Congestion-Approximators from the Bottom Up

Jason Li (CMU)
Joint with Satish Rao (Berkeley),
Di Wang (Google)

Max-Flow Problem

Given undirected (uncapacitated) graph $G=(V,E)$ and vector b in $\mathbb{R}^V$ with $\sum b_V=0$

Max-Flow Problem

Given undirected (uncapacitated) graph $G=(V,E)$ and vector b in $\mathbb{R}^V$ with $\sum b_v = 0$

s-t max-flow: $b = \mathbb{1}_s - \mathbb{1}_t = \begin{bmatrix} +1 \\ 0 \\ \vdots \\ -1 \end{bmatrix} \begin{matrix} s \\ \\ \\ t \end{matrix}$

Max-Flow Problem

Given undirected (uncapacitated) graph $G=(V,E)$ and vector b in $\mathbb{R}^V$ with $\sum b_v = 0$

s-t max-flow: $b = \mathbb{1}_s - \mathbb{1}_t = \begin{bmatrix} +1 \\ 0 \\ \vdots \\ -1 \end{bmatrix} \begin{matrix} s \\ \\ \\ t \end{matrix}$

Minimize congestion

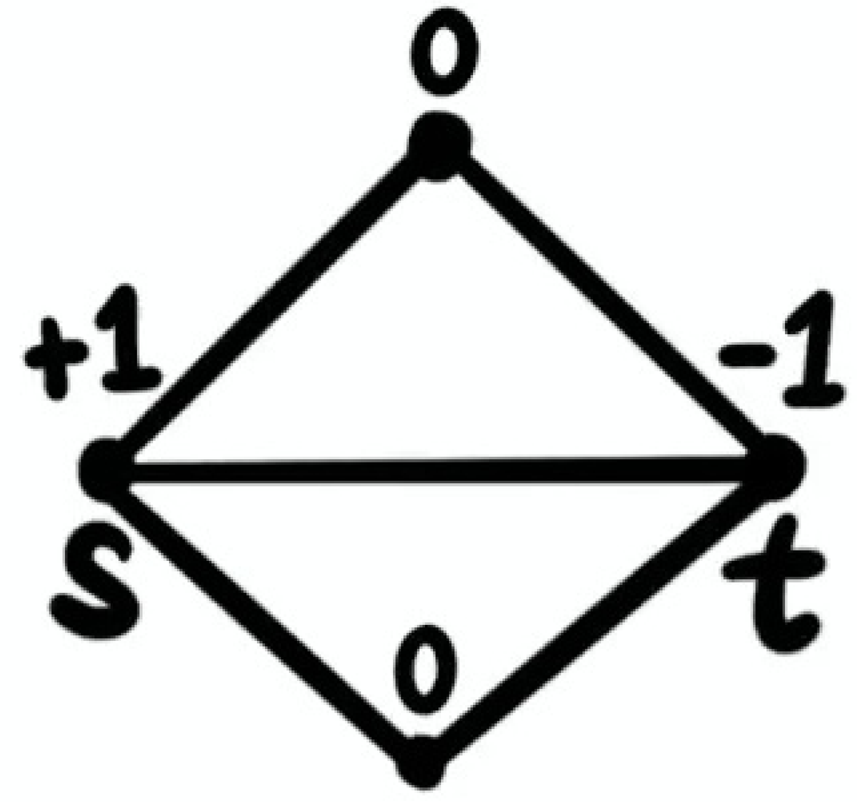
= max amount of flow sent along any edge

Max-Flow Problem

Given undirected (uncapacitated) graph

$G=(V,E)$ and vector b in $\mathbb{R}^V$ with $\sum b_v = 0$

s-t max-flow: $b = \mathbb{1}_s - \mathbb{1}_t = \begin{bmatrix} +1 \\ 0 \\ \vdots \\ -1 \end{bmatrix} \begin{matrix} s \\ \\ \\ t \end{matrix}$



$b = \mathbb{1}_s - \mathbb{1}_t$

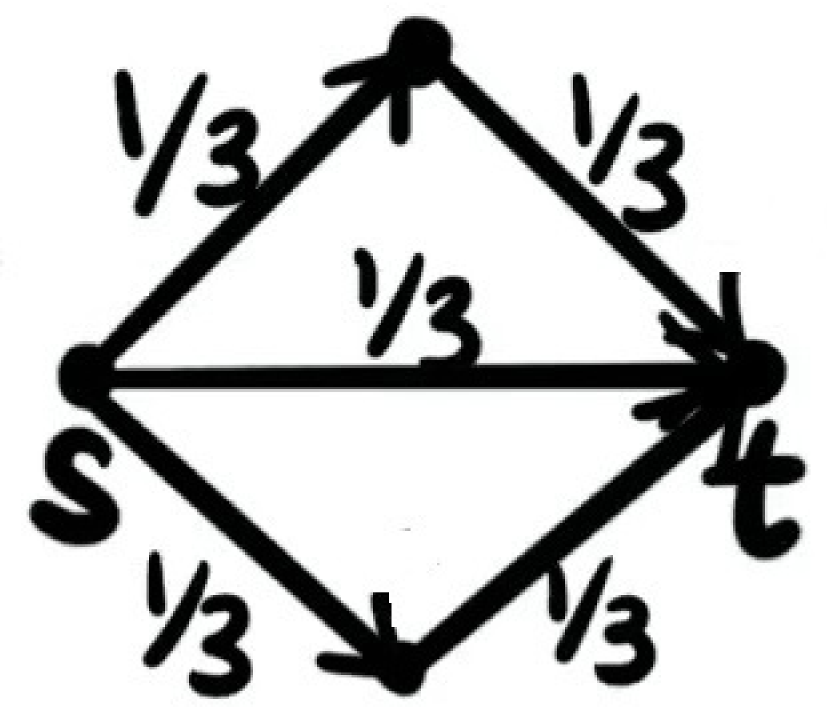
Minimize congestion

= max amount of flow sent along any edge

Max-Flow Problem

Given undirected (uncapacitated) graph $G=(V,E)$ and vector b in $\mathbb{R}^V$ with $\sum b_v = 0$

s-t max-flow: $b = \mathbb{1}_s - \mathbb{1}_t = \begin{bmatrix} +1 \\ 0 \\ \vdots \\ -1 \end{bmatrix} \begin{matrix} s \\ \\ \\ t \end{matrix}$



$b = \mathbb{1}_s - \mathbb{1}_t$

congestion = 1/3

Minimize congestion

= max amount of flow sent along any edge

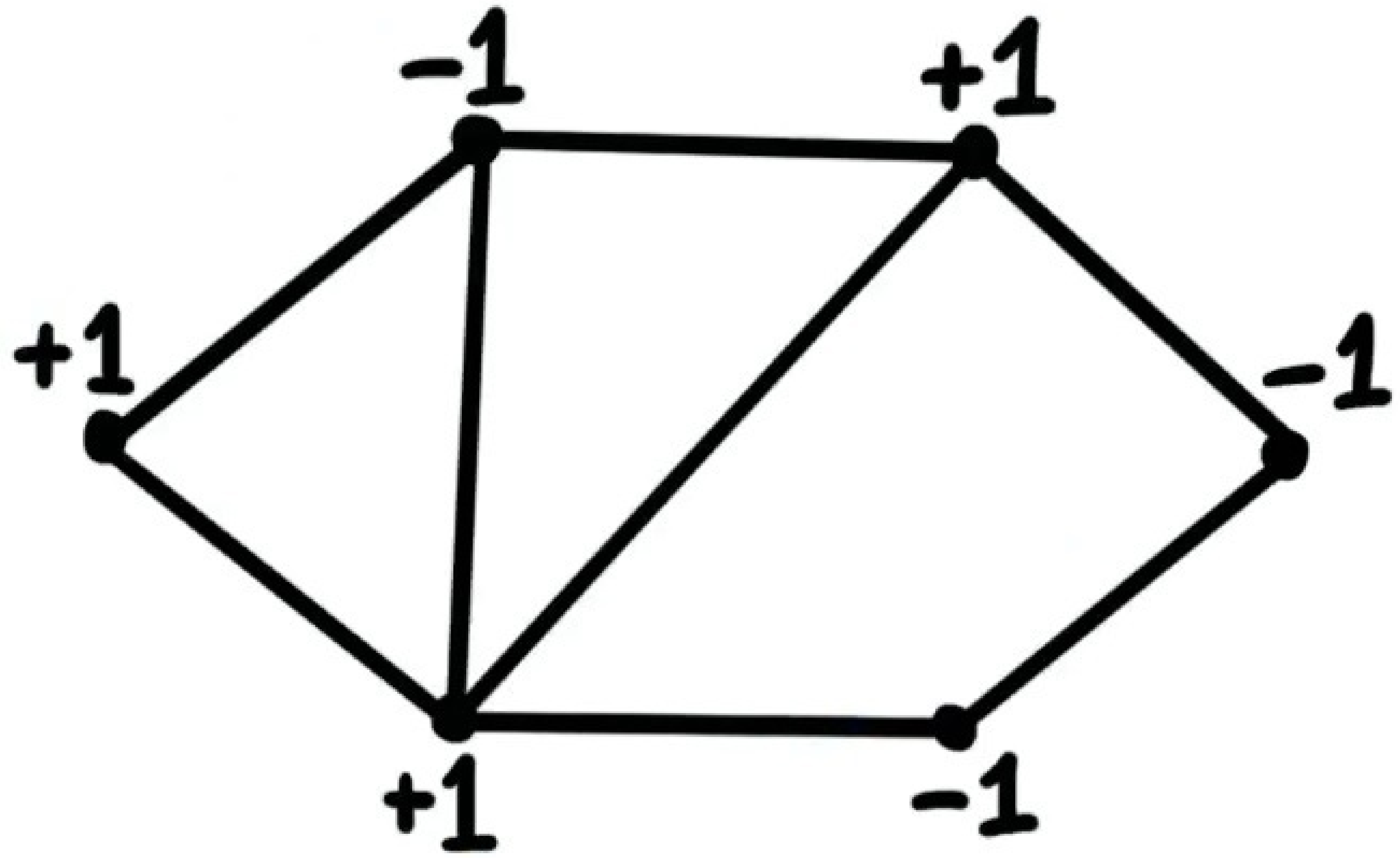
Flow/Cut Duality

Given a subset of vertices S, consider

$$\frac{|b(S)|}{\partial S} \quad \begin{array}{l} \leftarrow \left|\sum_{v \in S} b_v\right| \\ \leftarrow \text{size of boundary} \end{array}$$

Flow/Cut Duality

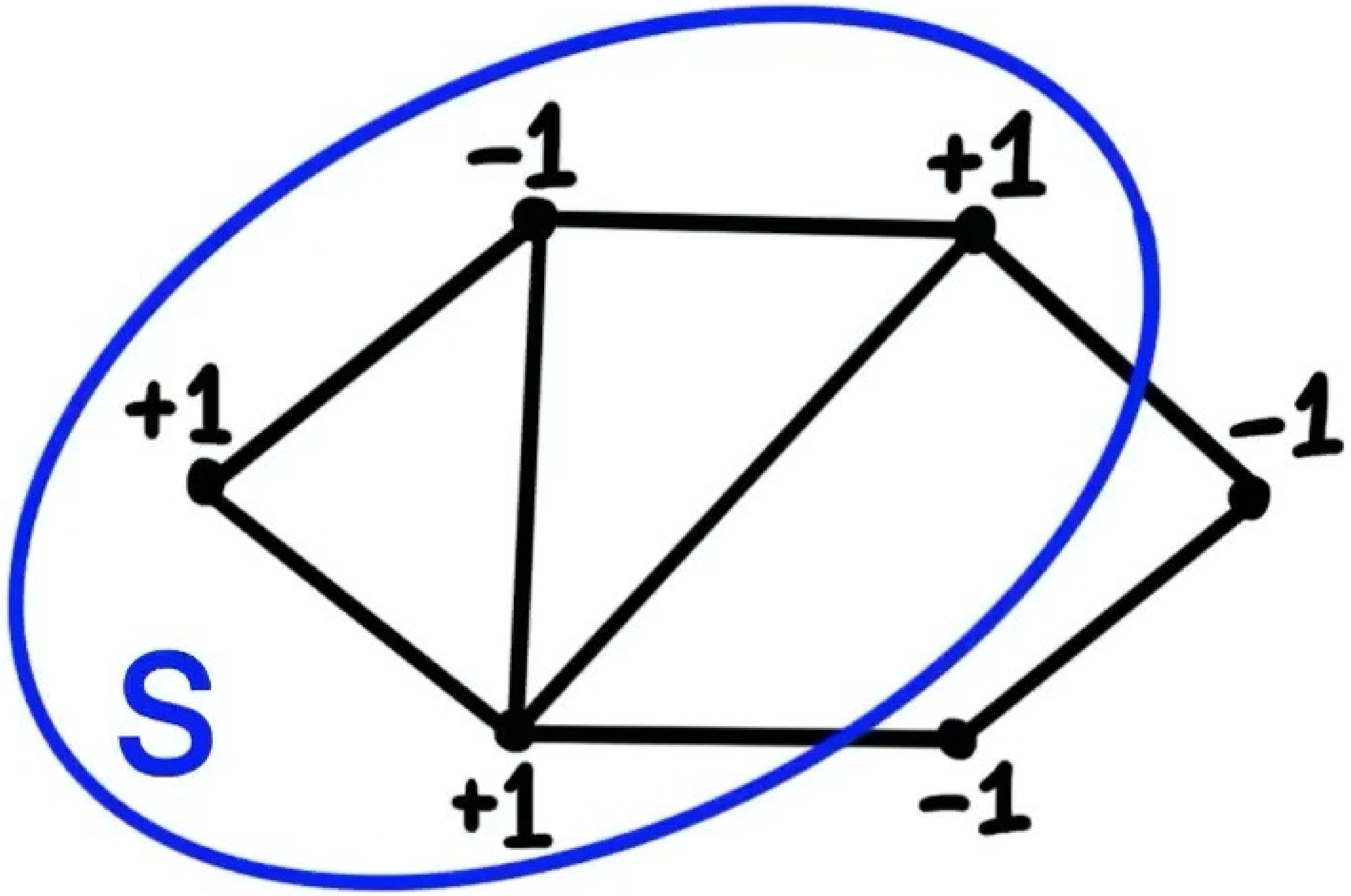
Given a subset of vertices S, consider



$$\frac{|b(S)|}{\partial S} \quad \begin{matrix} \leftarrow \left|\sum_{v\in S} b_v\right| \\ \leftarrow \text{size of boundary} \end{matrix}$$

Flow/Cut Duality

Given a subset of vertices S, consider

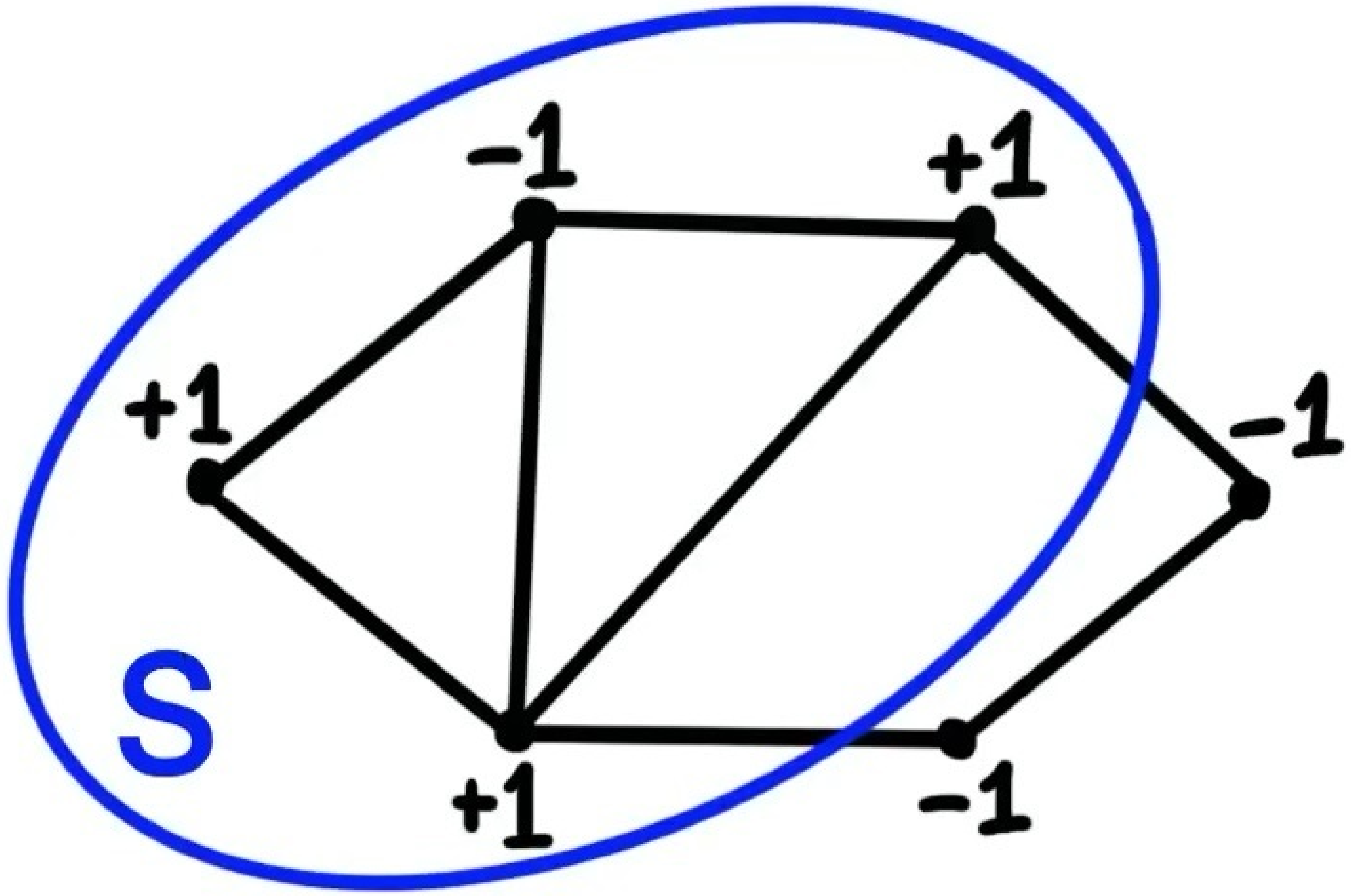


$$\frac{|b(S)|}{\partial S} \quad \begin{array}{l} \leftarrow \left|\sum_{v \in S} b_v\right| \\ \leftarrow \text{size of boundary} \end{array}$$

Flow/Cut Duality

Given a subset of vertices S, consider

$$\frac{|b(S)|}{\partial S} \quad \begin{array}{l} \leftarrow \left|\sum_{v \in S} b_v\right| \\ \leftarrow \text{size of boundary} \end{array}$$



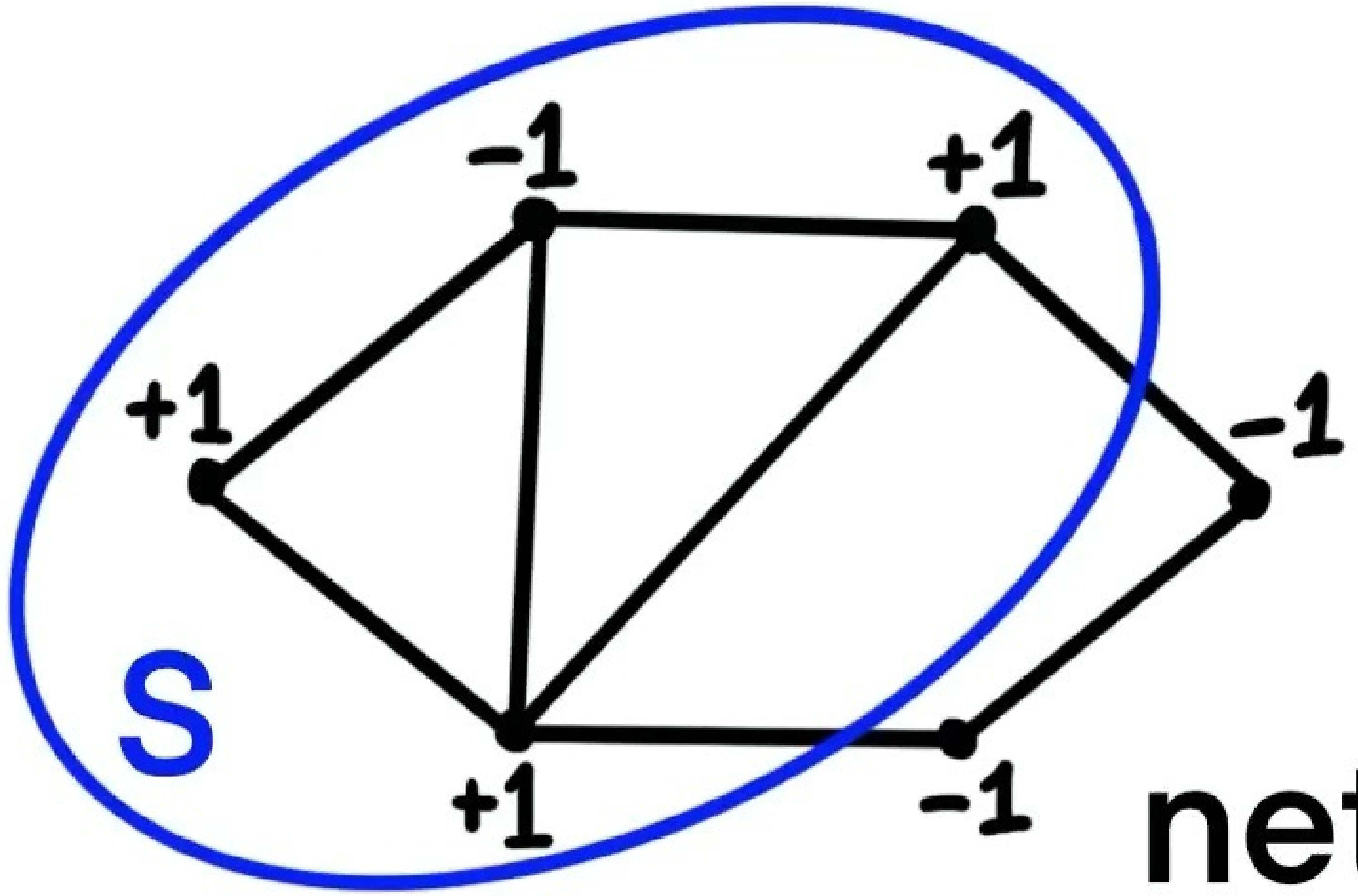
$\partial S = 2$

$|b(S)| = 2$

Flow/Cut Duality

Given a subset of vertices S, consider

$$\frac{|b(S)|}{\partial S} \quad \begin{matrix} \leftarrow \left|\sum_{v\in S} b_v\right| \\ \leftarrow \text{size of boundary} \end{matrix}$$



net |b(S)|=2 flow must
go across ∂S=2 edges

$\partial S = 2$

$|b(S)| = 2$

→ any flow has congestion >= 1

Flow/Cut Duality

For any S, congestion >= $\frac{|b(S)|}{\partial S}$

Flow/Cut Duality

For any S, congestion >= $\frac{|b(S)|}{\partial S}$

Flow/Cut Duality: congestion = $\max_{S \subseteq V} \frac{|b(S)|}{\partial S}$

Flow/Cut Duality

For any S, congestion >= $\frac{|b(S)|}{\partial S}$

Flow/Cut Duality: congestion = $\max_{S \subseteq V} \frac{|b(S)|}{\partial S}$

Congestion Approximator:

family $\mathcal{C}$ of subsets S s.t.

congestion <= $\alpha \cdot \max_{S \in \mathcal{C}} \frac{|b(S)|}{\partial S}$

↑ quality

Flow/Cut Duality

For any S, congestion >= $\frac{|b(S)|}{\partial S}$

Flow/Cut Duality: congestion = $\max_{S \subseteq V} \frac{|b(S)|}{\partial S}$

Congestion Approximator:

family $\mathcal{C}$ of subsets S s.t.

congestion <= $\alpha \cdot \max_{S \in \mathcal{C}} \frac{|b(S)|}{\partial S}$ for **all** demands b

($b(V)=0$)

↑ quality

Approximate Max-Flow

[Sherman '13, '17] Given polylog-quality congestion approximator, can solve (1+ε)-approximate congestion in $\tilde{O}(m/\varepsilon)$

^ with $\tilde{O}(m)$ representation size

Approximate Max-Flow

[Sherman '13, '17] Given polylog-quality congestion approximator, can solve $(1+\varepsilon)$-approximate congestion in $\tilde{O}(m/\varepsilon)$

with $\tilde{O}(m)$ ^ representation size

[Räcke, Shah, Taubig '14] Can compute polylog-quality congestion approximator by computing (1+1/polylog)-approximate max-flow on instances of total size $\tilde{O}(m)$

Approximate Max-Flow

[Sherman '13, '17] Given polylog-quality congestion approximator, can solve $(1+\varepsilon)$-approximate congestion in $\tilde{O}(m/\varepsilon)$

with $\tilde{O}(m)$ ^ representation size

[Räcke, Shah, Taubig '14] Can compute polylog-quality congestion approximator by computing (1+1/polylog)-approximate max-flow on instances of total size $\tilde{O}(m)$

Cyclic dependency! Resolved by [Peng '16] using recursive graph sparsification. Complicated algorithm

Approximate Max-Flow

[Sherman '13, '17] Given polylog-quality congestion approximator, can solve $(1+\varepsilon)$-approximate congestion in $\tilde{O}(m/\varepsilon)$
with $\tilde{O}(m)$ representation size

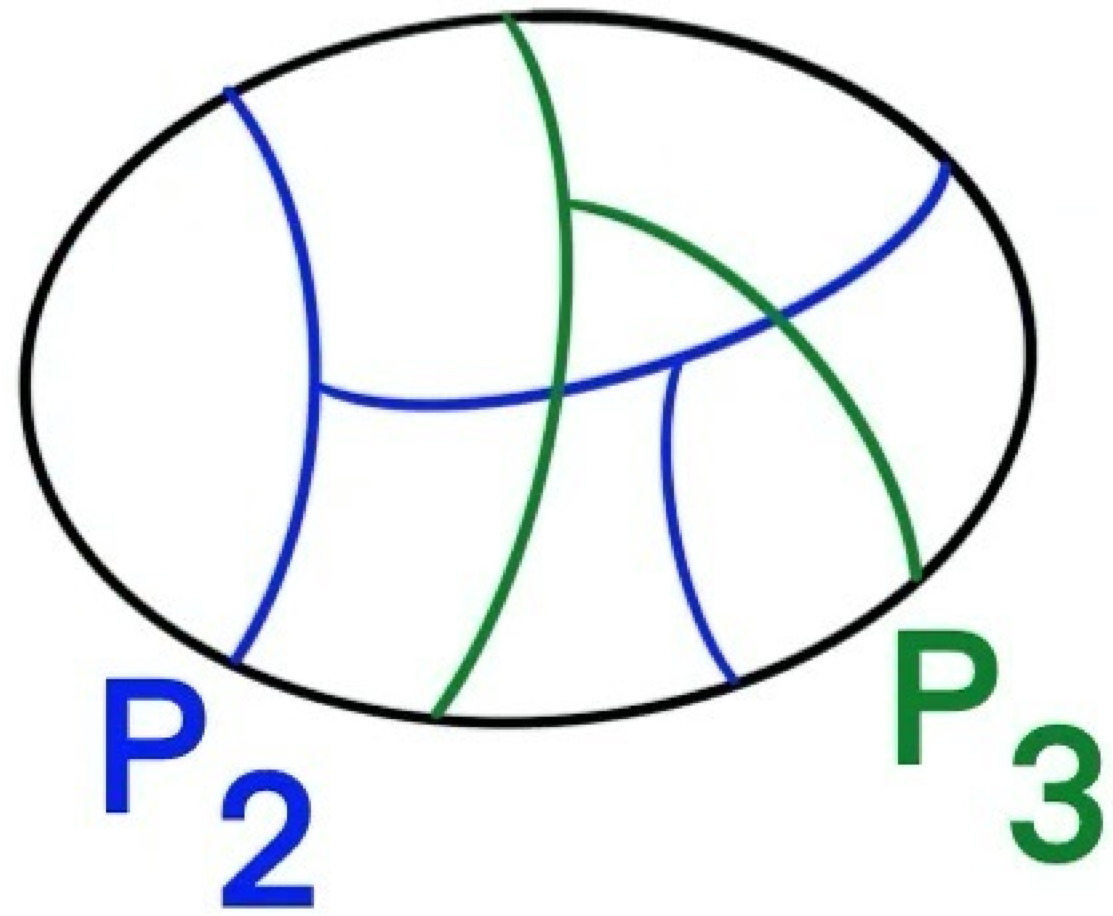
[Räcke, Shah, Taubig '14] Can compute polylog-quality congestion approximator by computing (1+1/polylog)-approximate max-flow on instances of total size $\tilde{O}(m)$

Cyclic dependency! Resolved by [Peng '16] using recursive graph sparsification. Complicated algorithm

This work: polylog-quality congestion approximator without recursive max-flow

Our Congestion Approximator Construction

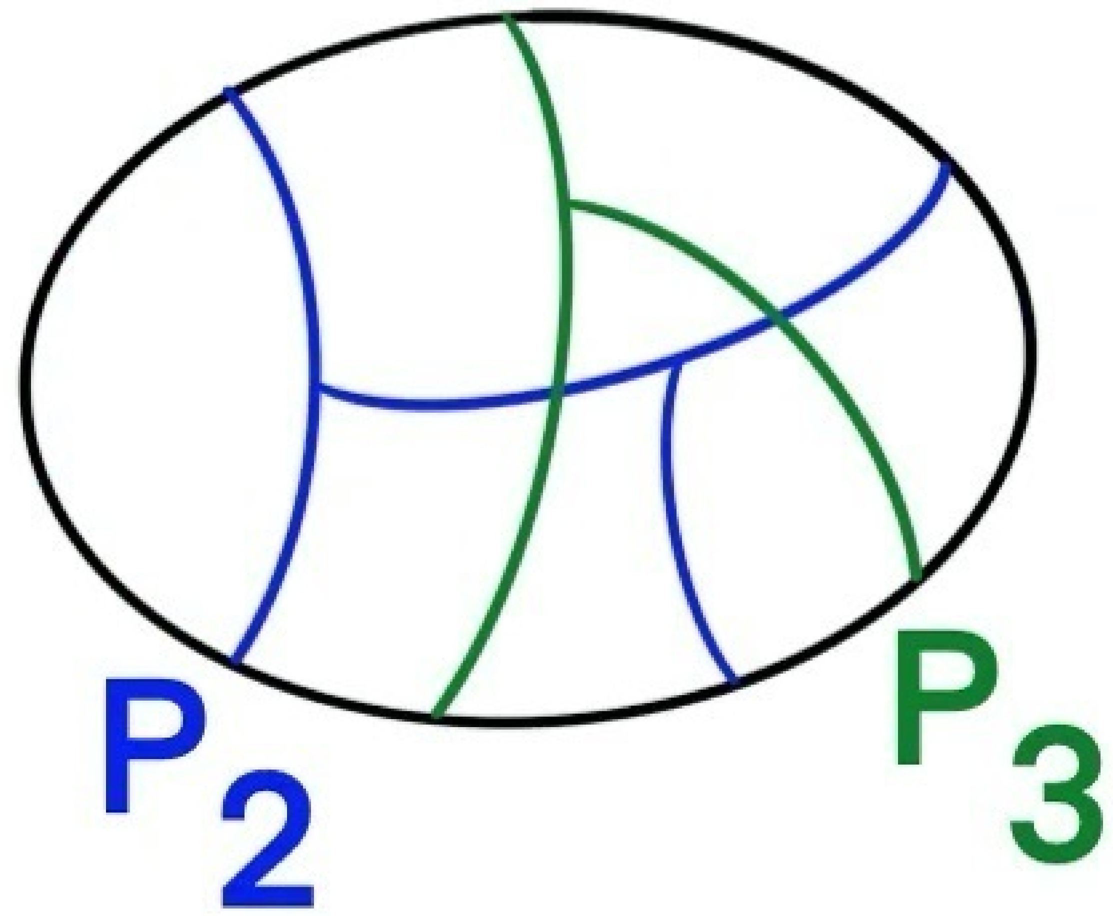
Theorem: Consider partitions $P_1, P_2, \ldots P_L$ of V s.t.



Our Congestion Approximator Construction

Theorem: Consider partitions $P_1, P_2, \ldots P_L$ of V s.t.

(1) $P_1=\{\{v\}:v\in V\}$ is singletons and $P_L=\{V\}$ is single cluster

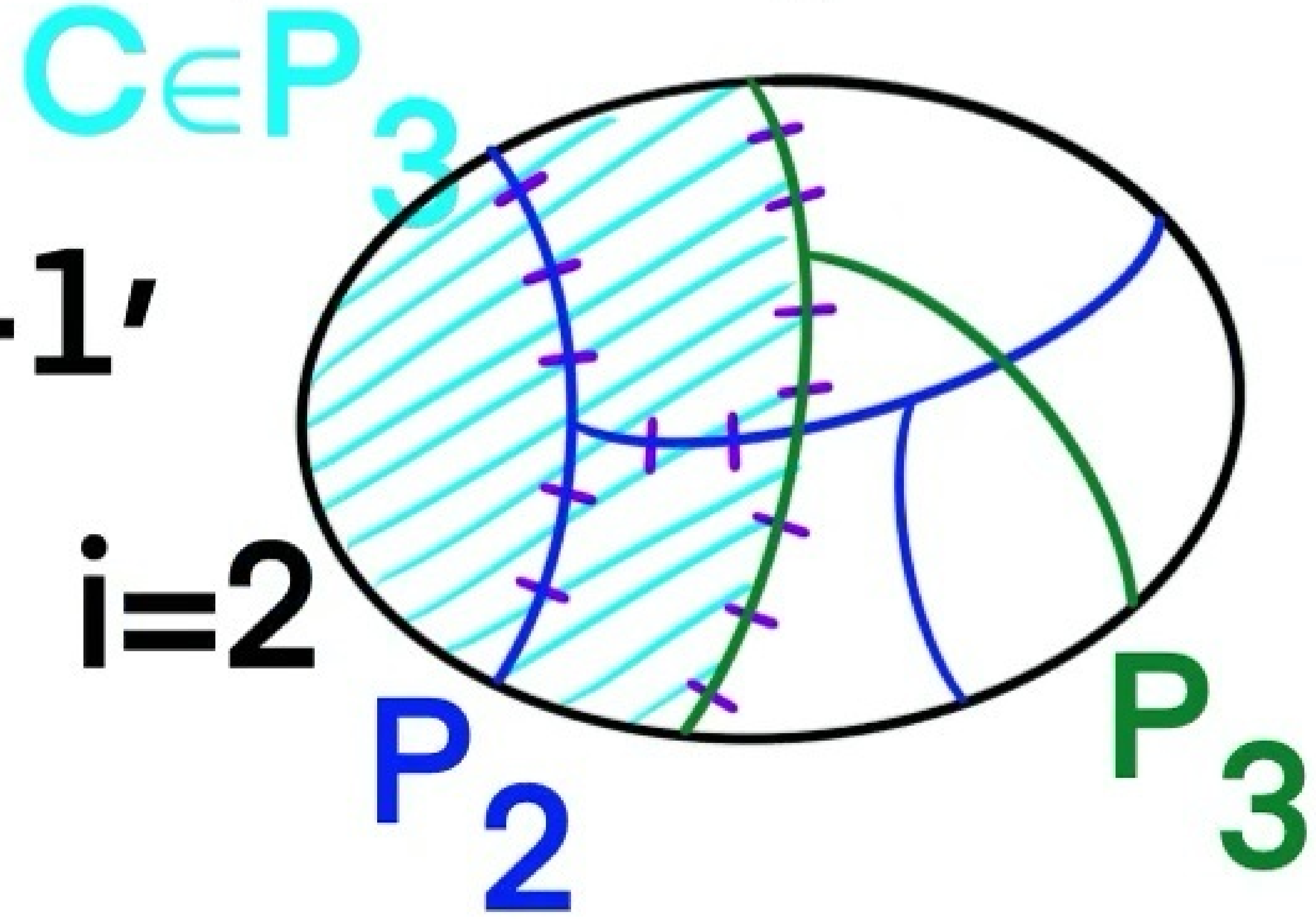


Our Congestion Approximator Construction

Theorem: Consider partitions $P_1, P_2, \ldots P_L$ of V s.t.

(1) $P_1=\{\{v\}:v\in V\}$ is singletons and $P_L=\{V\}$ is single cluster

(2) For each i and each cluster $C\in P_{i+1}$, $\partial P_i \cap E[C] \cup \partial C$ "mixes" in G[C] with congestion polylog

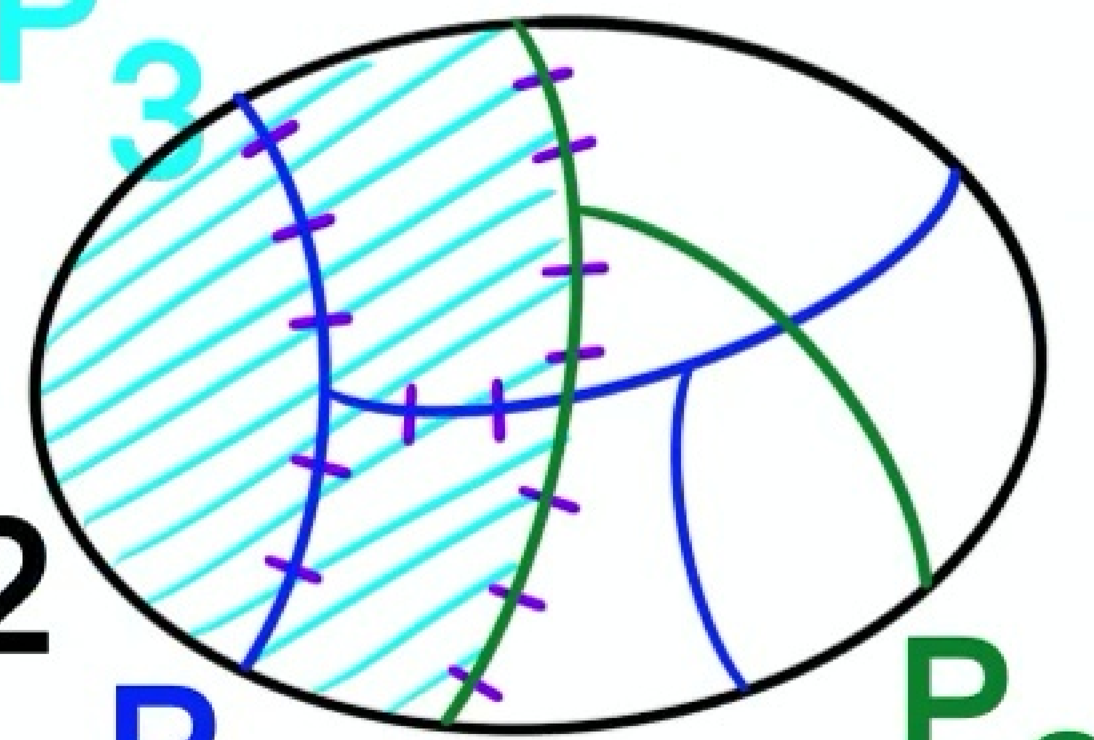


Our Congestion Approximator Construction

Theorem: Consider partitions $P_1, P_2, \ldots P_L$ of V s.t.

(1) $P_1 = \{\{v\} : v \in V\}$ is singletons and $P_L = \{V\}$ is single cluster

(2) For each i and each cluster $C \in P_{i+1}$,
$\partial P_i \cap E[C] \cup \partial C$ "mixes" in $G[C]$
with congestion polylog



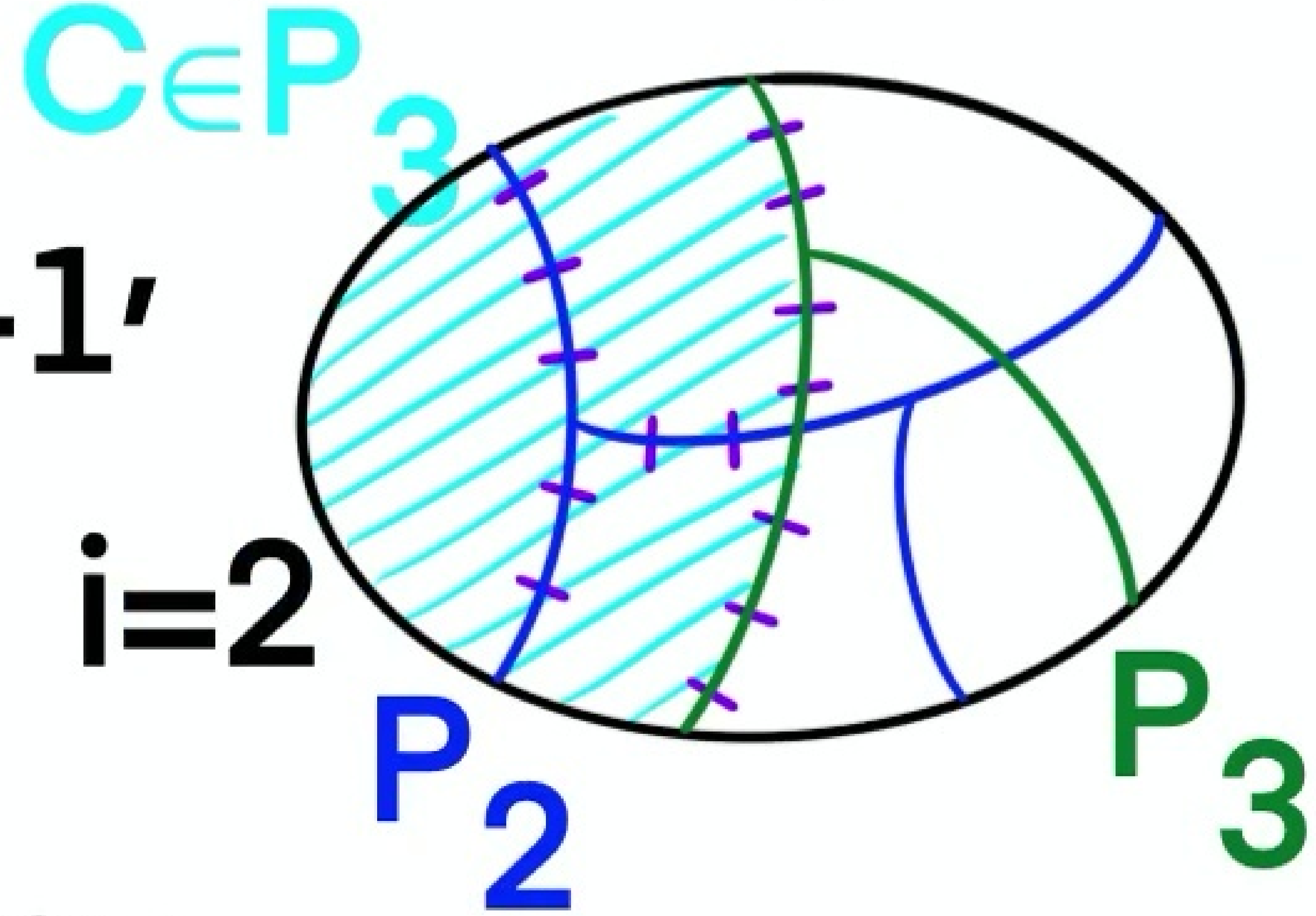
(3) For each i, there is a flow $\partial P_{i+1} \rightarrow \partial P_i$
with congestion polylog s.t.
each $e \in P_i$ sends 1 flow and
each $e \in P_{i+1}$ receives <=1/2 flow

Our Congestion Approximator Construction

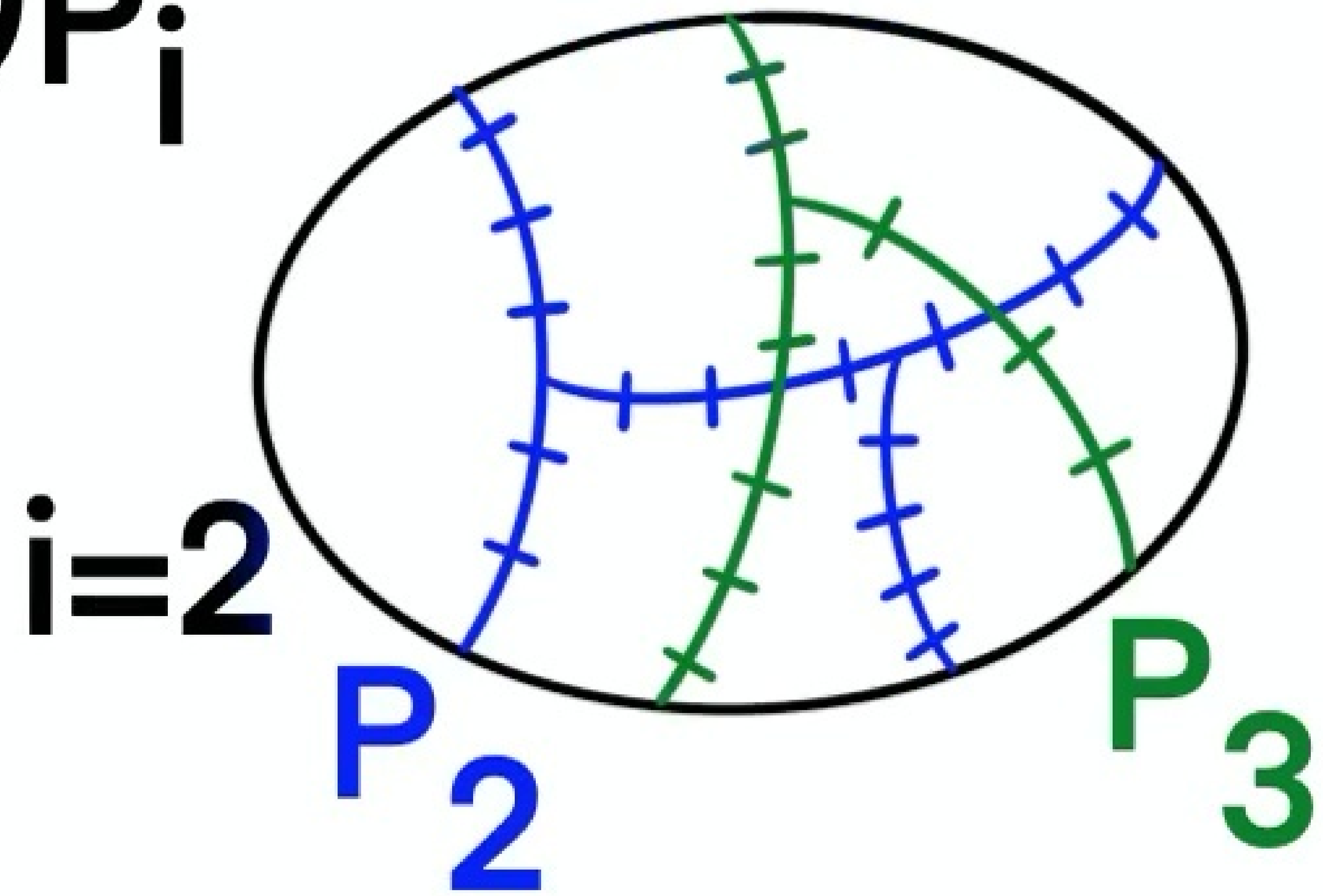
Theorem: Consider partitions $P_1, P_2, \ldots P_L$ of V s.t.

(1) $P_1=\{\{v\}:v\in V\}$ is singletons and $P_L=\{V\}$ is single cluster

(2) For each i and each cluster $C\in P_{i+1}$, $\partial P_i \cap E[C] \cup \partial C$ "mixes" in G[C] with congestion polylog



(3) For each i, there is a flow $\partial P_{i+1} \rightarrow \partial P_i$ with congestion polylog s.t. each $e\in P_i$ sends 1 flow and each $e\in P_{i+1}$ receives <=1/2 flow

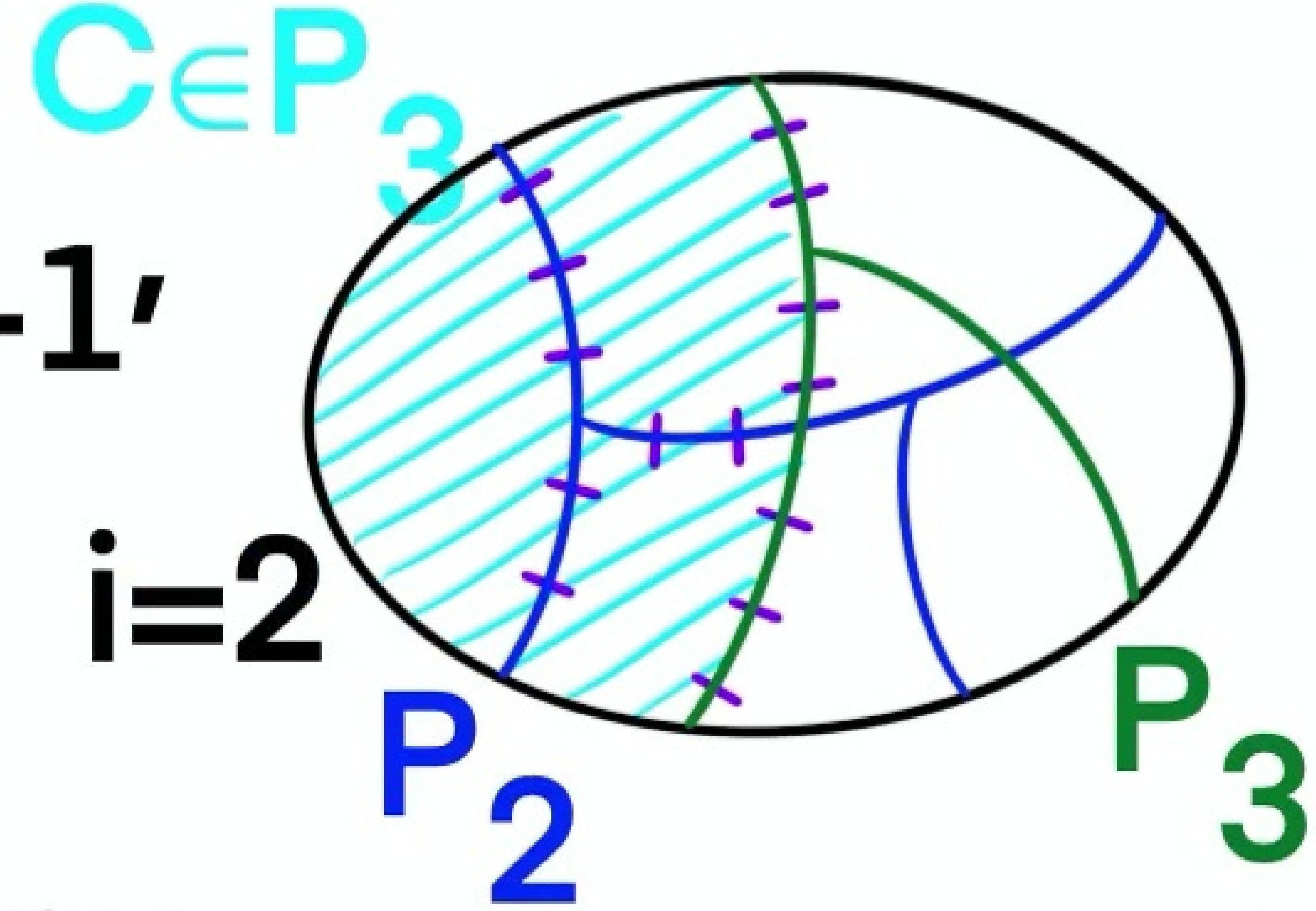


Our Congestion Approximator Construction

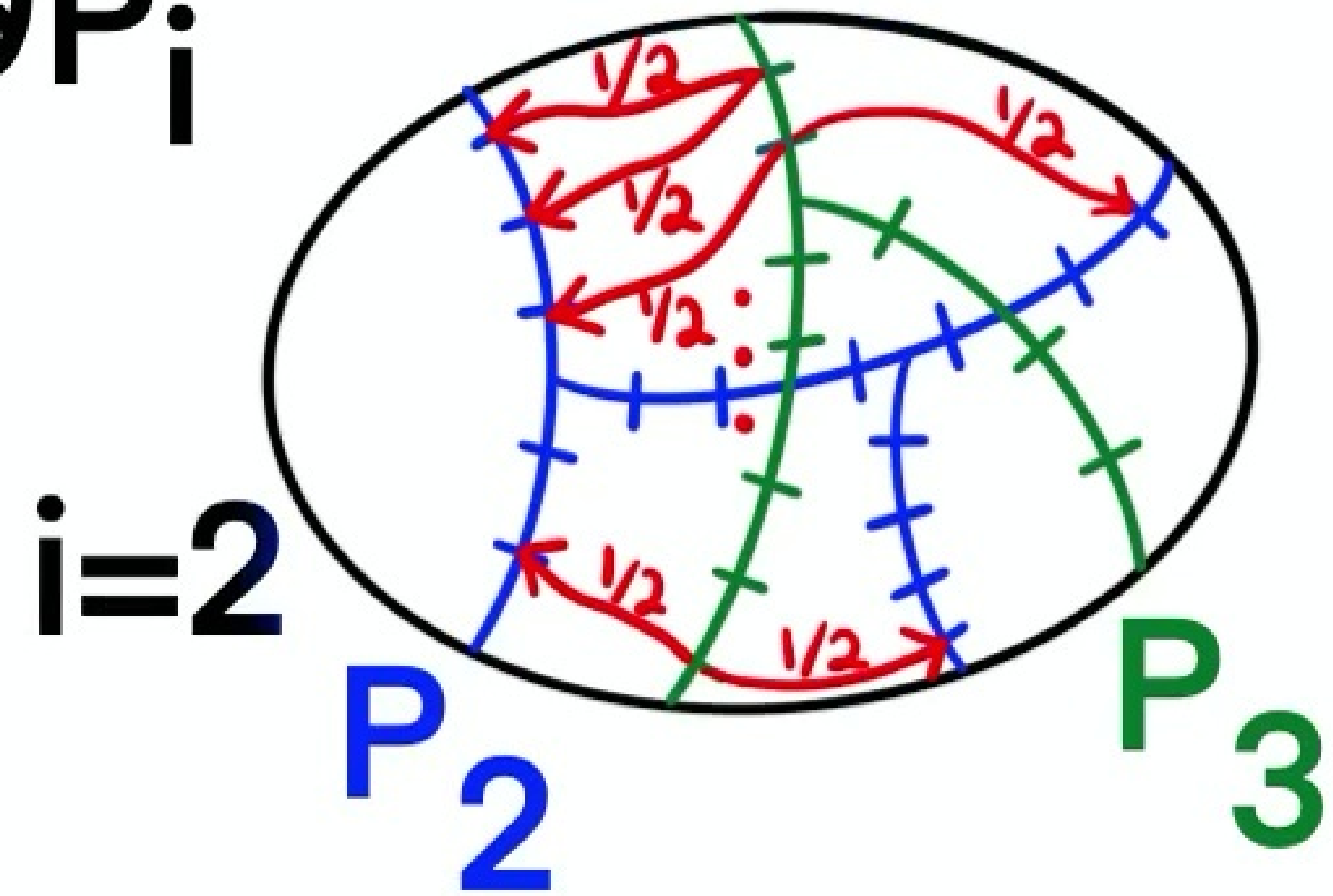
Theorem: Consider partitions $P_1, P_2, \ldots P_L$ of V s.t.

(1) $P_1=\{\{v\}:v\in V\}$ is singletons and $P_L=\{V\}$ is single cluster

(2) For each i and each cluster $C\in P_{i+1}$, $\partial P_i \cap E[C] \cup \partial C$ "mixes" in G[C] with congestion polylog



(3) For each i, there is a flow $\partial P_{i+1} \rightarrow \partial P_i$ with congestion polylog s.t. each $e\in P_i$ sends 1 flow and each $e\in P_{i+1}$ receives <=1/2 flow



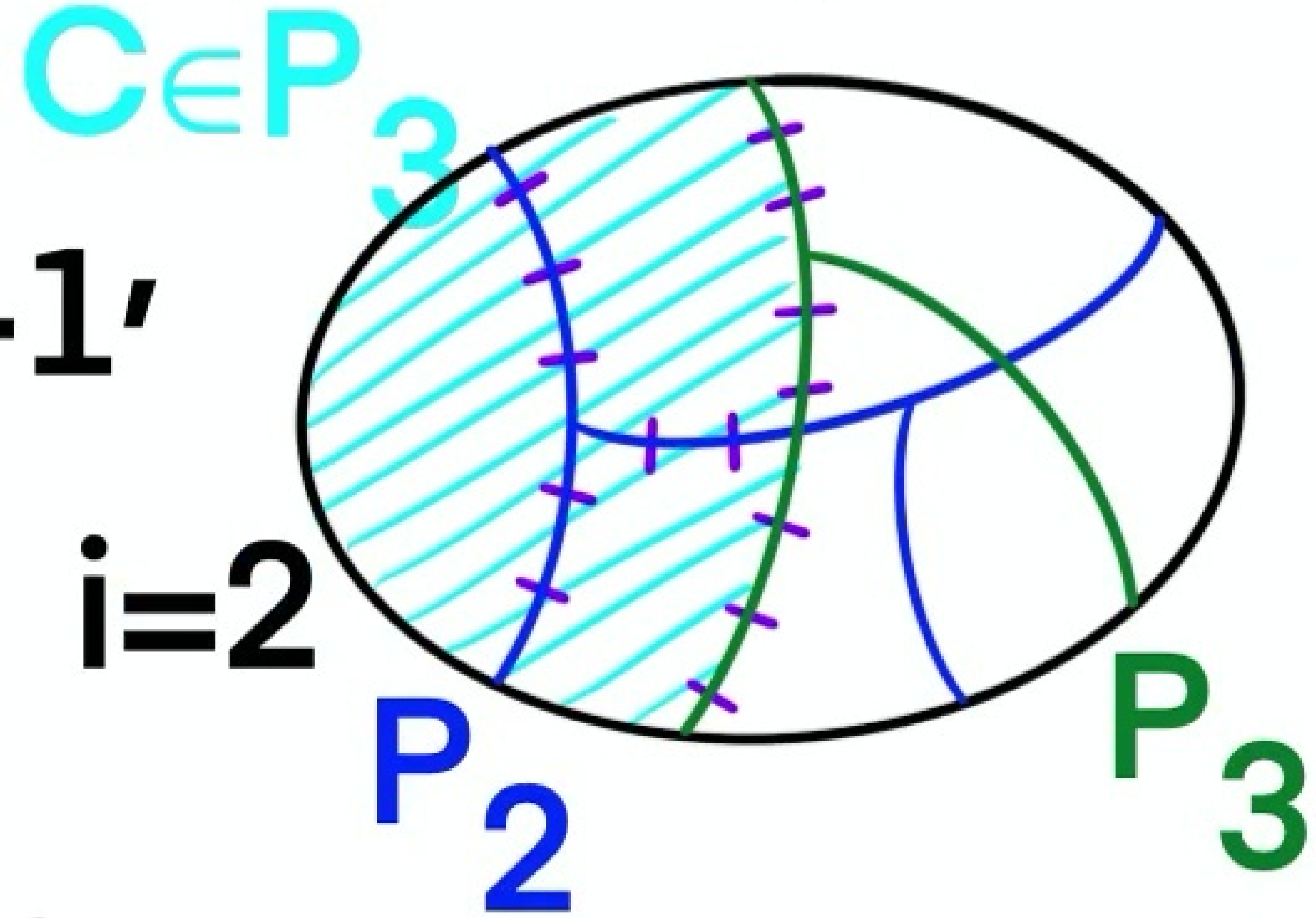
Our Congestion Approximator Construction

Theorem: Consider partitions $P_1, P_2, \ldots P_L$ of V s.t.

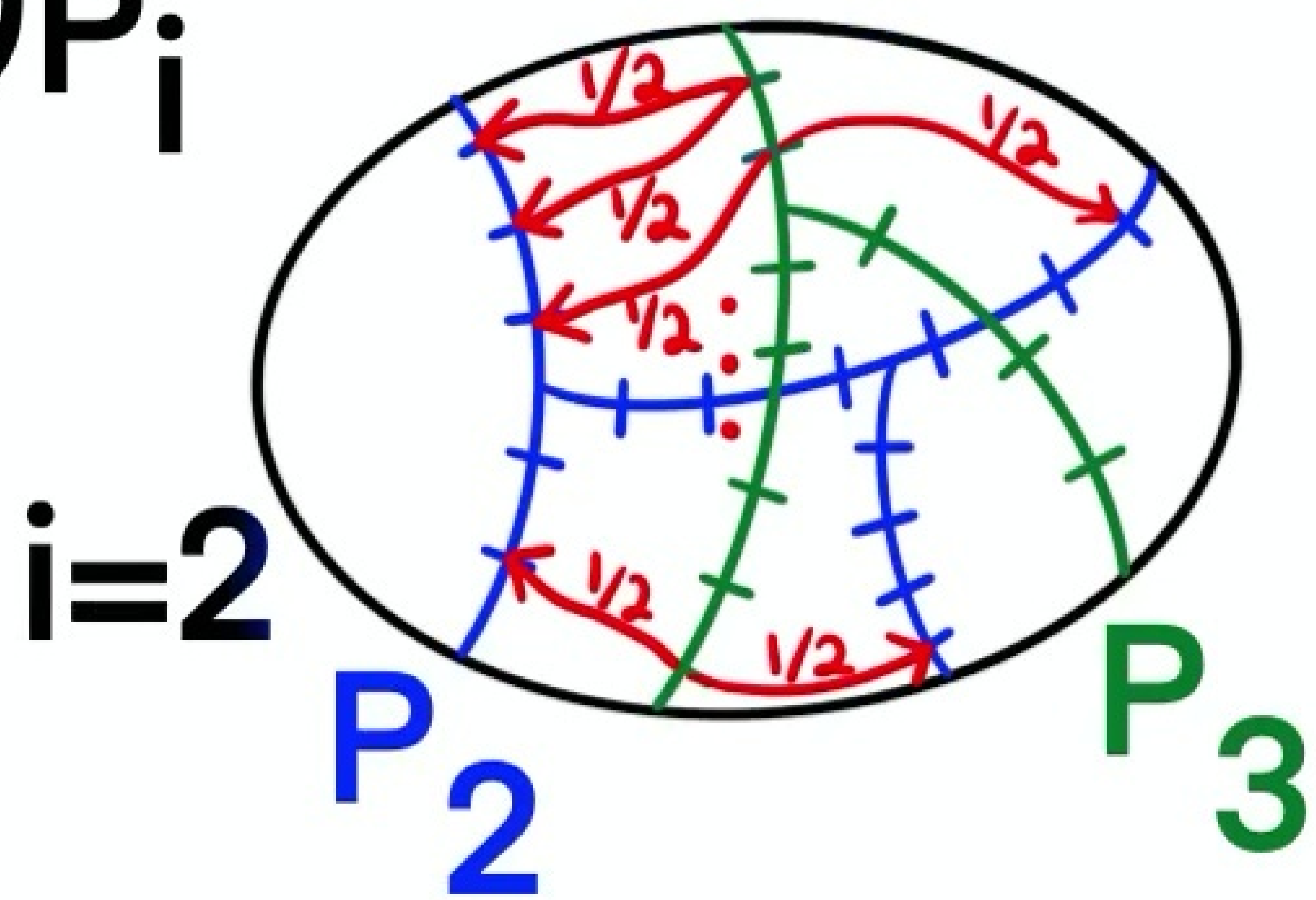
(1) $P_1=\{\{v\}:v\in V\}$ is singletons and $P_L=\{V\}$ is single cluster

(2) For each i and each cluster $C\in P_{i+1}$, $\partial P_i \cap E[C] \cup \partial C$ "mixes" in G[C] with congestion polylog

Actual: slightly weaker



(3) For each i, there is a flow $\partial P_{i+1} \rightarrow \partial P_i$ with congestion polylog s.t.
each $e\in P_i$ sends 1 flow and
each $e\in P_{i+1}$ receives <=1/2 flow



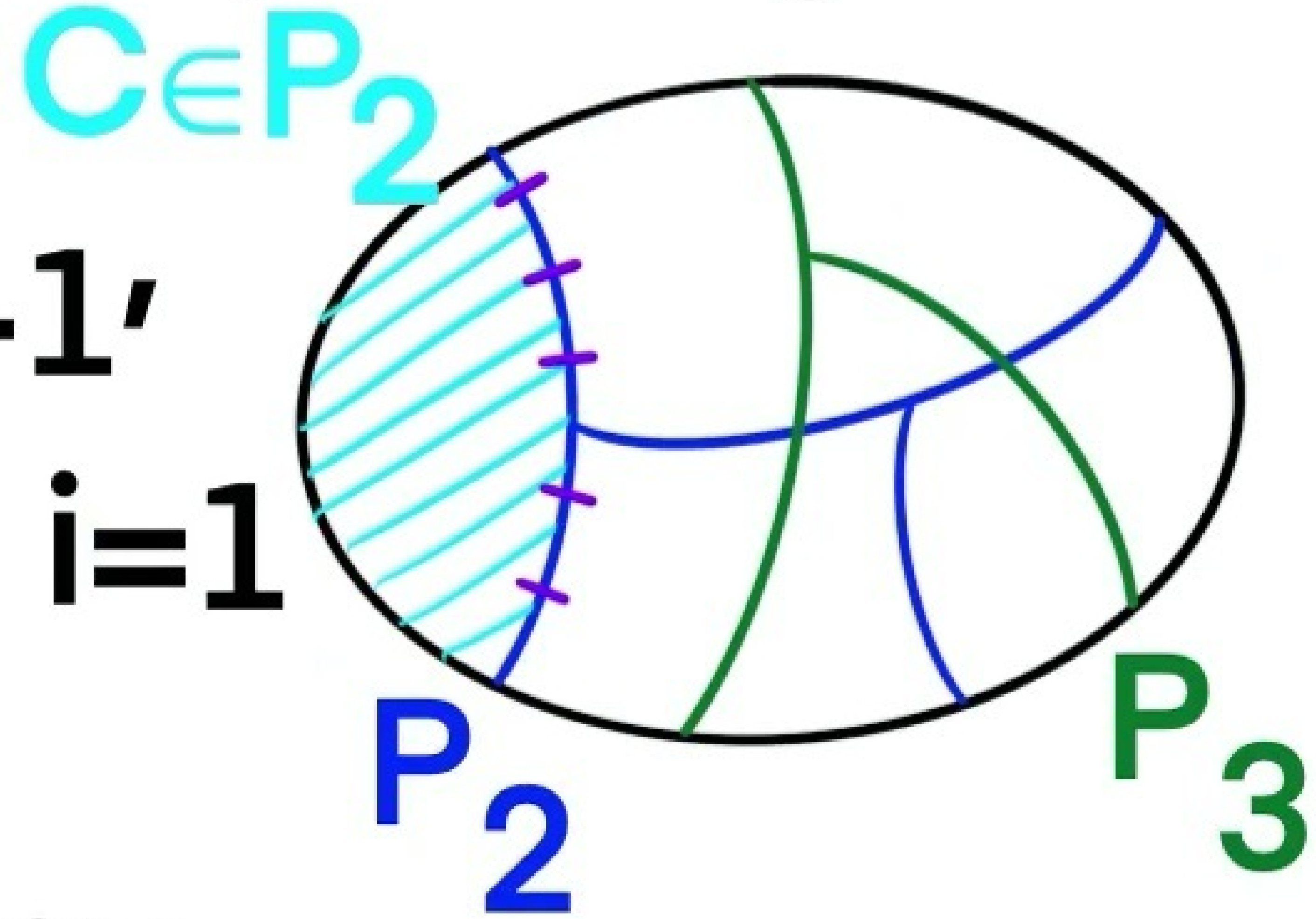
Our Congestion Approximator Construction

Theorem: Consider partitions $P_1, P_2, \ldots P_L$ of V s.t.

(1) $P_1=\{\{v\}:v\in V\}$ is singletons and $P_L=\{V\}$ is single cluster

(2) For each i and each cluster $C\in P_{i+1}$, $\partial P_i \cap E[C] \cup \partial C$ "mixes" in G[C] with congestion polylog

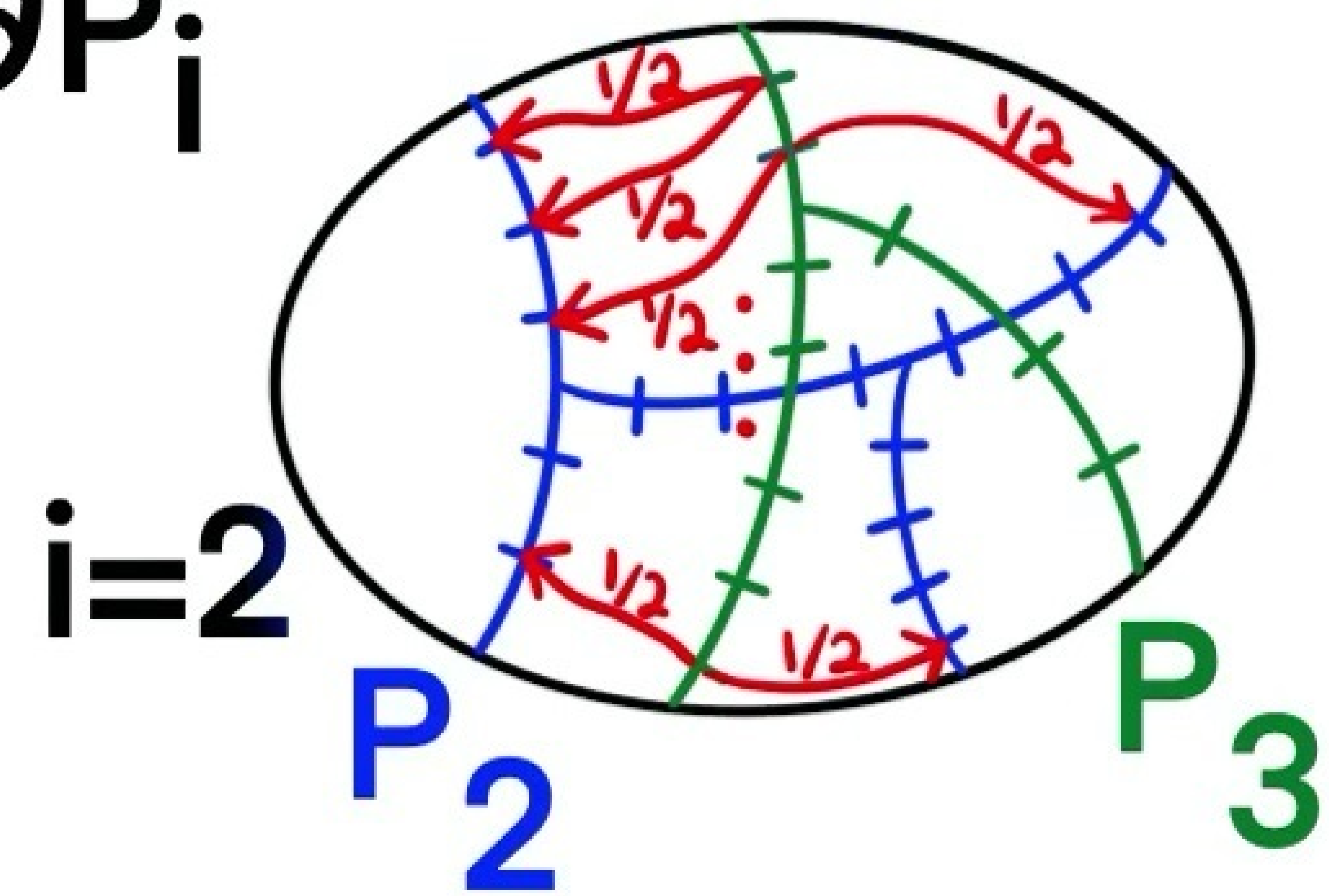
Actual: slightly weaker



(3) For each i, there is a flow $\partial P_{i+1}\rightarrow\partial P_i$ with congestion polylog s.t.

each $e\in P_i$ sends 1 flow and

each $e\in P_{i+1}$ receives <=1/2 flow



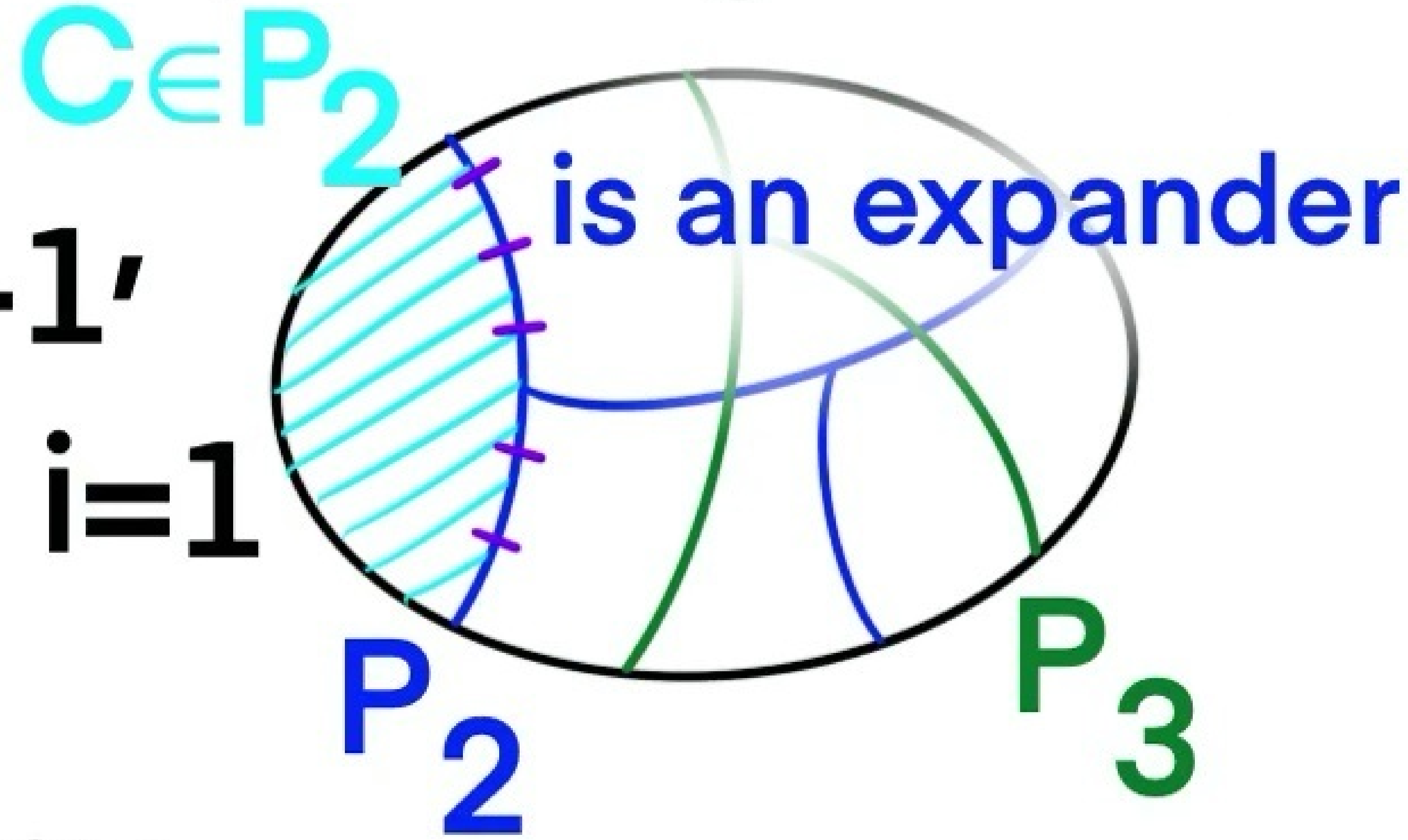
Our Congestion Approximator Construction

Theorem: Consider partitions $P_1, P_2, \ldots P_L$ of V s.t.

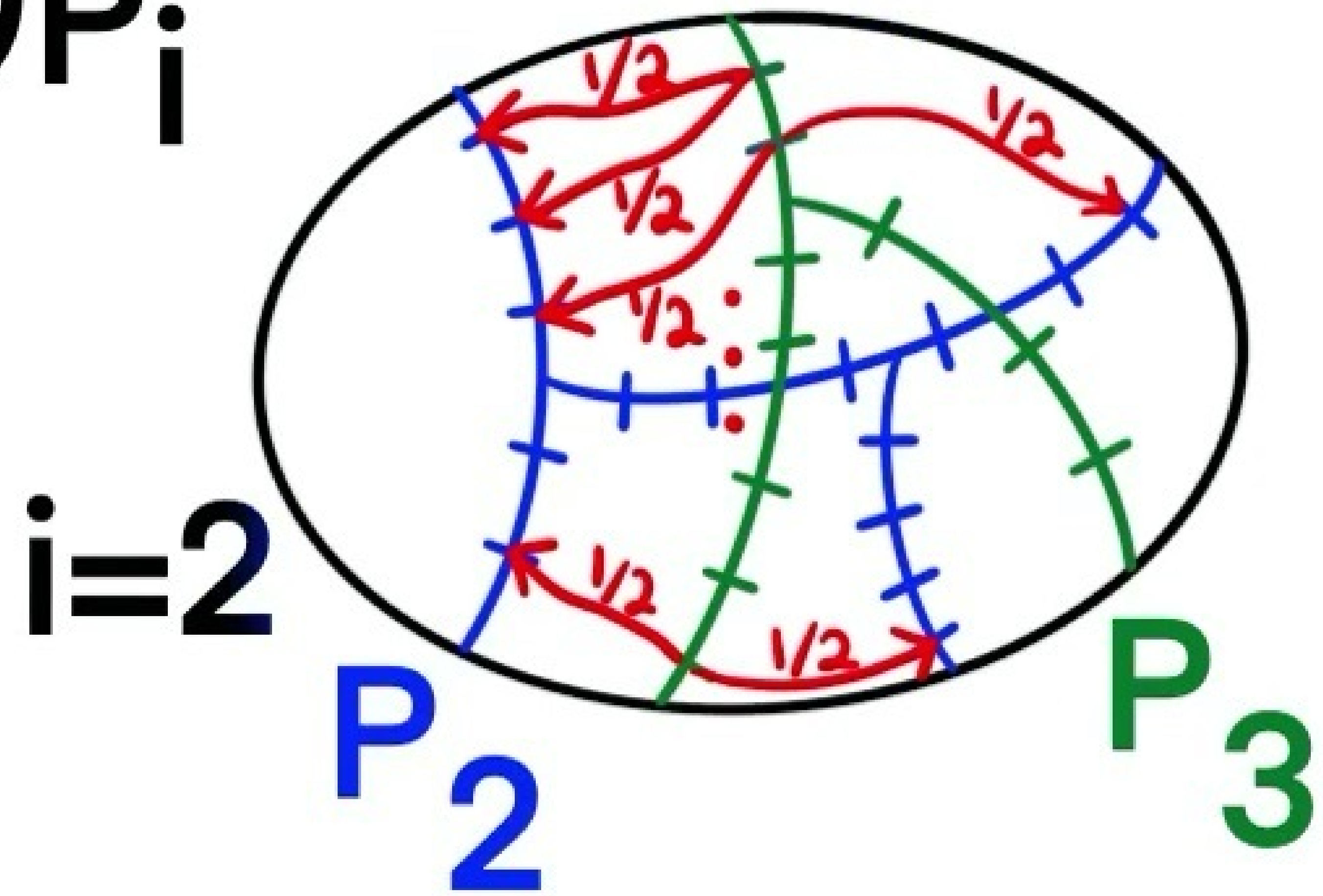
(1) $P_1=\{\{v\}: v \in V\}$ is singletons and $P_L=\{V\}$ is single cluster

(2) For each i and each cluster $C \in P_{i+1}$,
$\partial P_i \cap E[C] \cup \partial C$ "mixes" in $G[C]$
with congestion polylog

Actual: slightly weaker



(3) For each i, there is a flow $\partial P_{i+1} \rightarrow \partial P_i$
with congestion polylog s.t.
each $e \in P_i$ sends 1 flow and
each $e \in P_{i+1}$ receives <=1/2 flow



Our Congestion Approximator Construction

Theorem: Consider partitions $P_1, P_2, \ldots P_L$ of V s.t.

⋮

For each i, let $R_{>=i}$ be the common refinement of partitions $P_i, P_{i+1}, \ldots, P_L$

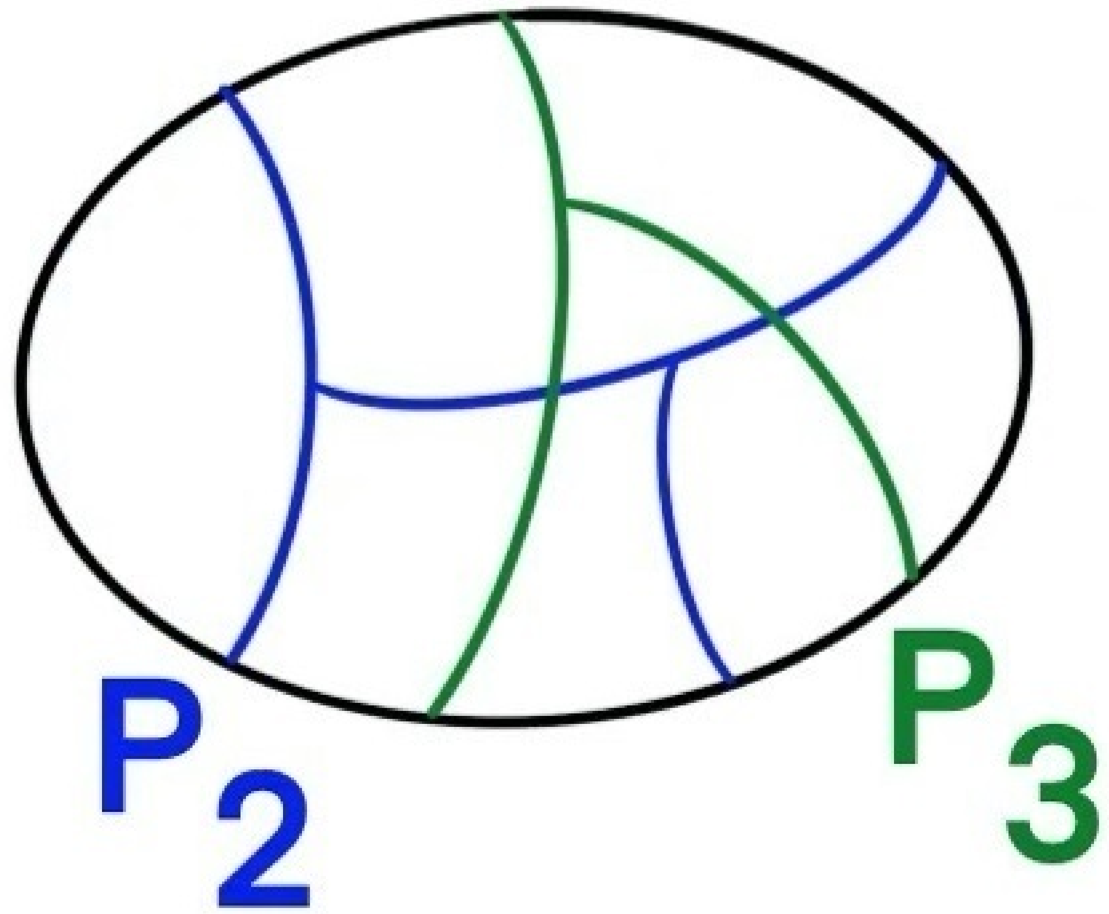
Then $\mathcal{C} = \cup_i R_{>=i}$ is a congestion approximator with quality polylog.

Our Congestion Approximator Construction

Theorem: Consider partitions $P_1, P_2, \dots P_L$ of V s.t.

⋮

For each i, let $R_{>=i}$ be the common refinement of partitions $P_i, P_{i+1}, \dots, P_L$



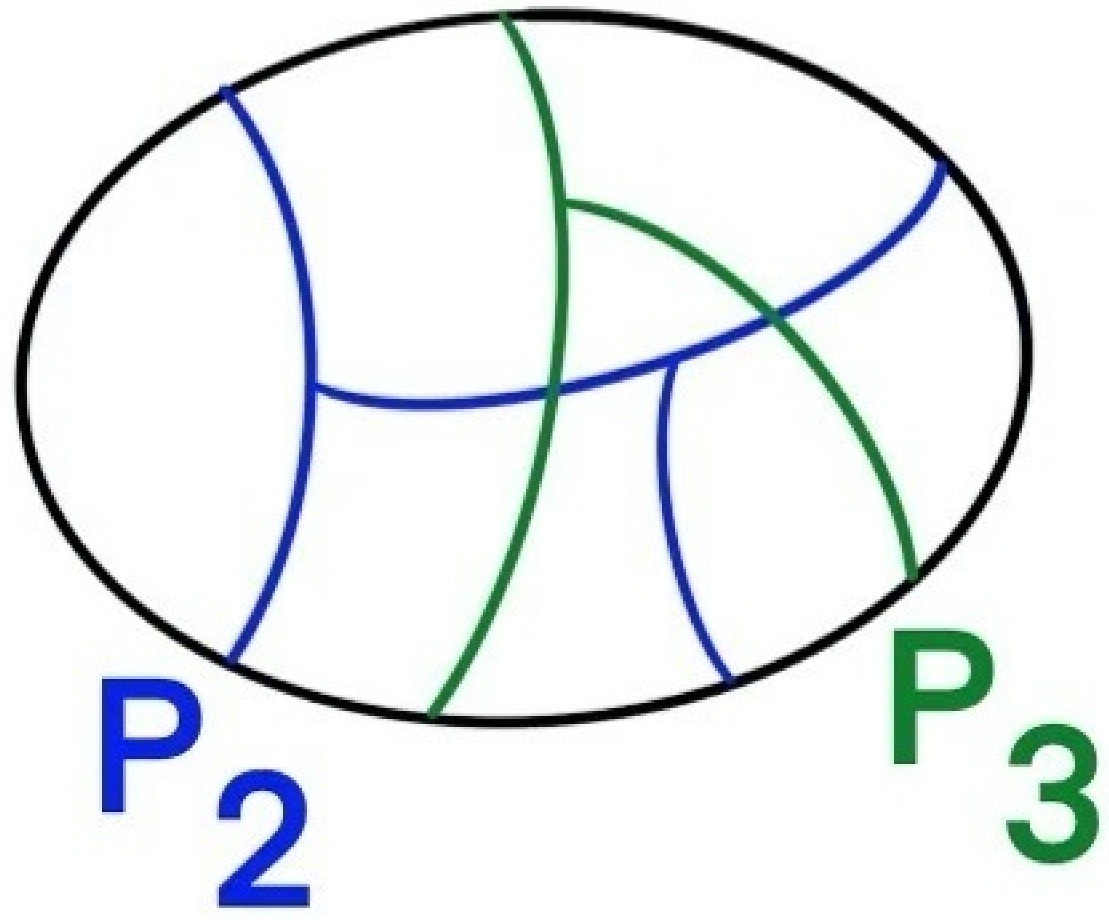
Then $\mathcal{C} = \cup_i R_{>=i}$ is a congestion approximator with quality polylog.

Our Congestion Approximator Construction

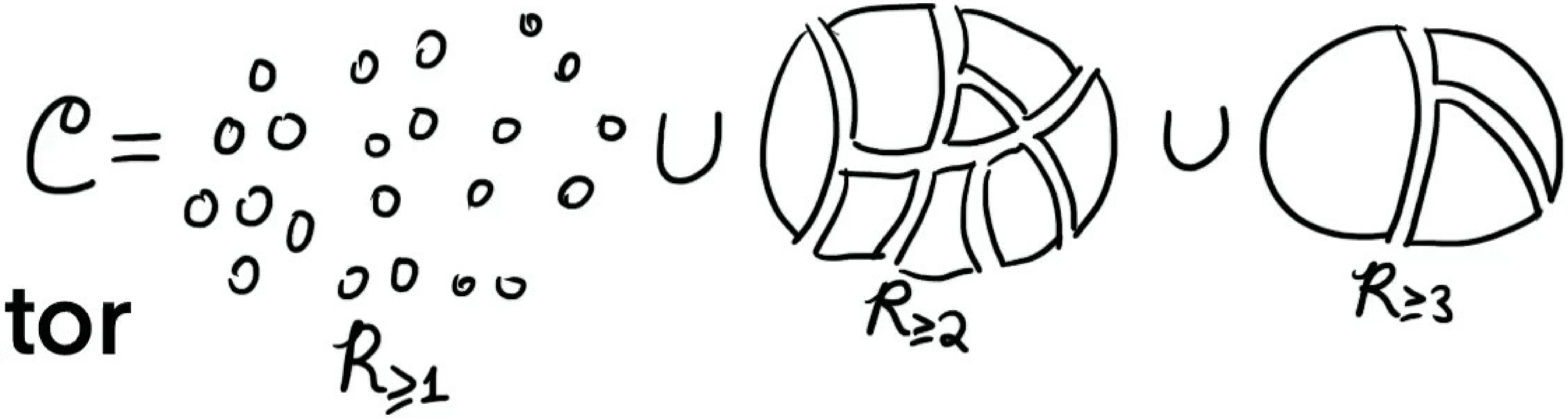
Theorem: Consider partitions $P_1, P_2, \ldots P_L$ of V s.t.

⋮

For each i, let $R_{>=i}$ be the common refinement of partitions $P_i, P_{i+1}, \ldots, P_L$



Then $\mathcal{C} = \cup_i R_{>=i}$ is a congestion approximator with quality polylog.



Constructing the Congestion Approximator

P_2 is an expander decomposition (doable)

Constructing the Congestion Approximator

P_2 is an expander decomposition (doable)

In general, use $P_1, \ldots, P_{i-1}$ to construct P_i

Constructing the Congestion Approximator

P_2 is an expander decomposition (doable)

In general, use $P_1, \ldots, P_{i-1}$ to construct P_i

Max-flow calls required, but use structure of $P_1, \ldots, P_{i-1}$ to build "pseudo"-congestion approximator sufficient for the specialized max-flow calls

Conclusion

Bottom-up congestion approximator construction,
in contrast to top-down of [Räcke, Shah, Taubig '14]

Conclusion

Bottom-up congestion approximator construction,
in contrast to top-down of [Räcke, Shah, Taubig '14]

No recursive max-flow required $\rightarrow$ cleaner algorithm

Conclusion

Bottom-up congestion approximator construction,
in contrast to top-down of [Räcke, Shah, Taubig '14]

No recursive max-flow required → cleaner algorithm

Number of polylog factors still high (unspecified).
Open question: reduce number of polylog factors?