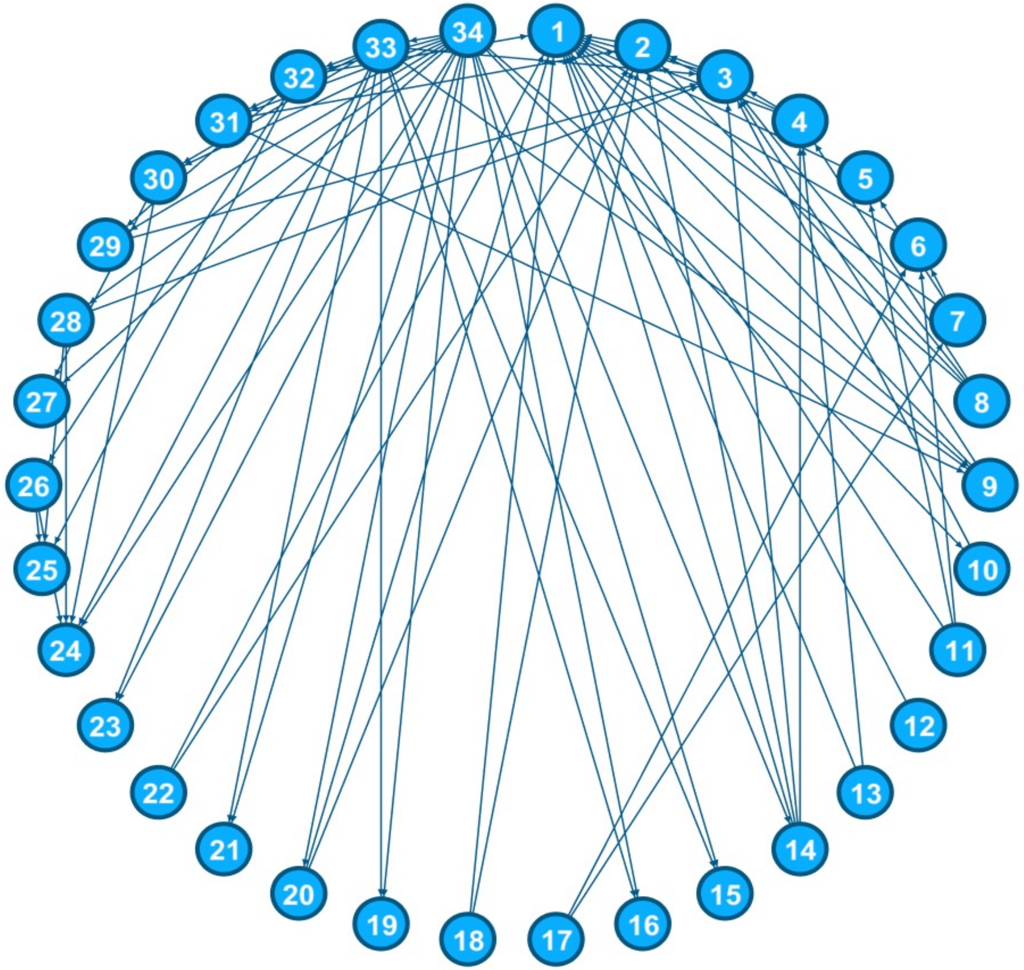
Dynamic Schur Complement of Graph Laplacian

Yu Gao
Georgia Tech

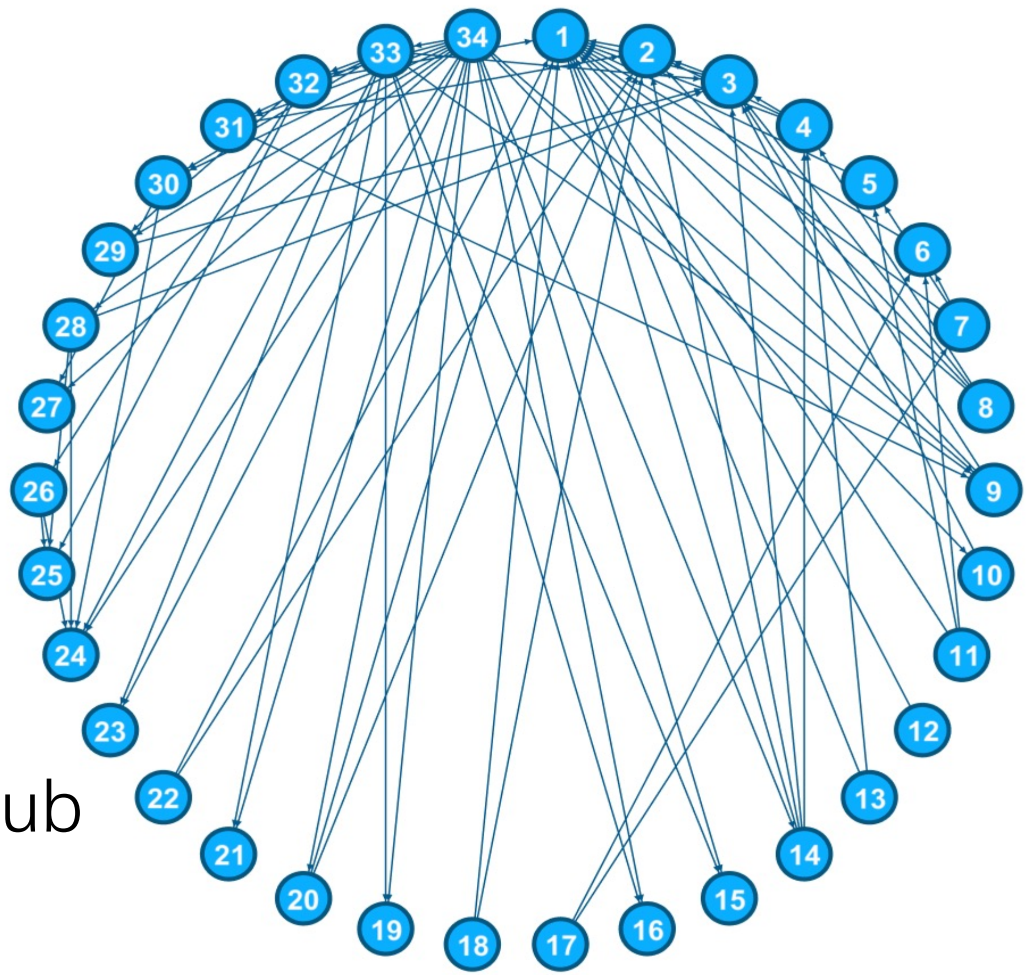
Georgia Tech

- Karate Club and Graph Laplacian
- Dynamic Laplacian by Schur Complement
- Dynamic Laplacian for Planar Graphs
- Dynamic Laplacian for General Graphs

Example of Graph: Zachary’　s Karate Club



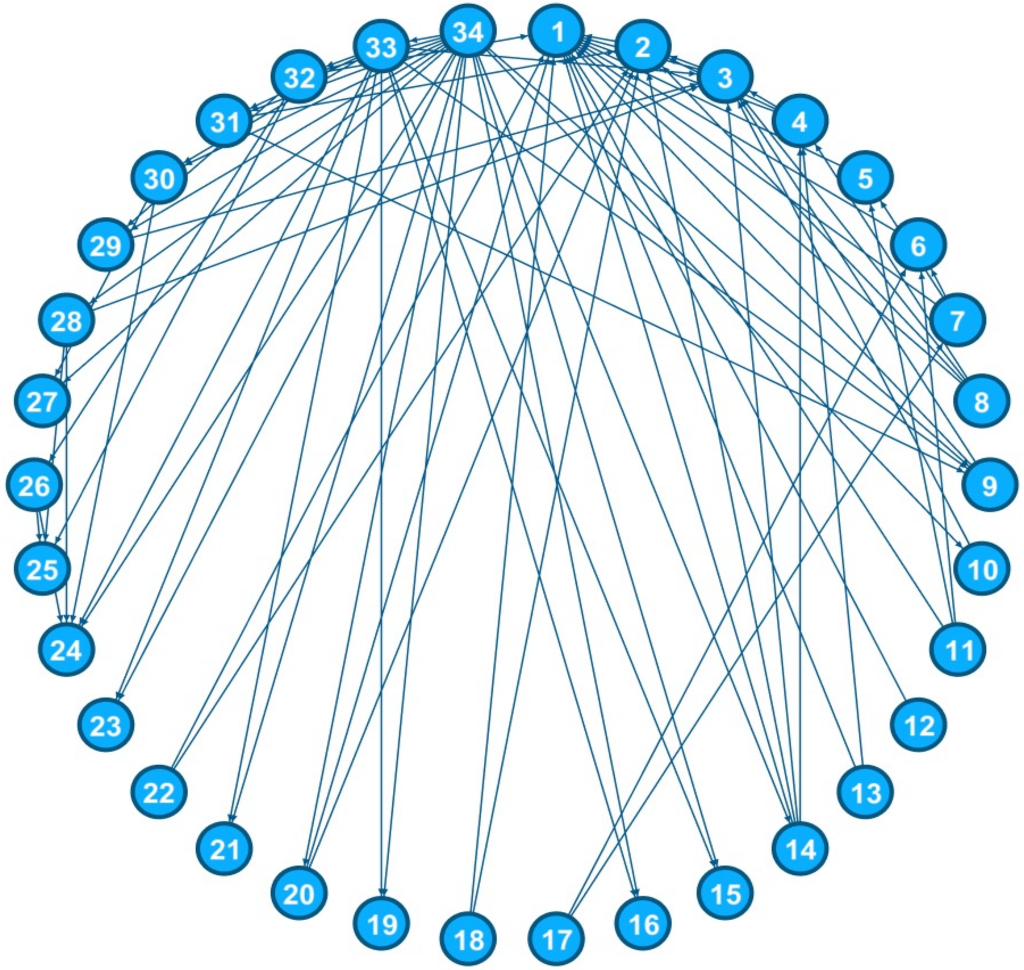
Example of Graph: Zachary’ s Karate Club

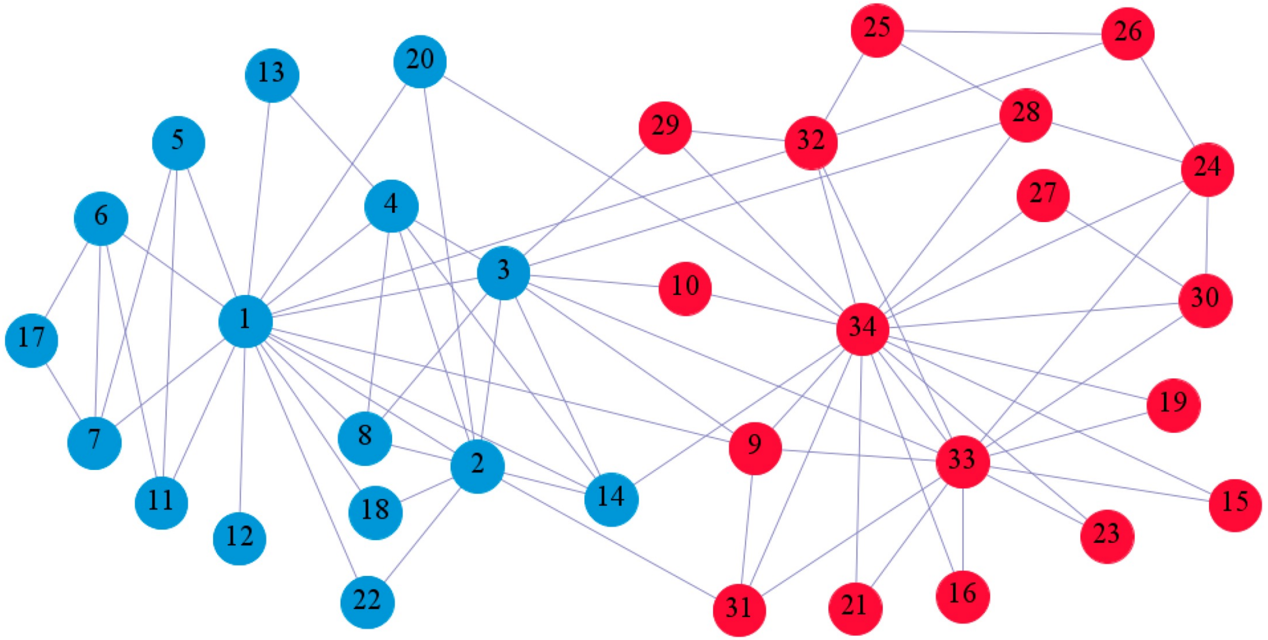


Vertex 1~34: 34 club members

Edge: two people interacted outside the club

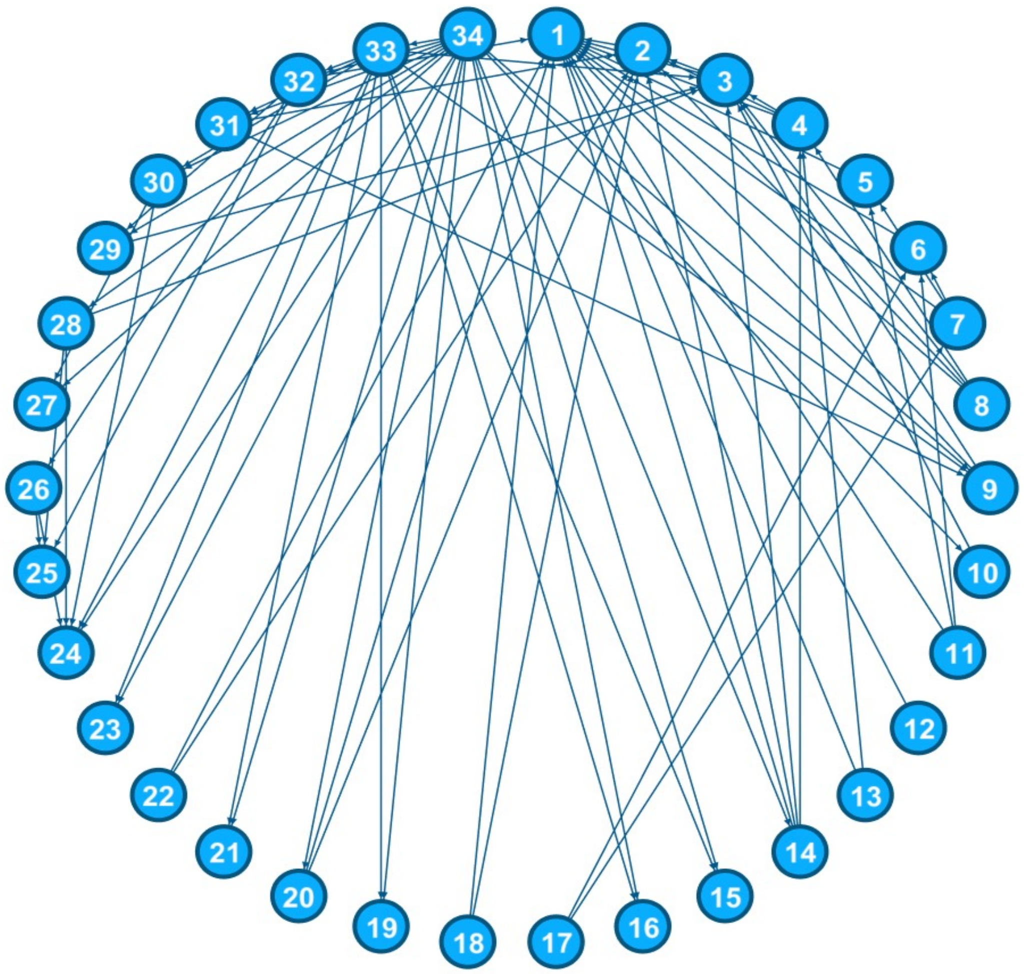
Example of Graph: Zachary's Karate Club





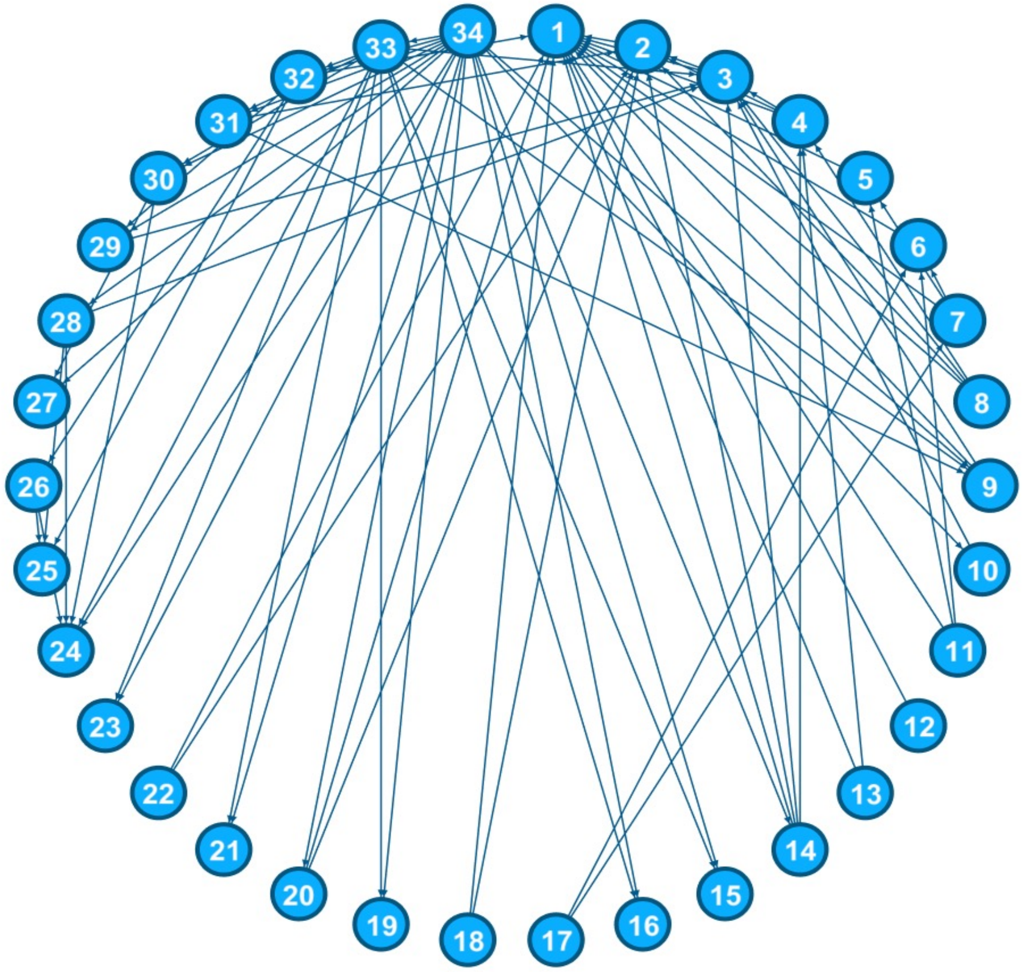
How to Label the Graph

- $v_1 = 1$
- $v_{34} = 0$
- Task: Define $v_2, \ldots, v_{33}$.



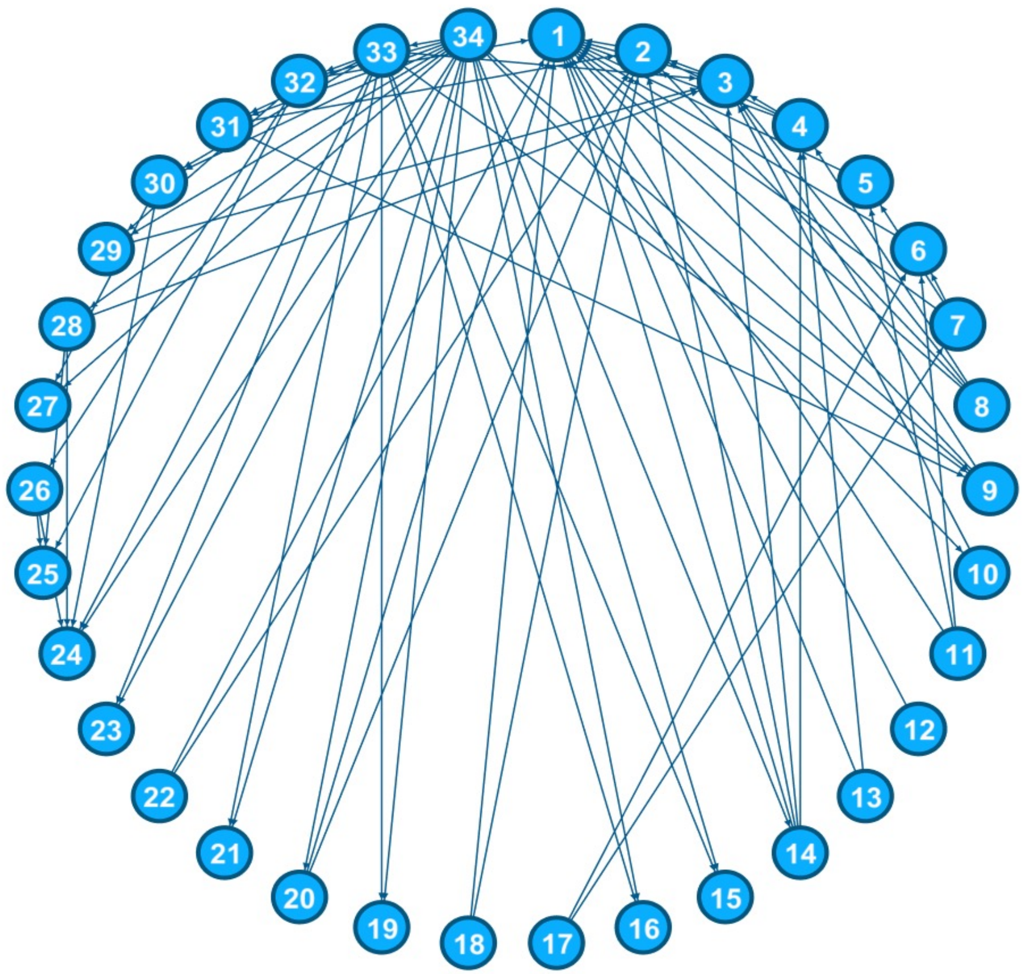
How to Label the Graph

- $v_1 = 1$
- $v_{34} = 0$
- Task: Define $v_2, \ldots, v_{33}$.
- $v_x =$ average of v_y for $y\sim\mathrm{x}$?



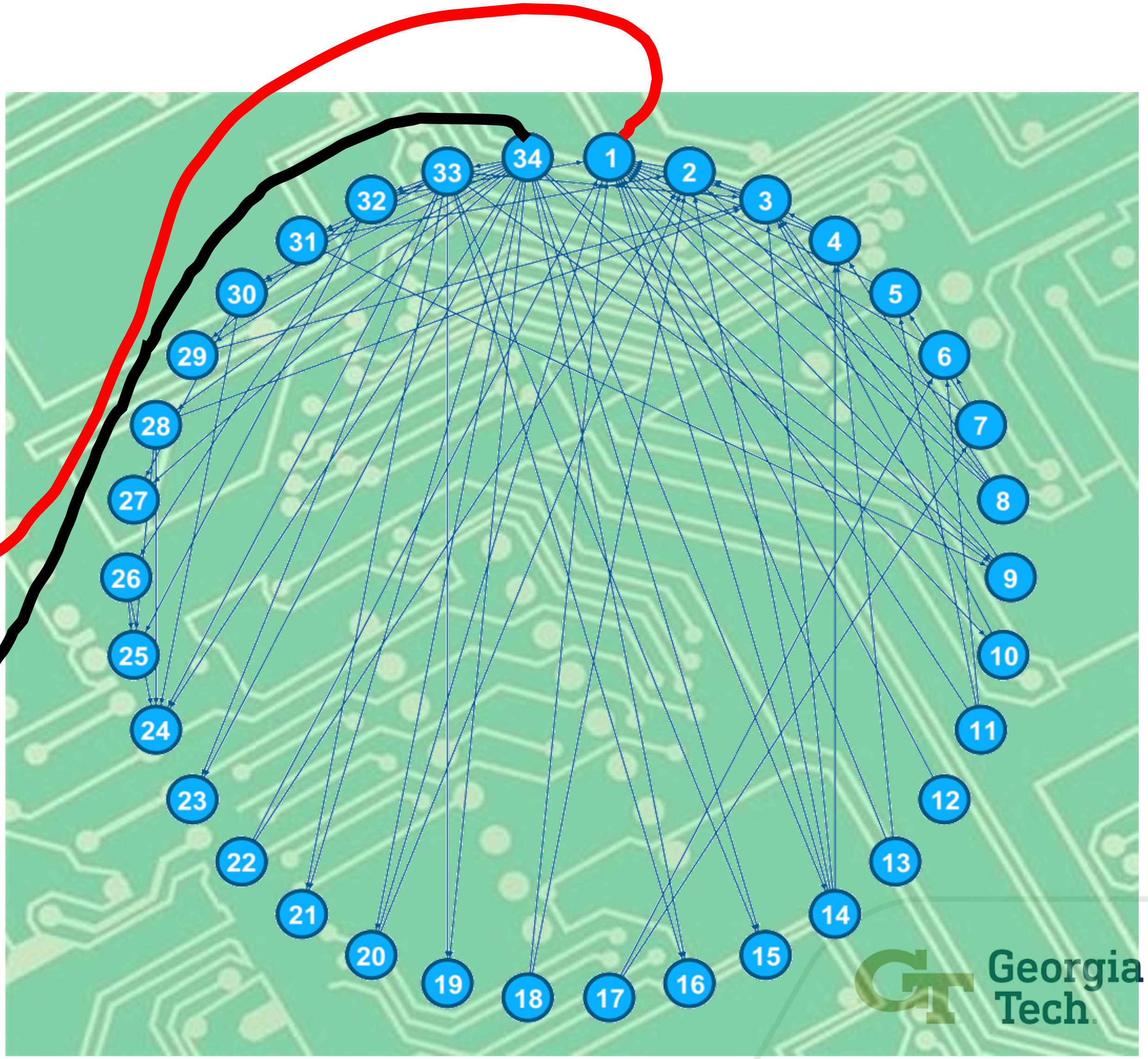
How to Label the Graph

- $v_1 = 1$
- $v_{34} = 0$
- $v_x =$ average of v_y for $y \sim \mathrm{x}$
- v: Electric potentials



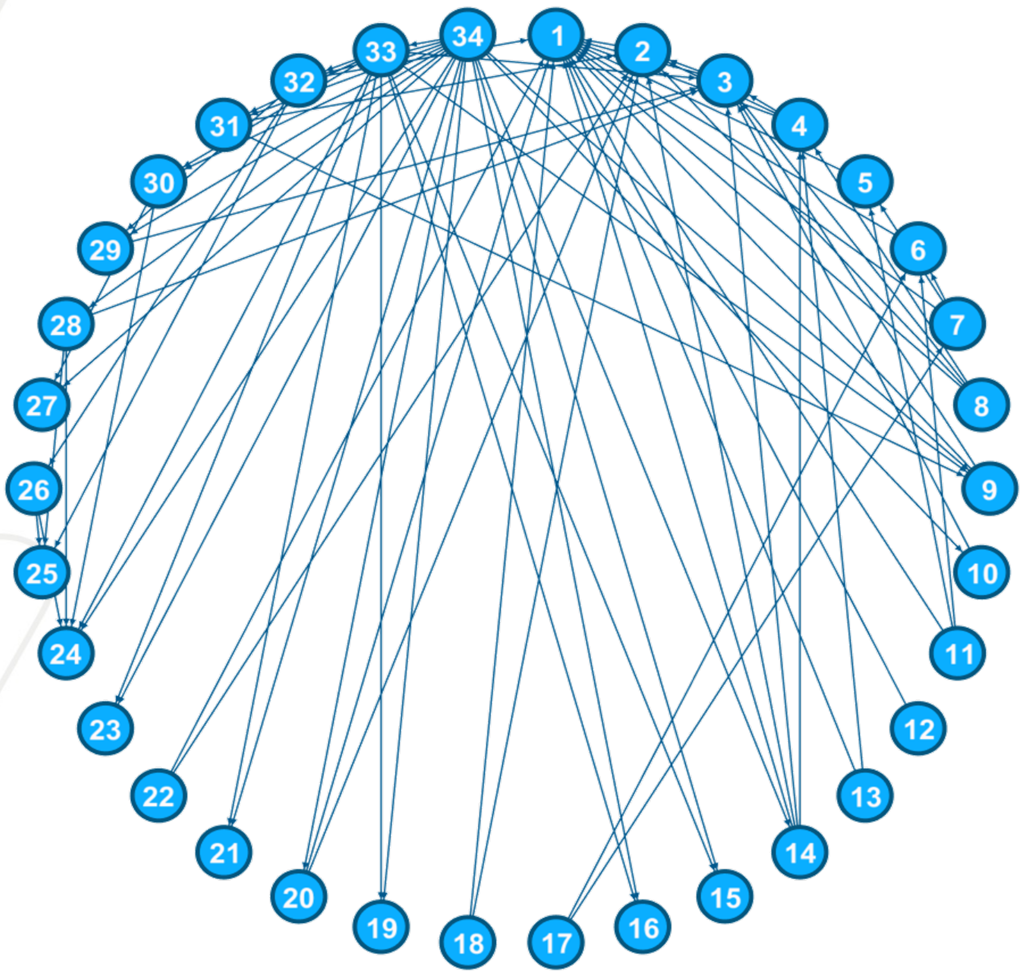
ɉia

Electric Flow from Instructor to Administrator



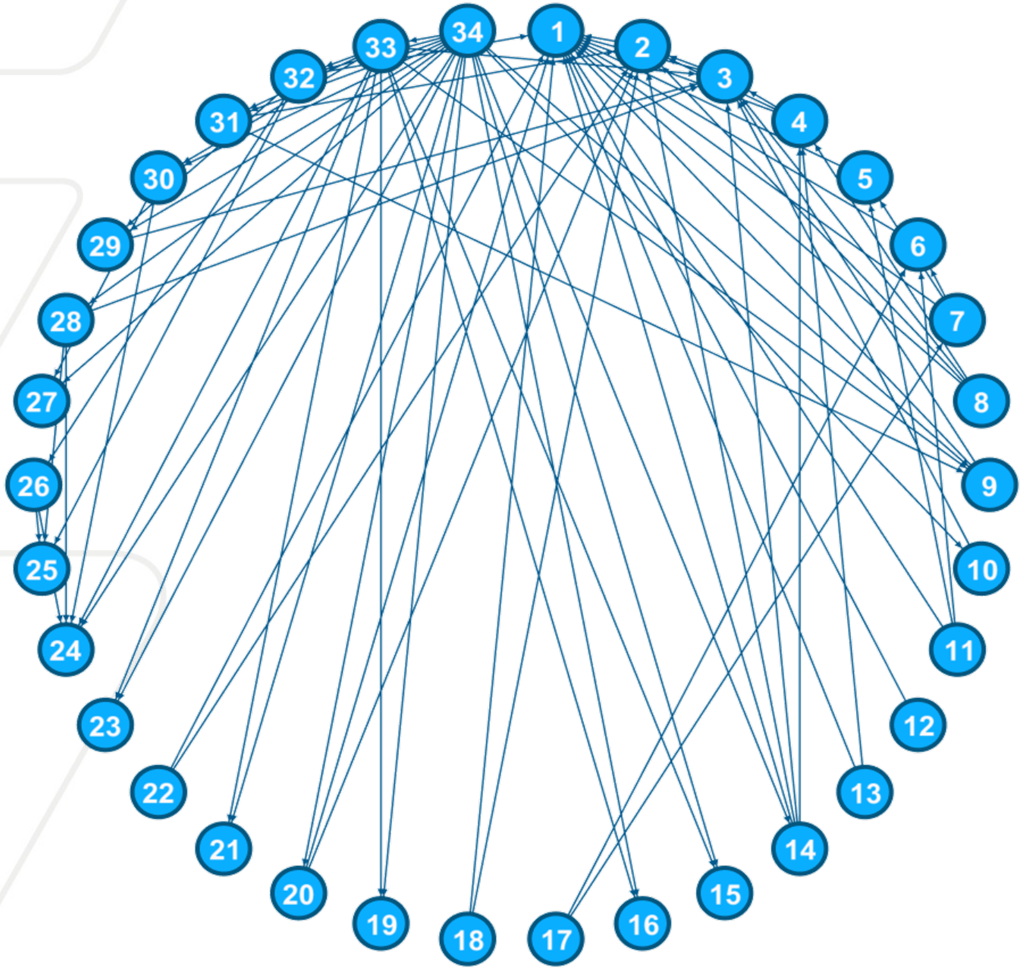
Vertices: Pins
Edges: Resistors connecting pins

Club Members Sorted by Labels



Person	Potential
34	0
27	0.052312
21	0.102497
19	0.102497
16	0.102497
15	0.102497
30	0.104625
24	0.161193
33	0.204993
23	0.204993
28	0.237501
10	0.255444
26	0.258846
25	0.277923
29	0.28277
31	0.323029
32	0.337422
9	0.407782
3	0.510887
20	0.559781
14	0.583623
2	0.679342
4	0.727888
8	0.729529
22	0.839671
18	0.839671
13	0.863944
17	1
11	1
7	1
6	1
5	1
12	1
1	1

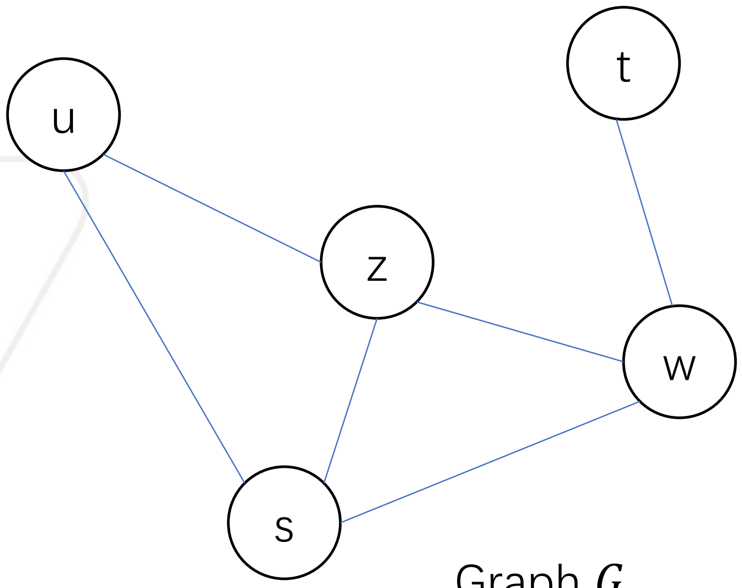
Verify the Result



Person	Potential	Outcome
34	0	2
27	0.052312	2
21	0.102497	2
19	0.102497	2
16	0.102497	2
15	0.102497	2
30	0.104625	2
24	0.161193	2
33	0.204993	2
23	0.204993	2
28	0.237501	2
10	0.255444	2
26	0.258846	2
25	0.277923	2
29	0.28277	2
31	0.323029	2
32	0.337422	2
9	0.407782	1
3	0.510887	1
20	0.559781	1
14	0.583623	1
2	0.679342	1
4	0.727888	1
8	0.729529	1
22	0.839671	1
18	0.839671	1
13	0.863944	1
17	1	1
11	1	1
7	1	1
6	1	1
5	1	1
12	1	1
1	1	1

Graph Laplacian: Solving Electric Flow

- $v_x = \frac{\sum_{y \sim x} v_y}{deg(x)}$, $\deg(x)$: degree of x
- $\deg(x)\, v_x - \sum_{y \sim x} v_y = 0$



Graph G

	s	t	u	z	w
s	3	0	-1	-1	-1
t	0	1	0	0	-1
u	-1	0	2	-1	0
z	-1	0	-1	3	-1
w	-1	-1	0	-1	3

$$\cdot \begin{bmatrix} \mathrm{v_s} = 1 \\ \mathrm{v_t} = -1 \\ v_u \\ v_z \\ v_w \end{bmatrix} = \begin{bmatrix} d \\ -\mathrm{d} \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

voltages demands

Graph Laplacian
$\mathrm{L}(G) = D(G) - A(G)$

Classical and Theoretical Applications

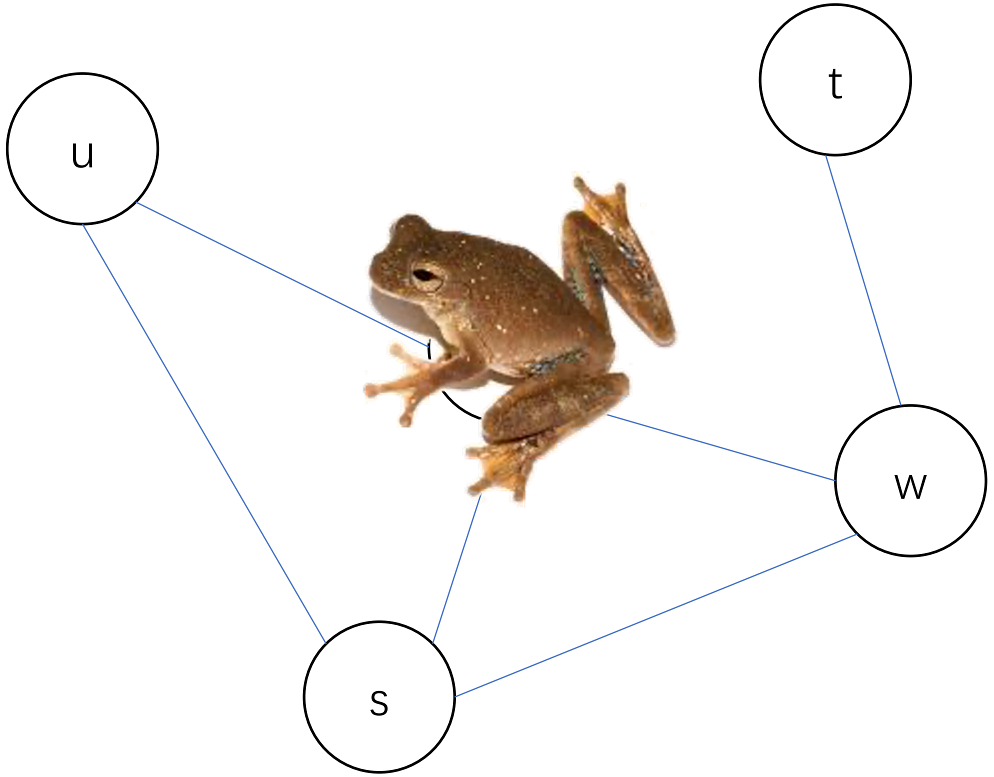
- Semi-supervised learning in larger social networks

Laplacian Regularization term [Zhu, Ghahramani, Lafferty ICML ’03]

- Graph clustering
- Network flows (maxflow, mincost flow…)

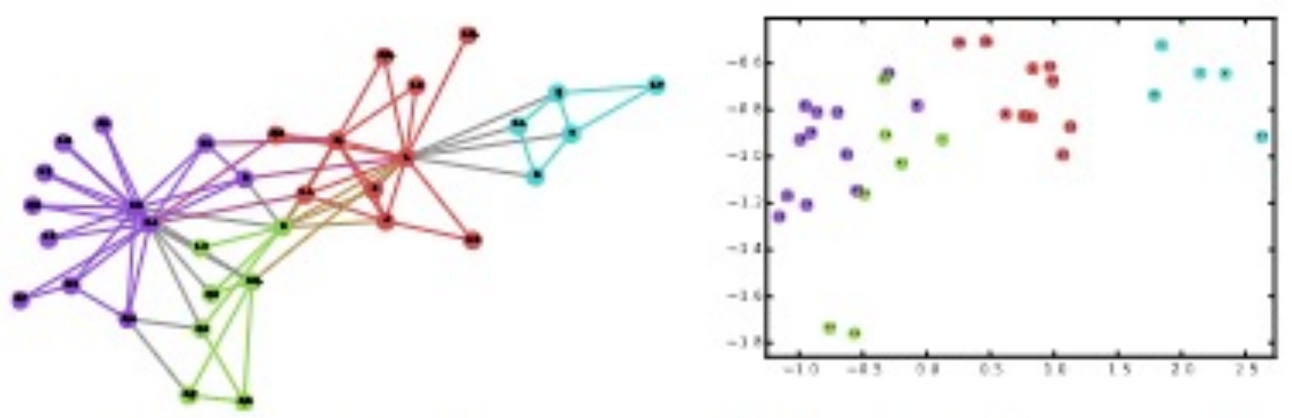
Sparsifying random walk matrices

- [Perozzi-Al-Rfou-Skiena KDD’ 14] DeepWalk
- Learns embeddings of a graph by short random walks



Sparsifying random walk matrices

- [Perozzi-Al-Rfou-Skiena KDD' 14] DeepWalk
- Learns embeddings of a graph by short random walks



(a) Input: Karate Graph (b) Output: Representation

Sparsifying random walk matrices

- [Cheng-Cheng-Liu-Peng-Teng COLT' 15] Sparsifying random walk matrices:

Theorem [CCLPT' 15]: All length-T random walks in a graph can be sparsified in $\tilde{O}(T^2 m)$ time.

Georgia Tech

Sparsifying random walk matrices

- [Qiu-Dong-Ma-Li-Wang-Wang WWW' 19] NetSMF: Large-Scale Network Embedding as Sparse Matrix Factorization
- 24 hours to generate embeddings of the OAG dataset (895,368,962 edges)
- Best paper in WWW' 19

Graph Laplacian

- Found in machine learning, network science, scientific computing, ⋯
- Can be solved in nearly-linear-time by Spielman-Teng

Spectral sparsification of graphs. *SIAM J. Computing* 40:981-1025, 2011.

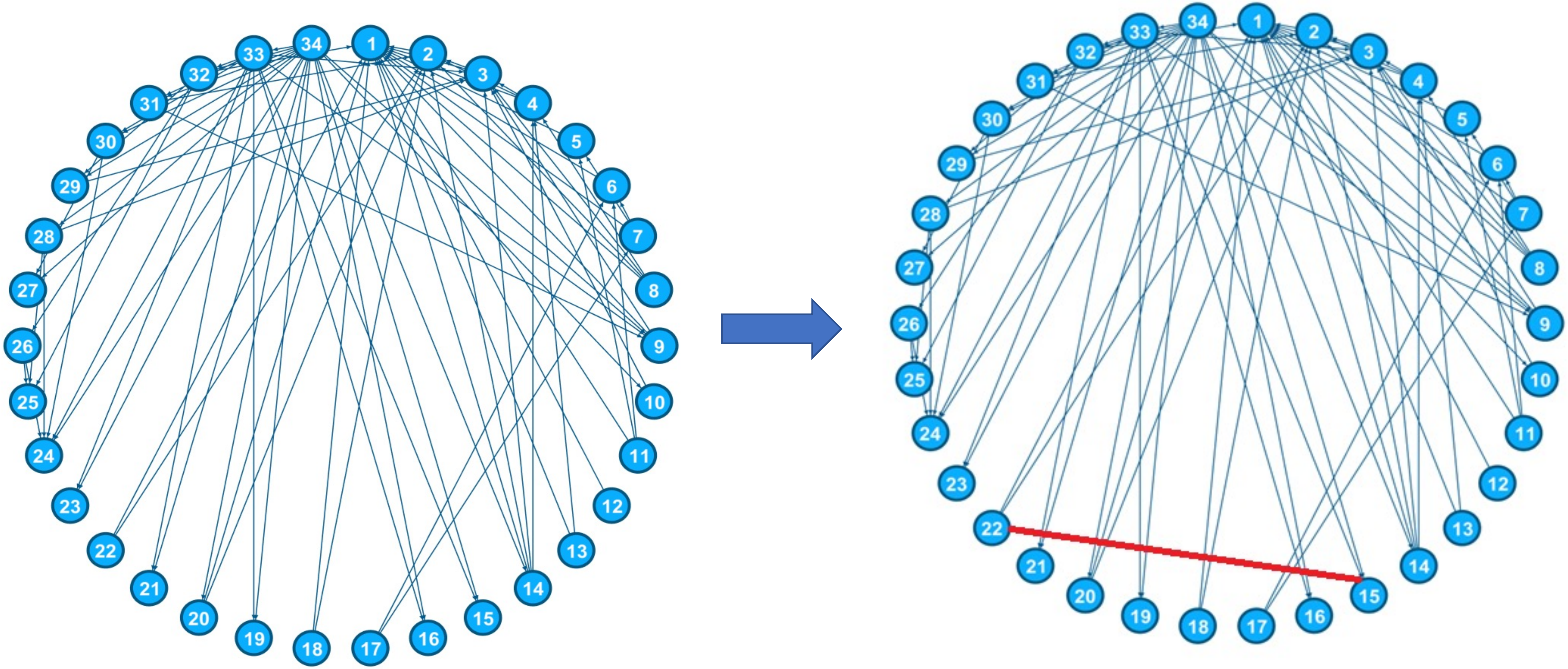
A local clustering algorithm for massive graphs and its application to nearly linear time graph partitioning. *SIAM J. Computing* 42:1-26, 2013.

Nearly linear time algorithms for preconditioning and solving symmetric, diagonally dominant linear systems. *SIAM J. Matrix Anal. Appl.* 35:835-885, 2014.

Their works on nearly-linear-time Laplacian solvers resolved an outstanding open problem in numerical linear algebra: solving symmetric diagonally dominant linear systems in nearly linear time. This result delivered a new and extremely powerful algorithmic primitive: nearly linear time electrical flow computations.

- Karate Club and Graph Laplacian
- Dynamic Laplacian by Schur Complement
- Dynamic Laplacian for Planar Graphs
- Dynamic Laplacian for General Graphs

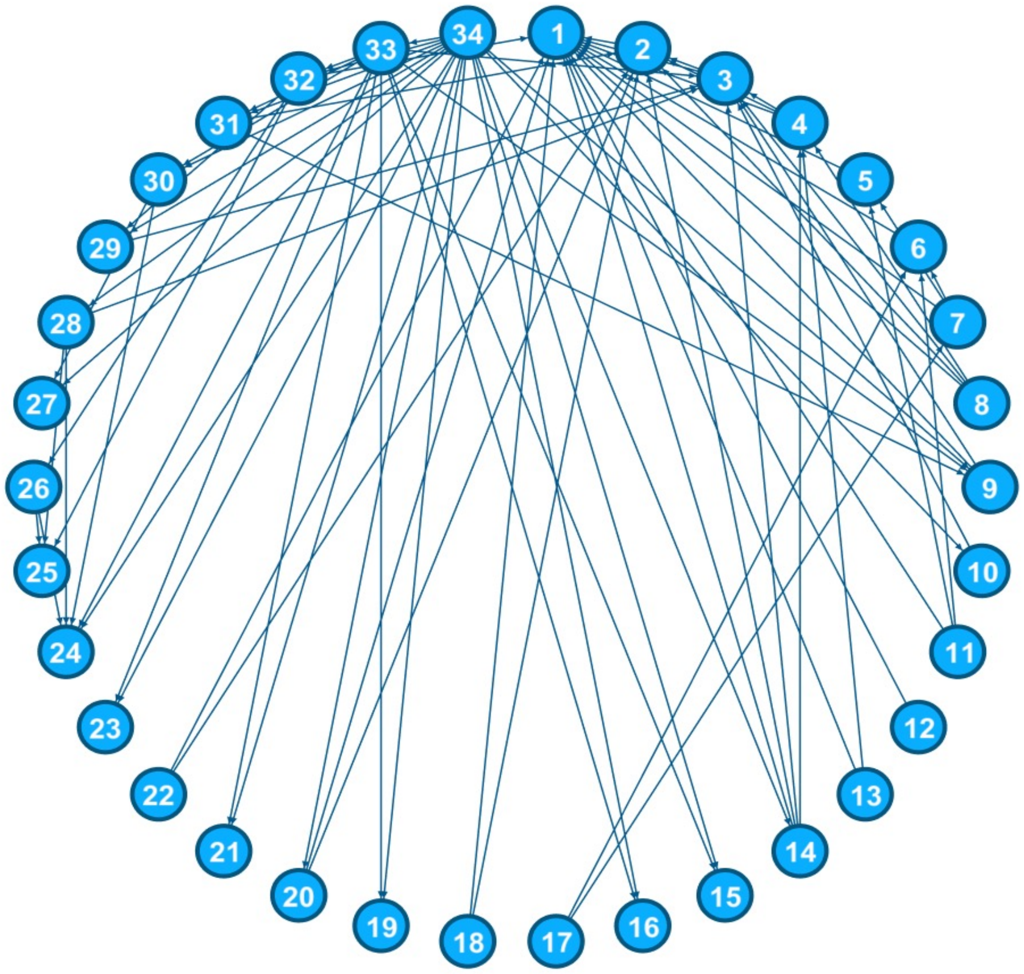
Dynamic Laplacian



Dynamic Laplacian

- A graph G
- Update: Add or delete an edge
- Query: Output electric potential of a vertex

(We can also support outputting electric flow on some edge, outputting vertices with large potential changes, …)

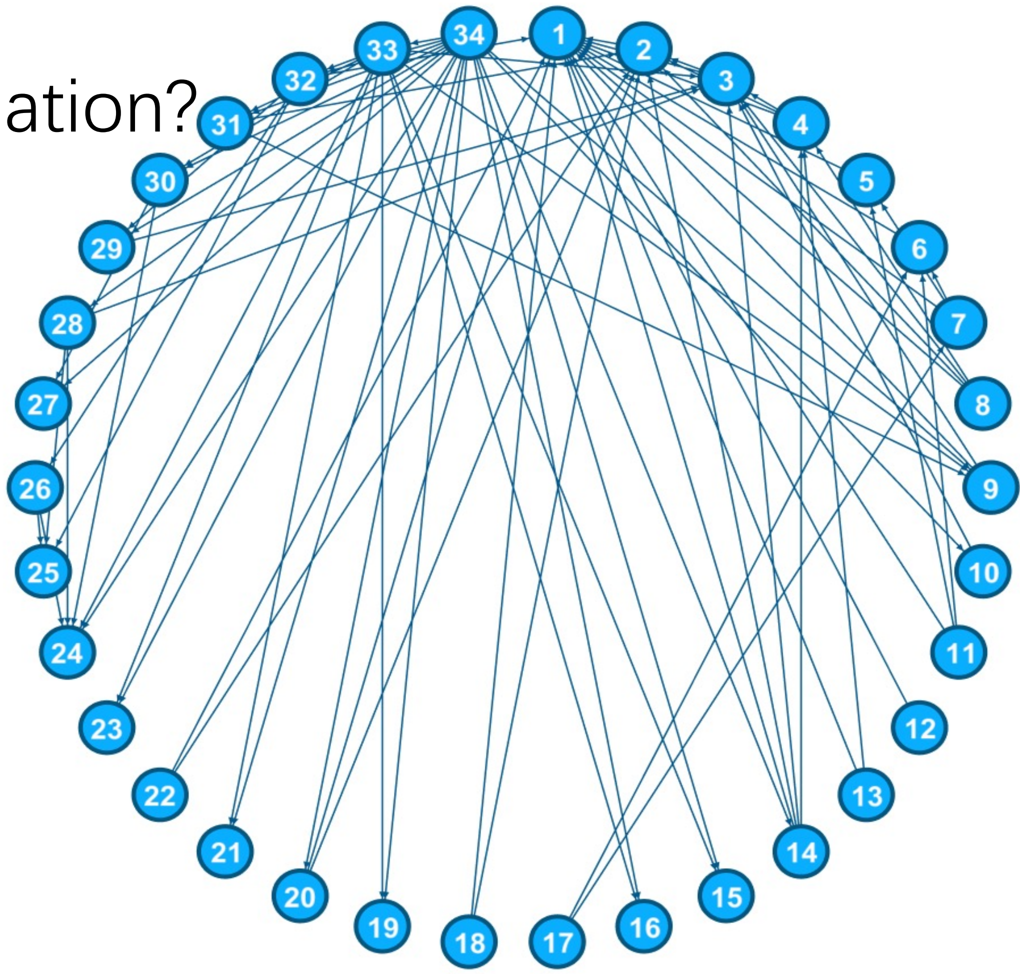


Dynamic Laplacian

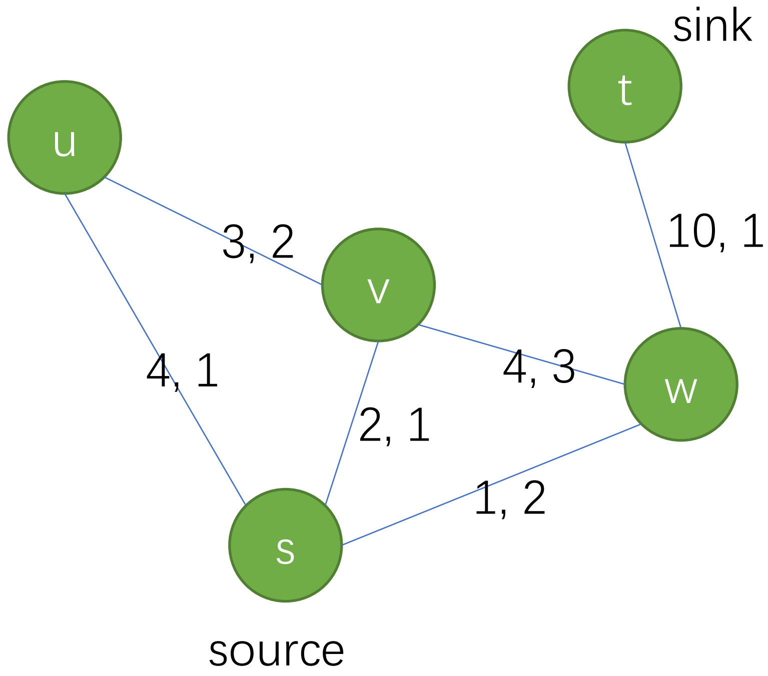
- Update labels when the graph changes
- Social representation w. temporal information?
- Network flow problems:

  Maximum flow

  Minimum cost flow



Application: Planar mincost flow



- Given
  - Graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$
  - Capacities of the edges
  - Costs of the edges
  - A source and a sink

- Q: How many units of flow can we send from source to sink? What is the minimum cost of it?

Georgia Tech

Application: Planar mincost flow

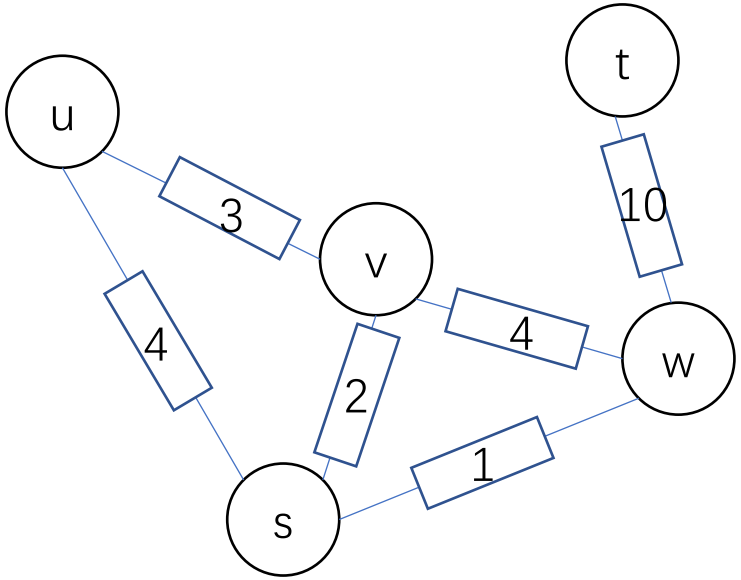
- Theorem [DGGLPSY' 21]: Let G be a planar graph with n edges. Assume all demands, costs and capacities are bounded by M. ∃ Algorithm computes a minimum cost flow in $O\left(n \log^{O(1)} n \log M\right)$ time.
- Previously, the best planar mincost flow algorithm is the $O\left(n^{1.5} \log^{O(1)} n \log^2 M\right)$ algorithm for all (planar and nonplanar) graphs.

Schur Complement -- Elimination

- $Lx = b$
- $L = \begin{bmatrix} L_{FF} & L_{FC} \\ L_{CF} & L_{CC} \end{bmatrix}$
- $Sc(L, C) = L_{CC} - L_{CF} L_{FF}^{-1} L_{FC}$
- If $b_F = 0$, $Sc(L, C) x_C = b_C$

Georgia Tech

Graph Laplacian – Electric Network



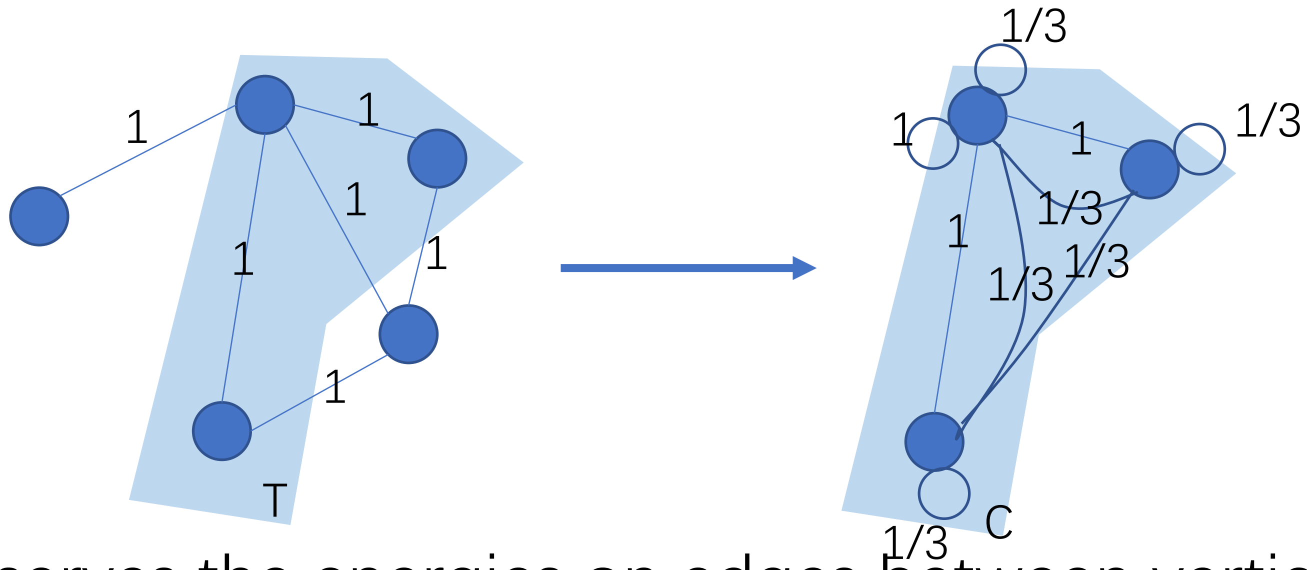
- Edge uv: conductance w_{uv}

  (resistance $r_{uv} = 1/w_{uv}$)
- Vertex v: potential ϕ_v
- Edge orientation $u \to v$: current flow $C_{u\to v}$
- Kirchnoff's Law:

  $\forall$ vertex v, flow-in = flow-out
- Ohm's Law: $\forall$ edge uv, $C_{u\to v} = \frac{\phi_u - \phi_v}{r_{uv}}$

Schur Complement – Equivalent Electric Network

- Let $\mathbf{C}$ be a subset of vertices. Suppose we only care about energies of edges in $\mathbf{C}$.



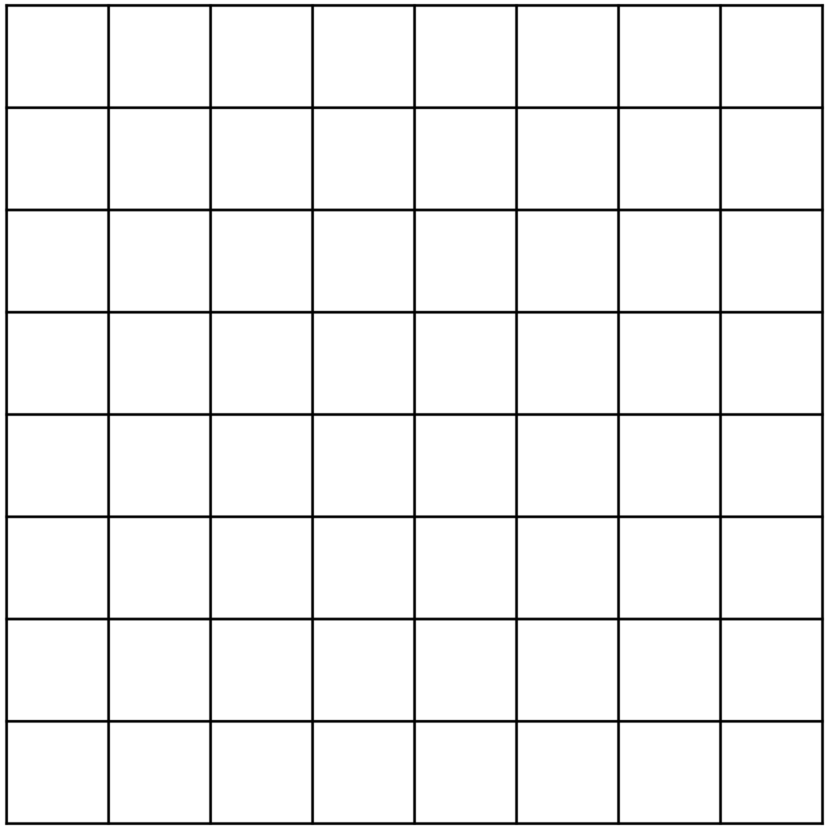
- $\mathbf{Sc(G, C)}$ preserves the energies on edges between vertices in $\mathbf{C}$
- **$\mathbf{SC(G, C)}$ is still a graph!**

- Karate Club and Graph Laplacian
- Dynamic Laplacian by Schur Complement
- Dynamic Laplacian for Planar Graphs (By separator tree)
- Dynamic Laplacian for General Graphs

Schur complements on planar graph

- Planar graph G
- Update an edge
- Query vertex potentials on the boundary

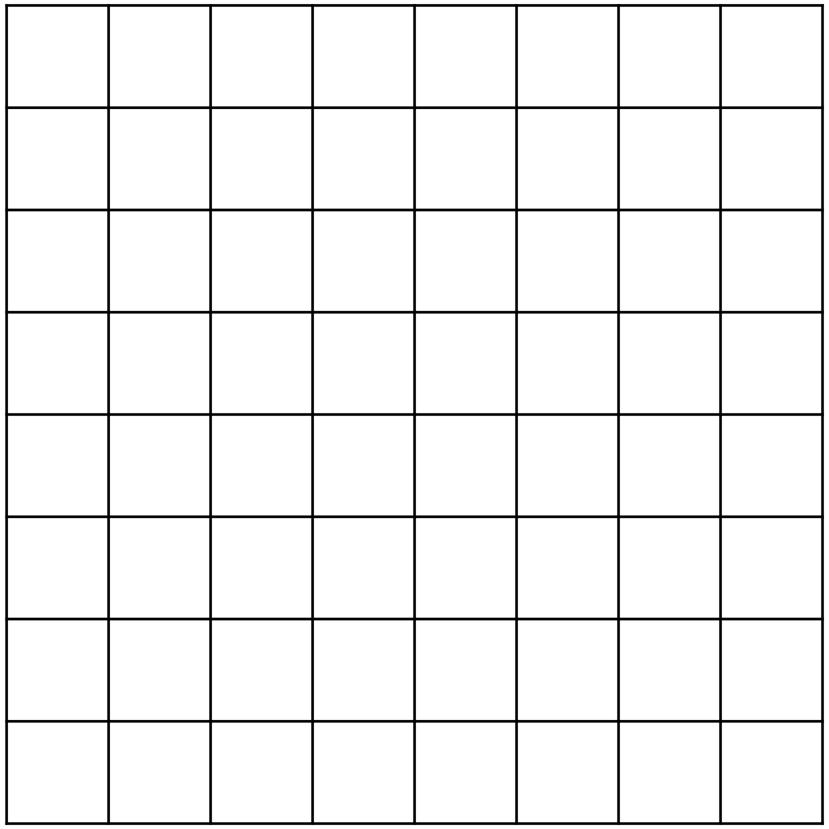
- Schur complement of G onto the boundary vertices



A Separator Theorem

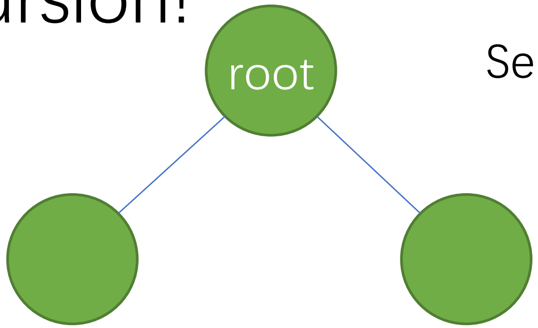
- Planar graph can be separated evenly by $\sqrt{n}$ nodes

- Theorem [Ungar' 51, Lipton-Tarjan' 79] $\exists\ O(\sqrt{n})$ vertices s.t. removing them partitions a planar graph into disjoint subgraphs with at most $2n/3$ vertices each.

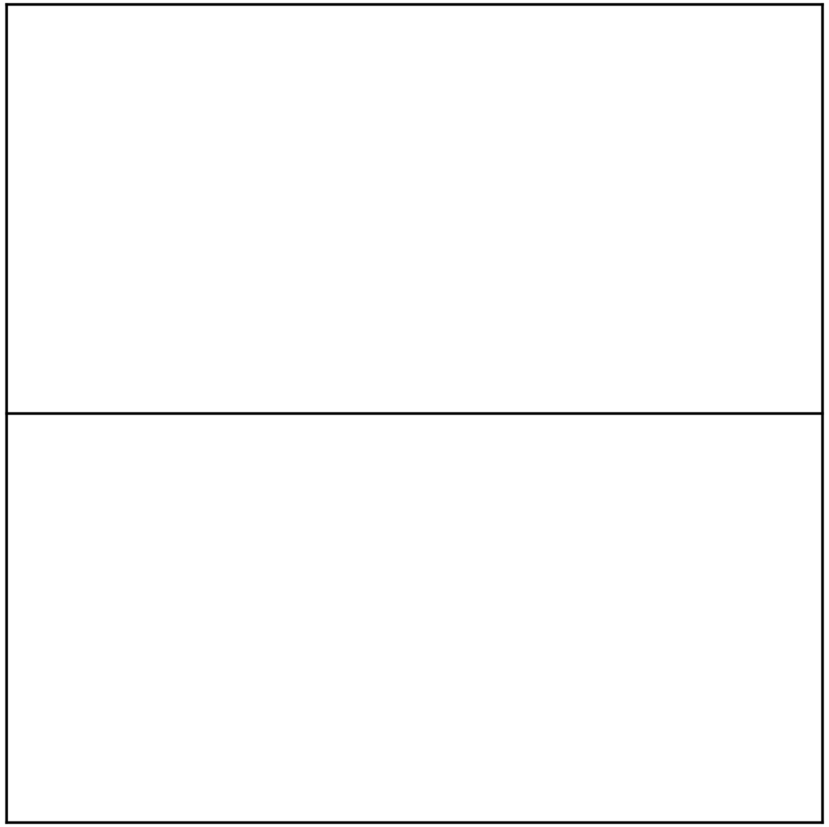


Recursively Partition the Graph

- Each region is still a planar graph
- Recursion!

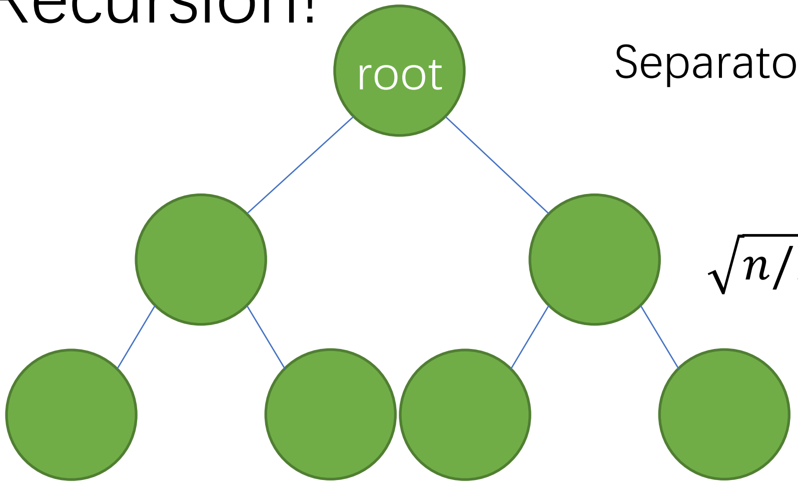


Separator size: $\sqrt{n}$



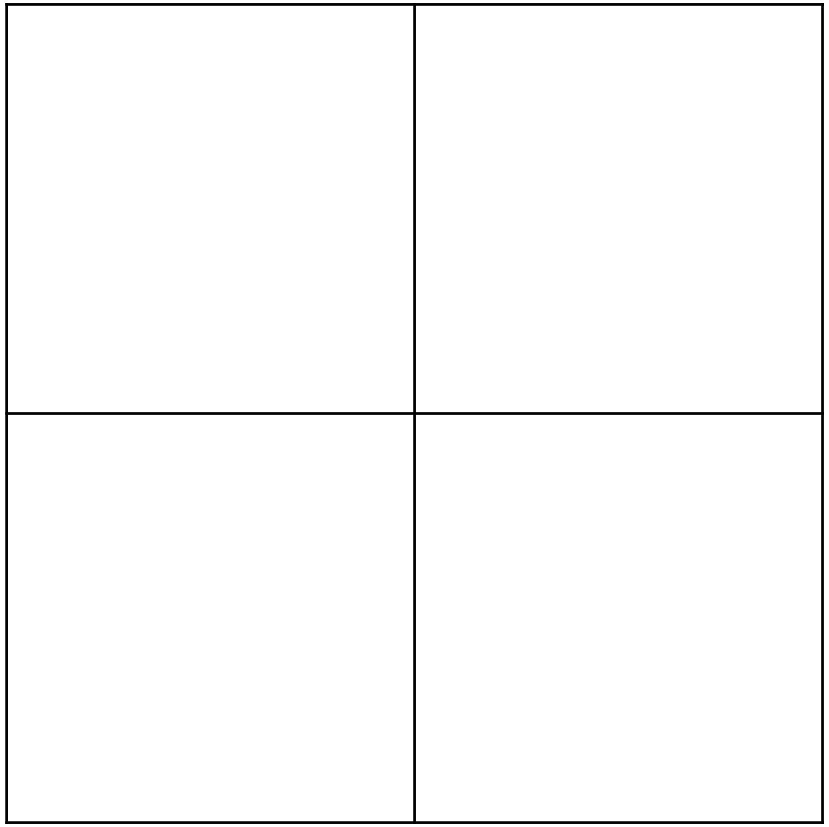
Recursively Partition the Graph

- Each region is still a planar graph
- Recursion!



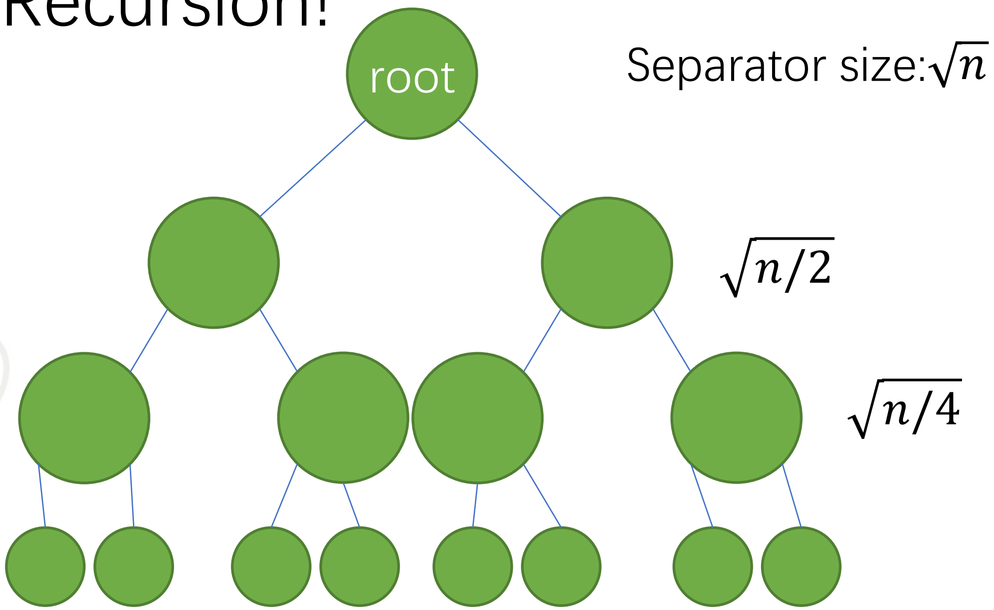
Separator size: $\sqrt{n}$

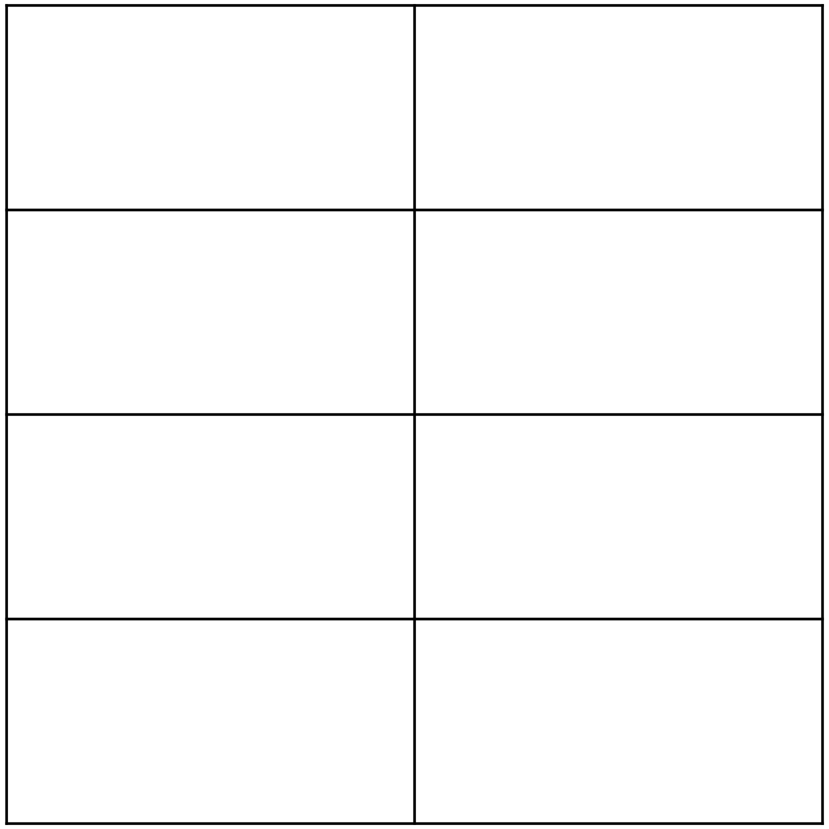
$\sqrt{n/2}$



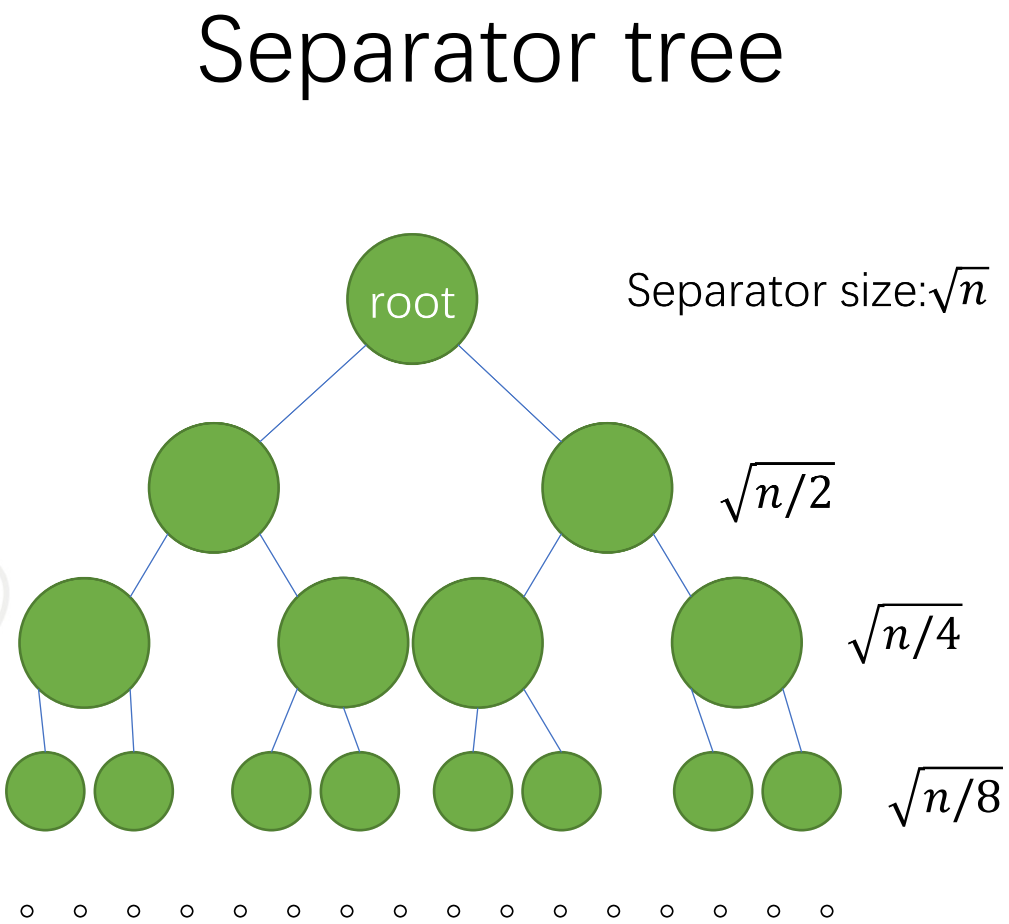
Recursively Partition the Graph

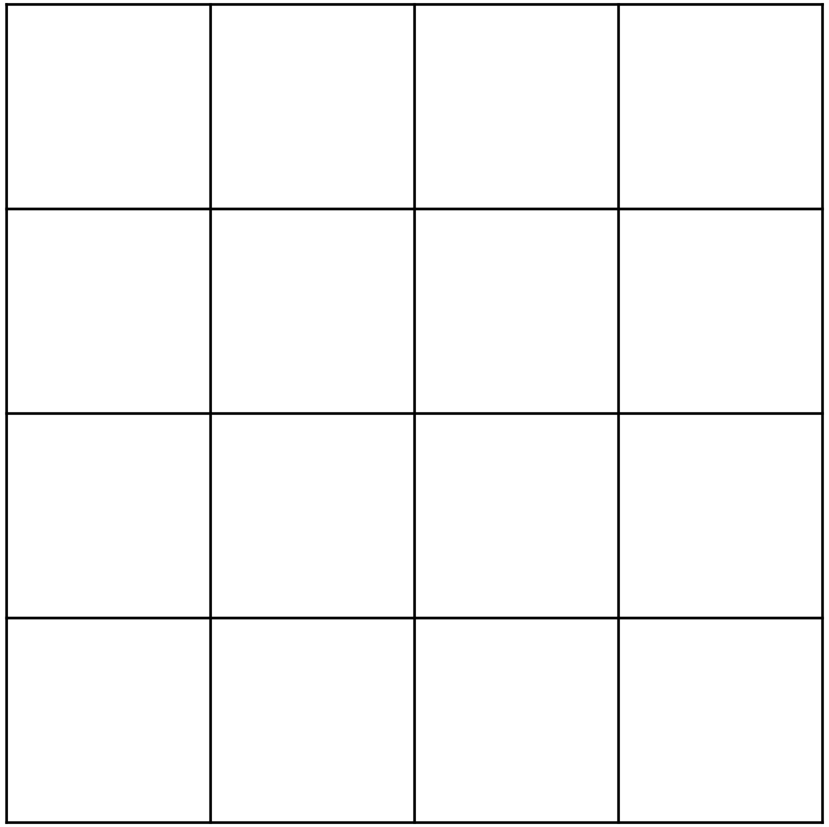
- Each region is still a planar graph
- Recursion!





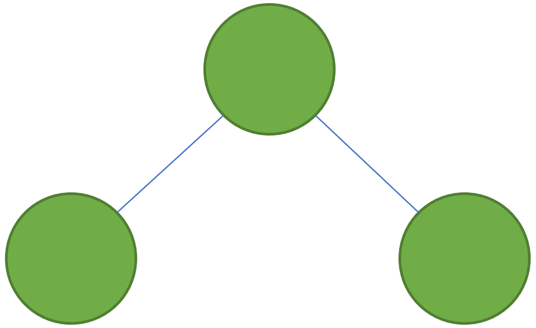
Recursively Partition the Graph

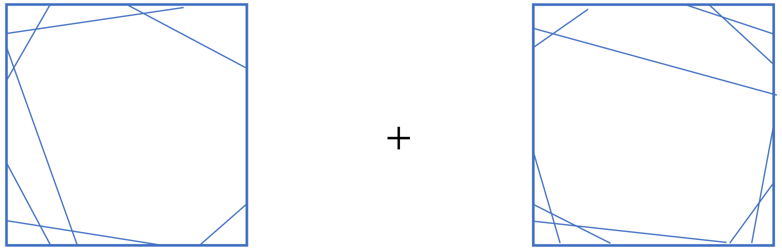




Schur Complement Formula

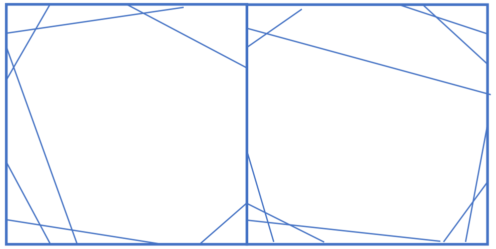
- $Sc(H, \delta H) = Sc\big(Sc\big(L(H), \delta L(H)\big) + Sc\big(R(H), \delta R(H)\big), \delta H\big)$





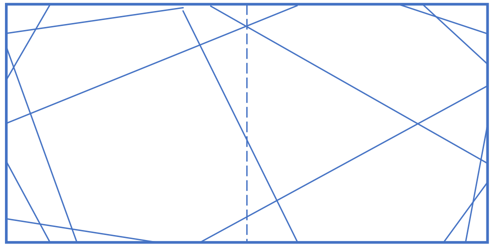
$Sc\big(L(H), \delta L(H)\big)$ $+$ $Sc\big(R(H), \delta R(H)\big)$

$=$



$Sc\big(L(H), \delta L(H)\big) + Sc\big(R(H), \delta R(H)\big)$

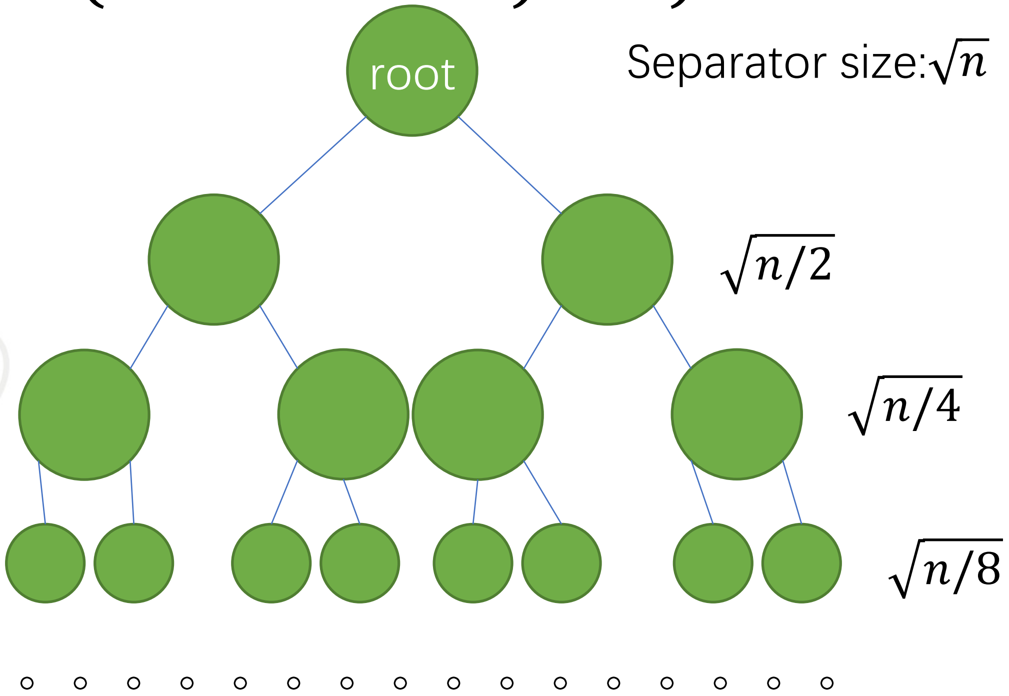
$\Rightarrow$

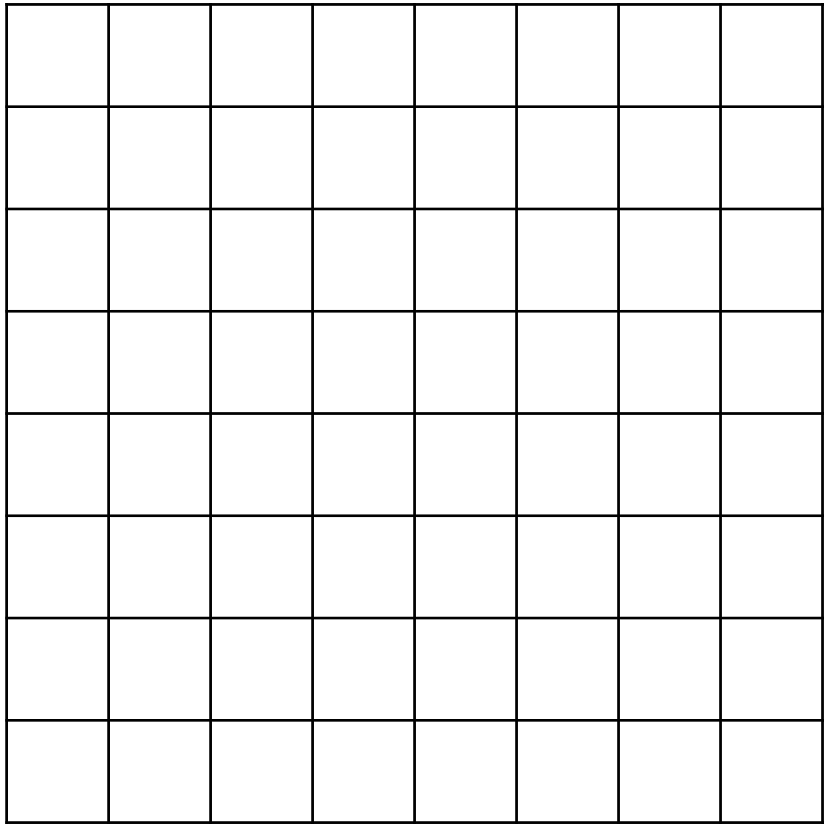


$Sc(H, \delta H)$

Update on the Separator Tree

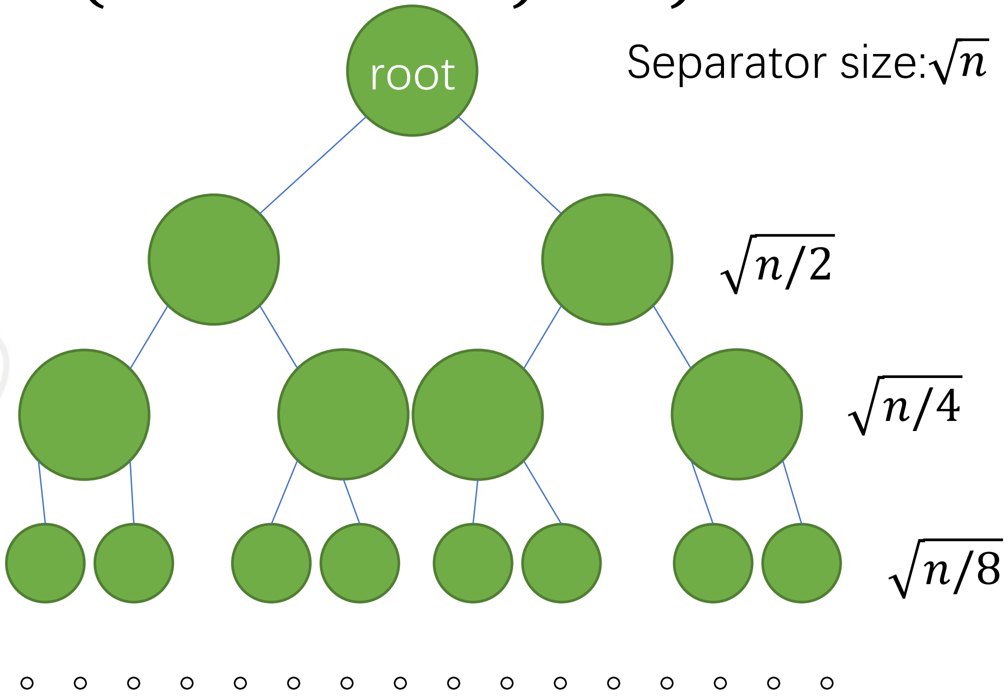
- $Sc(H, \delta H) = Sc\Big(Sc\big(L(H), \delta L(H)\big) + Sc\big(R(H), \delta R(H)\big), \delta H\Big)$

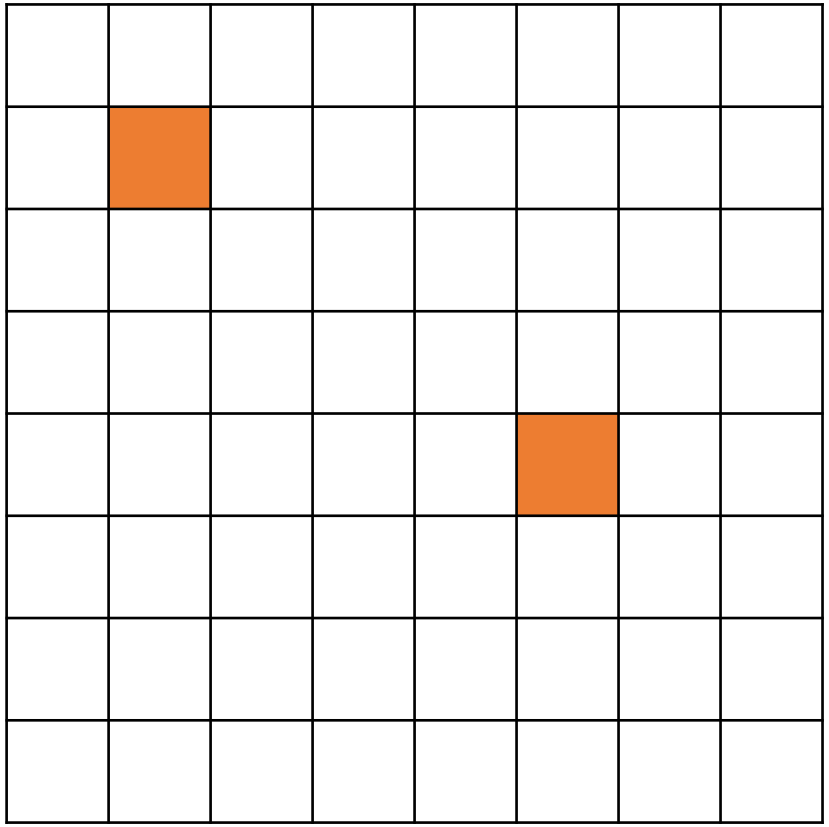




Update on the Separator Tree

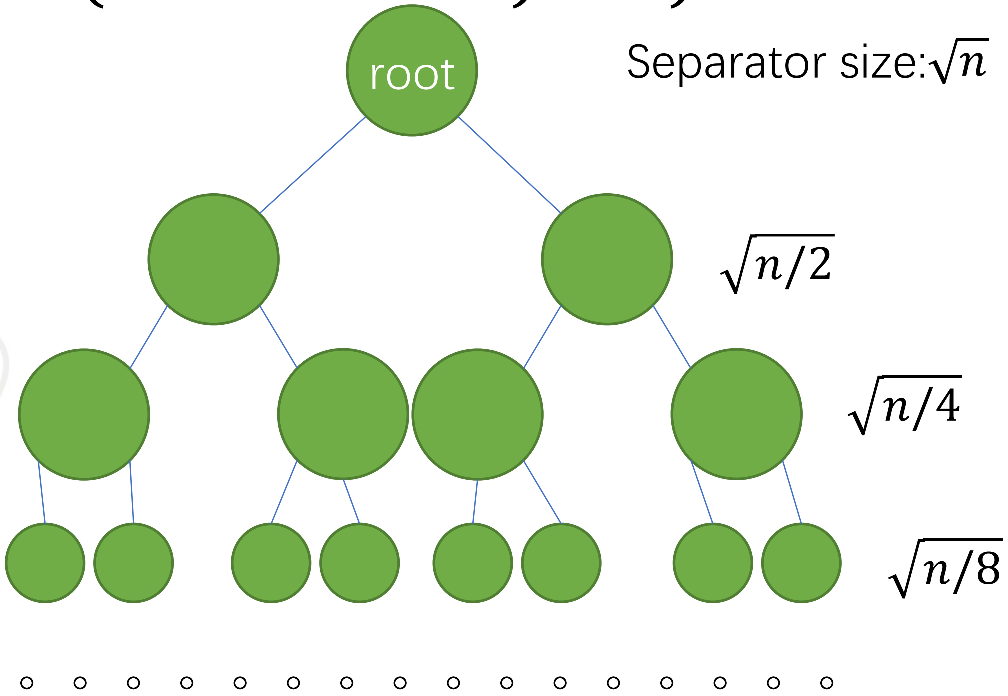
- $Sc(H, \delta H) = Sc\big(Sc\big(L(H), \delta L(H)\big) + Sc\big(R(H), \delta R(H)\big), \delta H\big)$

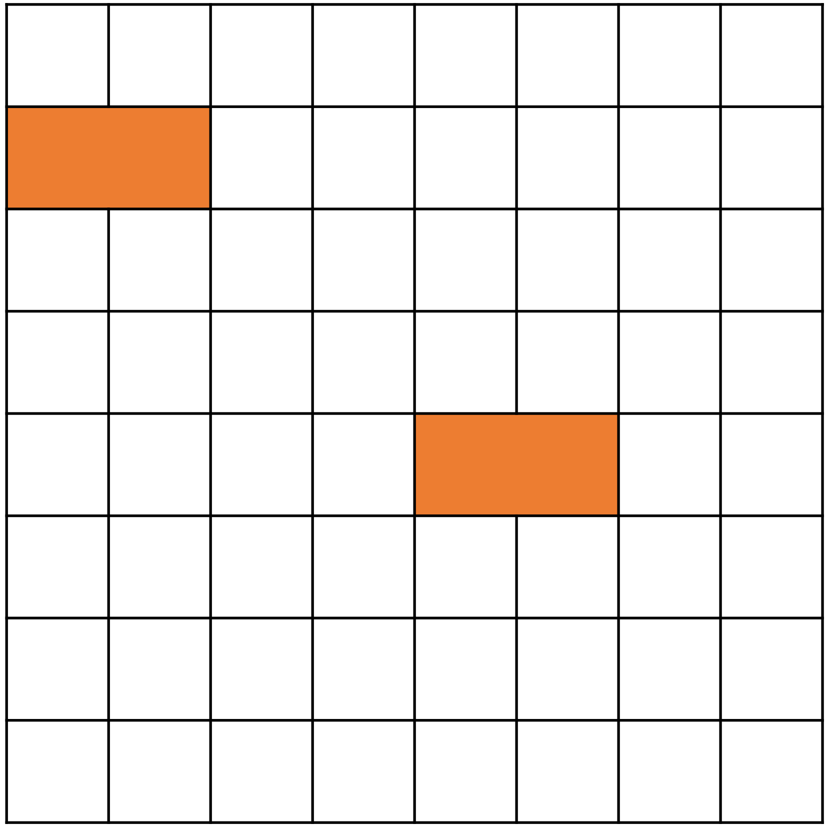




Update on the Separator Tree

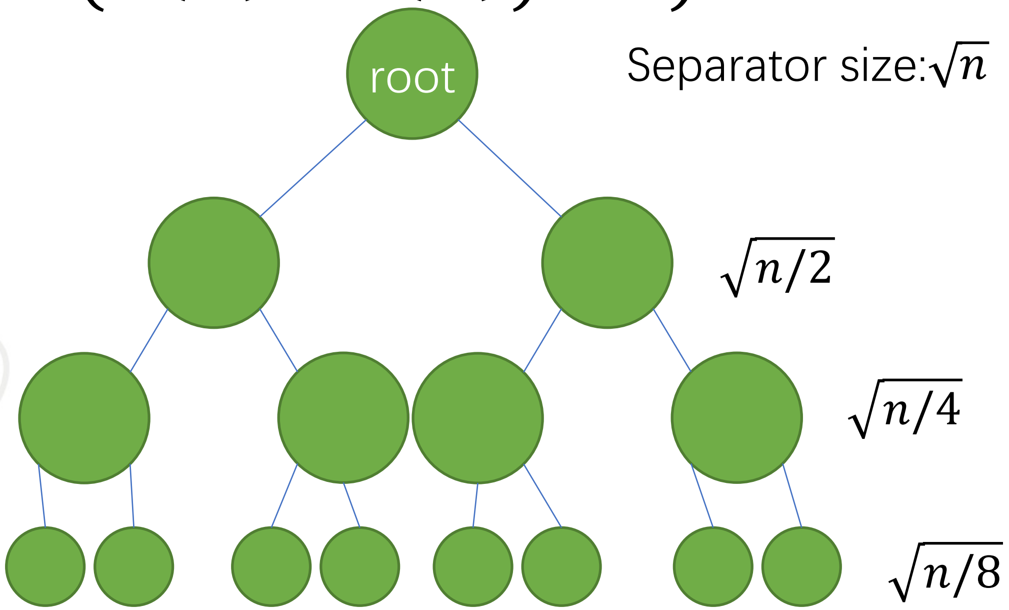
- $Sc(H, \delta H) = Sc\big(Sc\big(L(H), \delta L(H)\big) + Sc\big(R(H), \delta R(H)\big), \delta H\big)$

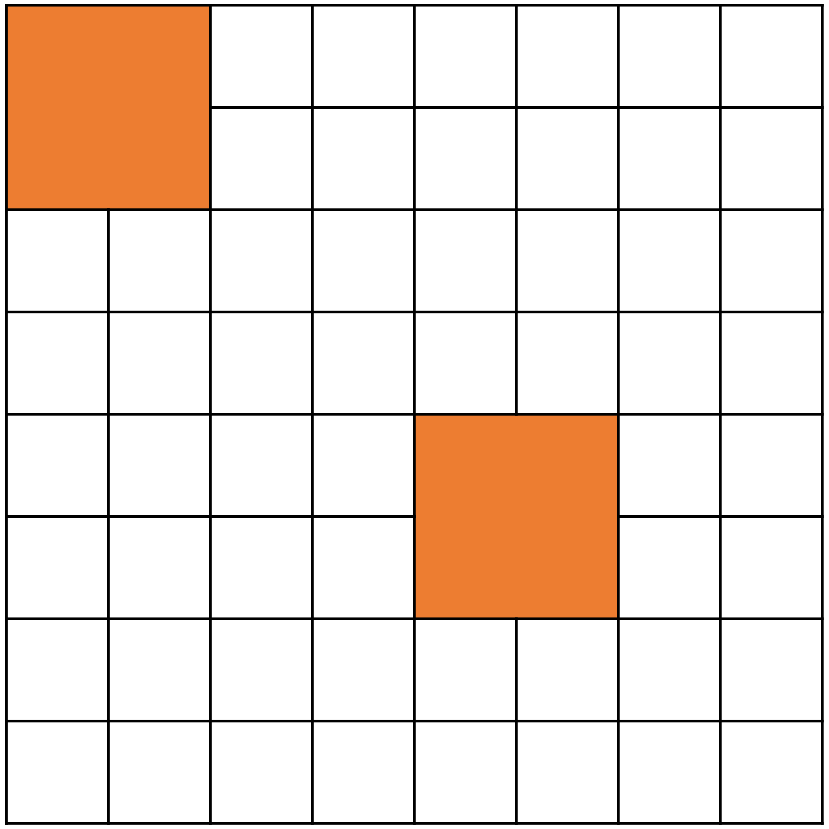




Update on the Separator Tree

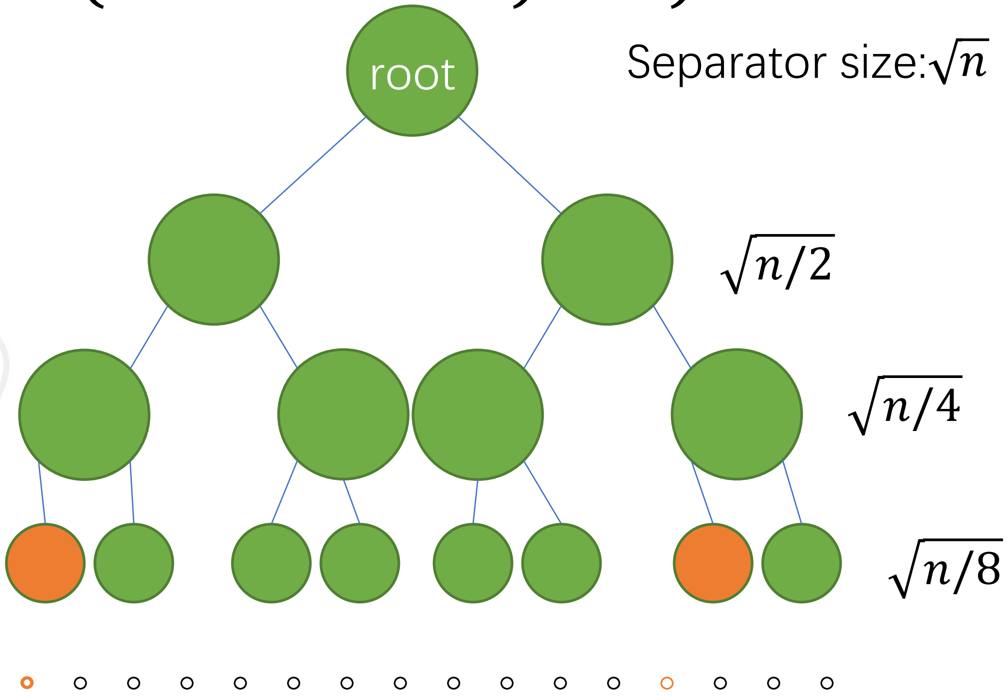
- $Sc(H, \delta H) = Sc\Big(Sc\big(L(H), \delta L(H)\big) + Sc\big(R(H), \delta R(H)\big), \delta H\Big)$

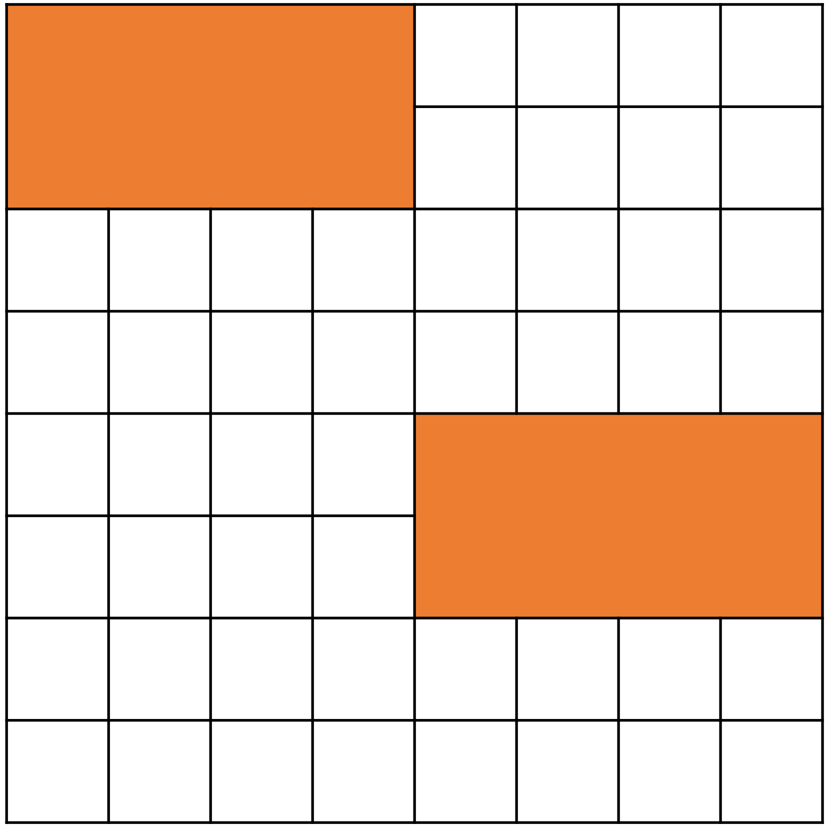




Update on the Separator Tree

- $Sc(H, \delta H) = Sc\big(Sc\big(L(H), \delta L(H)\big) + Sc\big(R(H), \delta R(H)\big), \delta H\big)$

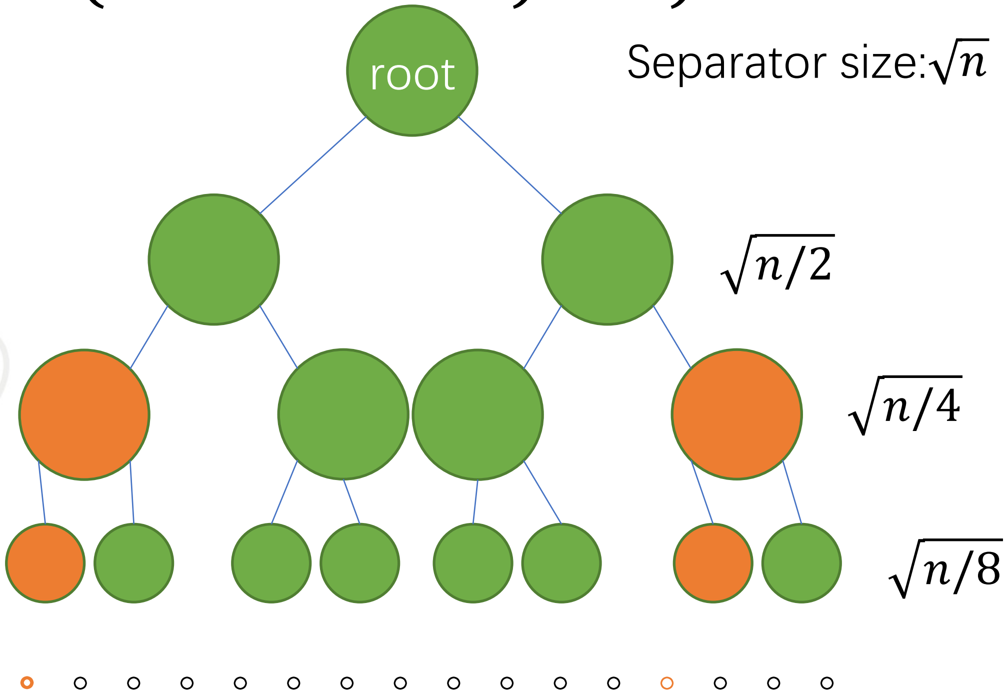


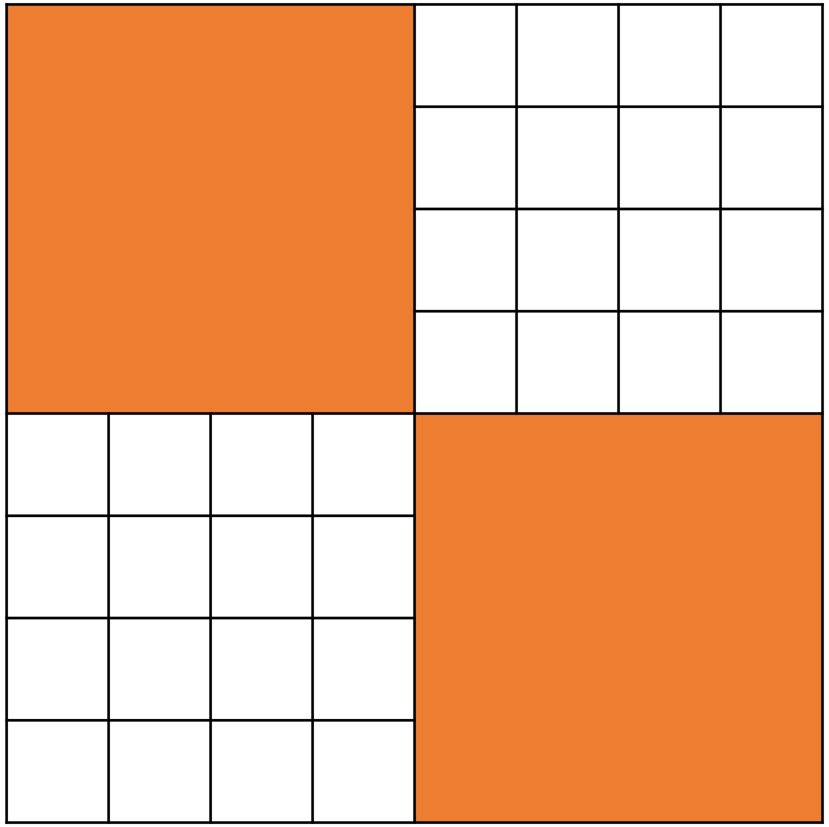


Georgia Tech

Update on the Separator Tree

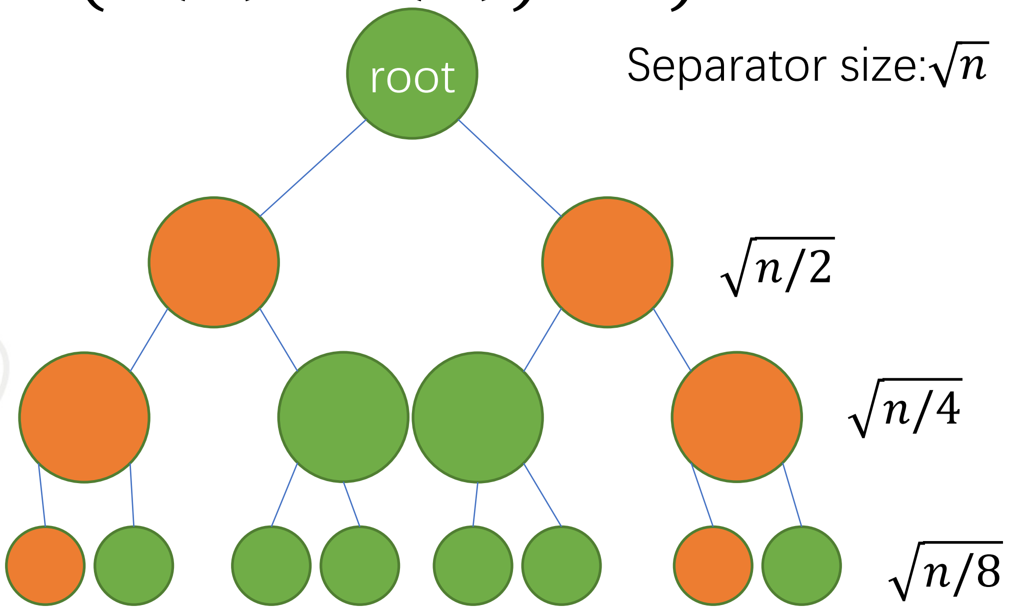
- $Sc(H, \delta H) = Sc\big(Sc\big(L(H), \delta L(H)\big) + Sc\big(R(H), \delta R(H)\big), \delta H\big)$





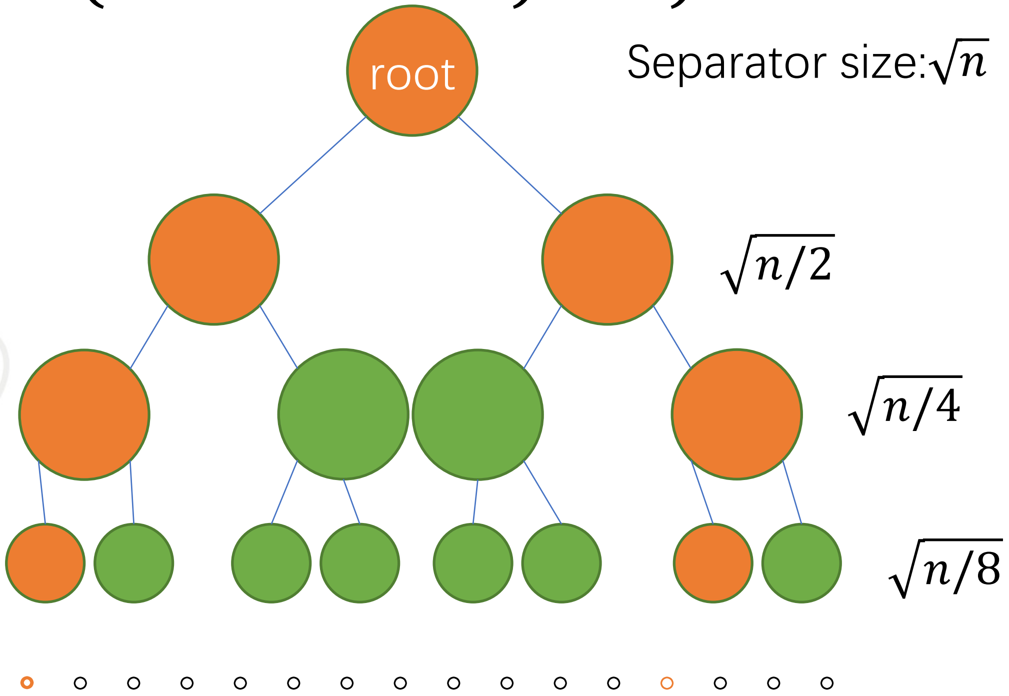
Update on the Separator Tree

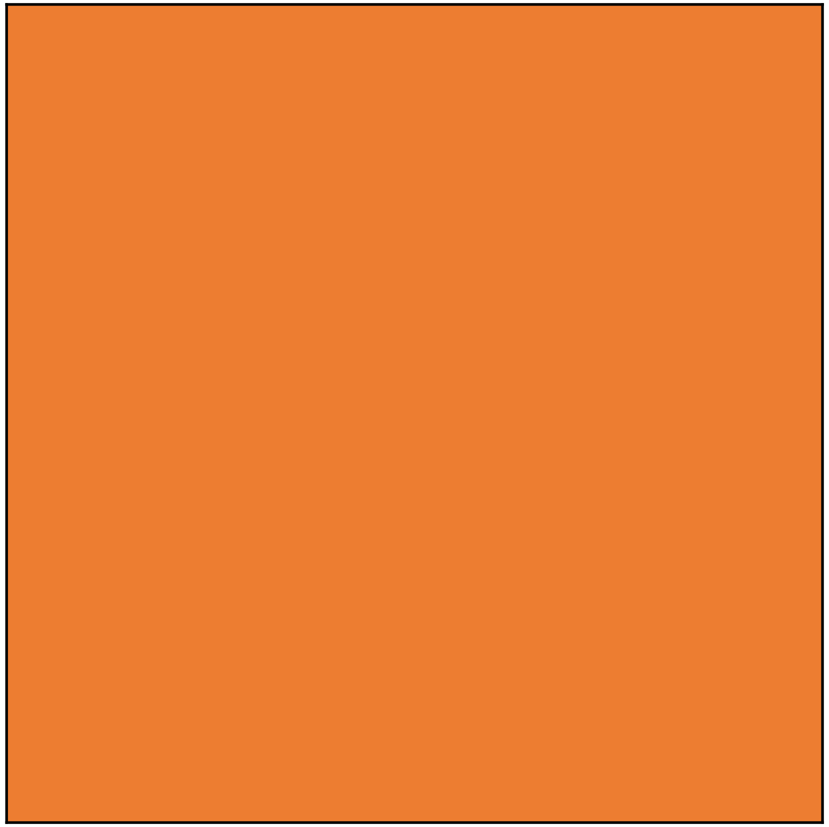
- $Sc(H, \delta H) = Sc\big(Sc\big(L(H), \delta L(H)\big) + Sc\big(R(H), \delta R(H)\big), \delta H\big)$



Update on the Separator Tree

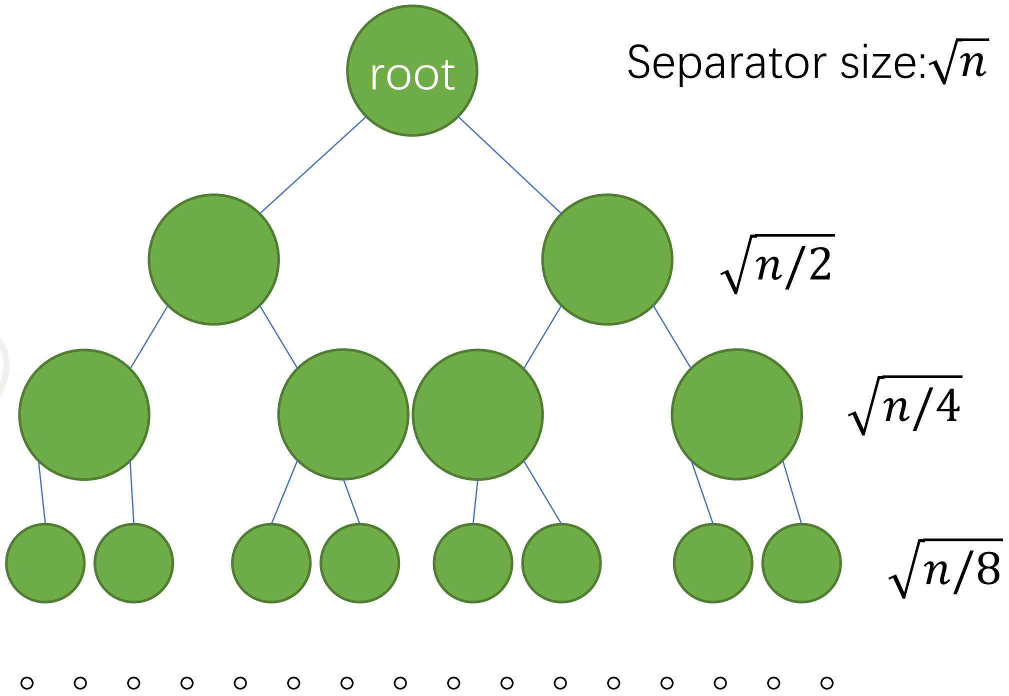
- $Sc(H, \delta H) = Sc\big(Sc\big(L(H), \delta L(H)\big) + Sc\big(R(H), \delta R(H)\big), \delta H\big)$

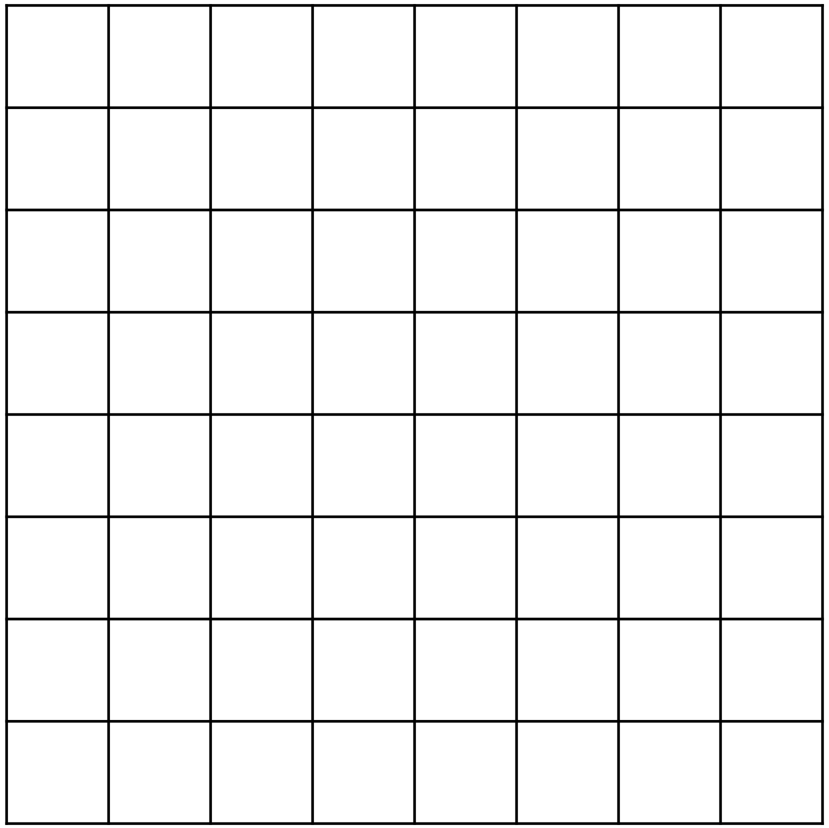




Update time

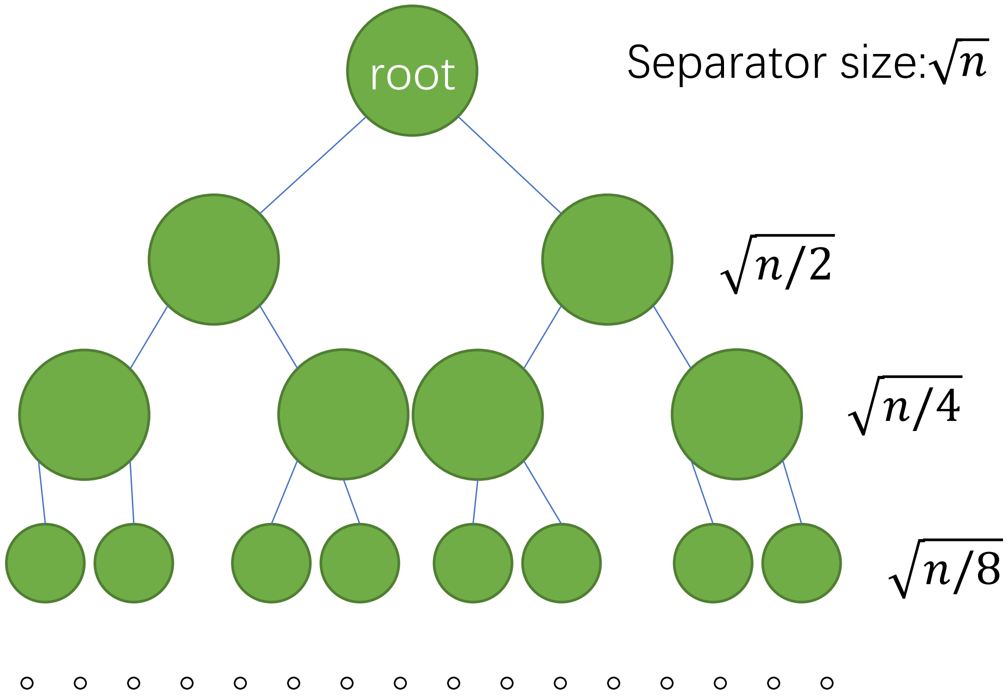
- Theorem: Modifying k edges costs only $O(\sqrt{nk})$ time

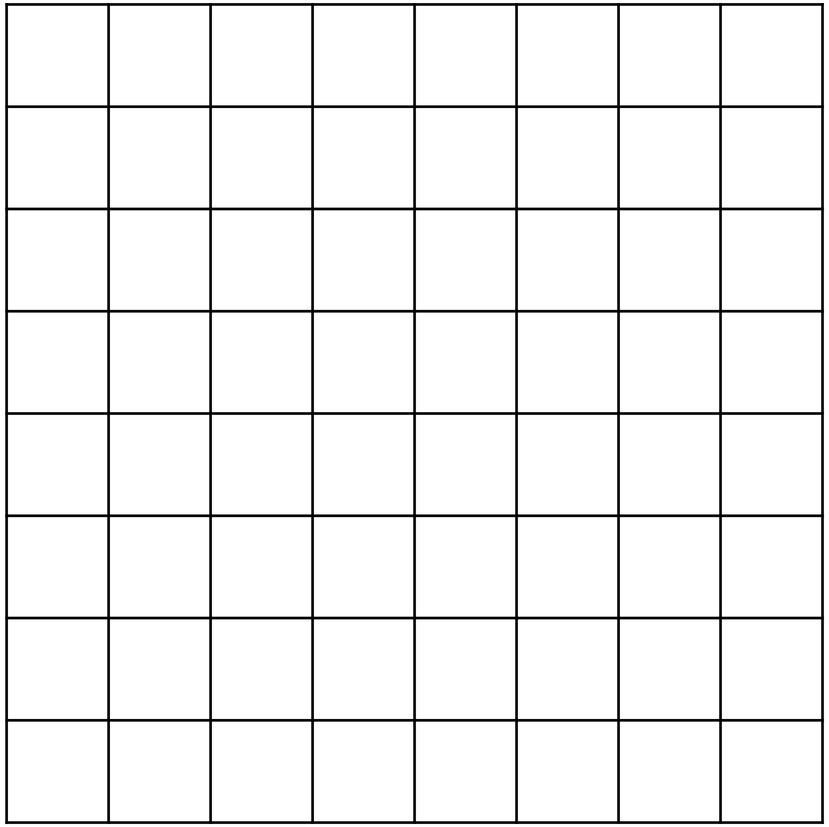




Update time

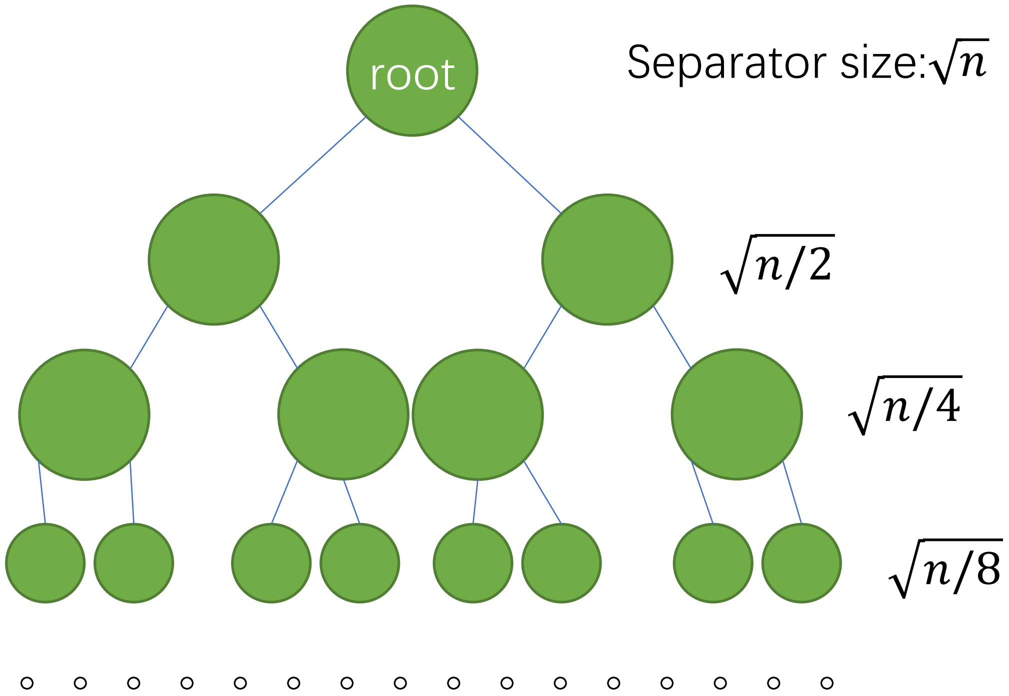
- Corollary: Modifying 1 edges costs $O(\sqrt{n})$ time

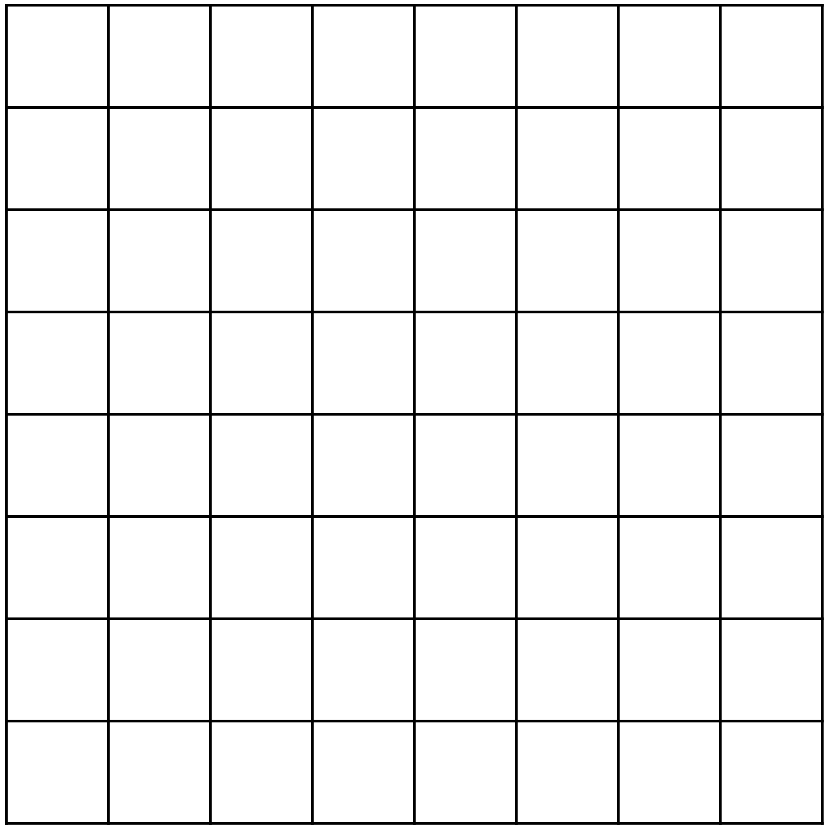




Update time

- Corollary: Modifying n edges costs $O(n)$ time





Application: Planar mincost flow

- Theorem [DGGLPSY' 21]: Let G be a planar graph with n edges. Assume all demands, costs and capacities are bounded by M. ∃ Algorithm computes a minimum cost flow in $O\left(n \log^{O(1)} n \log M\right)$ time.

- Interior point method: $\sqrt{n/k}$ batches of k updates each

- Dynamic electric flow: k updates in $\sqrt{nk}$ time

Georgia Tech

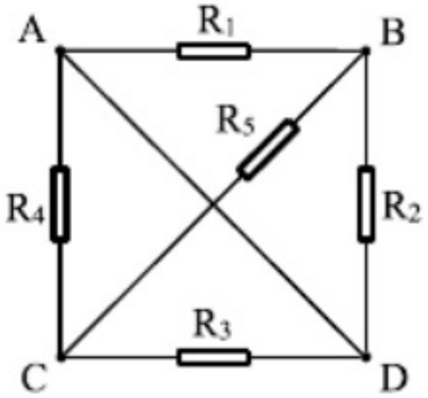
- Karate Club and Graph Laplacian
- Dynamic Laplacian by Schur Complement
- Dynamic Laplacian for Planar Graphs
- Dynamic Laplacian for General Graphs (By random walks)

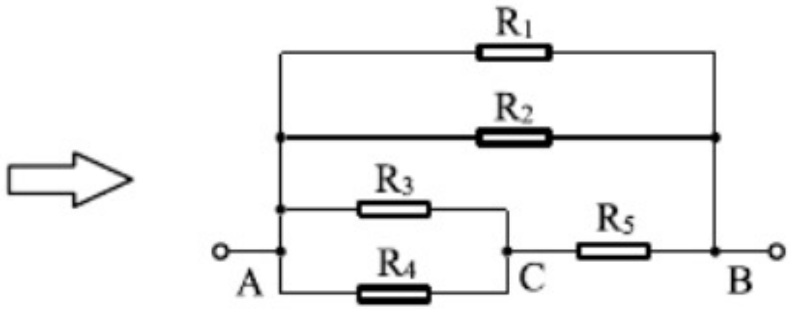
Dynamic Laplacian on General Graphs

(1) Update$(\mathbf{e}, \mathbf{r}_\mathbf{e}^{\mathbf{new}})$: Change the resistance of $\mathbf{e}$ to $\mathbf{r}_\mathbf{e}^{\mathbf{new}}$

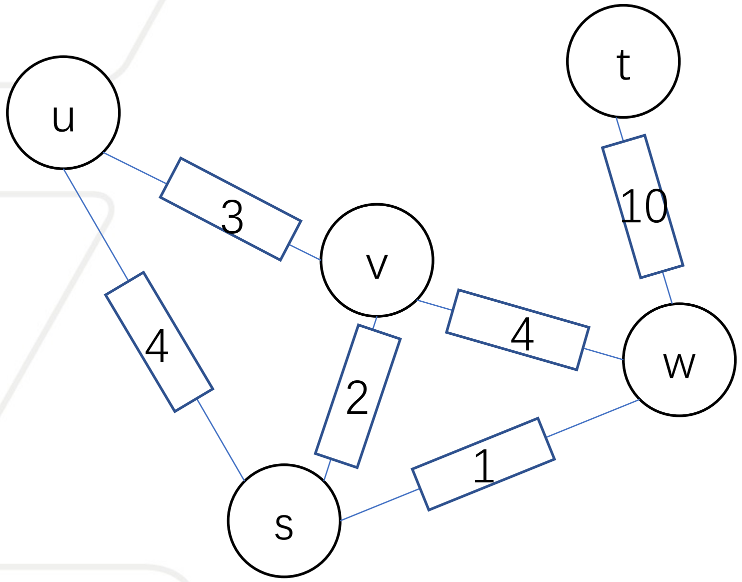
(2) Query($\mathbf{v}$): Output the potential of any vertex v in the unit $s-\mathrm{t}$ electrical flow

What is the potential of A?





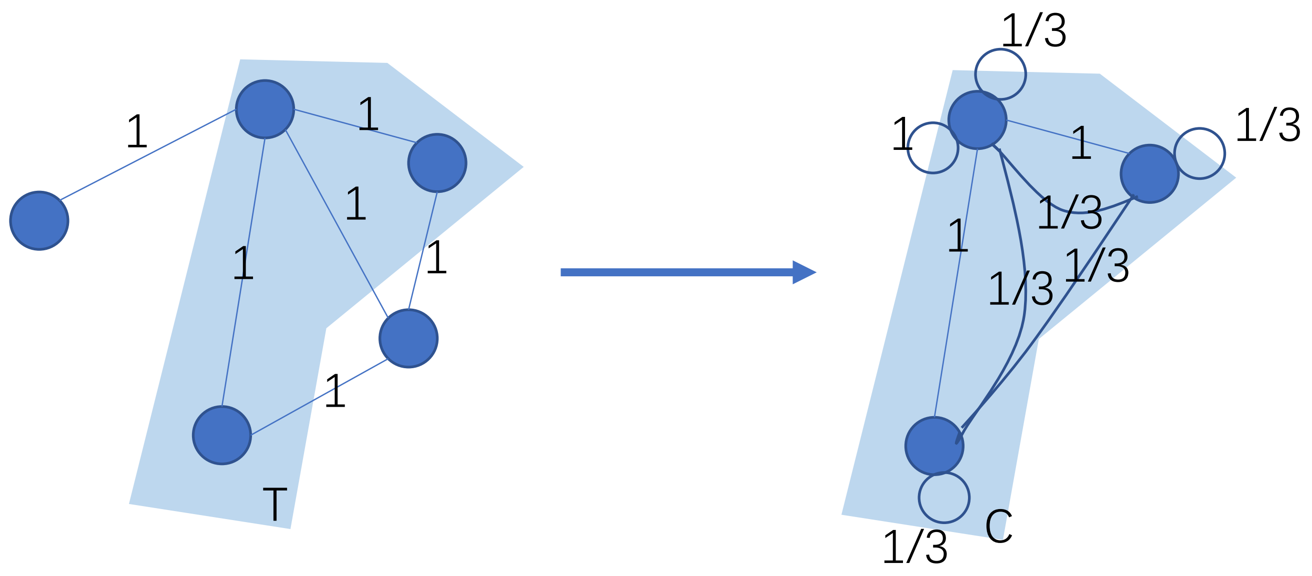
Graph Laplacian – Random Walks



- [Doyle-Snell `84] Unit electrical flow form s to t is the expected trajectory of a random walk from s to t, with cancellations, w.r.t. to the edge conductances.
- Flow $\mathbf{f_e}$ on edge $\mathrm{e = uv}$: $\mathrm{E_{rand\ walk}[\#\ of\ uv\ -\ \#\ of\ vu]}$
- Potential $\boldsymbol{\phi}_\mathrm{v}$ of vertex v: $\propto$ $\displaystyle \Pr_{\mathrm{r.w.\ from\ v}}[\mathrm{s\ is\ visited\ before\ t}]$

Schur Complement – Equivalent Electric Network

- Let $\mathbf{C}$ be a subset of vertices. Suppose we only care about energies of edges in $\mathbf{C}$.



- $\mathbf{Sc(G, C)}$ preserves the energies on edges between vertices in $\mathbf{C}$

Schur Complement – Compressed Random Walk

- If a random walk goes outside, take it back with the correct probability distribution over vertices in $\mathbf{C}$

- $\mathrm{Sc}(G,C) =$

$$\sum_{u^{(0)},u^{(l)}\in C,\forall 1\leq i<l,u^{(i)}\notin C} \frac{\Pi_{0\leq j<k}\mathbf{w}_{u^{(j)}u^{(j+1)}}}{\Pi_{0<j<k}\deg(u^{(j)})}$$

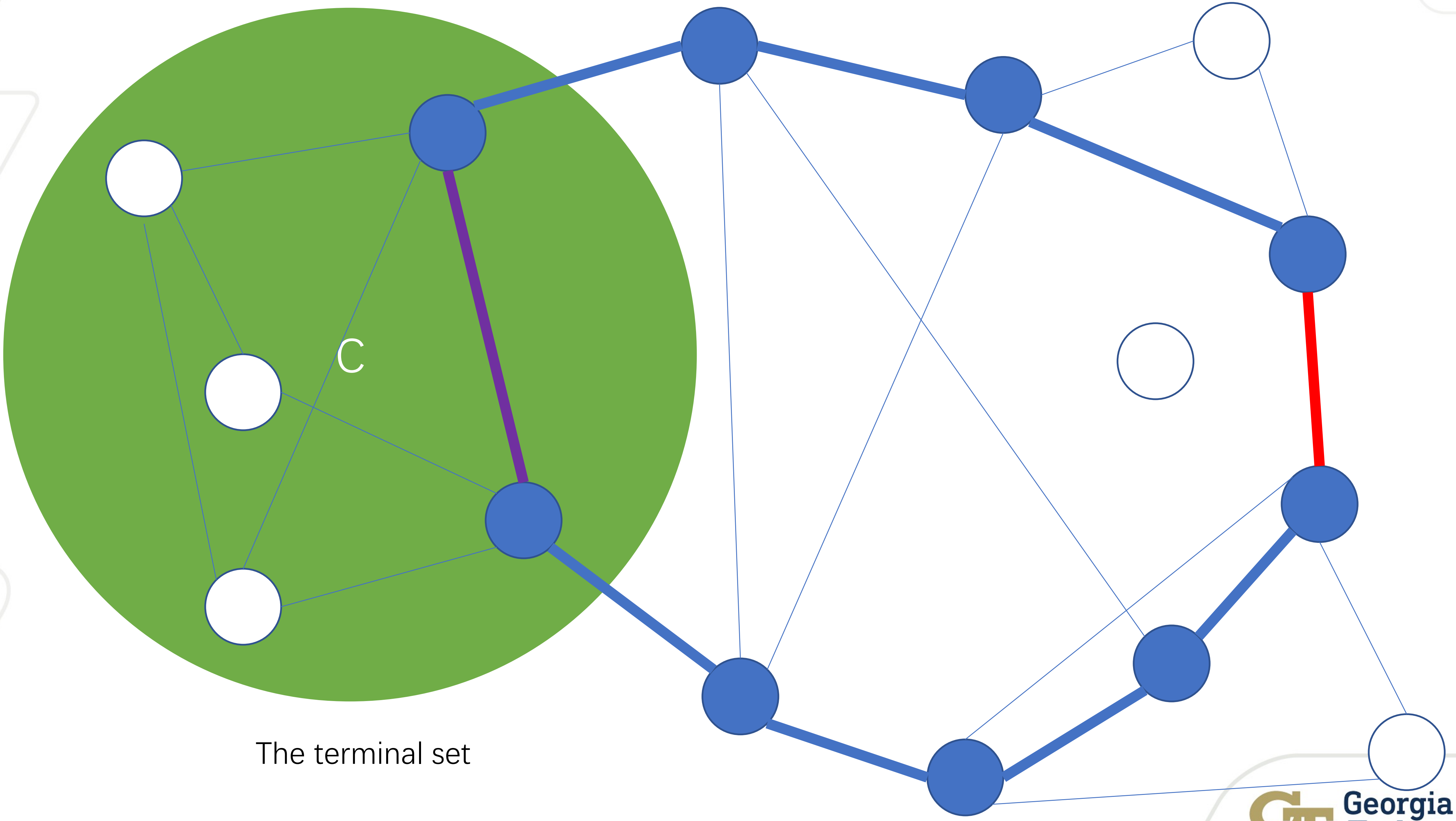
(Sum over all random walks from $\mathbf{C}$ to $\mathbf{C}$ whose interior is disjoint from $\mathbf{C}$)

Schur Complement: Static Approximation

- [DGGP `19]Theorem: Let $\mathbf{C}$ be a subset of vertices. For each edge $\mathbf{uv} = \mathbf{e} \in \mathbf{E}$, repeat $\rho = \widetilde{O}(\epsilon^{-2})$ times:
  1. Sample a random walk from $\mathbf{u}$ until it hits $\mathbf{C}$ at some $\mathbf{w}$.
  2. Sample a random walk from $\mathbf{v}$ until it hits $\mathbf{C}$ at some $\mathbf{z}$.
  3. Connect the random walks above by the edge $\mathbf{uv}$ into one random walk $\mathbf{W}$.
  4. Add an edge between $\mathbf{wz}$ with resistance $\rho \sum_{\mathbf{e} \in \mathbf{W}} \mathbf{r_e}$ to $\mathbf{H}$

Then $\mathbf{H}$ is an ϵ-approximation of $\mathbf{Sc(G, C)}$

Edge energies are preserved upto $(1 \pm \epsilon)$

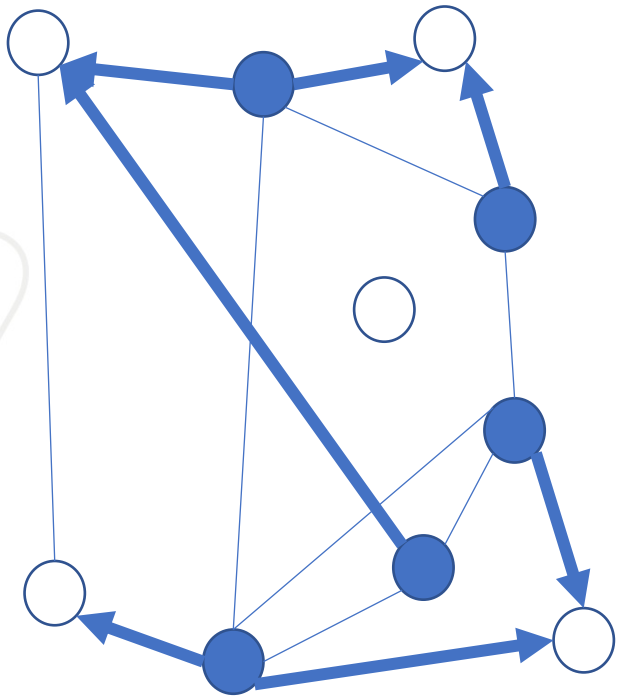


The terminal set

Sample random walk: Morris walk

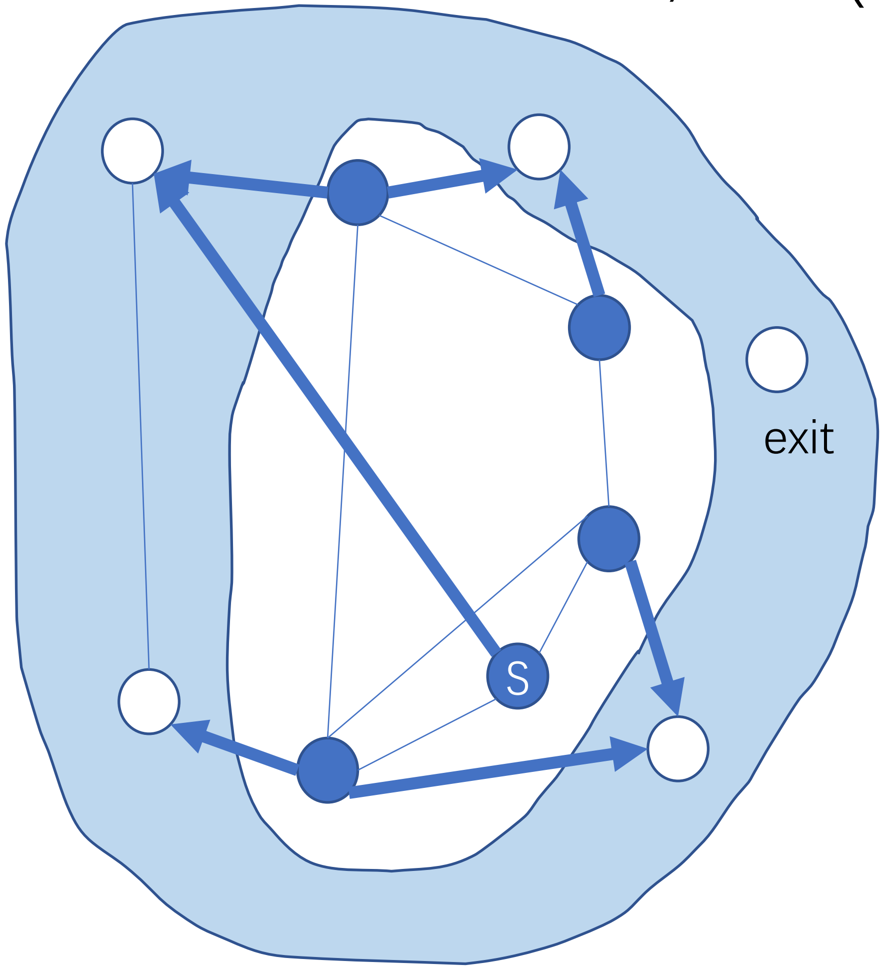
- Need: First k distinct vertices visited and length of walk in between

- Repeat: Given the visited vertex, find (sample) the next new vertex



Sample random walk: Morris walk

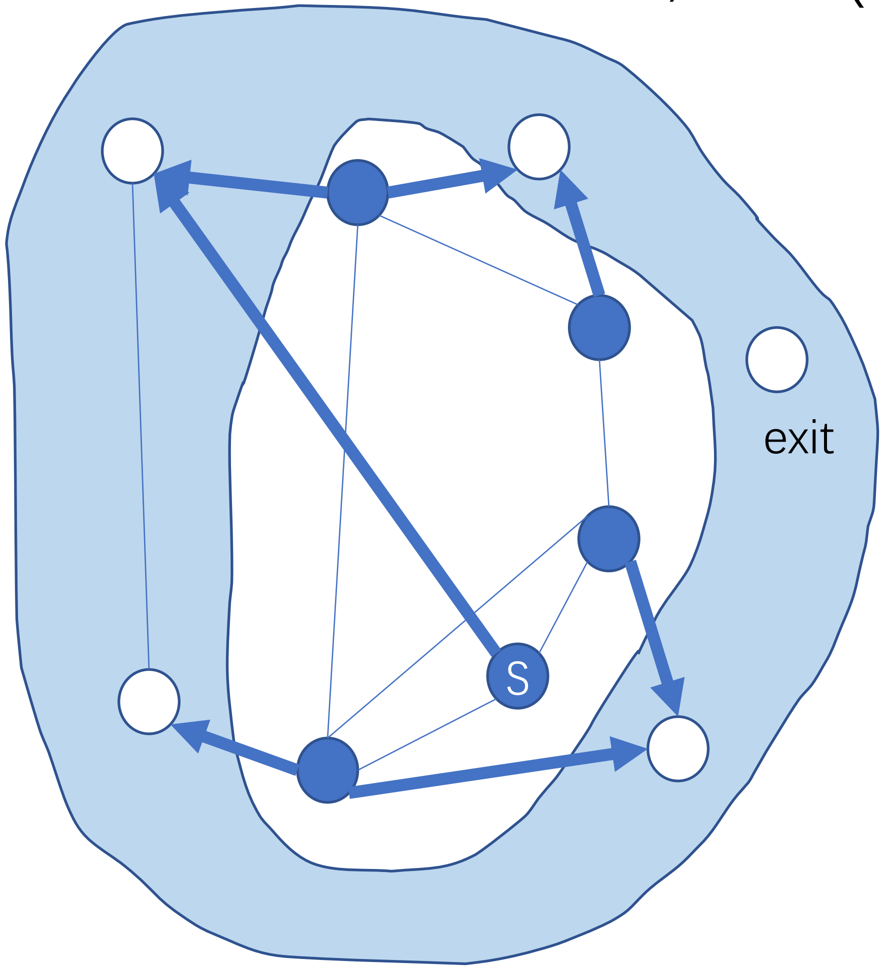
- Given the visited vertex, find (sample) the next new vertex



States: $U \cup N(U)$
Exit states: $N(\mathrm{U})$
Non exit states: U
Goal: Sample the next exit state

Sample random walk: Morris walk

- Given the visited vertex, find (sample) the next new vertex

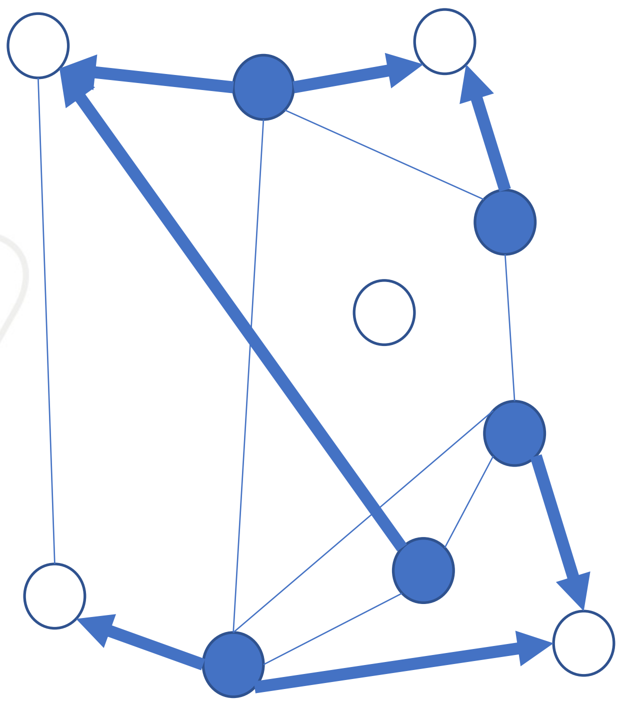


Electric current on e
= expected trajectory on e
$= \Pr[e \text{ is the last edge before exit}]$

Laplacian solver √

Sample random walk: Morris walk

- Given the visited vertex, **sample** the next new vertex and **length**
- Dynamic programming: Can only get expectation of length



Solution: Morris counter [Morris' 1978]

The counter stores $\mathrm{x} = \log n$

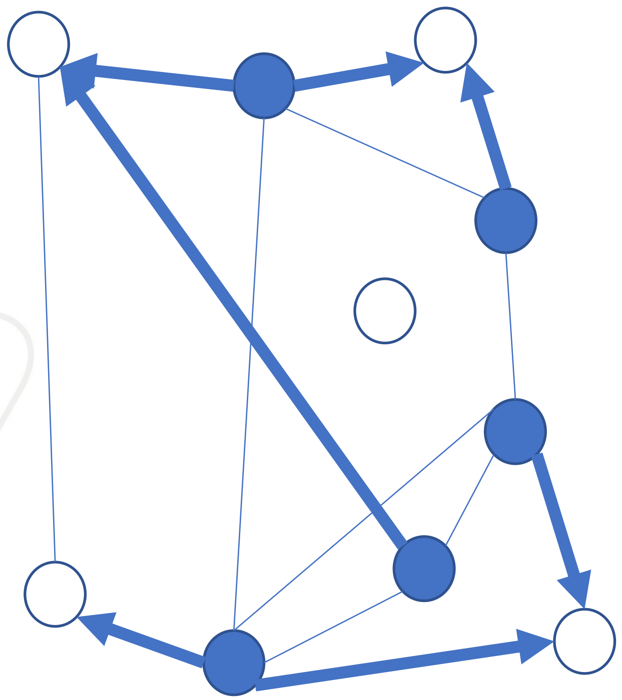
Increase Counter: $x \leftarrow x + 1$ with probability $1/2^x$

Property: By increasing the counter 2^x times, x is increased by 1 in expectation.

Georgia Tech

Sample random walk: Morris walk

- Given the visited vertex, **sample** the next new vertex and **length**



Solution: Morris counter [Morris' 1978]

The counter stores $\mathrm{x} = \log_{1+\epsilon} \epsilon n + 1$

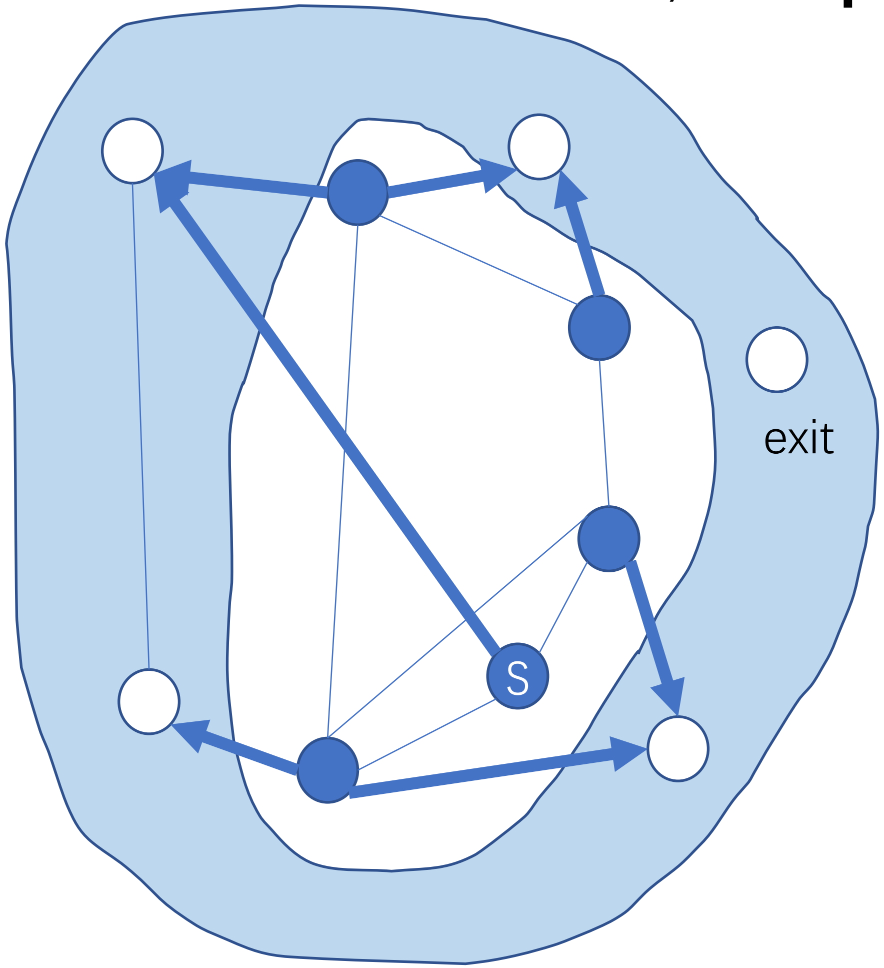
Increase Counter: with probability $1/(1+\epsilon)^x$

Theorem: $\frac{1}{\epsilon}\left((1+\epsilon)^X - 1\right)$ is w.h.p. $(1+\epsilon)$-approximation of true counter

Georgia Tech

Sample random walk: Morris walk

- Given the visited vertex, **sample** the next new vertex and **length**



Current Morris counter value: x

States: $V \times \mathbb{Z}$

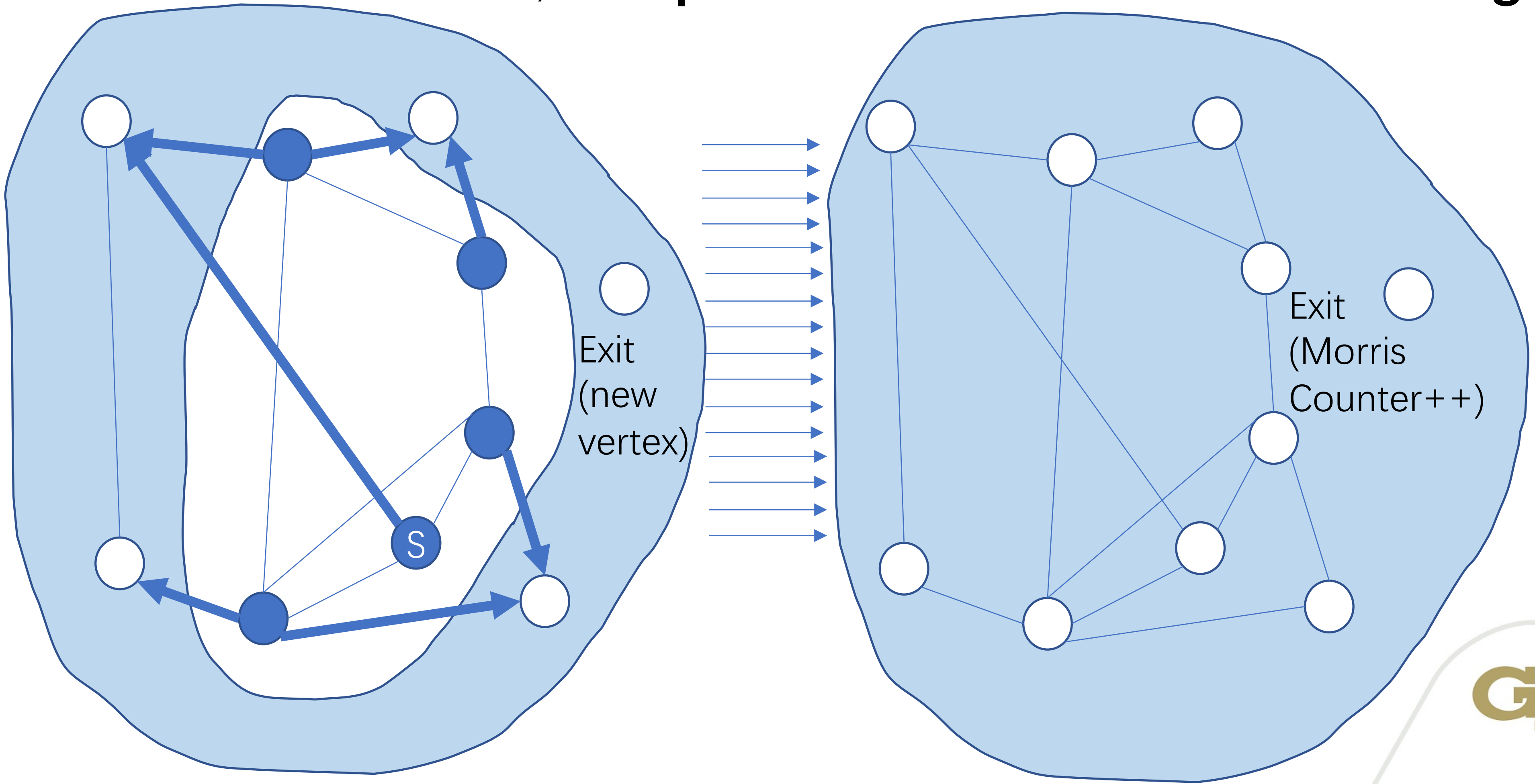
Exit states: $N(\mathrm{U}) \times \{x\}$ and $V \times \{\geq x+1\}$

Non exit states: $U \times \{\mathrm{x}\}$

Goal: Sample the next exit state

Sample random walk: Morris walk

- Given the visited vertex, **sample** the next new vertex and **length**



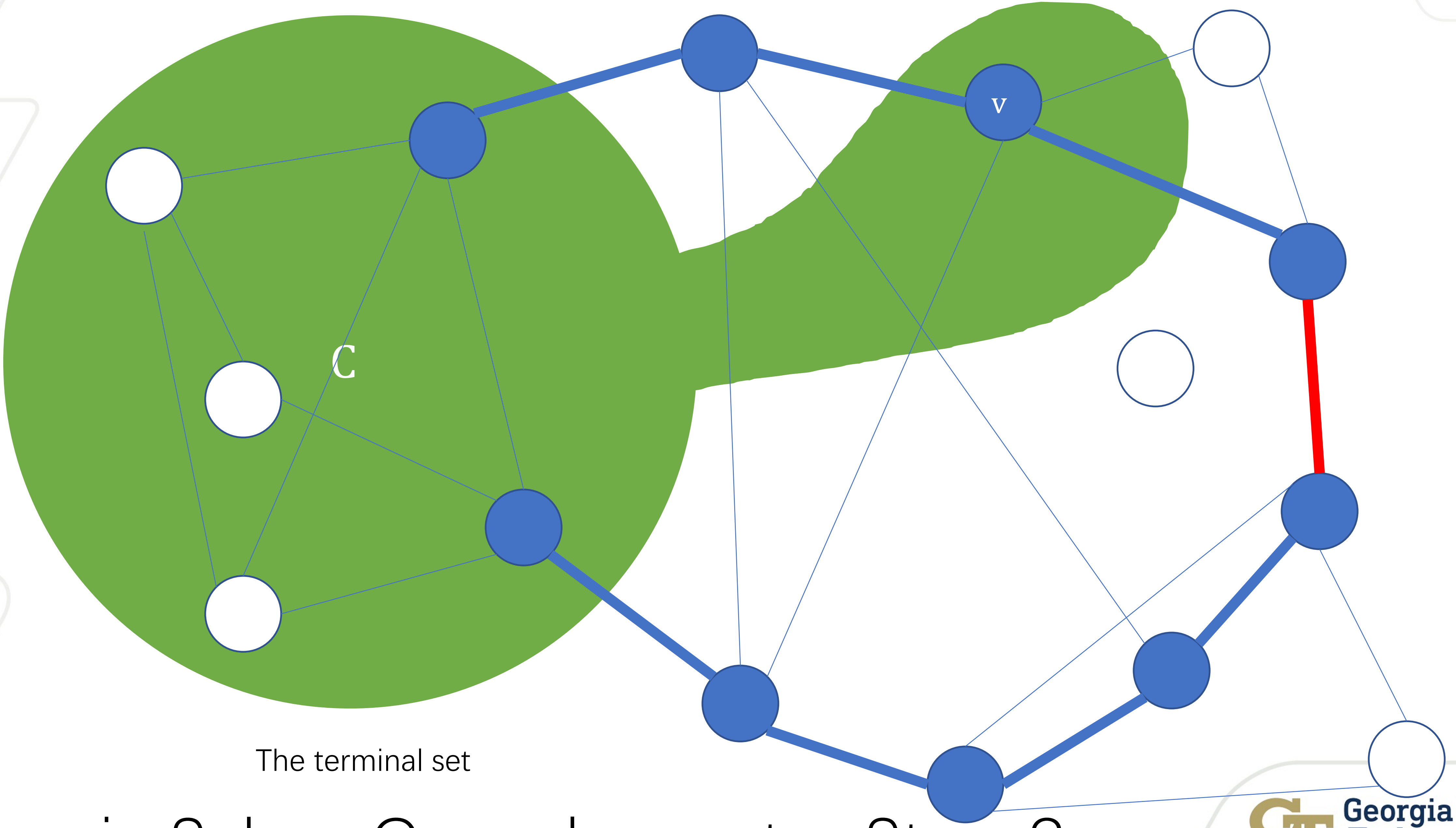
Dynamic Schur Complement – Step 1

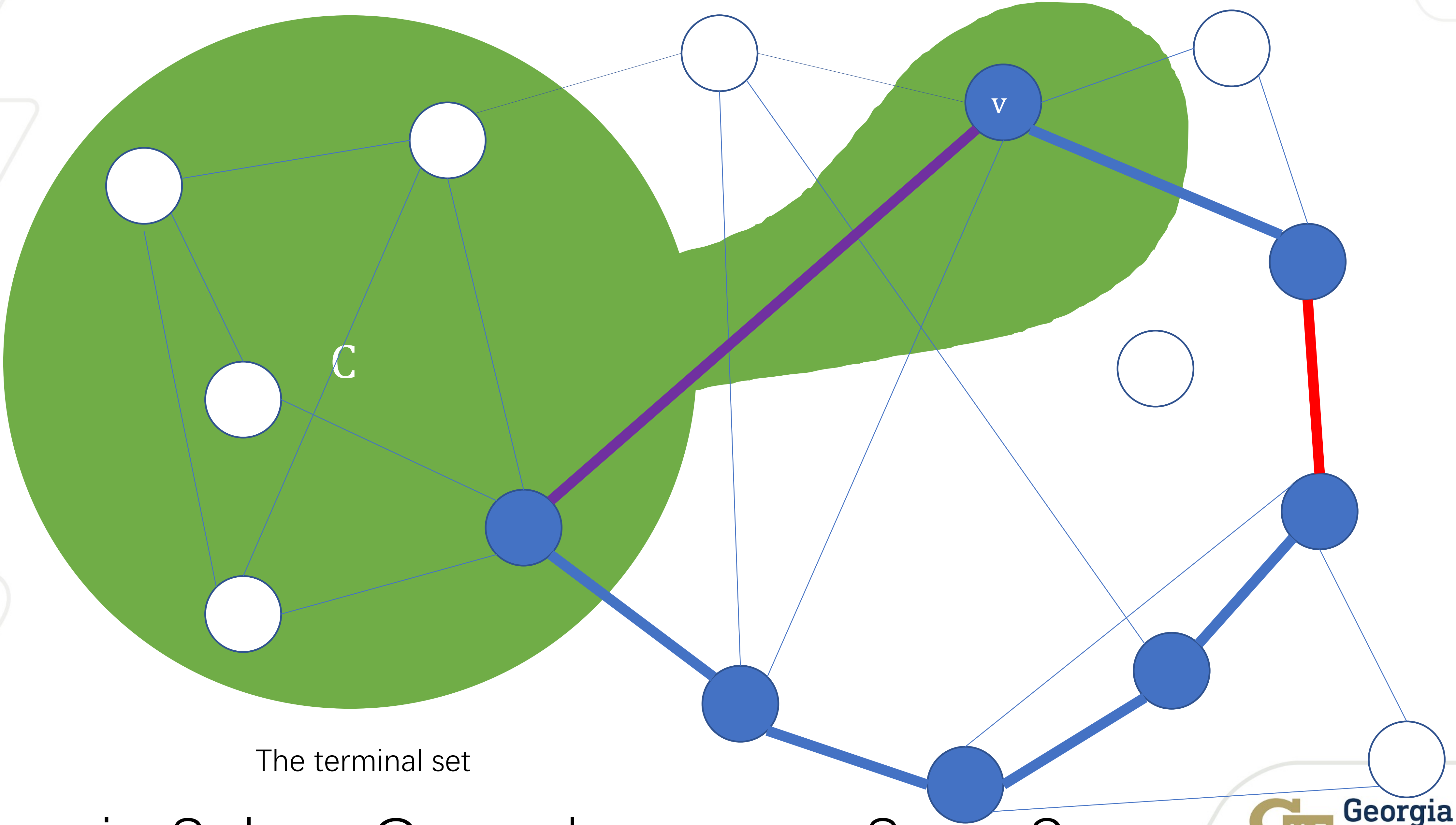
- Let C be a subset of vertices. For each edge $uv \subseteq C$, repeat $\rho = \tilde{O}(\epsilon^{-2})$ times:
  1. A random walk from u hits C at u.
  2. A random walk from v hits C at v.
  3. Connect the random walks above by the edge uv into one random walk $W = (uv)$.
  4. Add an edge between uv with resistance ρr_{uv} to H.

- Theorem: Let C be a subset of vertices. For each edge $uv = e \in E$, repeat $\rho = \tilde{O}(\epsilon^{-2})$ times:
  1. Sample a random walk from u until it hits C at some w.
  2. Sample a random walk from v until it hits C at some z.
  3. Connect the random walks above by the edge uv into one random walk W.
  4. Add an edge between wz with resistance $\rho \sum_{e \in W} r_e$ to H

Then H is an ϵ-approximation of $Sc(G, C)$

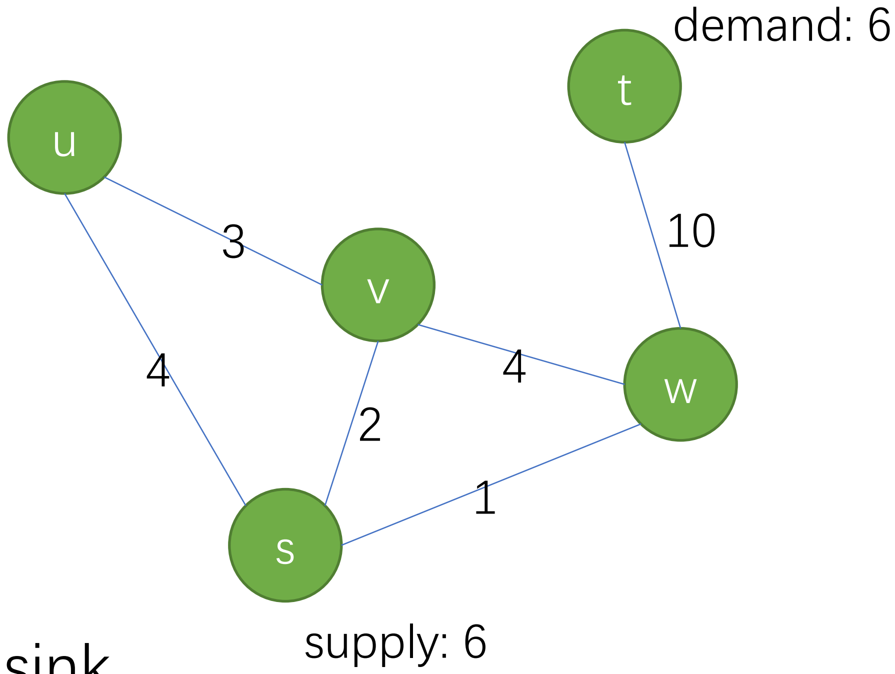
Dynamic Schur Complement – Step 2





Dynamic Schur Complement – Step 2

Application: Maxflow



- Given
  - Graph $G = (V, E)$
  - Capacities of the edges
  - Demand or supply of the source and sink

- Q: Can we fulfill the demand/supply by a flow not exceeding the capacities?

Pseudocode of Maxflow by IPM+Electric flow

while(more than 1 unit of flow remaining)

Determine edge resistances r by flows and capacities

Calculate electric flow from s to t by r

Route $1/\sqrt{m}$ fraction of flow from s to t

Electric flow to accelerate maxflow

- Theorem [GLP' 21]: Let G be a graph with m edges. Assume all demands and capacities are bounded by M. ∃ Algorithm computes a minimum cost flow in $O\left(m^{3/2-1/328}\log^{O(1)} m \log M\right)$ time.

- Improve over the 20-year-old $O\left(m^{3/2}\log^{O(1)} m \log M\right)$ result by [Goldberg-Rao' 98]

Capacity Releasing Diffusion [WFHMR' 17]

- Flow diffusion: a process that spreads mass among vertices by sending mass along edges

Thank you!