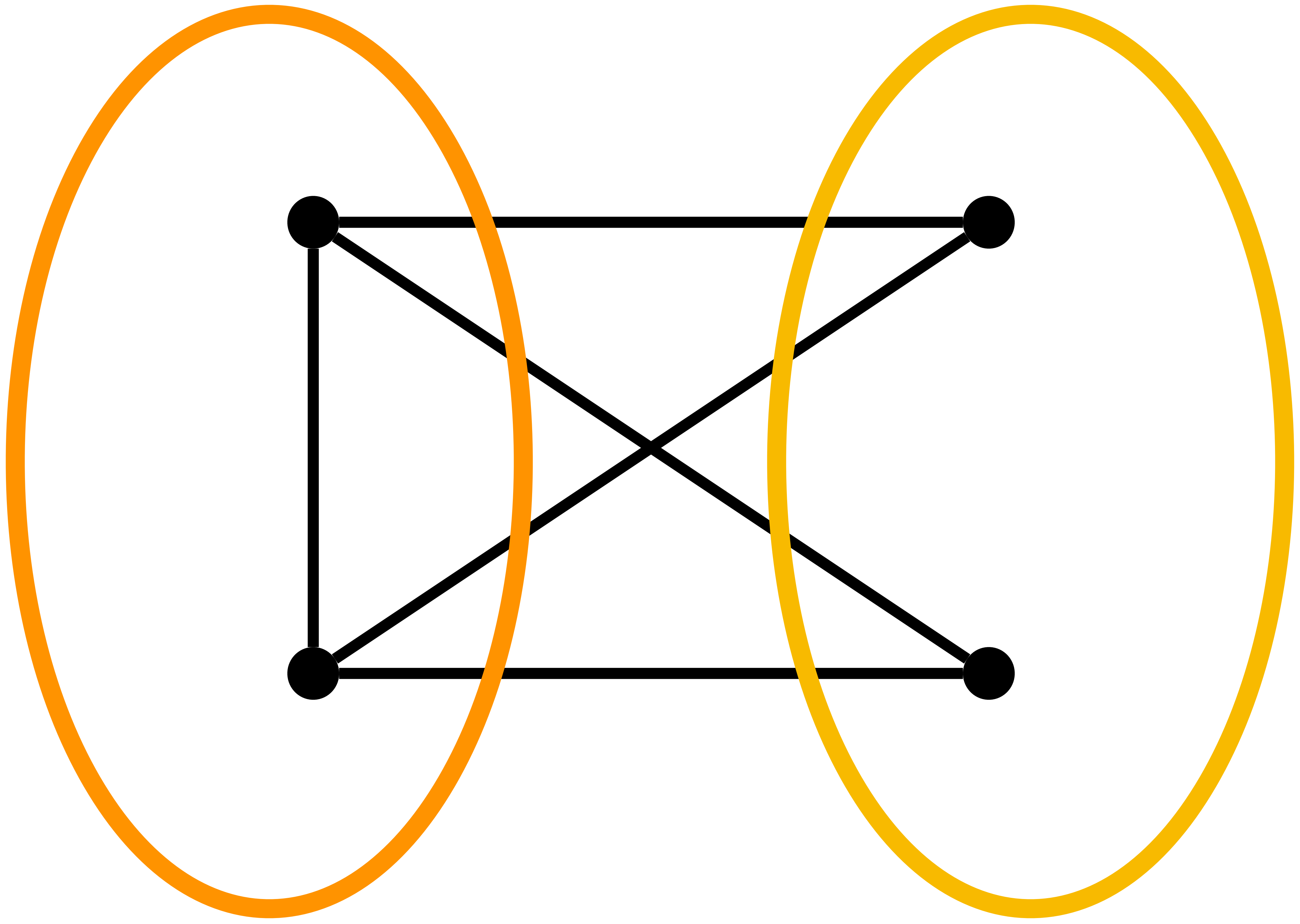
Improved streaming approximations for Max-DICUT via local snapshots

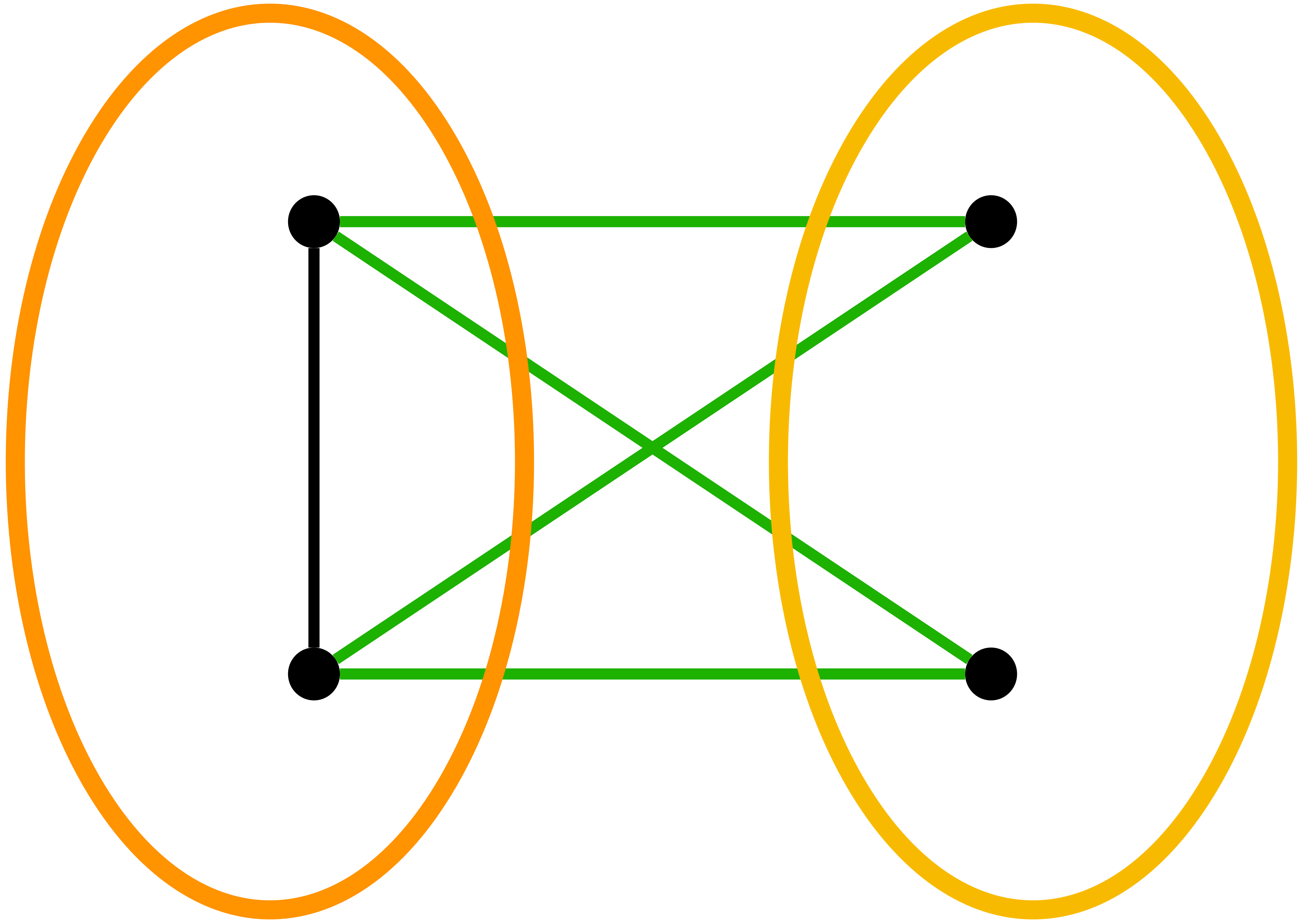
Santhoshini Velusamy (TTIC)

Based on joint works with Raghuvansh Saxena, Noah Singer, and Madhu Sudan

Max-CUT



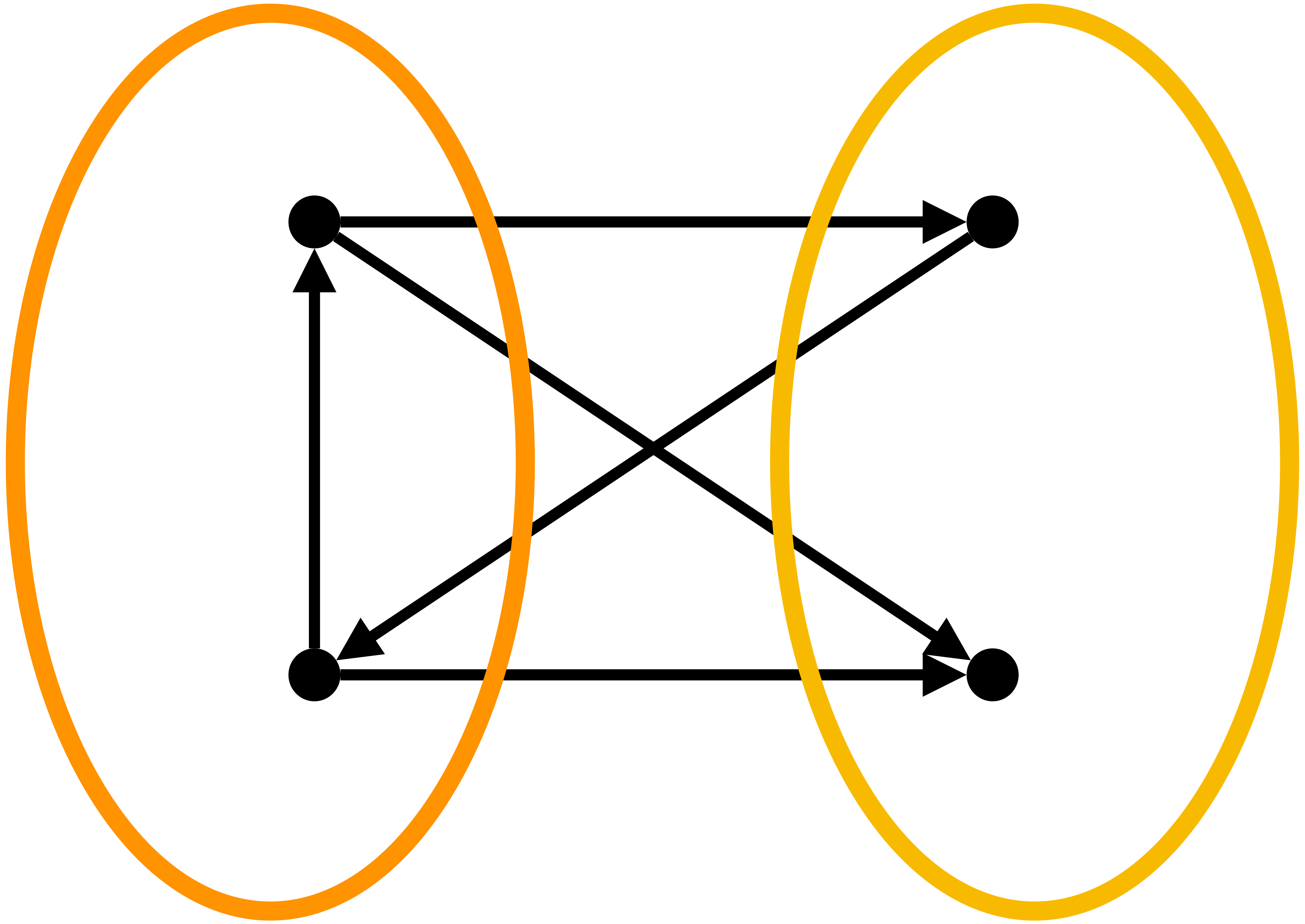
Max-CUT



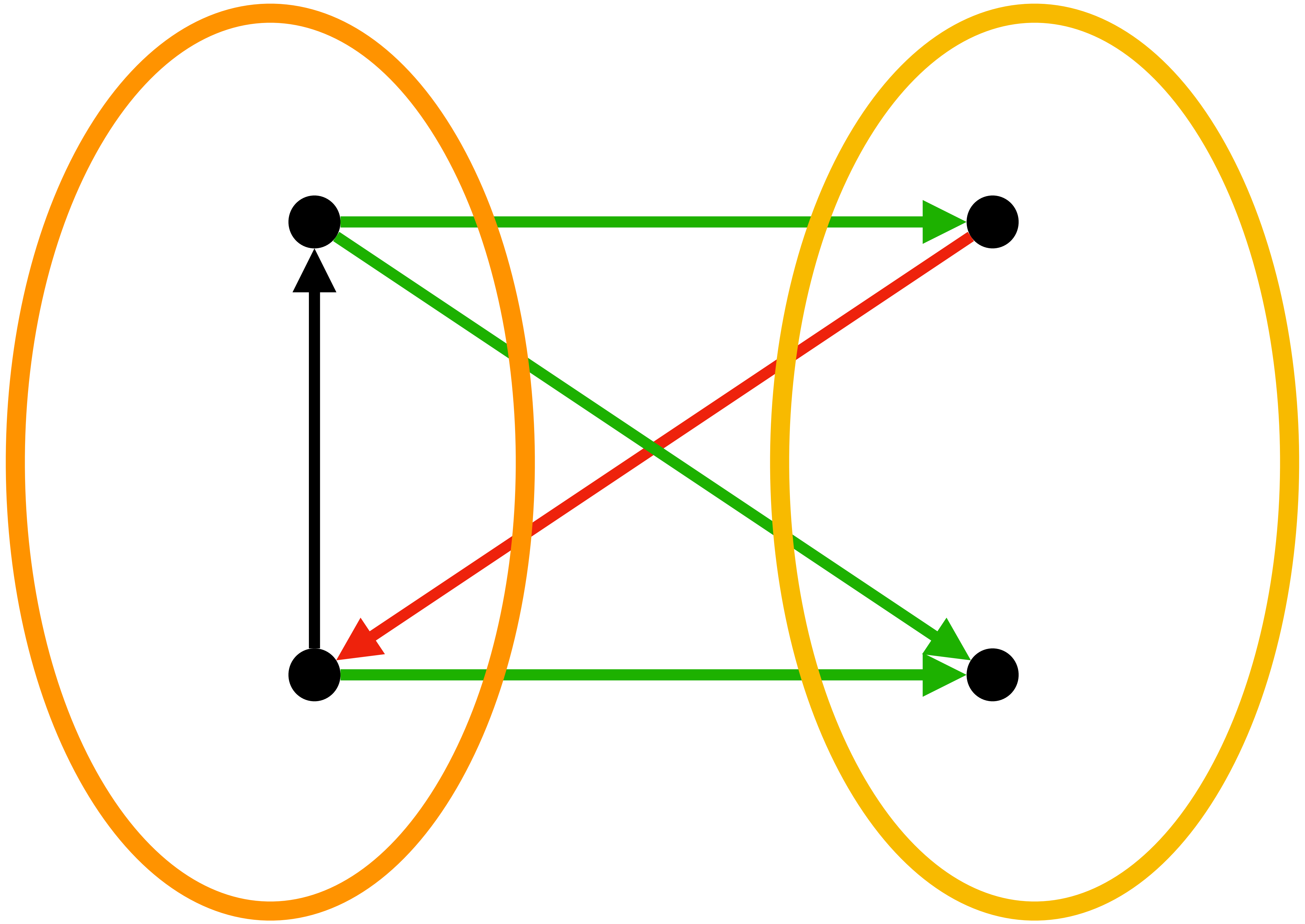
CUT size = 4

Output largest CUT size

Max-DICUT



Max-DICUT

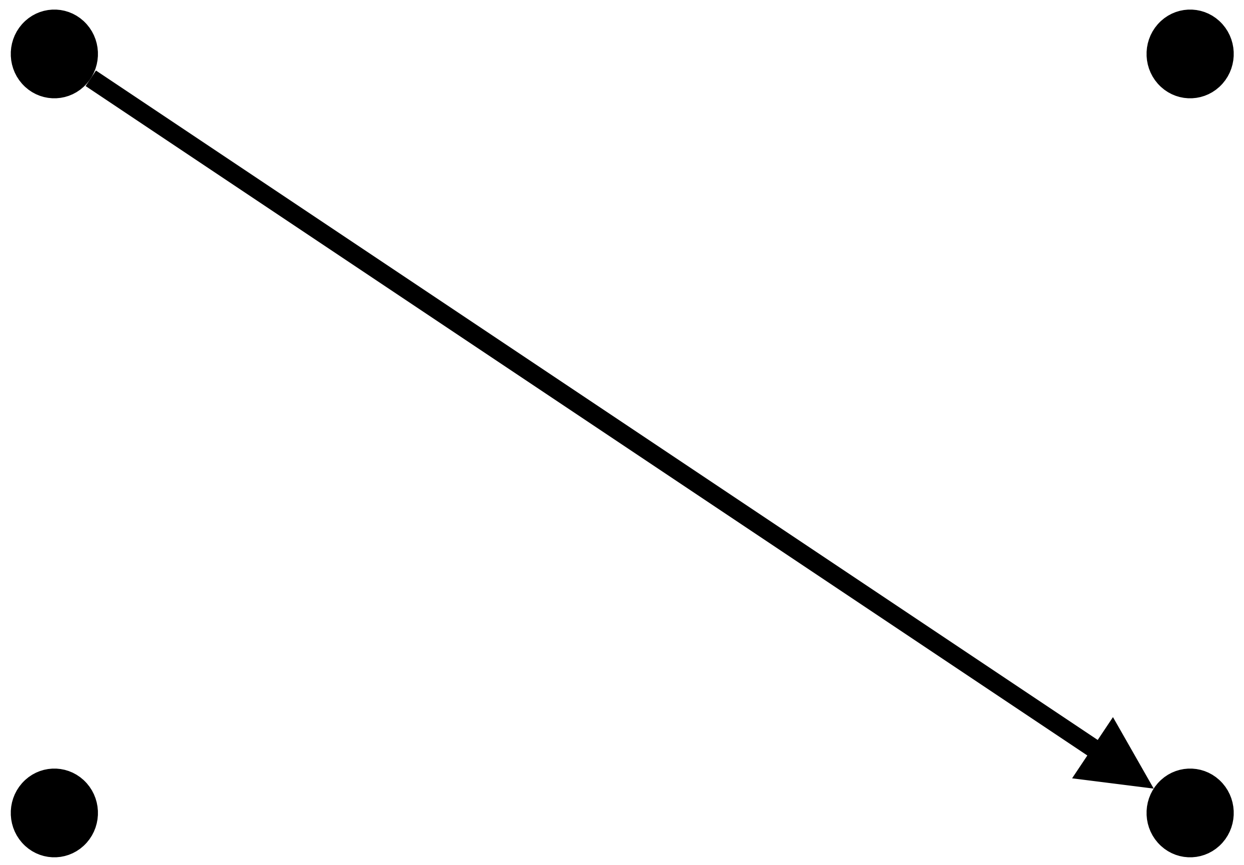


DICUT size = 3

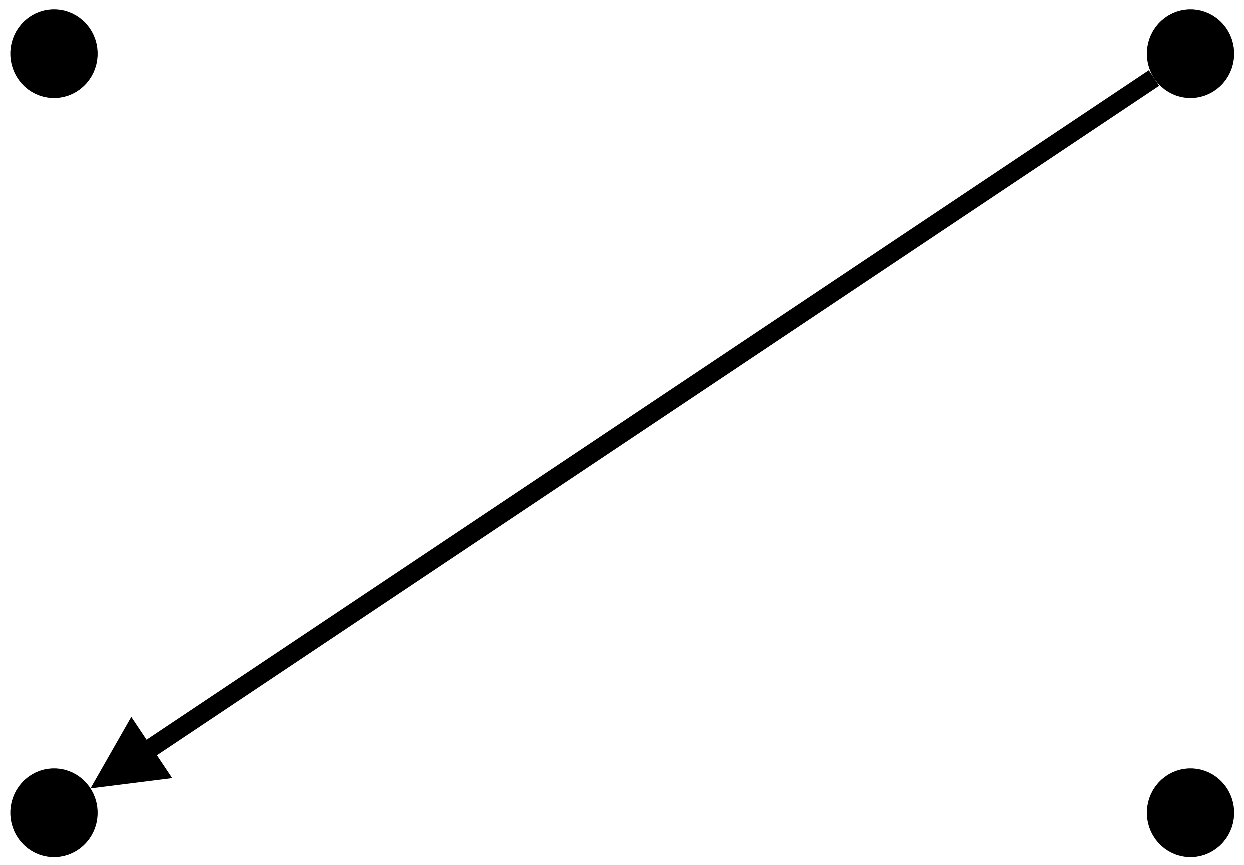
Output largest DICUT size

Streaming setting

Streaming setting



Streaming setting



Streaming setting

Streaming setting

Compute the size of the largest CUT/DICUT in small space

$0 < \alpha < 1$, α approximation: outputs T such that: $\alpha \cdot \text{OPT} \leq T \leq \text{OPT}$

Folklore approximations

- Arbitrary constant approximation in $\tilde{O}(n)$ space:
  - sample $O(n/\epsilon^2)$ random edges
  - compute Max-CUT/Max-DICUT value (possibly in exponential time) to obtain $(1-\epsilon)$ approximation
- Trivial approximation in $O(\log n)$ space:
  - Max-CUT: count the number of edges m and output $m/2$
  - Max-DICUT: output $m/4$

What approximations are possible in $o(n)$ space?

Max-CUT

- central problem for lower bounds
- [Kapralov Krachun 19] no non-trivial approximation in $o(n)$ space
- [Chou Golovnev Sudan V 22] extension to several CSPs

Max-DICUT

- central problem for algorithms
- [Guruswami Vellingker V 17] “bias-based” algorithm with non-trivially approximation in $O(\log n)$ space
- [Chou Golovnev V 20] “bias-based” algorithm is optimal in $o(\sqrt{n})$ space
- [Chou Golovnev Sudan V 21] extension to all CSPs

Goals of this talk

- [Saxena Singer Sudan V 23] beat "bias-based" algorithm for Max-DICUT in $\tilde{O}(\sqrt{n})$ space

- [Saxena Singer Sudan V *upcoming*] $1/2$-approximation for Max-DICUT in $o(n)$ space for bounded-degree graphs → optimal

Simulate local algorithms for Max-DICUT in the streaming setting by capturing local snapshots of the graph

Local algorithm I: oblivious algorithms

- Introduced by Feige and Jozeph in 2015
- Bias of a vertex $v = \dfrac{\text{out}(v) - \text{in}(v)}{\text{deg}(v)} \in [-1,1]$
- Oblivious algorithm $\mathcal{O}$: (randomly) assign every vertex based only on its bias
- Approximation ratio = $\dfrac{\mathbb{E}[\text{DICUT}(\mathcal{O})]}{\text{OPT}}$

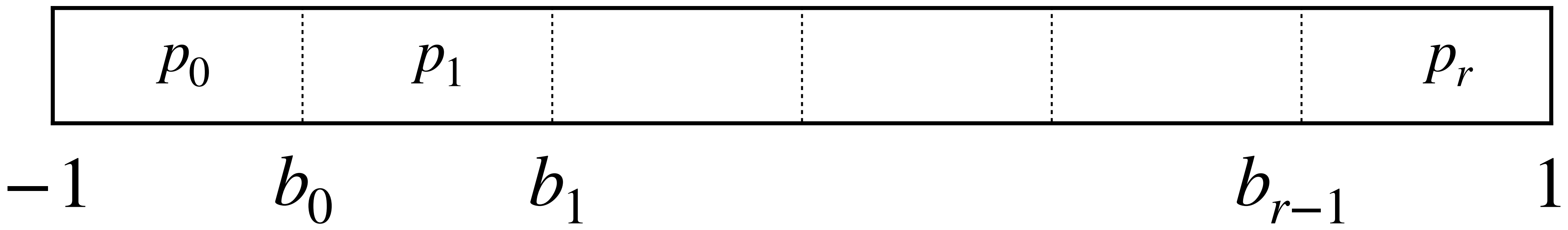
Oblivious approximations for Max-DICUT

- [FJ15] Oblivious algorithm with approximation ratio > 0.483
- [CGV20] Best $o(\sqrt{n})$ space streaming approximation for Max-DICUT is ≈ 0.44
- [SSSV23] Implement [FJ15] algorithm using only $\tilde{O}(\sqrt{n})$ space in the streaming setting

Streaming implementation

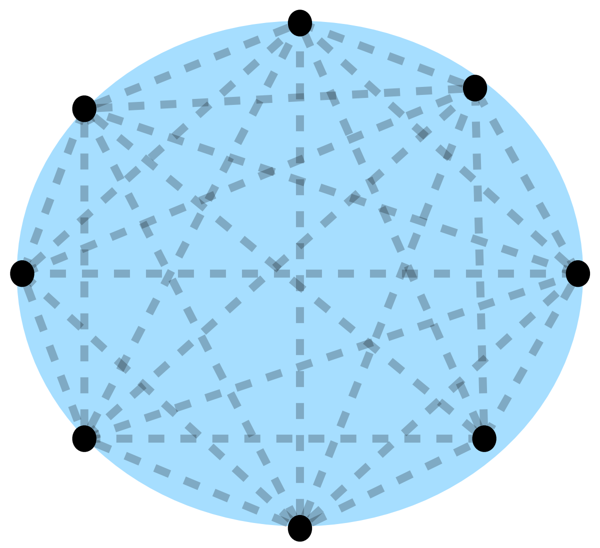
- [FJ15] algorithm has only constant number of bias classes
  - Bias intervals

$$-1 < b_0 < b_1 < \cdots < b_{r-1} < 1$$

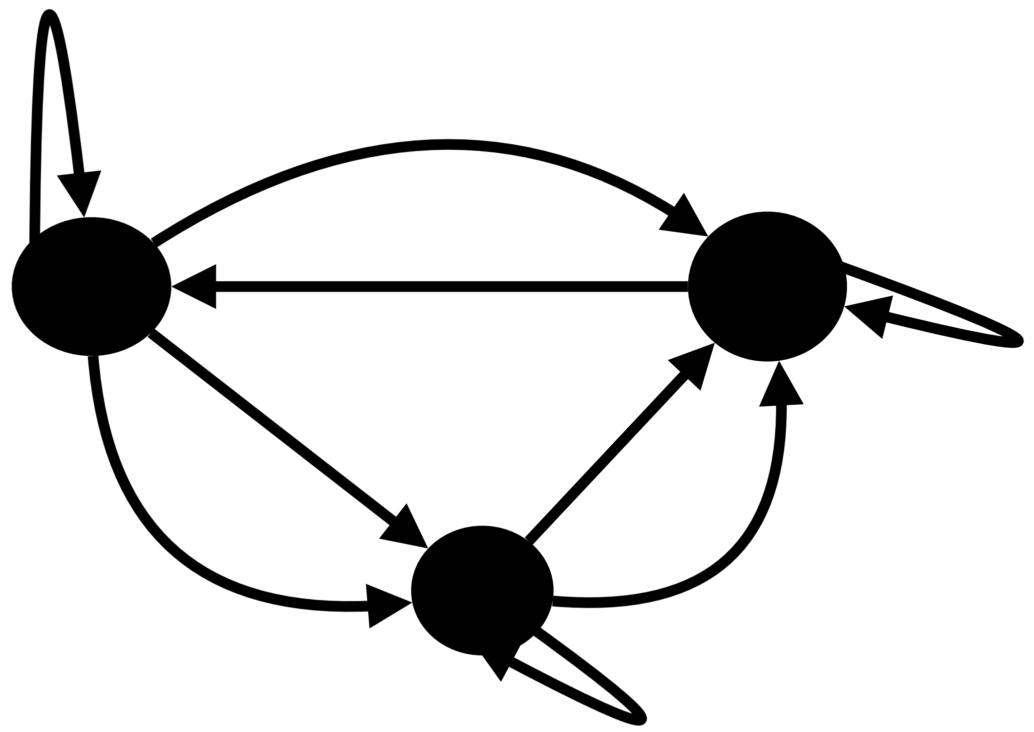


- Goal: Output $\mathbb{E}[\text{DICUT}(\mathcal{O})]$ to get > 0.483 approximation to Max-DICUT

- Suffices to compute a "constant-sized" snapshot of the graph

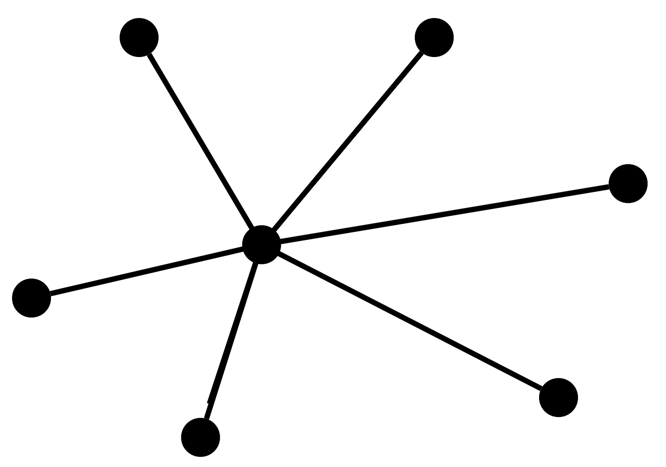


Collapse vertices within the **same bias interval**



Computing snapshots: special cases

- Edges are randomly ordered:
  - Take the first $O(1)$ edges. Store the induced subgraph and track the biases of the vertices. Compute snapshot.
- Bounded-degree graphs:
  - Sample $O(\sqrt{n})$ random (non-isolated) vertices. Store the induced subgraph and track the biases. Compute snapshot.
  - Bounded-degree assumption is crucial!



Snapshot estimation for arbitrary graphs

- "Layered Sampling of vertices"
  - Partition the degrees: $1 \leq d_0 < d_1 \cdots < d_{\log n} \leq n$, where $d_{i+1}/d_i = 2$
  - sample vertices of degree between $\frac{d}{2}$ and d with probability $\min\left\{\tilde{O}\left(\frac{d}{\sqrt{n}}\right), 1\right\}$ and store the induced subgraph
- Issue: We do not know the degree of the vertex when it first appears in the stream

Snapshot estimation for arbitrary graphs

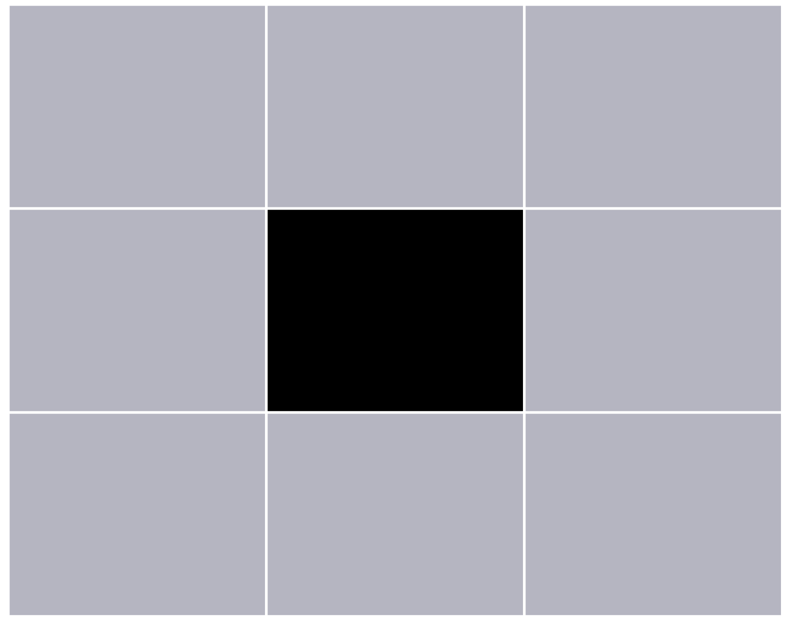
- “Layered sampling of edges”
  - Subsample edges with probability $\tilde{O}\left(\frac{1}{d}\right)$
  - Degree d vertices have degree $\tilde{O}(1)$ in the above graph
  - Ignore vertices with degree $\gg \tilde{O}(1)$ or $\ll \tilde{O}(1)$; Subsample every vertex with probability $\min\left\{\tilde{O}\left(\frac{d}{\sqrt{n}}\right), 1\right\}$ and store all the edges incident to it
- Issue: Can misclassify vertices and place them in wrong layers. Could also potentially make errors in bias computation

Snapshot estimation for arbitrary graphs

- Refined snapshot $M((i,j),(k,l))$: How many edges go from bias-interval B_i, degree-interval D_j to bias-interval B_k and degree-interval D_l?

- Smoothed snapshot $\widehat{M}$: Average of M over a "window" of size w

- Pointwise estimate $\widehat{M}$
  - More accurate estimates with increasing w
  - Harder to "retrieve" M if w is too large



Why smoothed snapshots suffice?

Why don't misclassification errors matter?

- Degree misclassification errors do not affect snapshot computation
  - If there is no error in bias computation, snapshot computed from M and $\widehat{M}$ are exactly the same
- Bias misclassification errors do not affect Max-DICUT value when w is small
  - Perturb the original graph G to get a new graph H with similar Max-DICUT value; Smoothed-snapshot$(G) \approx$ Refined-snapshot(H)
  - If bias of v is off by ϵ, create a new isolated vertex and modify bias by creating at most ϵd new edges; Max-DICUT value does not change by more than ϵm

Local algorithm II: distributed algorithms

- Message-passing model:
  - In every round, each vertex receives a message from each of its neighbors
  - After k rounds, each vertex “locally” computes its assignment
- [Censor-Hillel Levy Shachnai 17] [Buchbinder Feldman Seffi Schwartz 15] distributed algorithm for Max-DICUT - $\frac{1}{2}$ approximation - $(\Delta + 1)$ rounds

[CLS17] [BFSS15] Distributed algorithm $\mathscr{D}$

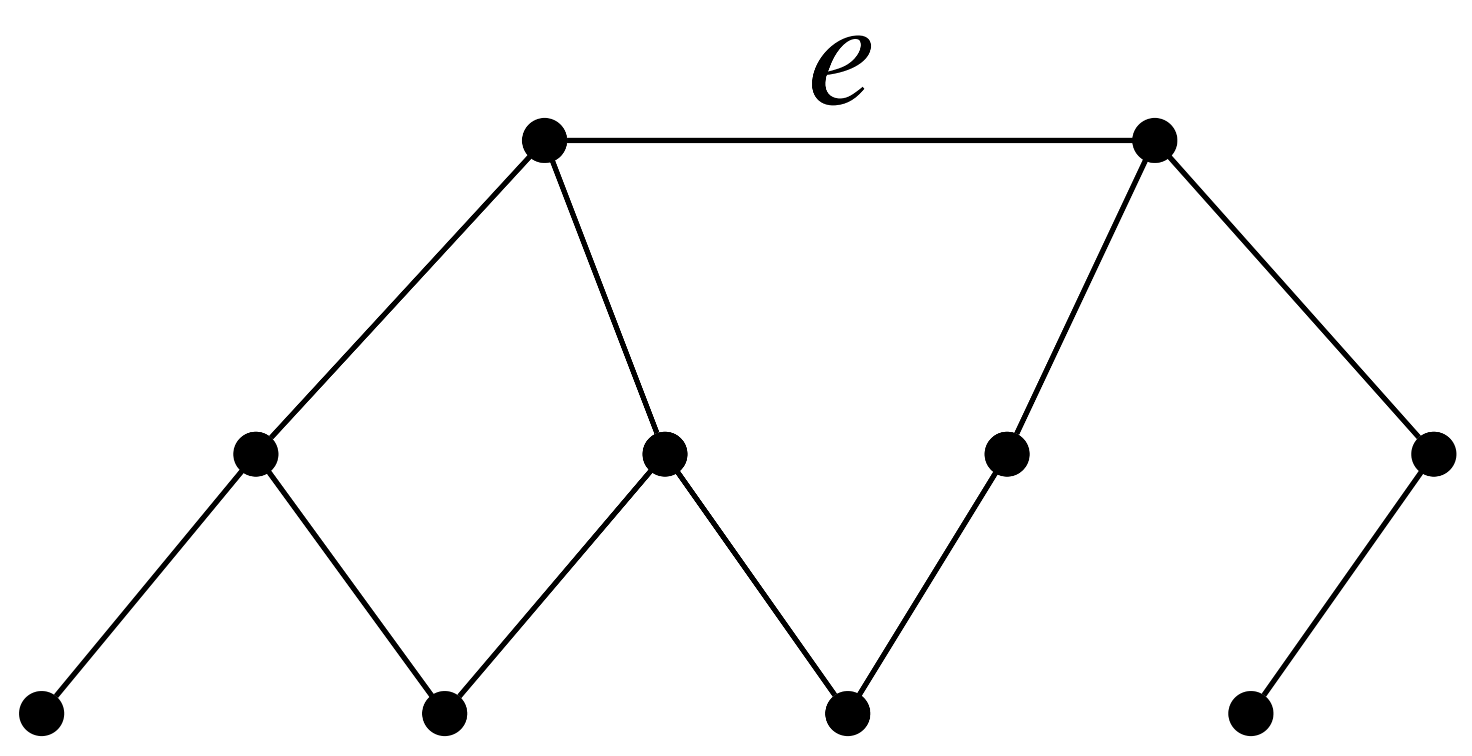
- Obtain a proper $\Delta + 1$ vertex coloring of the graph
- Assign vertices in batches
  - In the first round, assign vertices of color 1
  - In the second round, assign vertices of color 2, and so on
- At every round after a vertex is assigned, it sends a message to all higher color neighbors
- Approximation ratio = $\dfrac{\mathbb{E}[\text{DICUT}(\mathscr{D})]}{\text{OPT}}$

Streaming implementation

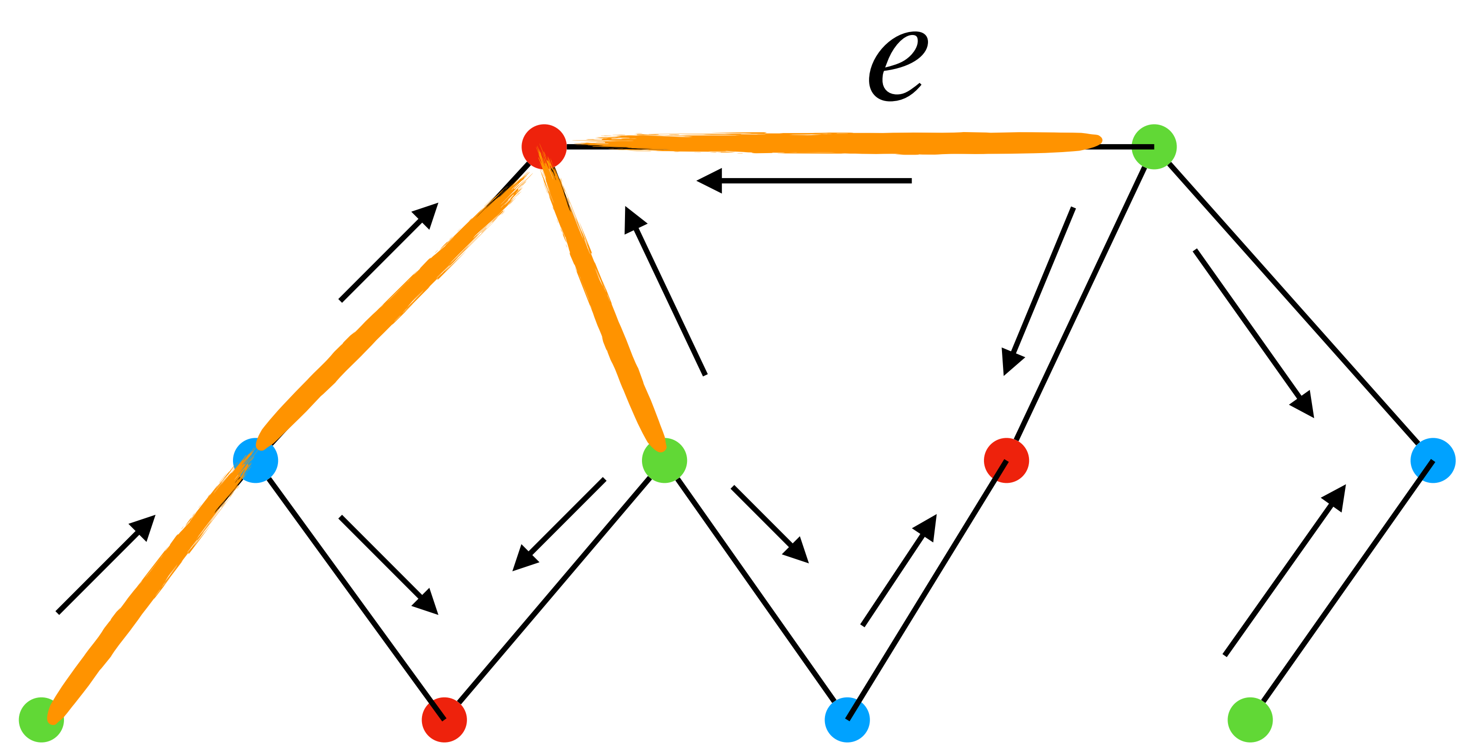
- Randomly color the graph using $O(1/\sqrt{\epsilon})$ colors
- With high probability, at most ϵ fraction of edges are monochromatic
- Delete the monochromatic edges
- Distributed algorithm $\mathscr{D}$: $\frac{1}{2} - \epsilon$ approximation in $O(1/\sqrt{\epsilon})$ rounds
- Want to compute $\mathbb{E}[\text{DICUT}(\mathscr{D})]$
  - $\mathbb{E}[\text{DICUT}(\mathscr{D})] = \sum_{e \in E} \Pr[e \in \text{DICUT}(\mathscr{D})]$
  - Sample $O(1)$ random edges and compute their average probability of belonging to $\text{DICUT}(\mathscr{D})$

Streaming implementation

- Radius-k neighborhood of an edge



- Observation: radius-$O(1/\sqrt{\epsilon})$ neighborhood of e suffices to compute the probability that e belongs to DICUT($\mathscr{D}$)



Red > Blue > Green

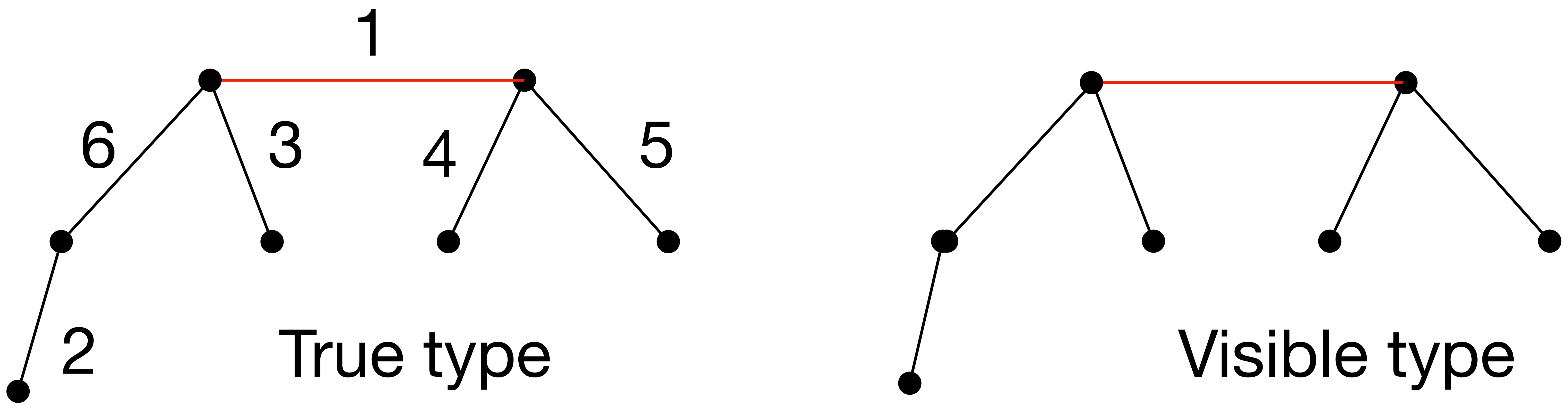
Streaming implementation

- Type of an edge: Radius-$O(1/\sqrt{\epsilon})$ neighborhood
- $\Delta \leq d$
  - $N(d, \epsilon)$ possible types: estimate the type distribution to compute $\mathbb{E}[\text{DICUT}(\mathscr{D})]$
- Sublinear space: For each type T
  - sample $O(n^{1-f(\epsilon,d,T)})$ random (non-isolated) vertices. Store the induced subgraph and the degrees of all the vertices.
  - consider edges whose entire radius-$O(1/\sqrt{\epsilon})$ neighborhood is contained within the stored graph
  - compute the number of edges of type T (and appropriately rescale)

Streaming implementation

- Random ordering $O(\log n)$ space
  - [Monemizadeh Muthukrishnan Peng Sohler 17]
  - High-level idea: Track the types of $O(1)$ random edges via BFS
    - compute the "visible" type distribution $\mathscr{V}$ and infer the "true" distribution $\mathscr{T}$ from $\mathscr{V}$

Streaming



- $\mathscr{V} = M\mathscr{T}$
- M is invertible
- $\mathscr{T} = M^{-1}\mathscr{V}$

Open problem I

- Extending $1/2$-approximation to general unbounded degree graphs
  - [SSSV *upcoming*] $O(1/\sqrt{\epsilon})$ pass $O(\log n)$ space algorithm
    - distributed algorithm: for high-degree vertices, suffices to sample messages from d random neighbors
  - Challenge: implementing in fewer passes, even with a randomly ordered stream

Open problem II

- Extension to other CSPs
- Max-k-AND
  - [CGSV 22] $o(n)$ space approximation is between $\left[\frac{1}{2}^k, \frac{1}{2}^{k-1}\right]$
  - [Boyland-Hwang-Prasad-Singer V 22] closed form expression for the approximation ratio achieved by "bias" algorithm
  - [Singer 23] established that for every k, there exists an oblivious algorithm that beats "bias" algorithm
  - Challenge: Is it possible to achieve $\frac{1}{2}^{k-1}$-approximation for Max-k-AND in sublinear space? smallest open case: Max-2AND (not sub-modular)

Thanks for your attention!