15-851 Final Presentation: Cut Sparsifiers in the Streaming Model

Oswaldo Ramirez & Lauren Sands

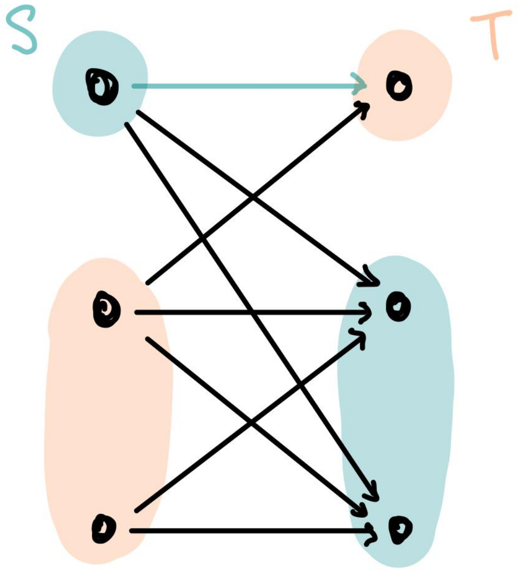
What is a Cut Sparsifier?

- **Sparsifier:** A data structure that gives a $(1 \pm \varepsilon)$ approximation of *all* cuts.
  - This data structure should use less than $O(n^2)$ space
- Two models to consider:
  - **For-all model:** We simultaneously need to preserve $(1 \pm \varepsilon)$ approximations for all cuts in the graph
  - **For-each model:** We only need a $(1 \pm \varepsilon)$ approximation for a specific cut queried at the end of the process

Existing Work

- For **undirected graphs**:
  - $\tilde{O}(n/\varepsilon^2)$ space 1-pass algorithm [1]
  - With deletions: $\tilde{O}(n/\varepsilon^2)$ space 1-pass algorithm [2]
- For **directed graphs:** [3]
  - In the worst case, require $O(n^2)$ edges stored

Worst Case n^2 Space for Directed Cut Sparsifier



Instead, consider **β-balanced graphs**: For every cut, the total weight in one direction is at most β times the weight in the other direction

- **For-each model:** matching $\widetilde{\Omega}(n\sqrt{\beta}/\varepsilon)$ space upper/lower bound [3]
- **For-all model:** matching $\Omega(n\beta/\varepsilon^2)$ space upper/lower bound [3]

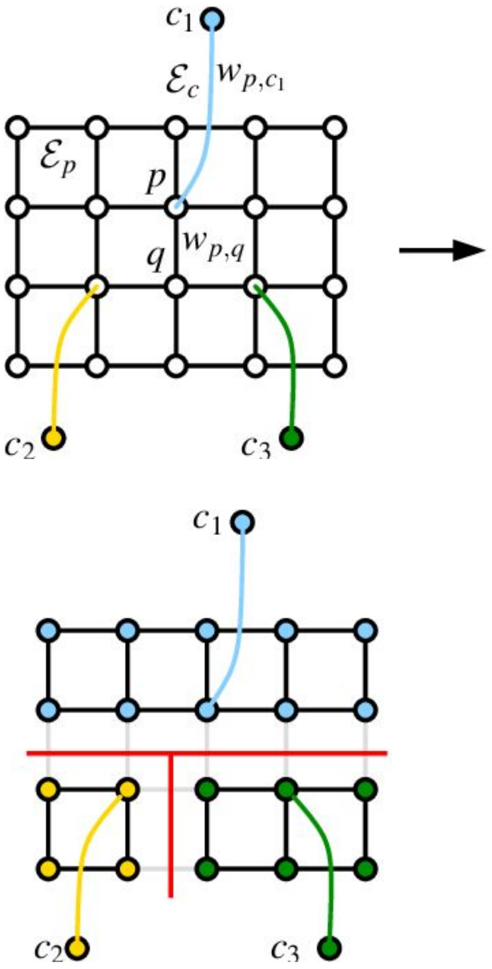
Question 0: Multiway-Cut on Directed Graphs

Multiway Cut Problem

Inputs:

- $G = (V, E)$, where edge e has weight w_e
- A set of terminals $S = \{s_1, s_2, \ldots, s_k\} \subseteq V$

A multiway cut is a set of edges that leaves each of the terminals in a separate component.



Issues

- In s-t cuts, the capacity is defined as edges from the S to T side
    - Here, there are multiple subsets of vertices, so what defines the "direction"?
- There are cases that require N^2 space in the regular s-t cut problem, and a β parameter is used to consider cases of more balanced graphs
    - How should β be defined here? There is not simply a ratio of "capacity in one direction" to "capacity in the other direction"
- This was likely too difficult to attempt in the time we have available

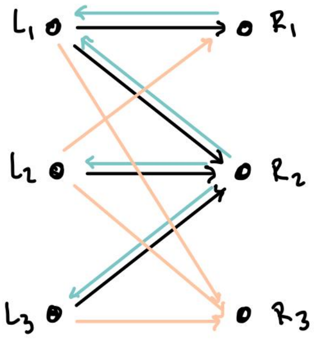
Question 1: Determining Beta in the Streaming Model

Determining Beta in the Streaming Model

- **Motivation:** There are algorithms to compute beta offline, but not in the streaming model
  - The papers that compute directed graph sparsifiers assume β is given.
- **Research Question**: What is the space complexity of computing β in the streaming model?
- **Conjecture**: Any streaming algorithm requires $\Omega(n^2)$ bits of memory to compute β

Intuition: Computing β takes n^2 space

- Consider the example to the right, where all edges in the first half point in one direction
    - We don't know if the second half will cancel out the edges, or keep pointing in the same direction
- Applying to sparsification:
    - At a point in the stream, β may appear to be n, so we use n^2 space at this step. But later it could turn out that β = 1



$(L_1, R_1), (L_1, R_2), (L_2, R_2), (L_3, R_2)$

$(R_1, L_1), (R_2, L_1), (R_2, L_2), (R_2, L_3)$] 1-balanced

OR

$(L_1, R_3), (L_2, R_1), (L_2, R_3), (L_3, R_3)$] ∞-balanced

Next Steps

- Formalize the adversary argument that requires n^2 space to compute β
- If we have trouble with this method, consider attempting a reduction to a communication complexity problem

Question 2/3: Preserving Only β-Balanced Cuts in a Non-β-Balanced Graph (Deterministically/randomly)

Preserving Only β-Balanced Cuts

- **Motivation:** The restriction for the directed cut sparsifier in [3] is strong in that *every* cut must be β-balanced.
  - What if a graph had most of its cuts β-balanced, with a few unbalanced cuts?
- **Question:** What is the lower bound to compute a directed cut sparsifier, where the whole graph is not guaranteed to be β-balanced, and we only need to preserve β-balanced cuts?
  - That is, we must return a $(1 \pm \varepsilon)$-approximation for queries on β-balanced cuts, but can fail if the cut is not β-balanced
  - We can consider this for deterministic and randomized cases

References

1. Graph Sparsification in the Semi-streaming Model
2. Single pass sparsification in the streaming model with edge deletions
3. Tight Lower Bounds for Directed Cut Sparsification and Distributed Min-Cut
4. Sparsification of Directed Graphs via Cut Balance