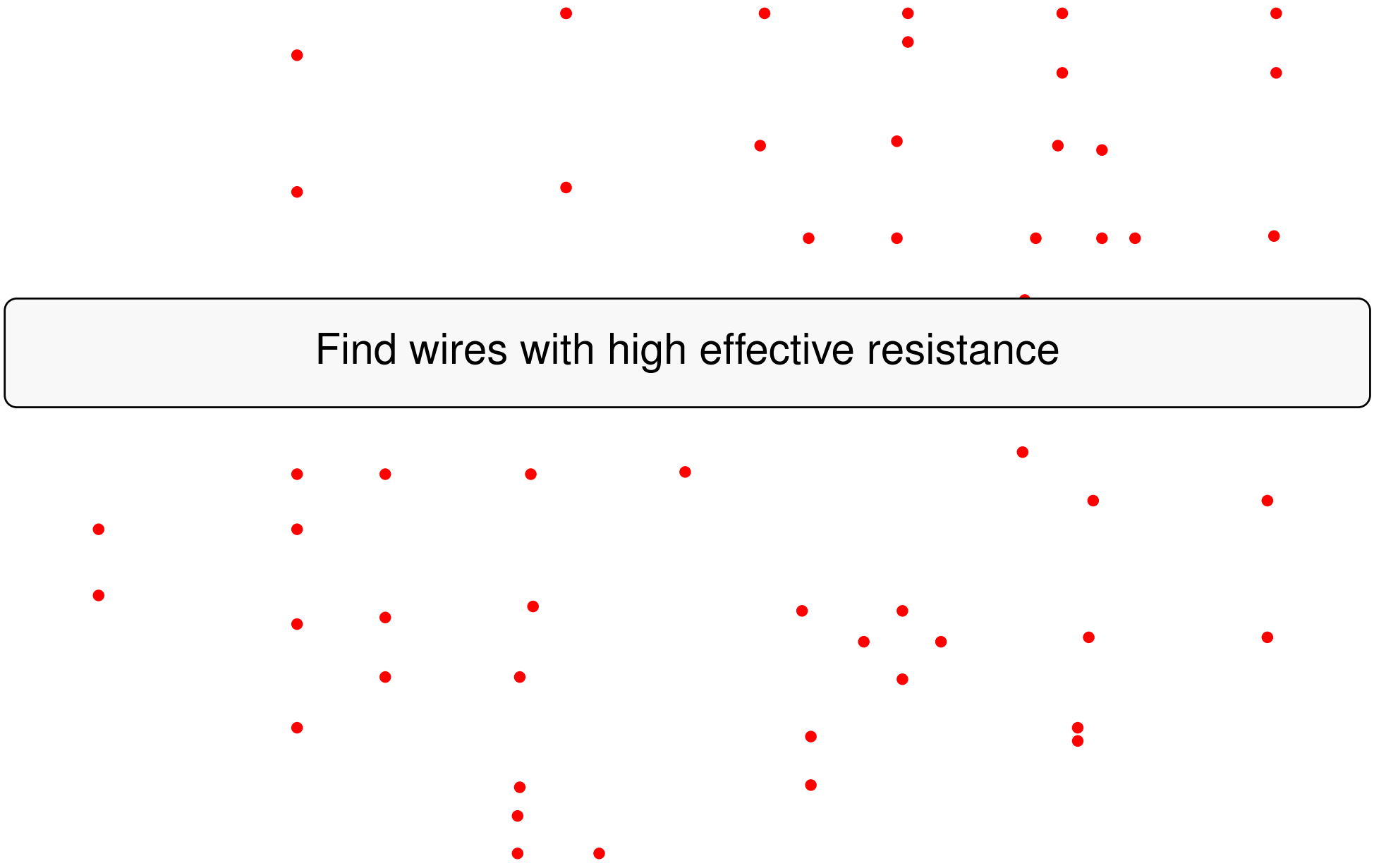
Find wires with high effective resistance

Find shortcuts in wiring



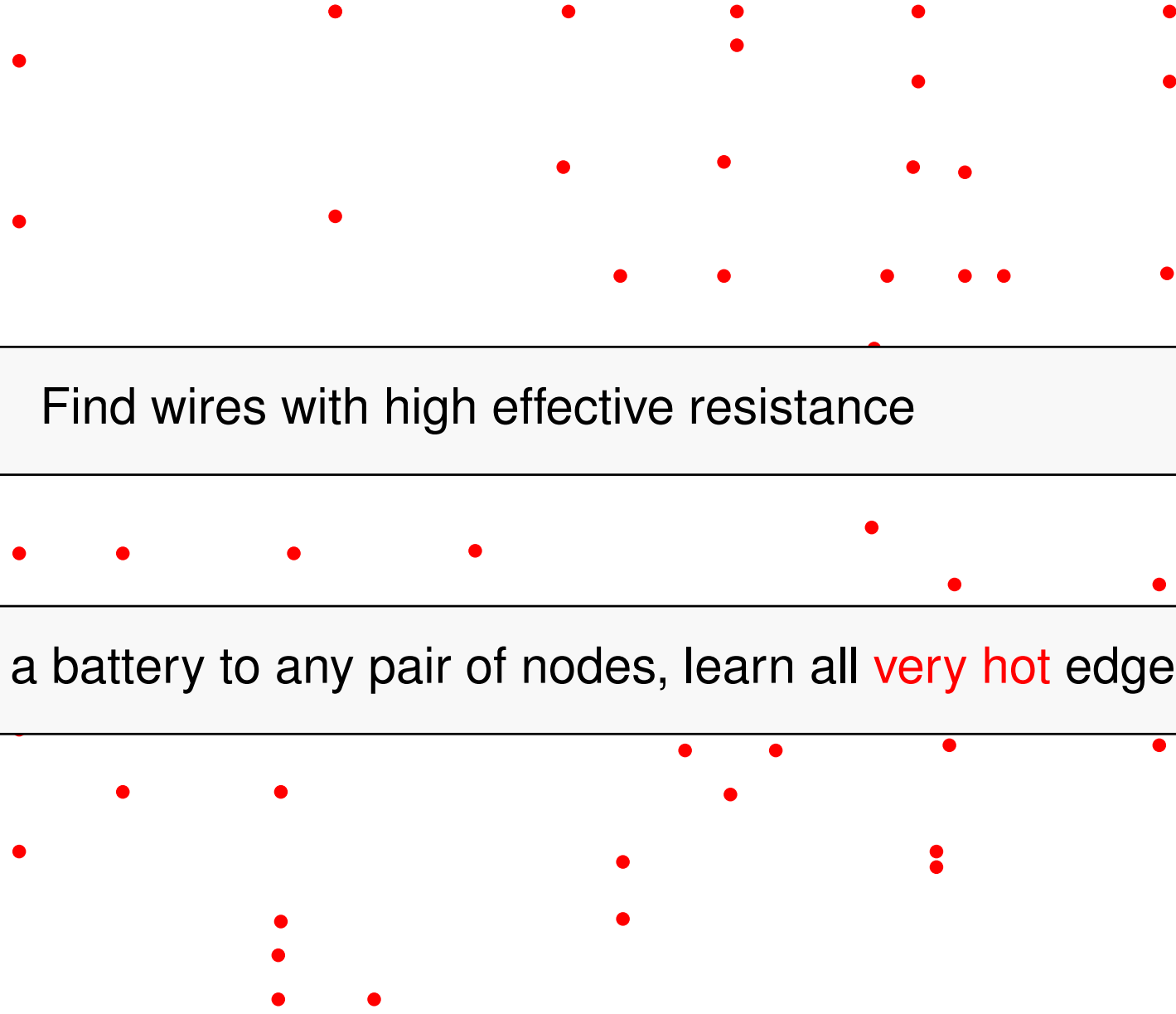
Find wires with high effective resistance

Find shortcuts in wiring



Find wires with high effective resistance

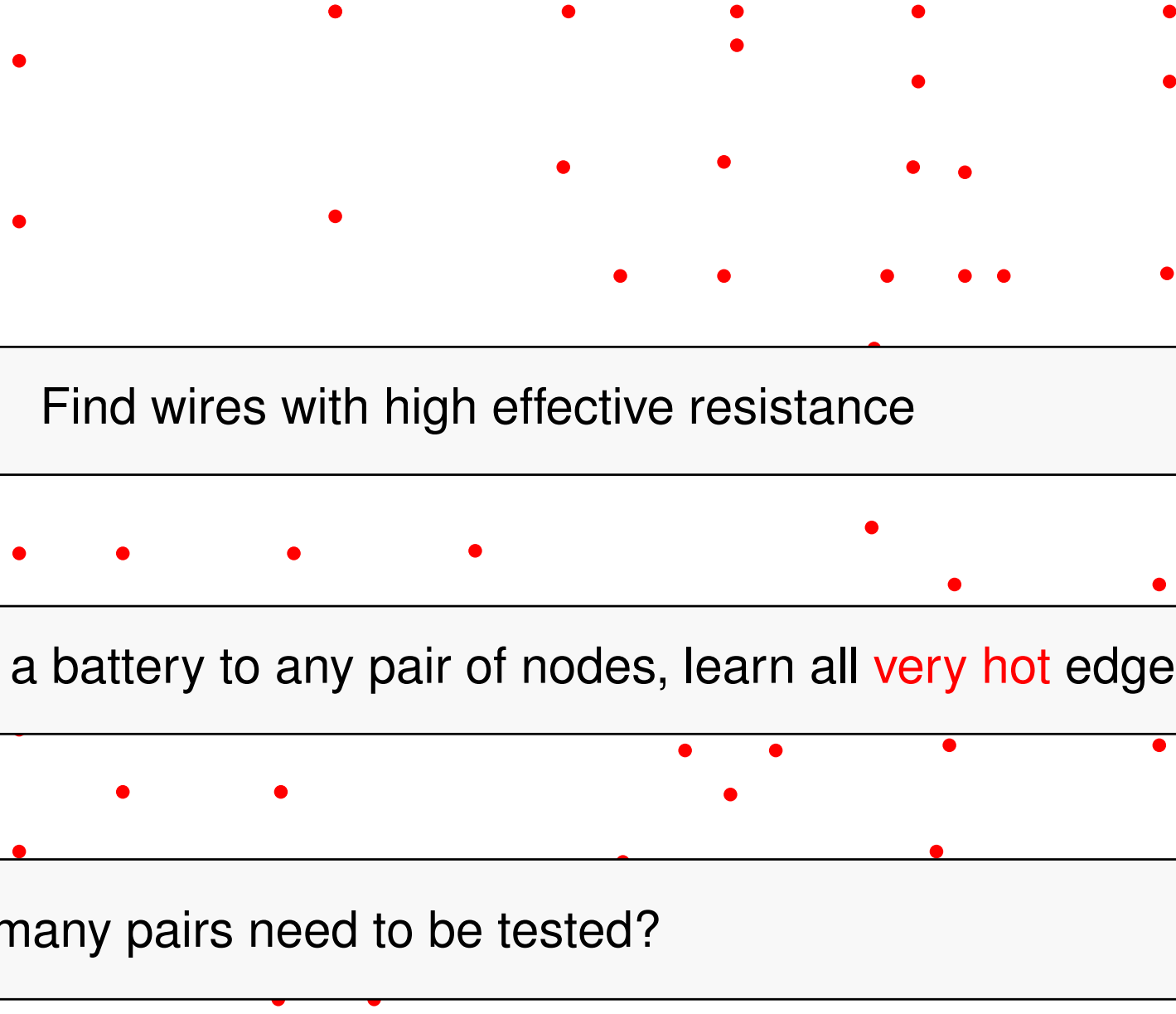
Find shortcuts in wiring



Find wires with high effective resistance

Can attach a battery to any pair of nodes, learn all very hot edges

Find shortcuts in wiring

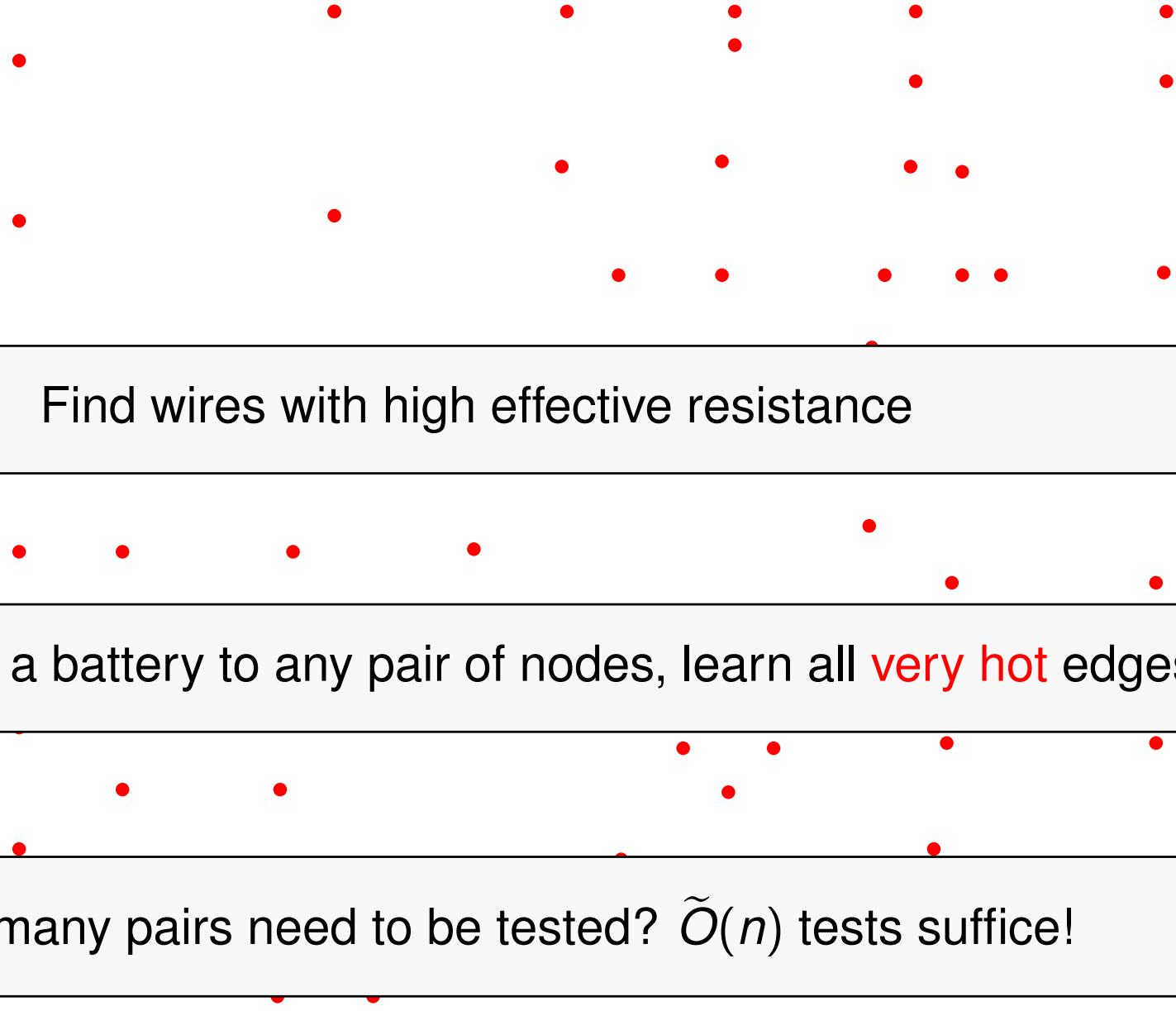


Find wires with high effective resistance

Can attach a battery to any pair of nodes, learn all very hot edges

How many pairs need to be tested?

Find shortcuts in wiring



Find wires with high effective resistance

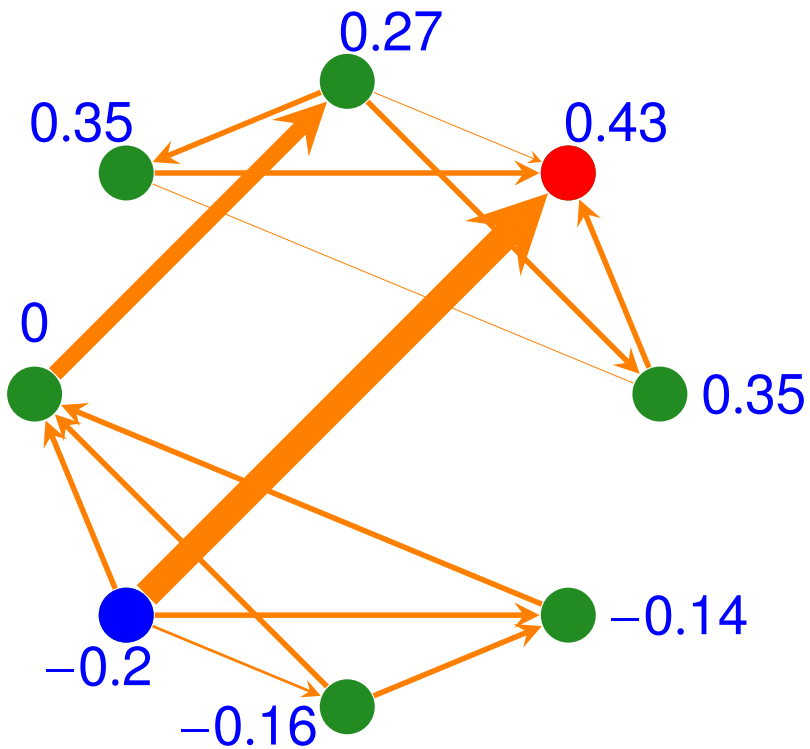
Can attach a battery to any pair of nodes, learn all very hot edges

How many pairs need to be tested? $\tilde{O}(n)$ tests suffice!

1. Spectral sparsification
2. Dynamic streaming and linear sketching
3. Spectral sparsification using small sketches
4. Fast decoding

1. Spectral sparsification
2. Dynamic streaming and linear sketching
3. Spectral sparsification using small sketches
4. **Fast decoding**

Effective resistance



Inject current at u, take out at v.

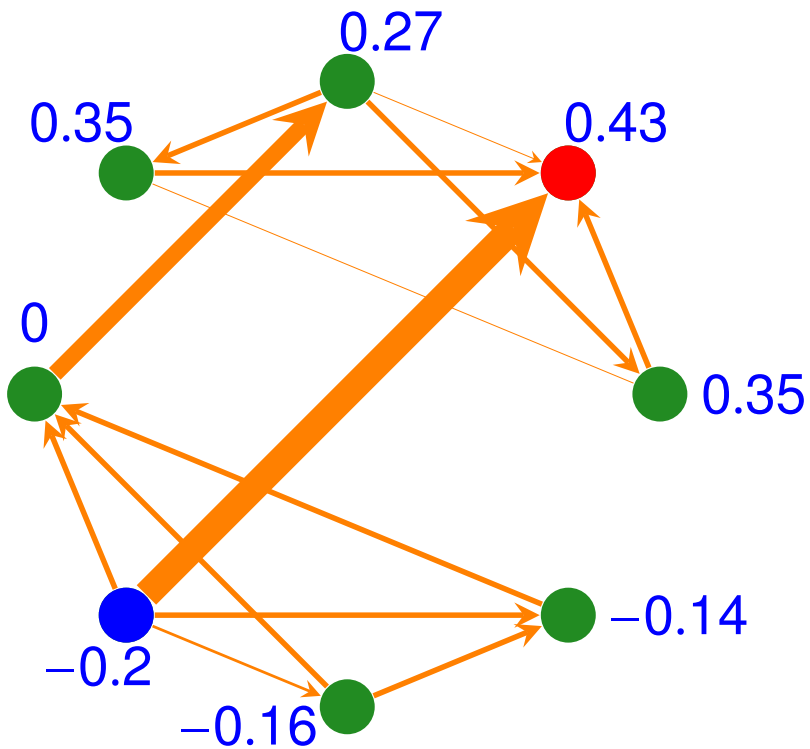
$\phi = L^+ \mathbf{b}_{uv}$=vertex potentials

$f = \mathbf{B}\phi$=currents on edges

We have

$$R_{uv} = \frac{f_{uv}^2}{||f||^2} = \frac{(\phi_u - \phi_v)^2}{R_{uv}}.$$

Effective resistance



Inject current at u, take out at v.

$\phi = L^+ \mathbf{b}_{uv}$=vertex potentials

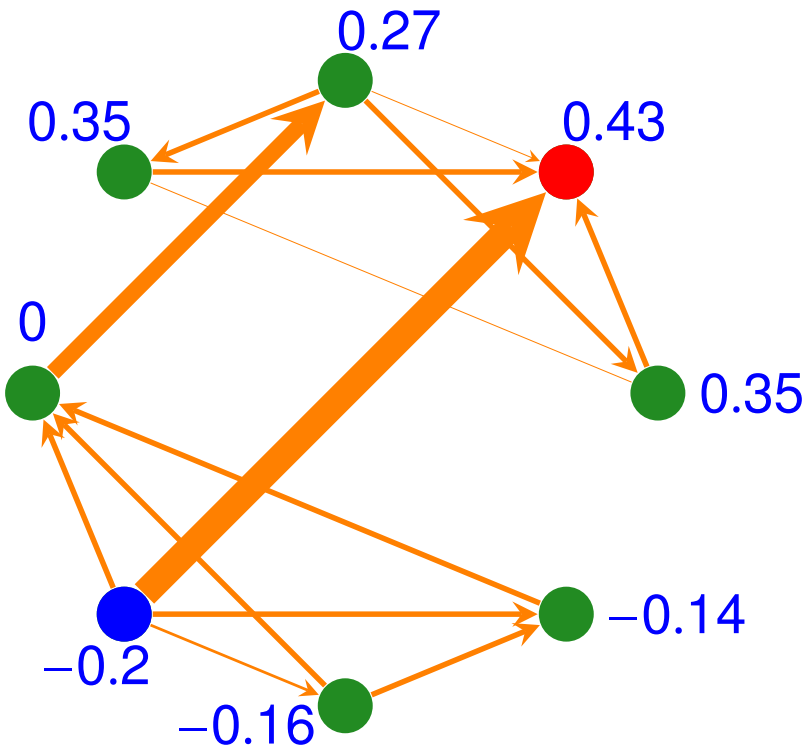
$f = \mathbf{B}\phi$=currents on edges

We have

$$R_{uv} = \frac{f_{uv}^2}{||f||^2} = \frac{(\phi_u - \phi_v)^2}{R_{uv}}.$$

Find pair $a, b \in V$ such that (u, v) carries a lot of $a \to b$ flow?

Effective resistance



Inject current at u, take out at v.

$\phi = L^+ \mathbf{b}_{uv}$=vertex potentials

$f = \mathbf{B}\phi$=currents on edges

We have

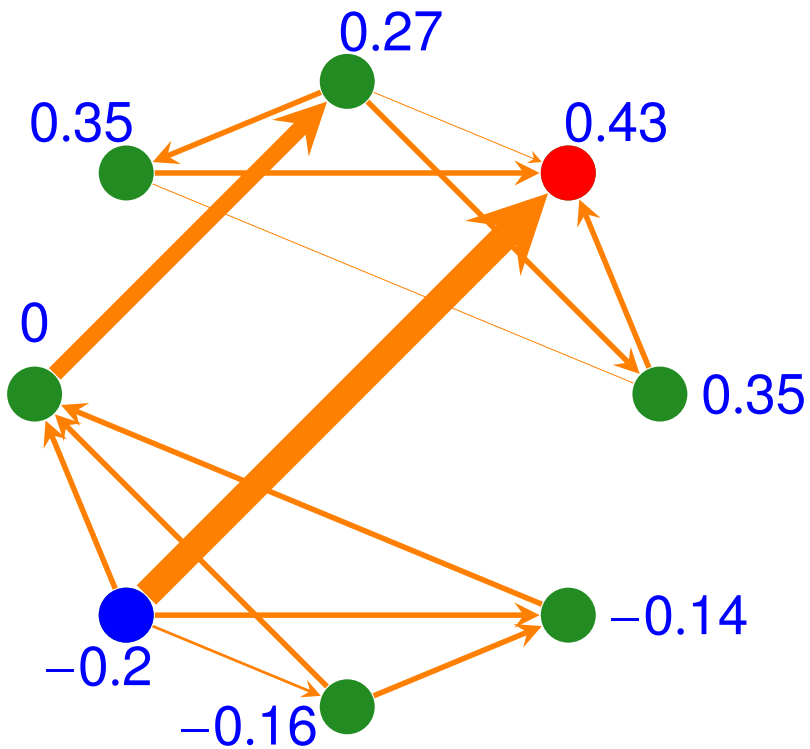
$$R_{uv} = \frac{f_{uv}^2}{||f||^2} = \frac{(\phi_u - \phi_v)^2}{R_{uv}}.$$

Find pair $a, b \in V$ such that (u, v) carries a lot of $a \to b$ flow?

$$\frac{(\tau_u - \tau_v)^2}{R_{ab}} \gtrsim 1/\log n$$

where $\tau = L^+ \mathbf{b}_{ab}$.

Effective resistance



Inject current at u, take out at v.

$\phi = L^+ \mathbf{b}_{uv}$=vertex potentials

$f = \mathbf{B}\phi$=currents on edges

We have

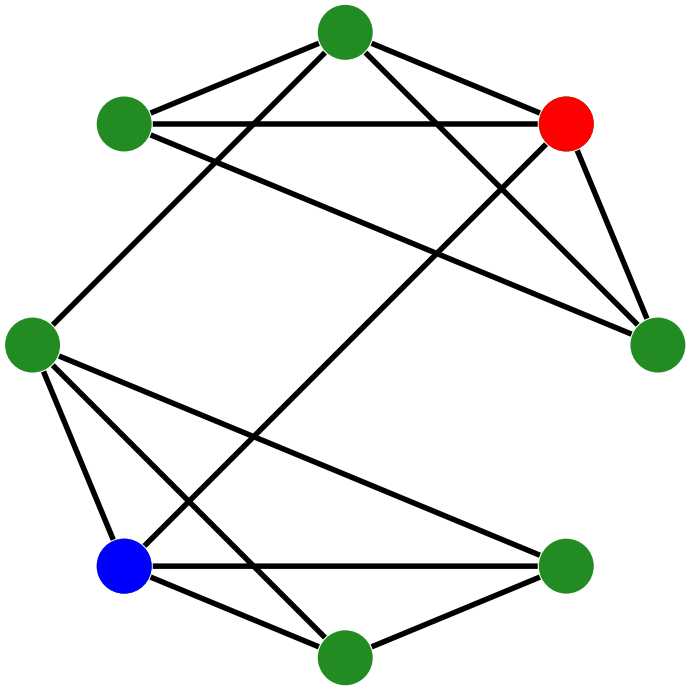
$$R_{uv} = \frac{f_{uv}^2}{||f||^2} = \frac{(\phi_u - \phi_v)^2}{R_{uv}}.$$

Find pair $a, b \in V$ such that (u, v) carries a lot of $a \to b$ flow?

$$\frac{(\tau_u - \tau_v)^2}{R_{ab}} \gtrsim \frac{(\phi_u - \phi_v)^2}{R_{uv}}$$

where $\tau = L^+ \mathbf{b}_{ab}$.

Effective resistance



Inject current at u, take out at v.

$\phi = L^+ \mathbf{b}_{uv}$=vertex potentials

$f = \mathbf{B}\phi$=currents on edges

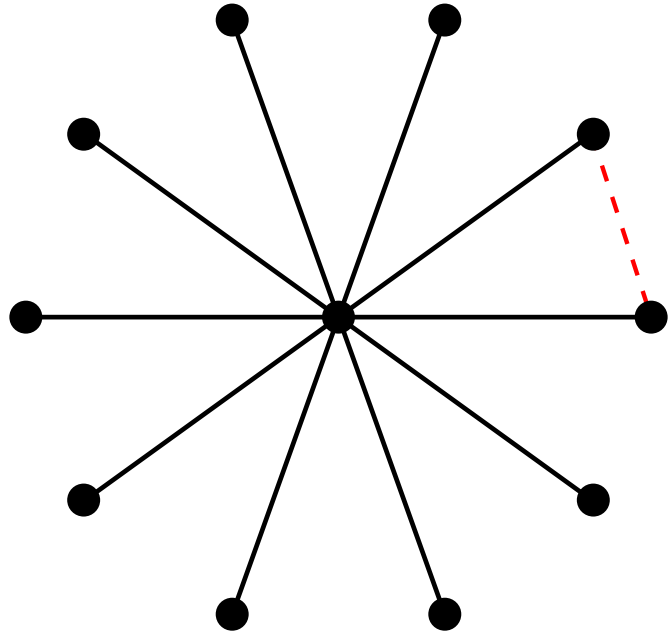
We have

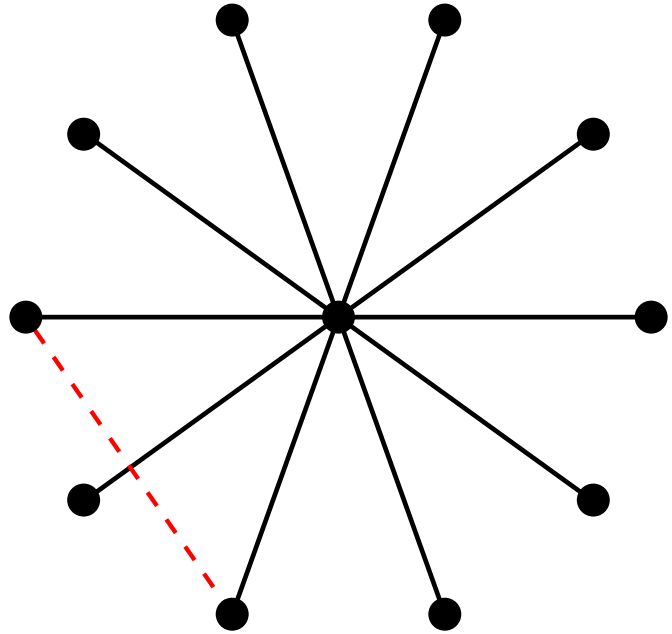
$$R_{uv} = \frac{f_{uv}^2}{||f||^2} = \frac{(\phi_u - \phi_v)^2}{R_{uv}}.$$

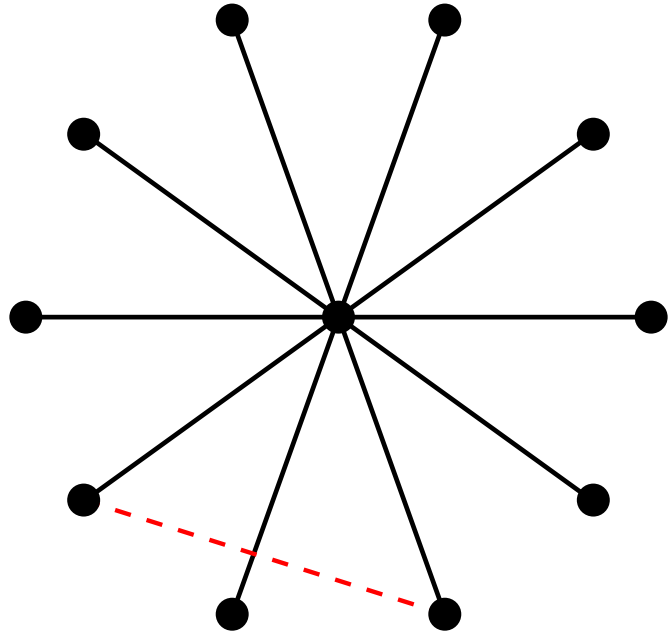
Find pair $a, b \in V$ such that (u, v) carries a lot of $a \to b$ flow?

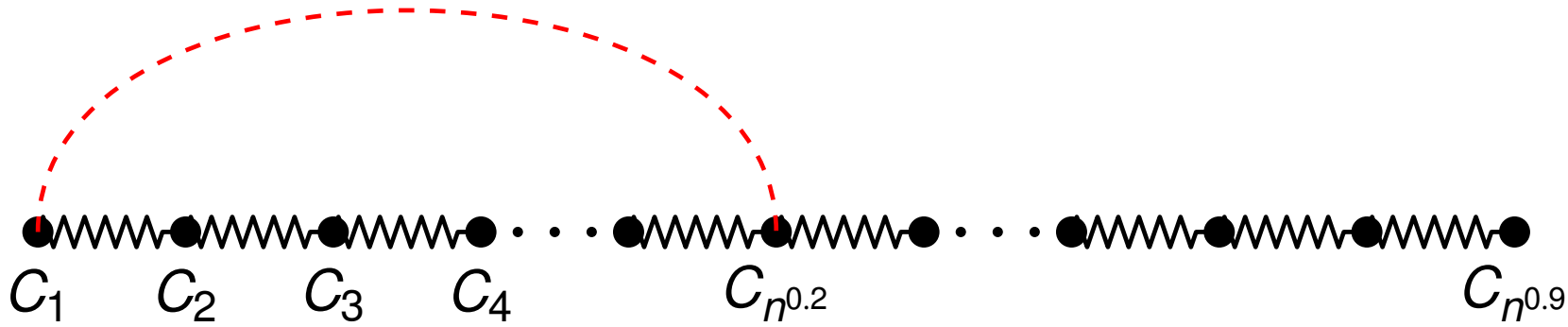
$$\frac{(\tau_u - \tau_v)^2}{R_{ab}} \gtrsim \frac{(\phi_u - \phi_v)^2}{R_{uv}}$$

where $\tau = L^+ \mathbf{b}_{ab}$.

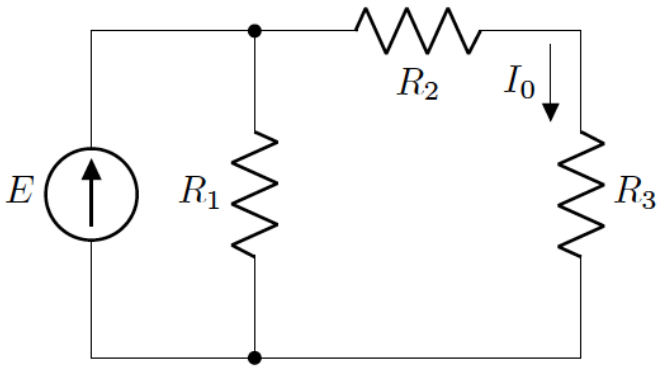




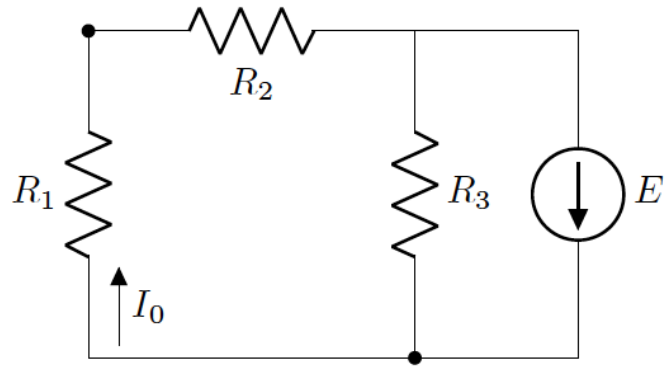



C_1
C_2
C_3
C_4
$C_{n^{0.2}}$
$C_{n^{0.9}}$

Reciprocity theorem



(a)



(b)

$u \to v$ flow results in potentials $\phi = L^{+}\mathbf{b}_{uv}$

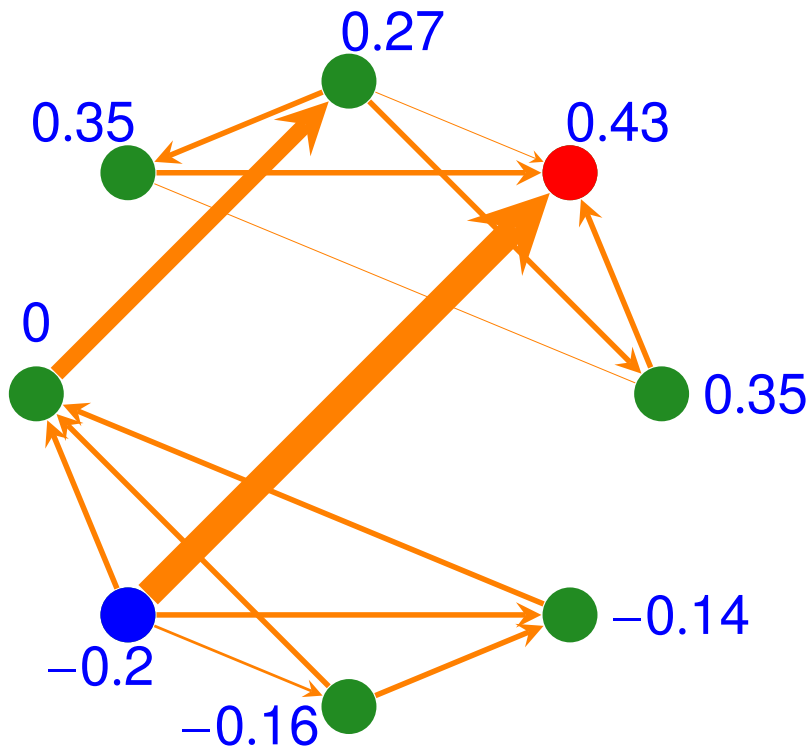
$a \to b$ flow results in potentials $\tau = L^{+}\mathbf{b}_{ab}$

$$\tau_v - \tau_u = \mathbf{b}_{uv}^T \tau = \mathbf{b}_{uv}^T L^{+} \mathbf{b}_{ab}$$

and

$$\phi_b - \phi_a = \mathbf{b}_{ab}^T \phi = \mathbf{b}_{ab}^T L^{+} \mathbf{b}_{uv}$$

Effective resistance

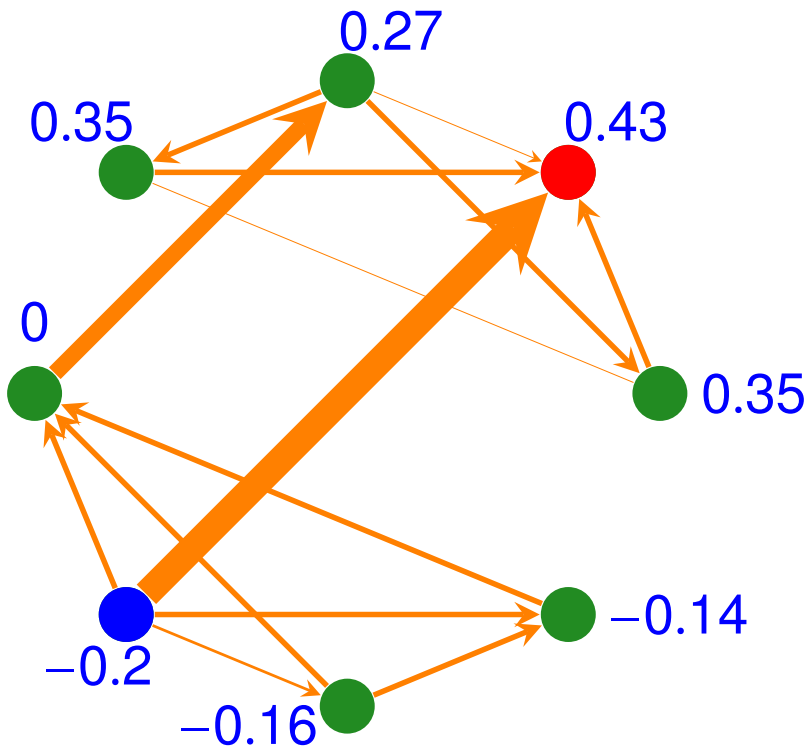


Find pair $a, b \in V$ such that (u, v) carries a lot of $a \to b$ flow?

$$\frac{(\tau_u - \tau_v)^2}{R_{ab}} \approx \frac{(\phi_u - \phi_v)^2}{R_{uv}}$$

where $\tau = L^+ \mathbf{b}_{ab}$.

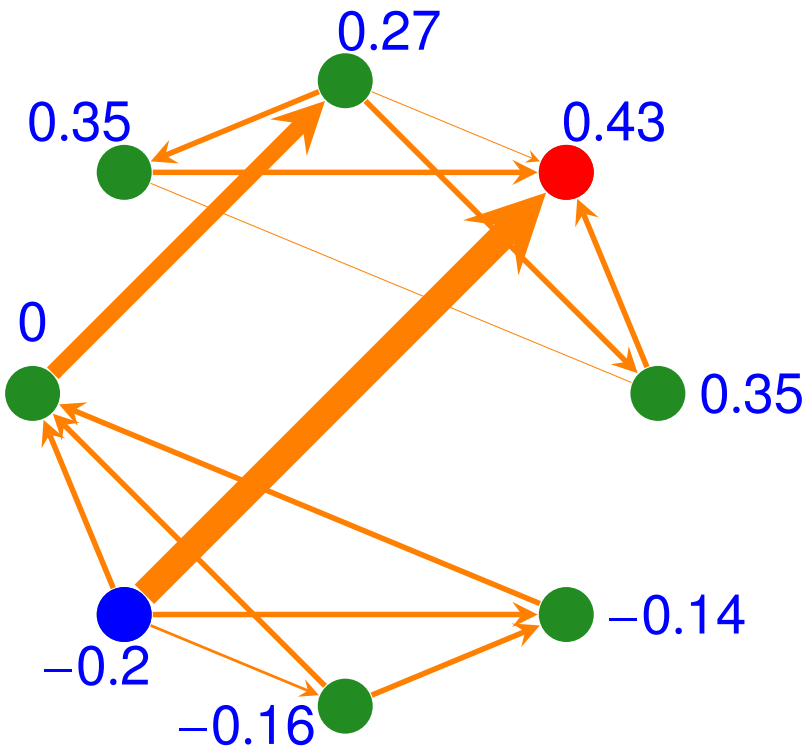
Potential embedding



$$R_{uv} = \frac{f_{uv}^2}{||f||^2} = \frac{(\phi_u - \phi_v)^2}{R_{uv}}$$

Potential embedding of (u, v):

Potential embedding



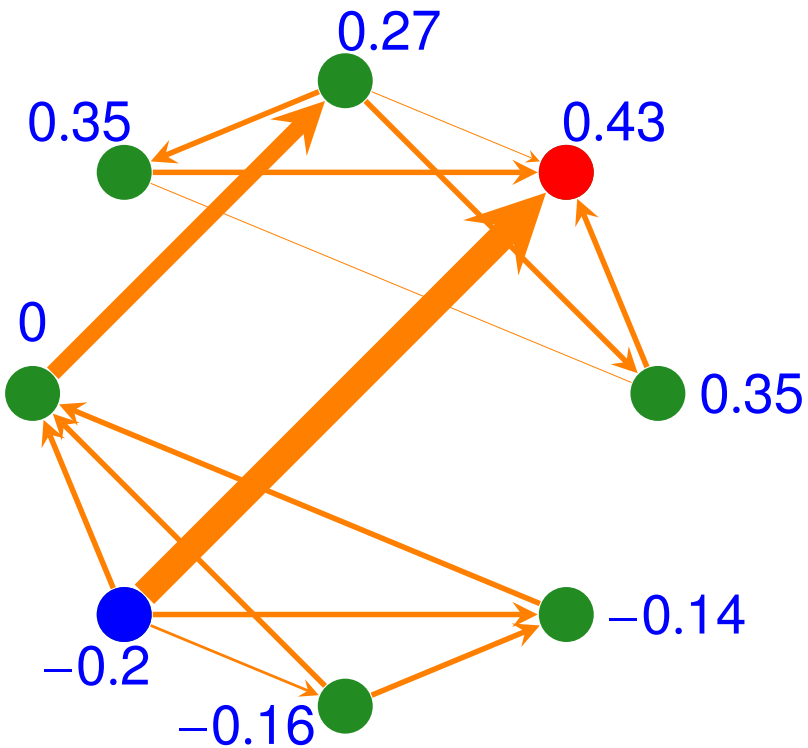
$$R_{uv} = \frac{f_{uv}^2}{||f||^2} = \frac{(\phi_u - \phi_v)^2}{R_{uv}}$$

Potential embedding of (u, v):

For $x \in V$

$$\phi_v - \phi_u = (\phi_v - \phi_x) + (\phi_x - \phi_u)$$

Potential embedding



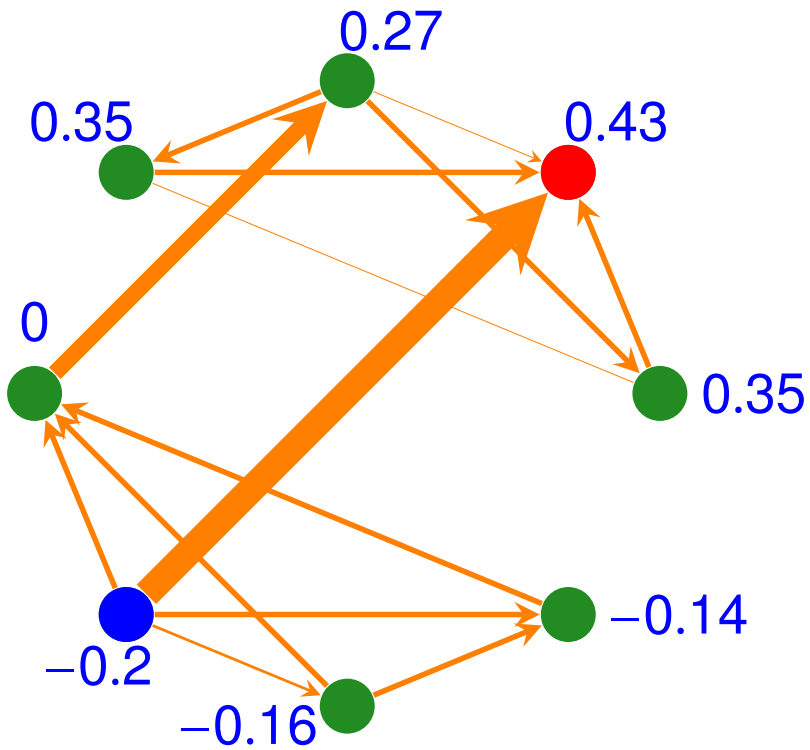
$$R_{uv} = \frac{f_{uv}^2}{||f||^2} = \frac{(\phi_u - \phi_v)^2}{R_{uv}}$$

Potential embedding of (u, v):

For $x \in V$

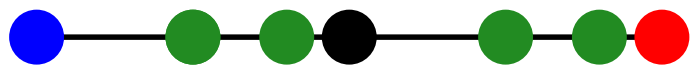
$$\phi_v - \phi_u = (\phi_v - \phi_x) + (\phi_x - \phi_u) = \mathbf{b}_{xv} L^+ \mathbf{b}_{uv} + \mathbf{b}_{ux} L^+ \mathbf{b}_{uv}$$

Potential embedding



$$R_{uv} = \frac{f_{uv}^2}{||f||^2} = \frac{(\phi_u - \phi_v)^2}{R_{uv}}$$

Potential embedding of (u, v):



For $x \in V$

$$\phi_v - \phi_u = (\phi_v - \phi_x) + (\phi_x - \phi_u) = \mathbf{b}_{uv}^T L^+ \mathbf{b}_{ux} + \mathbf{b}_{uv}^T L^+ \mathbf{b}_{xv}$$

Potential embedding

Potential embedding of (u, v):

Let $\tau = L^{+}\mathbf{b}_{xu}$ denote $x \to u$ flow potentials:

$$(\tau_u - \tau_v)^2 = (\mathbf{b}_{ux} L^{+} \mathbf{b}_{uv})^2 \geq \frac{1}{4}(\phi_u - \phi_v)^2$$

Potential embedding

Potential embedding of (u, v):

Let $\tau = L^+ \mathbf{b}_{xu}$ denote $x \to u$ flow potentials:

$$(\tau_u - \tau_v)^2 = (\mathbf{b}_{ux} L^+ \mathbf{b}_{uv})^2 \geq \frac{1}{4}(\phi_u - \phi_v)^2$$

So (u, v) carries a lot of $x \to u$ flow?

Potential embedding

Potential embedding of (u, v):

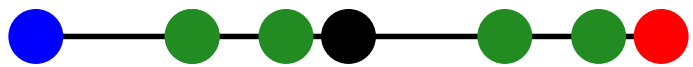
So

$$\frac{(\tau_u - \tau_v)^2}{R_{xu}} \gtrsim \frac{(\phi_u - \phi_v)^2}{R_{uv}} ?$$

Yes...

Potential embedding

Potential embedding of (u, v):



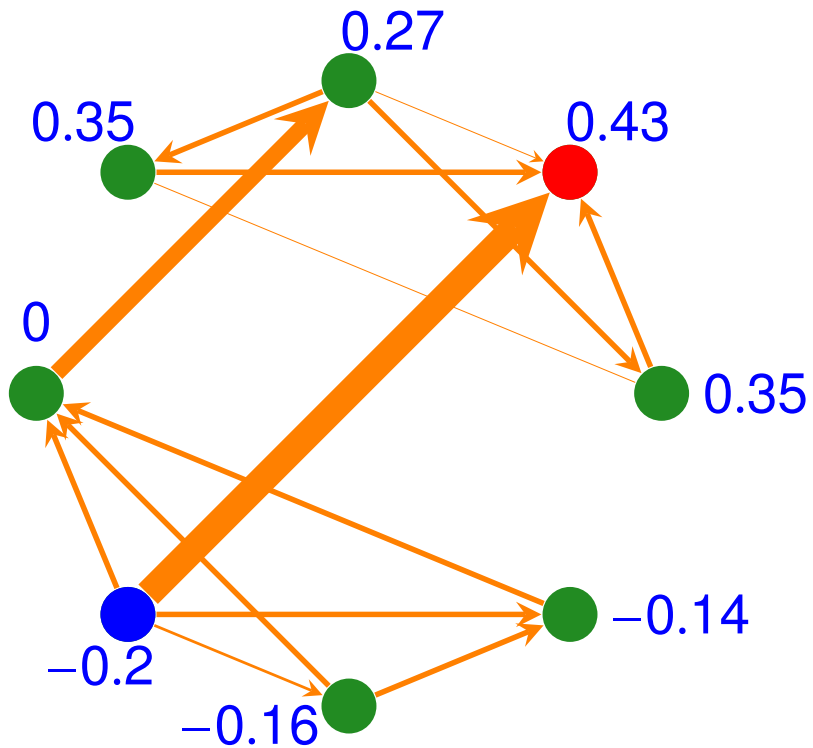
So

$$\frac{(\tau_u - \tau_v)^2}{R_{xu}} \gtrsim \frac{(\phi_u - \phi_v)^2}{R_{uv}}?$$

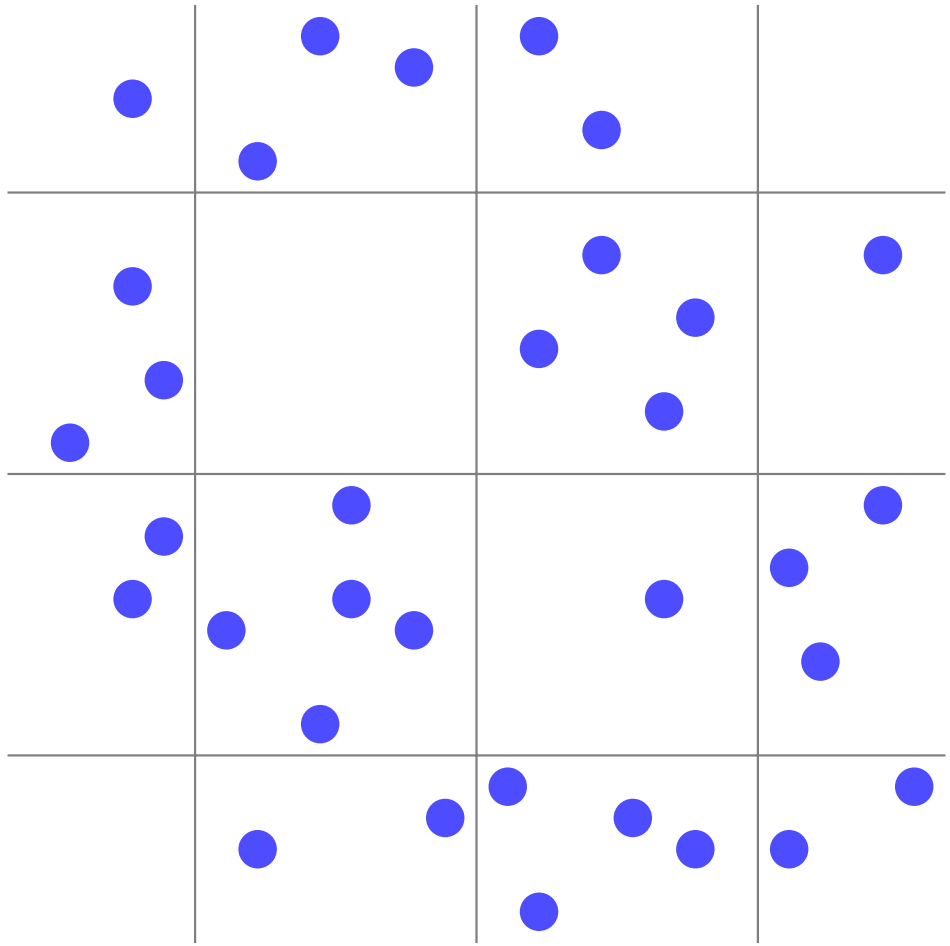
Yes... as long as $R_{xu} \lesssim R_{uv}$

To discover edge (u, v) with $R_{uv} \geq 1/\log n$:

1. Choose $x \in V$ with $R_{ux} \leq \log^{O(1)} n$
2. Send flow from x to u, list heavy edges
3. Send flow from x to v, list heavy edges

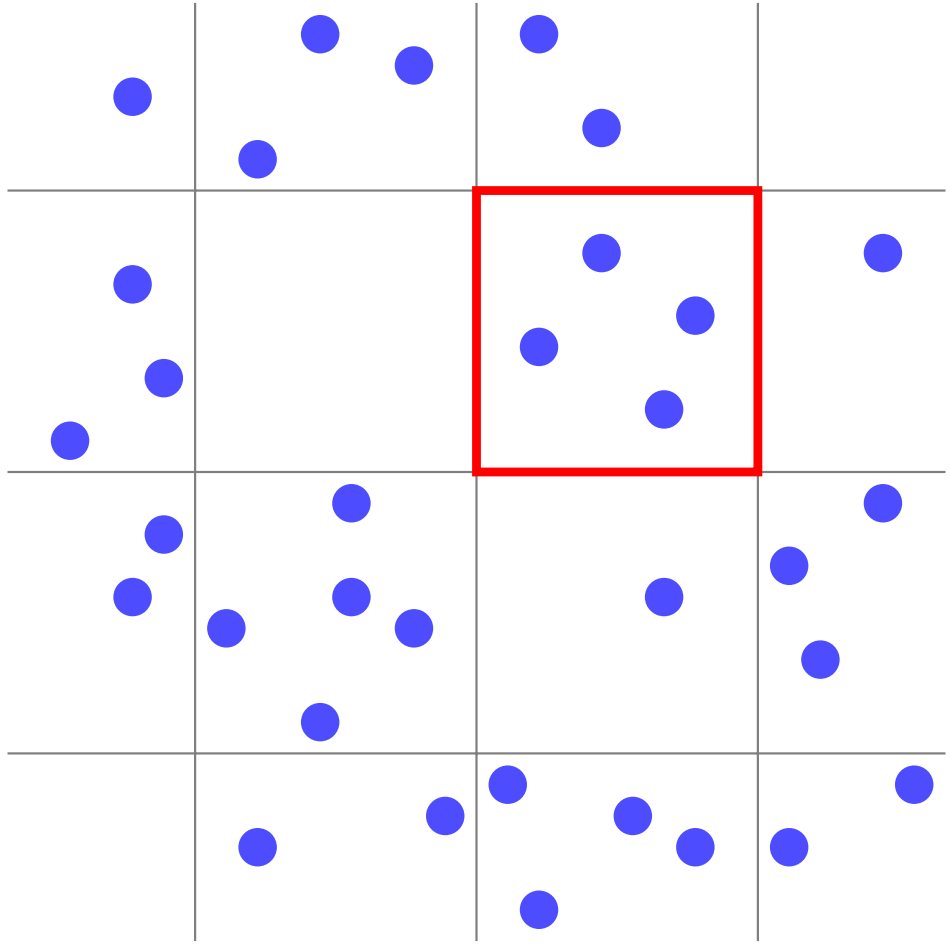


Our Approach



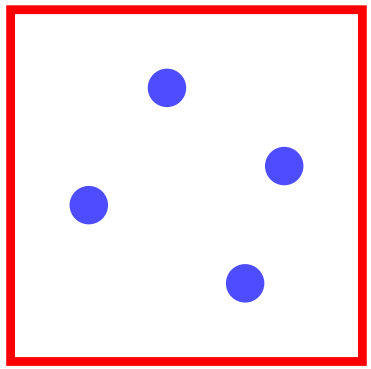
Partition vertices into clusters of bounded effective resistance diameter

Our Approach



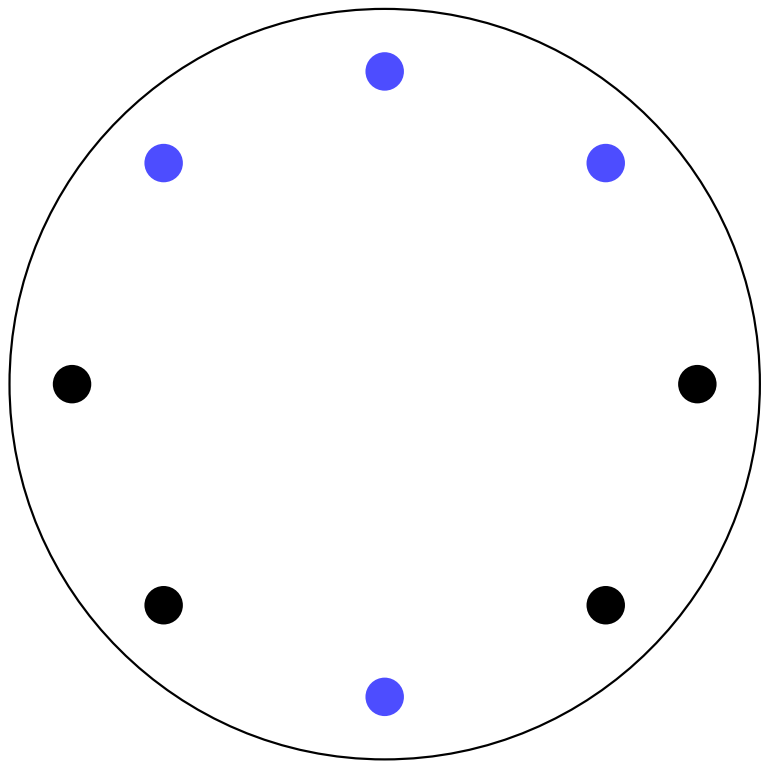
Partition vertices into clusters of bounded effective resistance diameter

Our Approach

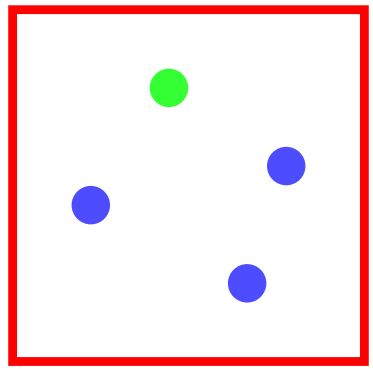


Pick representative vertex x in every cluster, try flows from x to all vertices in same cluster

This finds every heavy edge not cut by the partition

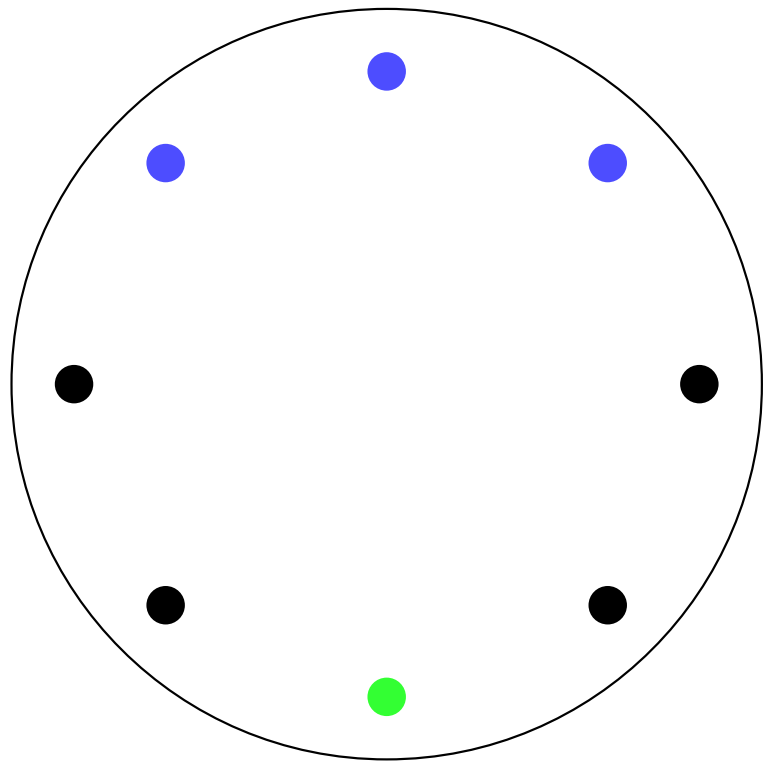


Our Approach

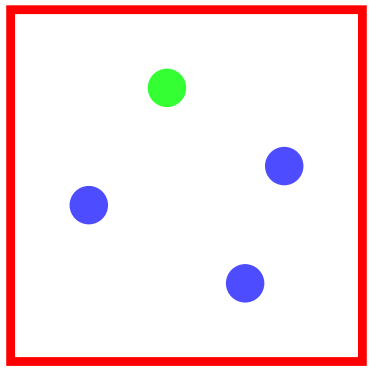


Pick representative vertex x in every cluster, try flows from x to all vertices in same cluster

This finds every heavy edge not cut by the partition

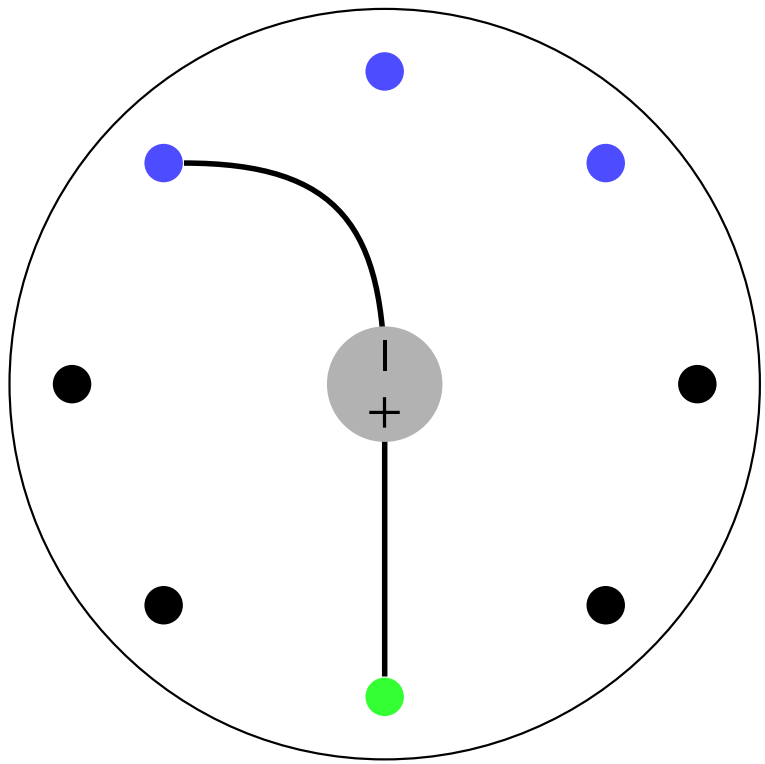


Our Approach

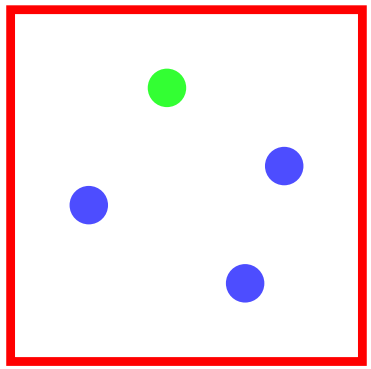


Pick representative vertex x in every cluster, try flows from x to all vertices in same cluster

This finds every heavy edge not cut by the partition

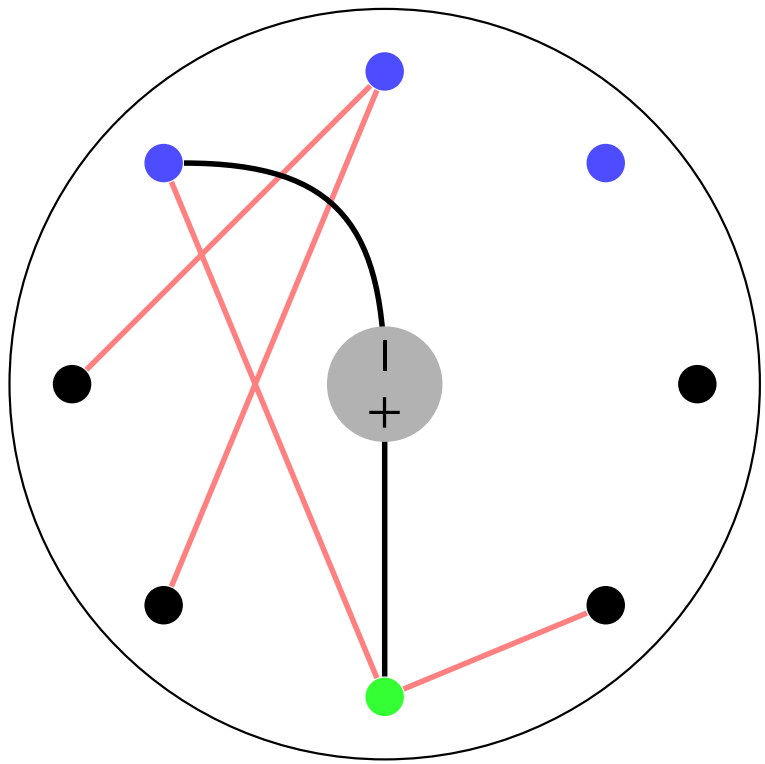


Our Approach

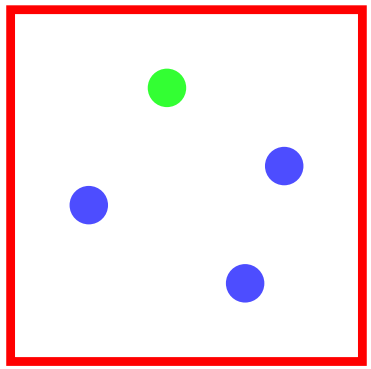


Pick representative vertex x in every cluster, try flows from x to all vertices in same cluster

This finds every heavy edge not cut by the partition

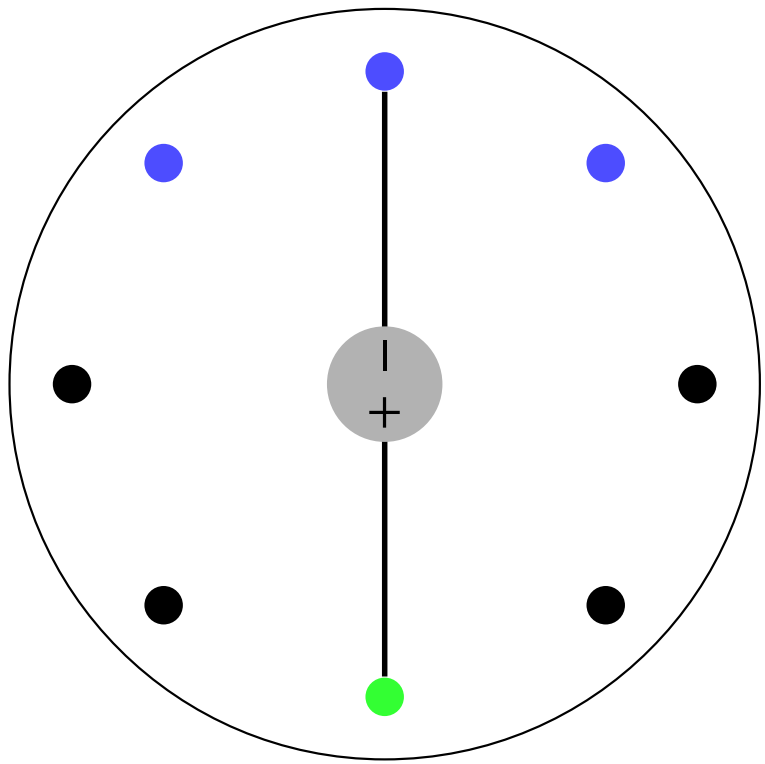


Our Approach

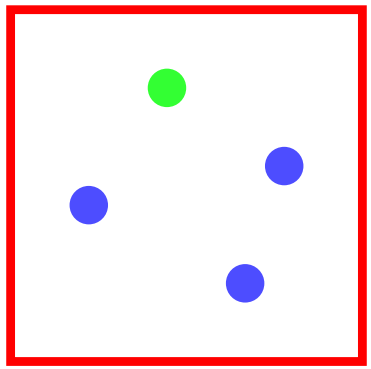


Pick representative vertex x in every cluster, try flows from x to all vertices in same cluster

This finds every heavy edge not cut by the partition

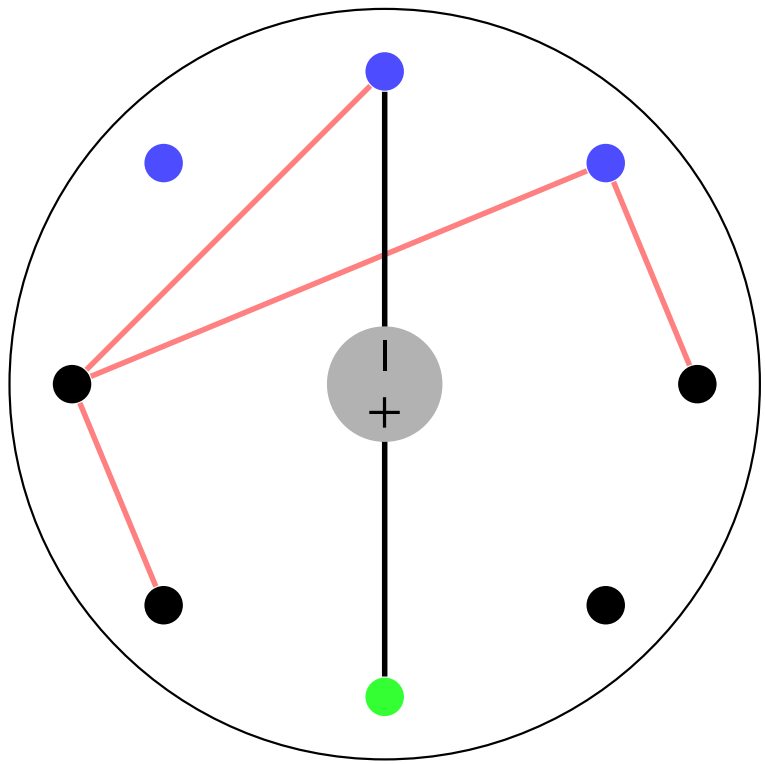


Our Approach

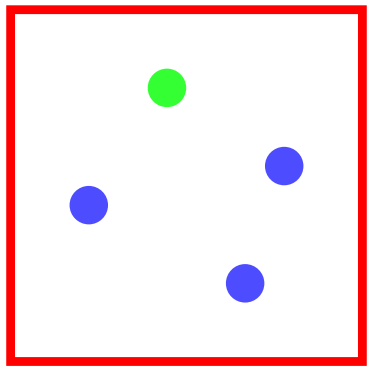


Pick representative vertex x in every cluster, try flows from x to all vertices in same cluster

This finds every heavy edge not cut by the partition

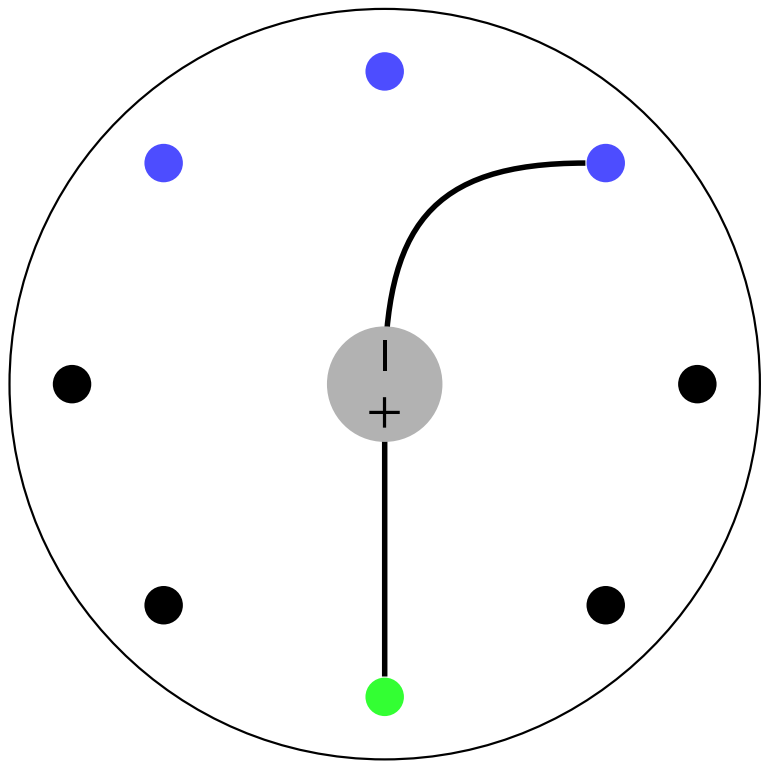


Our Approach

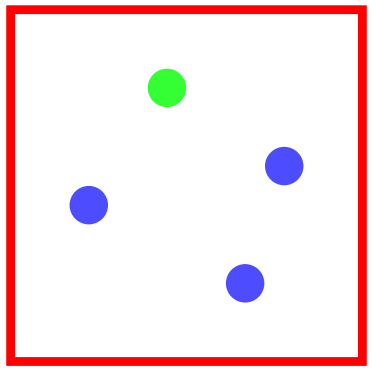


Pick representative vertex x in every cluster, try flows from x to all vertices in same cluster

This finds every heavy edge not cut by the partition

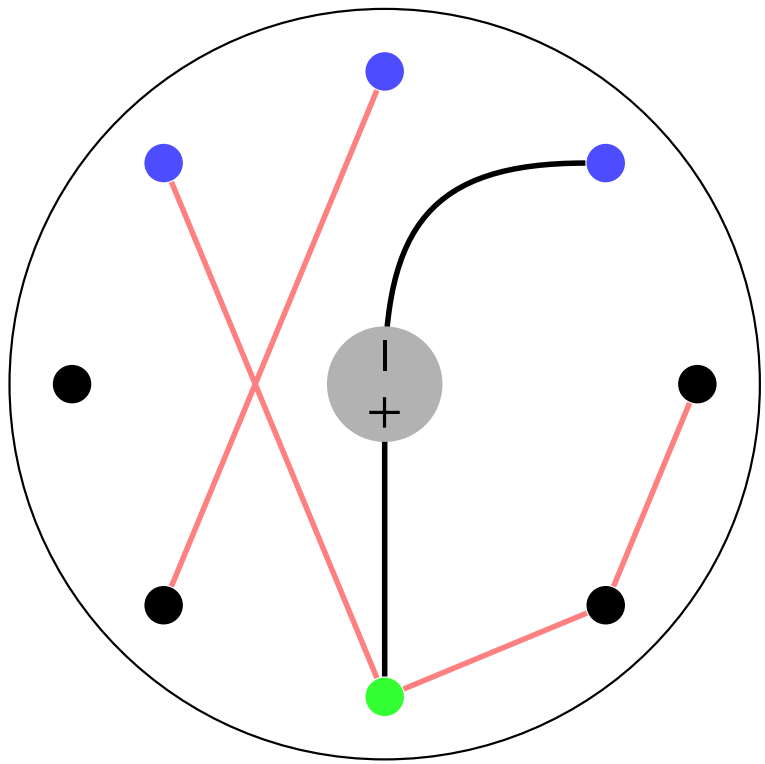


Our Approach



Pick representative vertex x in every cluster, try flows from x to all vertices in same cluster

This finds every heavy edge not cut by the partition



Partitioning via Locality Sensitive Hashing

Resistive embedding maps v to $\mathbb{R}^{\binom{n}{2}}$:

$$y \to \mathbf{B}L^{+}\mathbf{1}_{y}$$

Then

$$R_{uv} = ||\mathbf{B}L^{+}\mathbf{b}_{uv}||_2^2 = ||\mathbf{B}L^{+}\mathbf{1}_{u} - \mathbf{B}L^{+}\mathbf{1}_{v}||_2^2.$$

Partitioning via Locality Sensitive Hashing

Resistive embedding maps v to $\mathbb{R}^{\binom{n}{2}}$:

$$y \to \mathbf{B}L^{+}\mathbf{1}_y$$

Then

$$R_{uv} = ||\mathbf{B}L^{+}\mathbf{b}_{uv}||_2^2 = ||\mathbf{B}L^{+}\mathbf{1}_u - \mathbf{B}L^{+}\mathbf{1}_v||_2^2.$$

Final algorithm*:

1. Construct resistive embedding (after dimensionality reduction)
2. Hash vertices using Locality Sensitive Hashing
3. In every bucket pick representative x, try flows to other vertices in bucket

Partitioning via Locality Sensitive Hashing

Resistive embedding maps v to $\mathbb{R}^{\binom{n}{2}}$:

$$y \to \mathbf{B}L^{+}\mathbf{1}_y$$

Then

$$R_{uv} = ||\mathbf{B}L^{+}\mathbf{b}_{uv}||_2^2 = ||\mathbf{B}L^{+}\mathbf{1}_u - \mathbf{B}L^{+}\mathbf{1}_v||_2^2.$$

Final algorithm*:

1. Construct resistive embedding (after dimensionality reduction)
2. Hash vertices using Locality Sensitive Hashing
3. In every bucket pick representative x, try flows to other vertices in bucket

Set 'near' distance to ≈ 1: a heavy edge is likely to not be cut, but diameter bounded by $\log^{O(1)} n$

Fast recovery for matrices+implications for numerical linear algebra and optimization?

Remarks, related results, open problems

Weighted graphs?

Can one support turnstile updates to the weights?

Weighted graphs?

Can one support turnstile updates to the weights?

Chen, Khanna, Li'22. – support turnstile updates to edge weights using $\approx n^{6/5}$ space

Weighted graphs?

Can one support turnstile updates to the weights?

Chen, Khanna, Li'22. – support turnstile updates to edge weights using $\approx n^{6/5}$ space

Chen, Khanna, Li'22. – lower bound of $\Omega(n^{21/20})$ for vertex incidence sketches!

Spanners?

A subgraph H is a t-spanner of G if for every $u, v \in V$ one has

$$d_G(u,v) \leq d_H(u,v) \leq t \cdot d_G(u,v)$$

Every graph G on n vertices admits a $(2k-1)$-spanner of size $\approx n^{1+1/k}$ (optimal assuming the Erdős girth conjecture).

Spanners?

A subgraph H is a t-spanner of G if for every $u, v \in V$ one has

$$d_G(u,v) \leq d_H(u,v) \leq t \cdot d_G(u,v)$$

Every graph G on n vertices admits a $(2k-1)$-spanner of size $\approx n^{1+1/k}$ (optimal assuming the Erdős girth conjecture).

Baswana-Sen'07: obtains a $(2k-1)$-spanner with $\widetilde{O}(n^{1+1/k})$ edges. Inherently very sequential (k rounds).

Spanners?

A subgraph H is a t-spanner of G if for every $u, v \in V$ one has

$$d_G(u,v) \leq d_H(u,v) \leq t \cdot d_G(u,v)$$

Every graph G on n vertices admits a $(2k-1)$-spanner of size $\approx n^{1+1/k}$ (optimal assuming the Erdős girth conjecture).

Baswana-Sen'07: obtains a $(2k-1)$-spanner with $\widetilde{O}(n^{1+1/k})$ edges. Inherently very sequential (k rounds).

Single pass sketching algorithm?

Spanners?

A subgraph H is a t-spanner of G if for every $u, v \in V$ one has

$$d_G(u,v) \leq d_H(u,v) \leq t \cdot d_G(u,v)$$

Every graph G on n vertices admits a $(2k-1)$-spanner of size $\approx n^{1+1/k}$ (optimal assuming the Erdős girth conjecture).

Baswana-Sen'07: obtains a $(2k-1)$-spanner with $\widetilde{O}(n^{1+1/k})$ edges. Inherently very sequential (k rounds).

Single pass sketching algorithm?

Theorem (Fitser-K.-Nouri'21)

A $\widetilde{O}(n^{2/3})$-spanner in $\widetilde{O}(n)$ space

Spanners?

A subgraph H is a t-spanner of G if for every $u, v \in V$ one has

$$d_G(u,v) \leq d_H(u,v) \leq t \cdot d_G(u,v)$$

Every graph G on n vertices admits a $(2k-1)$-spanner of size $\approx n^{1+1/k}$ (optimal assuming the Erdős girth conjecture).

Baswana-Sen'07: obtains a $(2k-1)$-spanner with $\widetilde{O}(n^{1+1/k})$ edges. Inherently very sequential (k rounds).

Single pass sketching algorithm?

Theorem (Fitser-K.-Nouri'21)

A $\widetilde{O}(n^{2/3})$-spanner in $\widetilde{O}(n)$ space(+space/stretch tradeoffs).

Algorithm: take any spectral sparsifier, drop weights on edges.

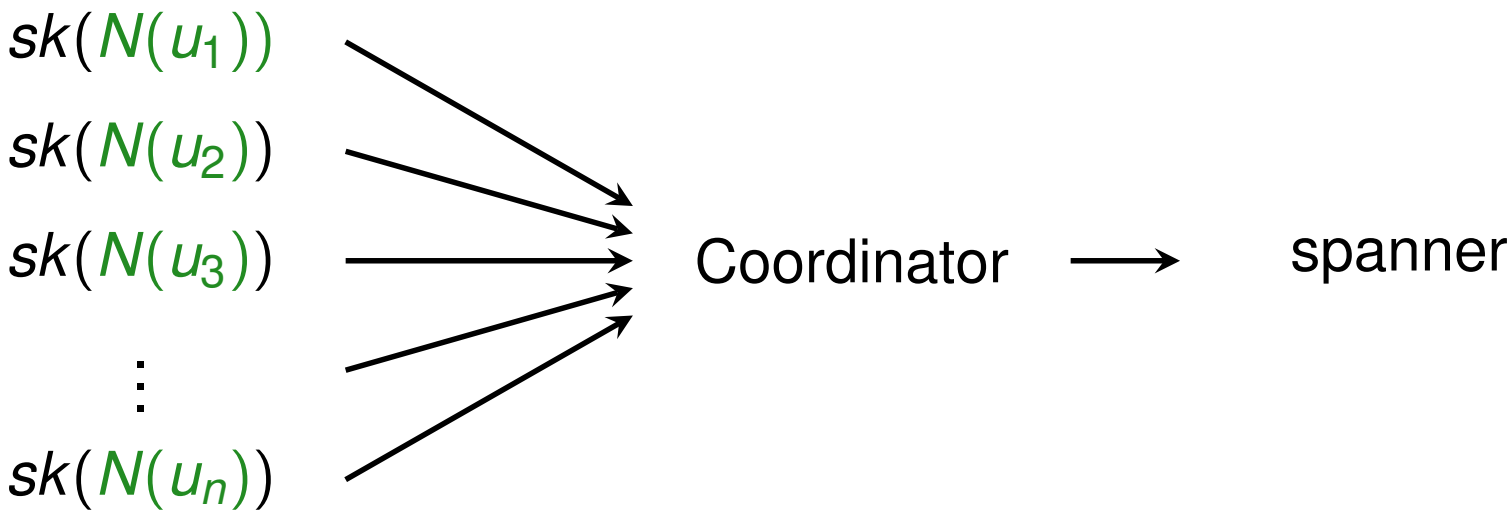
Lower bounds

Input: Each $u \in V$ holds list of its neighbors $N(u)$

Each $u \in V$ sends a sketch of $N(u)$ to coordinator

(assume shared randomness)

Output: $n^{2/3-\Omega(1)}$-spanner of G



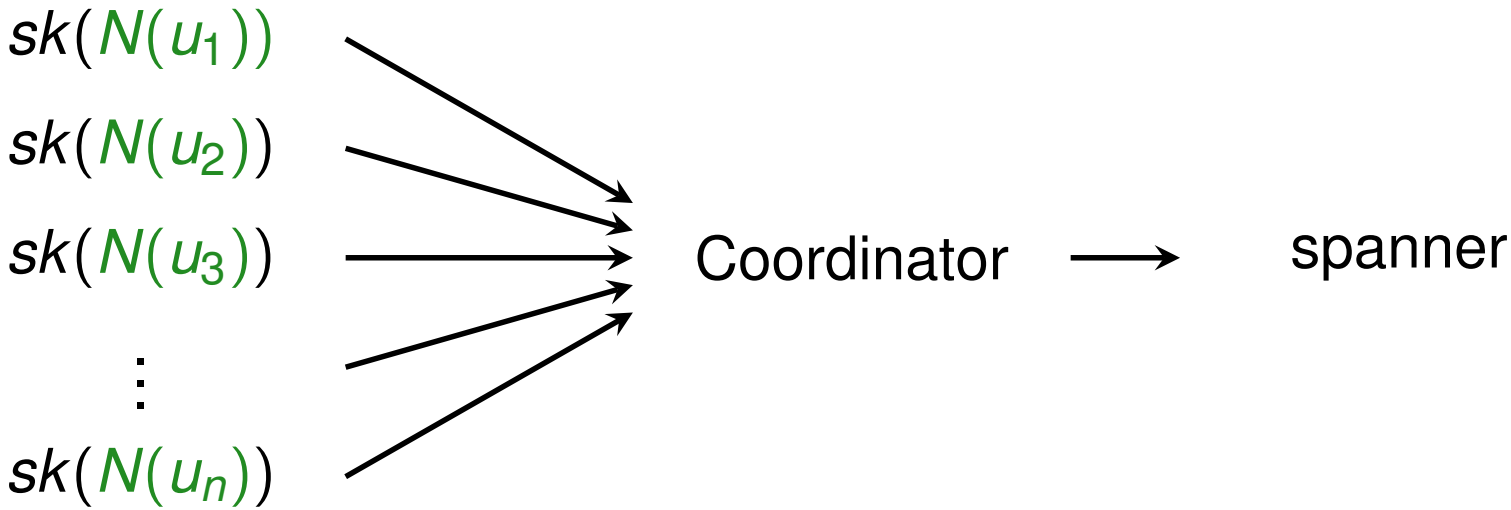
Lower bounds

Input: Each $u \in V$ holds list of its neighbors $N(u)$

Each $u \in V$ sends a sketch of $N(u)$ to coordinator

(assume shared randomness)

Output: $n^{2/3-\Omega(1)}$-spanner of G



Show that $n^{\Omega(1)}$ communication per node is needed?

Recent progress:Assadi-K.-Yu'23 prove that Filtser-K.-Nouri'21 tradeoffs are optimal for a natural class of sketches

Expander decompositions in sketching

Filtser-K.-Makarov'23 – expander decompositions by sketching

Filtser-K.-Makarov'23, Chen-K.-Makarov-Mazzali'?? – sparsity-independent expander decompositions!

Turn sketching algorithms into low space dynamic algorithms?

Prove that decoded solution is ‘stable’?