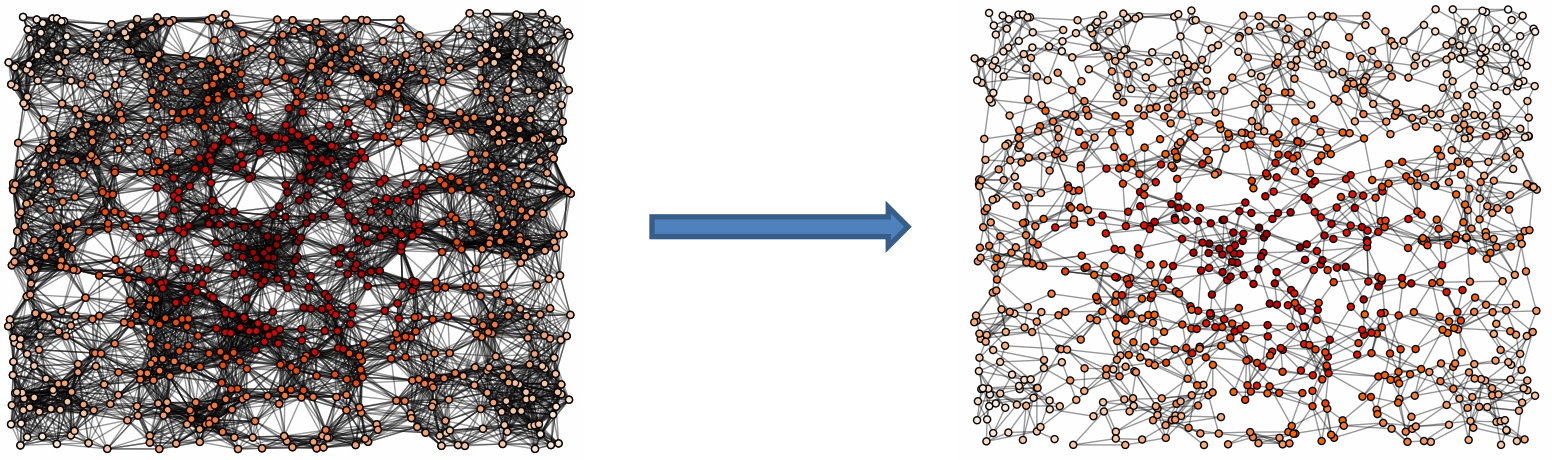
Graph Sparsification

Debmalya Panigrahi

Graph Sparsifiers



“Smaller” graph that (approximately) preserves the values of some set of graph parameters

Graph Sparsifiers

- Spanners
- Emulators
- Small stretch spanning trees
- Vertex sparsifiers
- …
- Spectral sparsifiers
- Cut sparsifiers

Spectral Sparsification

- Undirected graph **G = (V, E)**; error parameter **ε**
- Goal: $\mathbf{G_\varepsilon = (V, E_\varepsilon)}$ with $\mathbf{\tilde{O}(n/\varepsilon^2)}$ edges such that for all **n**-dimensional vectors **x**,

$$\mathbf{(1-\varepsilon)\, x^T L(G)\, x \;\le\; x^T L(G_\varepsilon)\, x \;\le\; (1+\varepsilon)\, x^T L(G)\, x}$$

- Graph Laplacian **L = D – A**, where
  - **D** = Diagonal Degree Matrix of the graph
  - **A** = Adjacency Matrix of the graph

Spectral Sparsification: Previous work

Running time of the sparsification algorithm	Number of edges in the sparsifier	
$O(n^3m)$	$O(n/\varepsilon^2)$	[Batson-Spielman-Srivastava '09]
$O(n^2m \log^3 n + n^4 \log n)$		[Zouzias '12]
$O(m \log^{O(1)} n)$	$O(n \log^{O(1)} n/\varepsilon^2)$	[Spielman-Teng '04]
$O(m \log^{O(1)} n)$	$O(n \log n/\varepsilon^2)$	[Spielman-Srivastava '08]
$O(m \log^3 n)$		SS + [Koutis-Miller-Peng '10, '11]
$O(m \log^2 n)$		[Koutis-Levin-Peng '12]
$O(m \log n)$	$O(n \log^3 n/\varepsilon^2)$	[Koutis-Levin-Peng '12]
$O(m)$???	???	

Spectral to Cut Sparsifiers

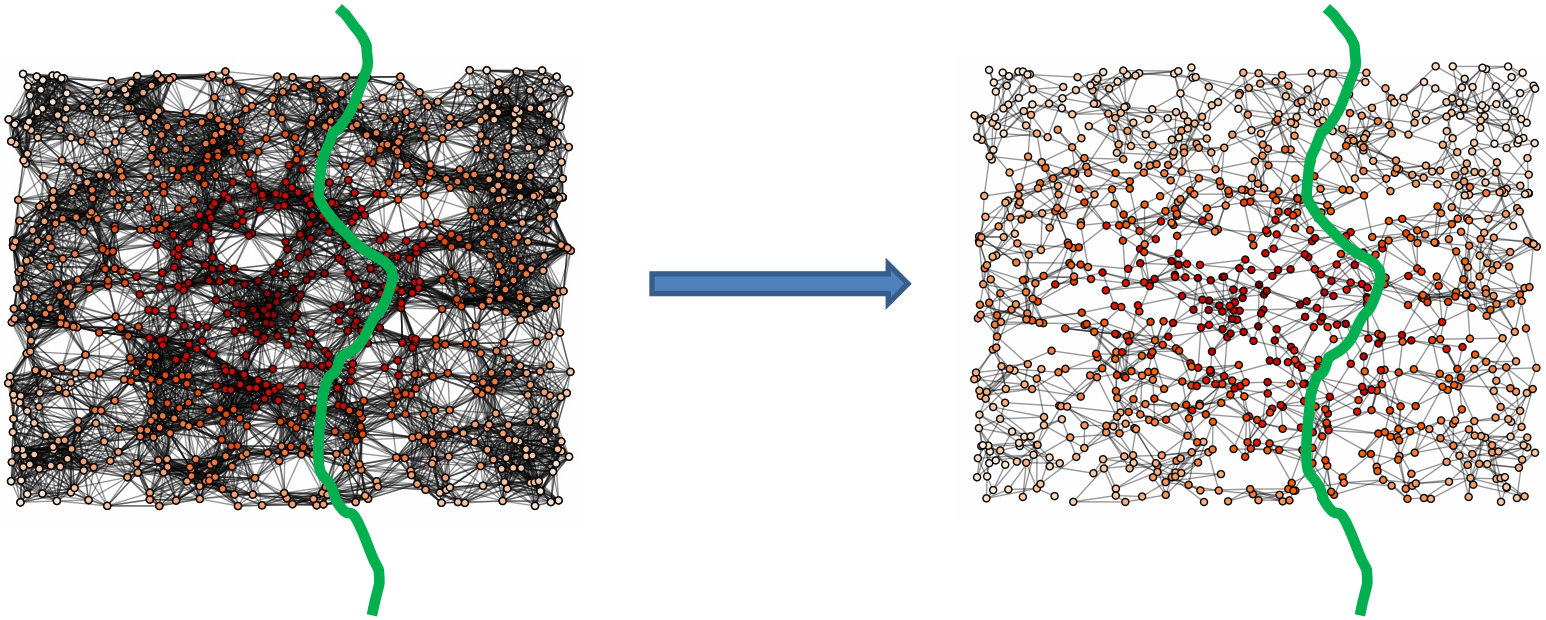
- $\mathbf{G_\varepsilon = (V, E_\varepsilon)}$ is a spectral sparsifier of $\mathbf{G = (V, E)}$ if for all $\mathbf{n}$-dimensional vectors $\mathbf{x}$,

$$\mathbf{(1-\varepsilon)\, x^T L(G)\, x \;\le\; x^T L(G_\varepsilon)\, x \;\le\; (1+\varepsilon)\, x^T L(G)\, x}$$

- $\mathbf{x^T L\, x = \Sigma_{(i,\, j) \in E} (x_i - x_j)^2}$
- Suppose $\mathbf{x \in \{0, 1\}^n}$; $\mathbf{S = \{i \in V: x_i = 1\}}$. Then,

$$\mathbf{x^T L\, x = \Sigma_{(i,\, j) \in E} (x_i - x_j)^2 = \Sigma_{(i,\, j) \in (S,\, V - S)} 1 = E(S)}$$

Cut Sparsification



Weight of <u>every cut</u> is preserved up to a multiplicative error of **(1 ± ε)**

Cut Sparsification

- Undirected (unweighted) graph **G = (V, E)**; error parameter **ε**
- Goal: **G_ε = (V, E_ε)** with **$O(n \log n/\varepsilon^2)$** edges such that for all cuts **(S, V – S)**,

$$(1-\varepsilon)\, E(S) \;\le\; E_\varepsilon(S) \;\le\; (1+\varepsilon)\, E(S)$$

- Introduced by Benczur-Karger ’96
  - **$O(m \log^2 n)$**-time algorithm to find a cut sparsifier (with high probability) containing **$O(n \log n/\varepsilon^2)$** edges in expectation

Fung-Hariharan-Harvey-P.:

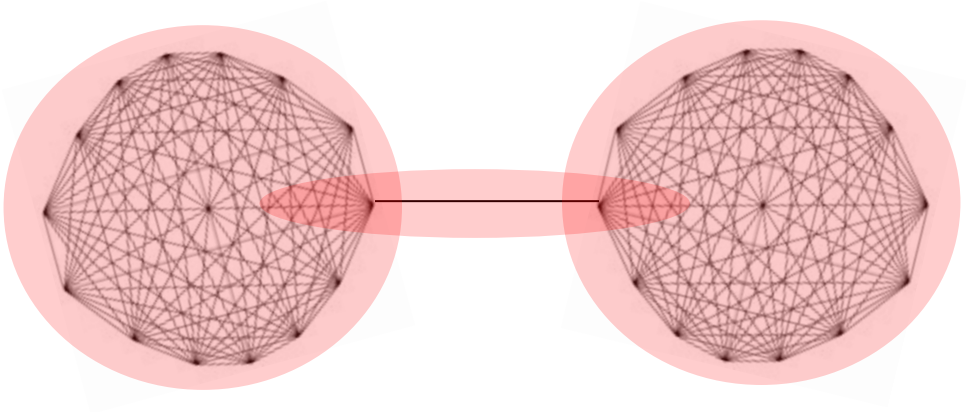
A linear-time, i.e. O(m), algorithm that produces a cut sparsifier (whp) containing $O(n \log n/\varepsilon^2)$ edges in expectation

Cut Sparsification by Sampling

Non

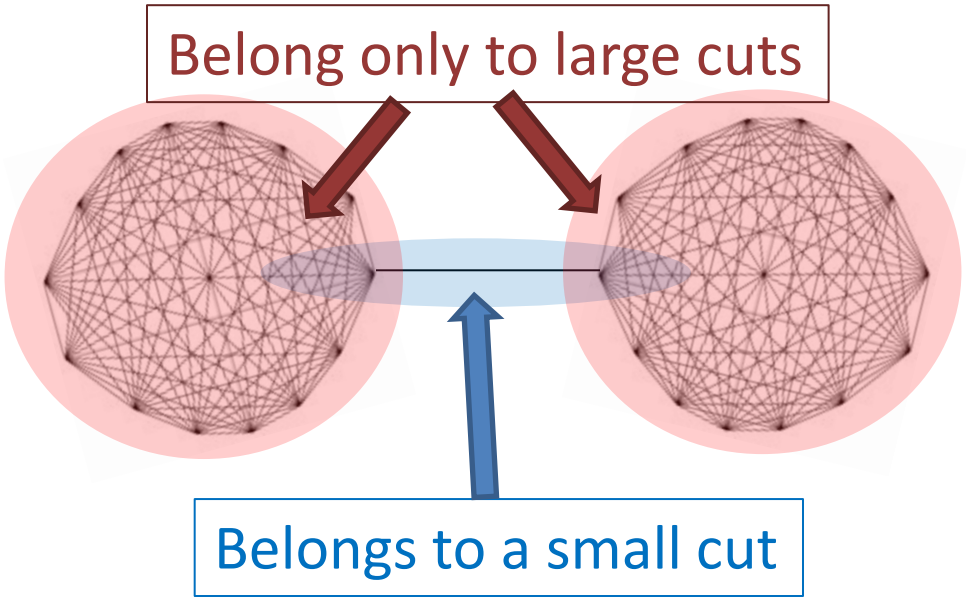
edge **e** with prob p_e

- **Uniformly** sample ~~all edges with prob $p \approx n/m$~~
  - Selected edge is given weight ~~$1/p$~~ $1/p_e$



p ≈ 1/n; graph gets disconnected

Sampling Probabilities



Edge Connectivity λ_e = size of smallest cut containing **e**

$$p_e = \log n/\lambda_e$$

Sampling by Edge Connectivity

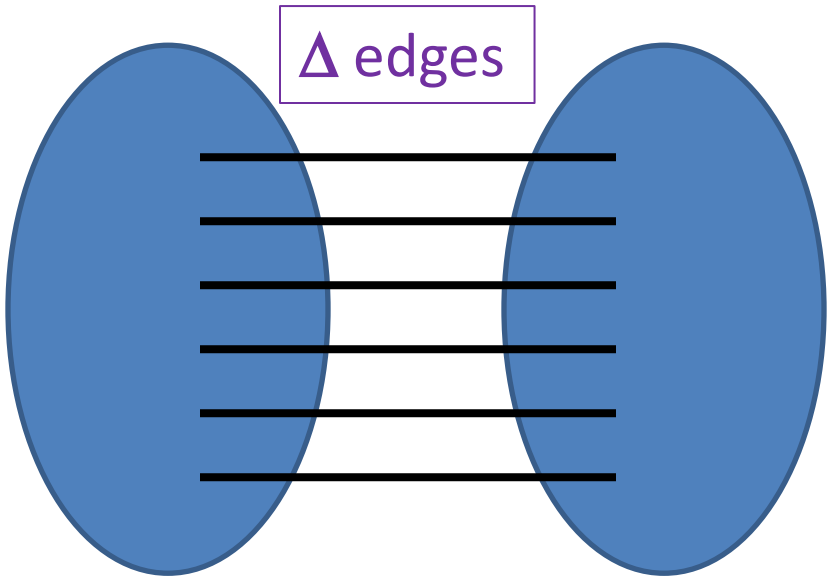
- Sample edge **e** independently (of other edges) with probability $\mathbf{p_e \approx \log n/\lambda_e}$
- If edge **e** is selected, it is given a weight of $\mathbf{1/p_e}$ in the sparsifier
- **Sparsifier has O(n log n) edges in expectation**

$$\lambda_e \geq 1/r_e \implies \Sigma_{e \in E}\, 1/\lambda_e \leq \Sigma_{e \in E}\, r_e = n - 1$$

- **Pr[$E_\varepsilon(S) \in (1 \pm \varepsilon)\, E(S)$** for all cuts **(S, V - S)]?**

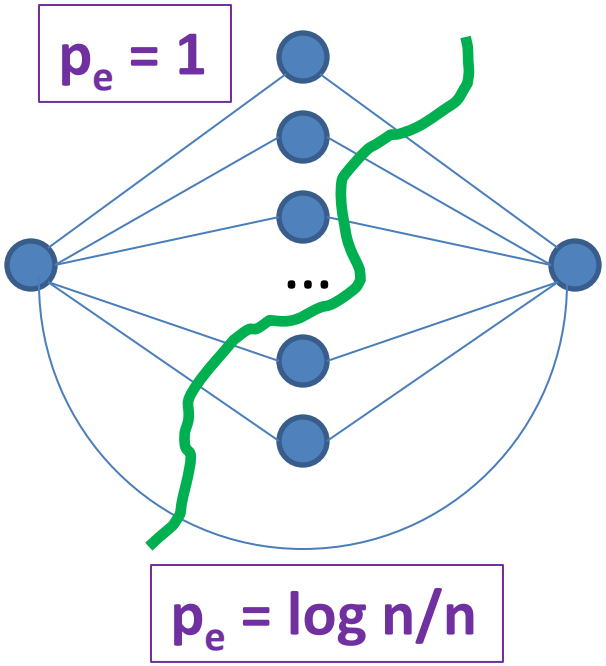
Bounding Deviation

- Expected number of edges in the cut ≥ **log n**
- Chernoff bounds: Probability of **εΔ** error ≤ **1/poly(n)**
- Exponential number of cuts!



$\lambda_e \leq \Delta$, i.e. $p_e \geq \log n/\Delta$

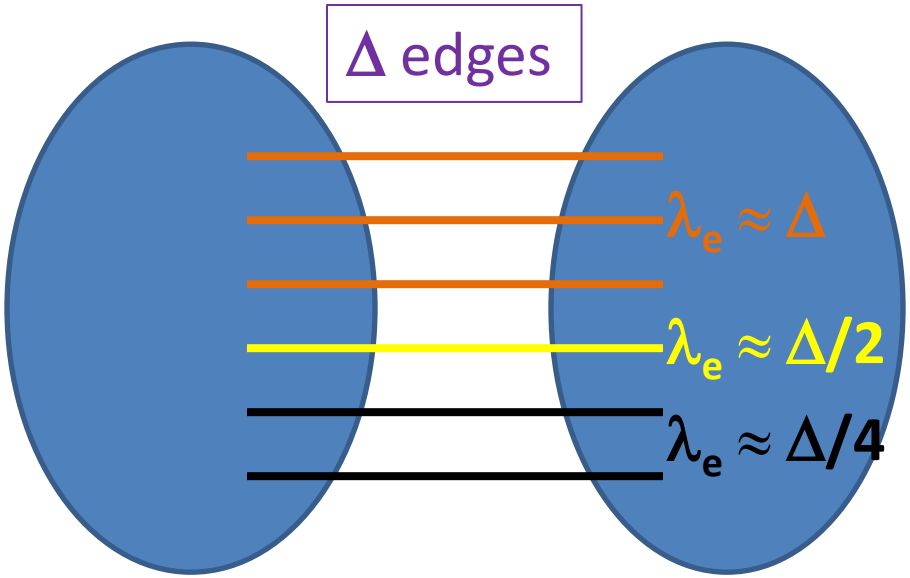
Bounding Deviation



- Error probability for single cut ≤ **1/poly(n)** but **exp(n)** cuts

- **Cut projections**
  Categorize edges in a cut according to the value of λ_e (i.e., p_e)

Bounding Deviation



- For $\lambda_e \approx \Delta/k$ cut projection, $p_e = k \log n/\Delta$
- Probability of $\varepsilon\Delta$ error $\leq \exp(-k \log n) = n^{-\Omega(k)}$

Cut Projections

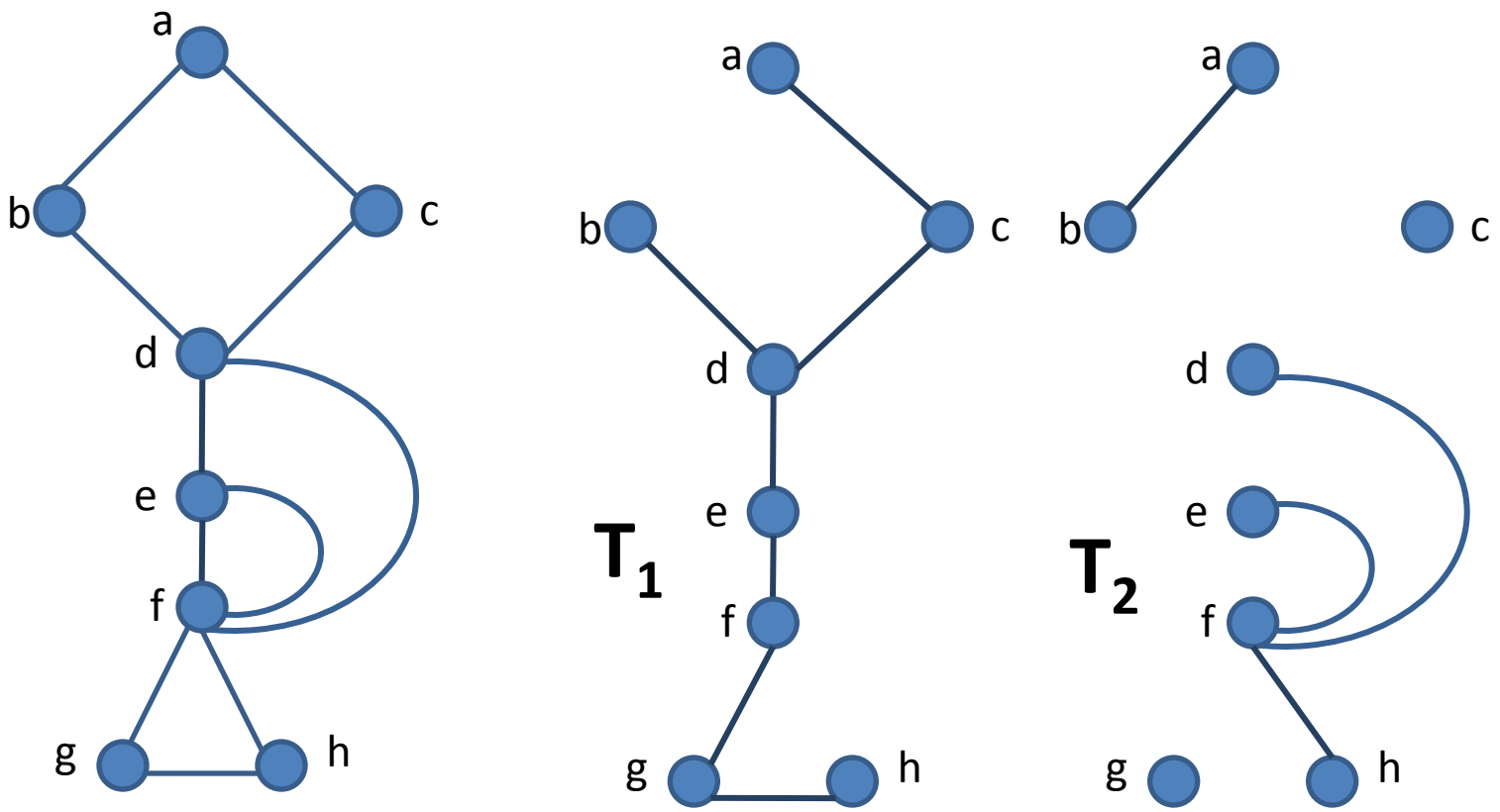
Lemma: There are $\leq n^{O(k)}$ distinct **(Δ, k)** cut projections in cuts of size **Δ**

union bound on **k, Δ**

Theorem: Sampling edge **e** with probability $\log^2 n / \lambda_e$ produces a cut sparsifier

Difficulty: Edge connectivities (λ_e) are time-consuming to calculate (Gomory-Hu tree takes $\tilde{O}(mn)$ time [Bhalgat-Hariharan-Kavitha-P., '07])

Greedy Spanning Forest packing



Sampling by NI Index

Nagamochi-Ibaraki (NI) index of edge e

$\mathbf{y_e}$ = index of **e** in an arbitrary but fixed greedy spanning forest packing

Proposed Cut Sparsification Algorithm

- Sample edge **e** with probability $\mathbf{p_e \approx \log^2 n / y_e}$
- If edge **e** is selected, it is given a weight of $\mathbf{1/p_e}$ in the sparsifier

Sampling by NI Index: Cut preservation

<u>Lemma</u>: The graph $G_\varepsilon = (V, E_\varepsilon)$ produced by sampling using NI indices is a cut sparsifier, i.e., with high probability, for all cuts (S, V-S)

$$(1-\varepsilon)\, E(S) \;\leq\; E_\varepsilon(S) \;\leq\; (1+\varepsilon)\, E(S)$$

- For each edge **e**, $\mathbf{y_e} \leq \lambda_e$ (if edge **e** is in $\mathbf{i}^{th}$ forest, then its endpoints are connected by disjoint paths in the previous **i-1** forests)
- Now piggyback on the proof for sampling using edge connectivities

Sampling by NI Index: Sparsification

<u>Lemma</u>: The sparsifier has $O(n \log^3 n)$ edges in expectation

$$\Sigma_{e \in E} 1/y_e = \Sigma_k |T_k|/k = (n-1) \Sigma_k 1/k$$
$$= O(n \log n)$$

Sampling by NI Index: Running time

<u>Lemma</u> [Nagamochi-Ibaraki ’92]: The running time of the sampling algorithm (i.e., time taken to estimate the NI indices of all edges) is O(m)

We have shown:

An **O(m)**-time algorithm that produces a cut sparsifier containing **$O(n \log^3 n)$** edges

We will now show:

An **O(m)**-time algorithm that produces a cut sparsifier containing **$O(n \log^2 n)$** edges

We had promised (see the paper):

An **O(m)**-time algorithm that produces a cut sparsifier containing **$O(n \log n)$** edges

Sampling by NI Index: New Algorithm

Previous Algorithm

- Sample edge **e** with probability $p_e \approx \log^2 n / y_e$
- If edge **e** is selected, it is given a weight of $1/p_e$ in the sparsifier

New Algorithm

- Sample edge **e** with probability $p_e \approx \log n / y_e$
- If edge **e** is selected, it is given a weight of $1/p_e$ in the sparsifier

Sampling by NI Index: New Algorithm

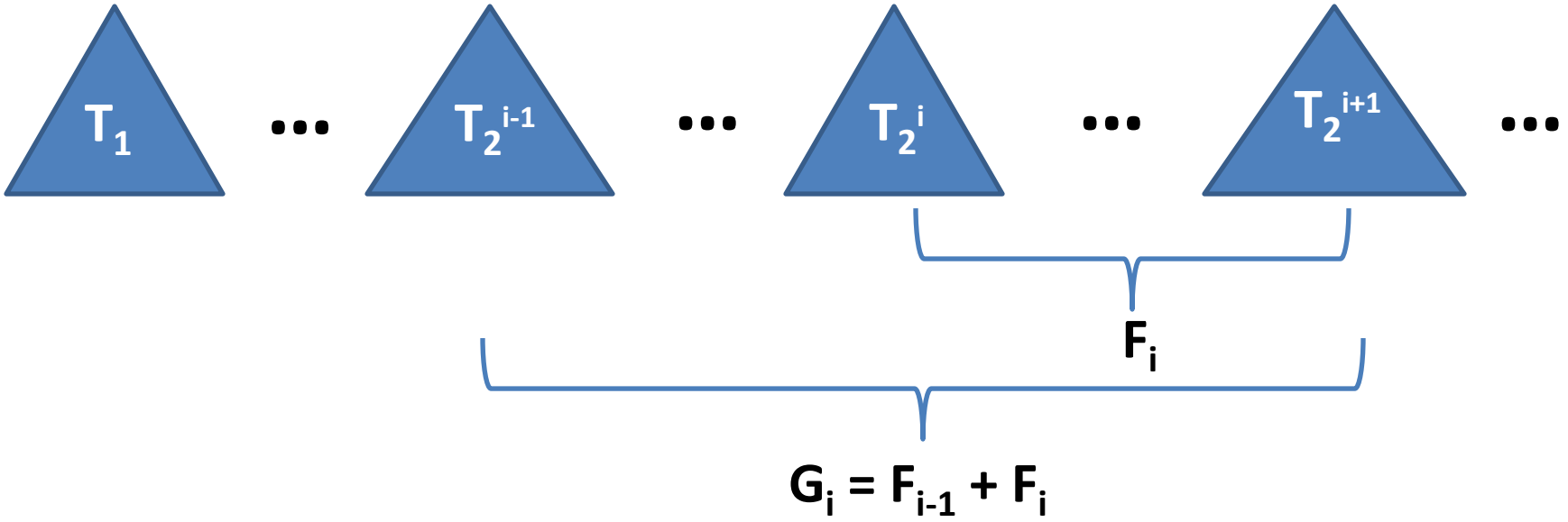
New Algorithm

- Sample edge **e** with probability $p_e \approx \log n / y_e$
- If edge **e** is selected, it is given a weight of $\mathbf{1/p_e}$ in the sparsifier

- Running time remains **O(m)**
- The expected number of edges is $\mathbf{O(n \log^2 n)}$
- **Is the sample a cut sparsifier?**

[Note: We can no longer piggyback on the analysis for sampling with edge connectivity]

Bucketing the forests



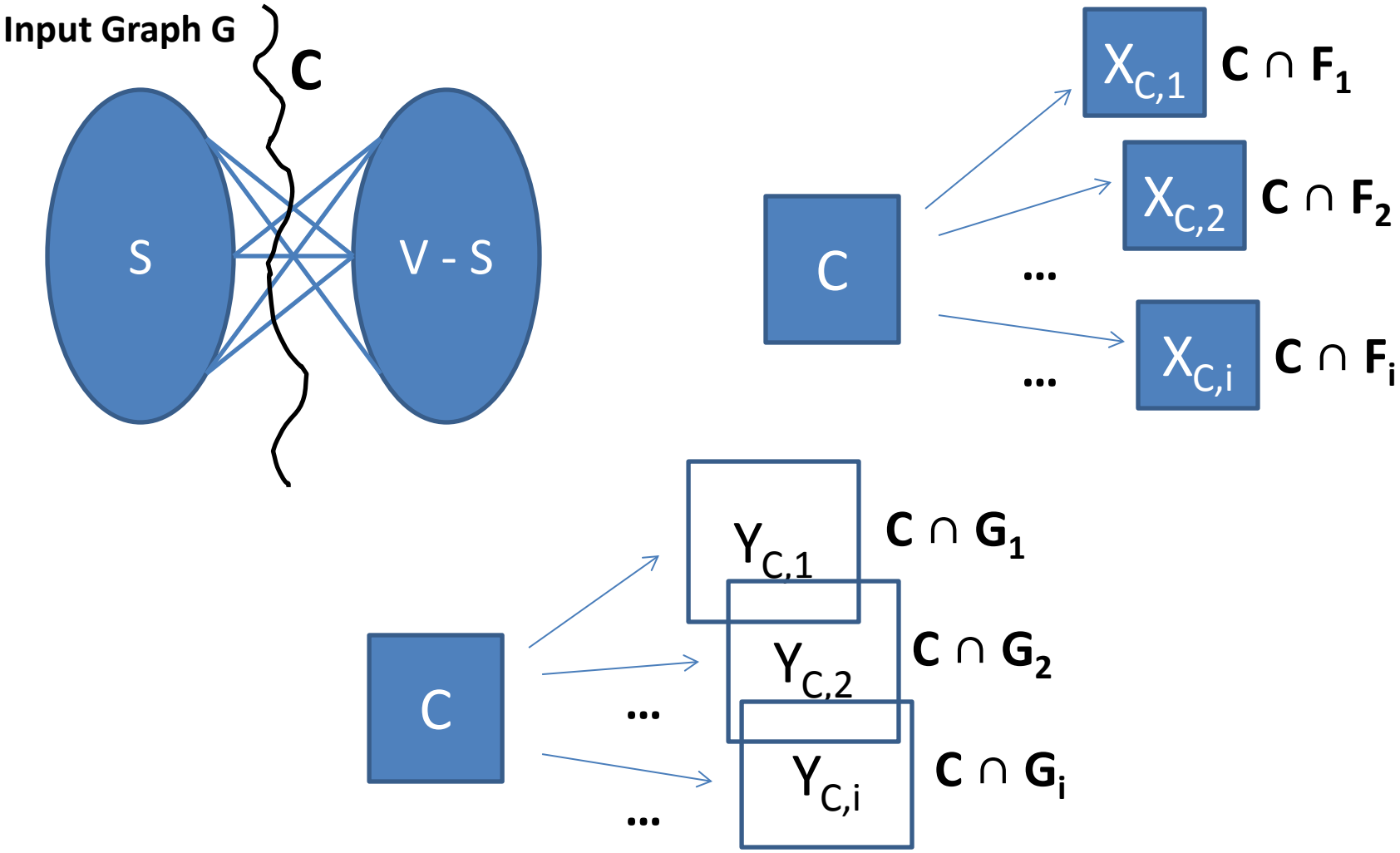
Properties of the bucketing

- Similarity property: All edges in $\mathbf{F_i}$ have sampling probability $\mathbf{p_e \approx \log n / 2^{i-1}}$ (up to a factor of 2)

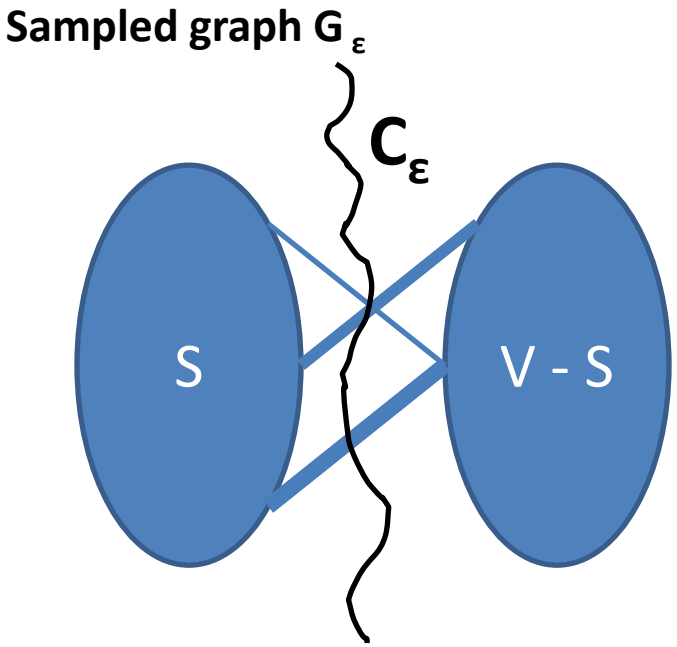
- Overlap property: Every edge appears in $\mathbf{G_i}$ for at most two values of **i**

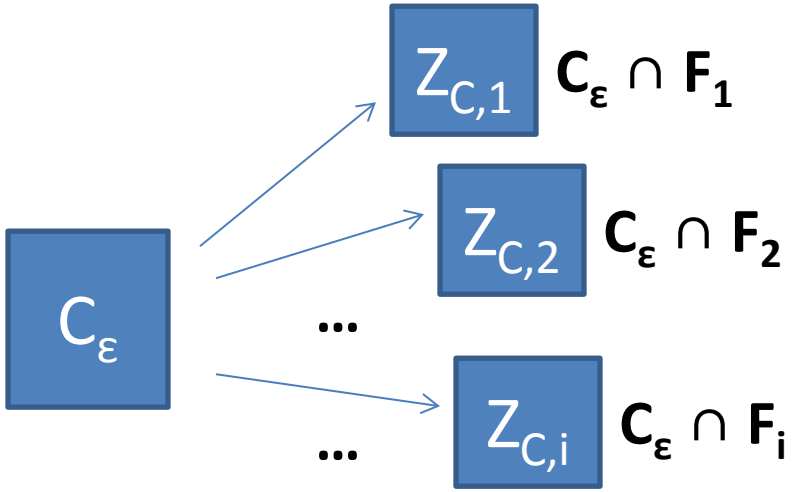
- Connectivity property: Every edge in $\mathbf{F_i}$ has edge connectivity $\geq \mathbf{2^{i-1}}$ in $\mathbf{G_i}$
  - The endpoints of the edge have $\mathbf{2^{i-1}}$ disjoint paths between them, one in each forest, in $\mathbf{G_i}$

Analysis of a cut



Tail Bounds on Deviation





Tail Bounds on Deviation

- Need to show: whp, $|C - C_\varepsilon| < \varepsilon C$ for all cuts C

- **whp,** $|X_{C,i} - Z_{C,i}| < \varepsilon X_{C,i}$ for all cuts C and all i

$\sum_i Y_{C,i} = 2C$ by the overlap property

Lemma: whp, $|X_{C,i} - Z_{C,i}| < \varepsilon Y_{C,i}$ for all cuts C and all i

Tail Bounds on Deviation

- **<u>Lemma</u>: whp, $|X_{C,i} - Z_{C,i}| < \varepsilon Y_{C,i}$ for all cuts C and all i**

- Let $\mathbf{C_k}$ be cuts for which $\mathbf{Y_{C,i} = |C \cap G_i| = 2^{i+k}}$
- By the connectivity property, every edge in $\mathbf{X_{C,i}}$ is $\mathbf{2^{i-1}}$-connected in $\mathbf{Y_{C,i}}$
- By Cut Projection Counting Lemma,

  There are at most $\mathbf{n^{2\wedge(i+k)/2\wedge i} = n^{2\wedge k}}$ distinct $\mathbf{X_{C,i}}$ in $\mathbf{C_k}$

Tail Bounds on Deviation

- **<u>Lemma</u>: whp, $|X_{C,i} - Z_{C,i}| < \varepsilon Y_{C,i}$ for all cuts C and all i**

- There are at most $n^{2\wedge k}$ distinct $X_{C,i}$ in C_k
- By the similarity property + Chernoff bounds,

$$\Pr[|X_{C,i} - Z_{C,i}| > \varepsilon Y_{C,i}] < \exp(-2^{i+k}(\log n / 2^i)) = n^{-2\wedge k}$$

union bound over distinct $X_{C,i}$ in C_k , all values of **k** and **i**

Open Problems

- Linear-time spectral sparsification algorithm
- (Near)-linear time construction of $\mathbf{O(n/\varepsilon^2)}$-sized cut/spectral sparsifiers
  - Edge sampling has fundamental limitations (connectivity of Erdos-Renyi random graph has a probability threshold of **log n/n**)
  - Cut/spectral sparsifiers from spanning trees?

    [Goyal-Rademacher-Vempala ’09, Fung-Harvey ’10]
  - Cut/spectral sparsifiers from spanners?

    [Kapralov-Panigrahy ’12, Koutis ’14]

Thank You

Questions?