Lecture on 02/17/2022

Lecturer: Xiaorui Sun *Scribe: Smrithi Balki*

1 Last Lecture's Review

In the last lecture, we discussed about the different ways to perform Expander Decomposition

2 Dynamic Connectivity in $n^{O(1)}$ time

Definition : Alot of systems want their algorithms to be deterministic so that they can give same result.Example :DataBases.Randomized algorithms are not good in some scenarios
Expander Decomposition : In Last lecture we have seen several ways to do it. $m^{1/2+O(1)}$ time for $(\phi, \phi.n^{O(1)})$

3 Expander Pruning

- **Definition** : If Graph G is a ϕ - expander and K edges are removed from G. There is an algorithm with $O(k \log m/\phi)$ to find S $\subseteq VS, t$.
  $3 - conditions$:

  - **1.** Vol(S) $\leq 8k/\phi$
  - **2.** E(S,$\overline{S}$)$\leq$ 4k
  - **3.** G($\overline{S}$)$is\phi/6$ expander after removing the edges

If some edges in the graph are remote , further remove the vertices , so that the remaining graph is a
To maintain the expander property addtional costs. There are 2 costs to handle.

1. **Edge Expansion**
2. **Remove Vertices**

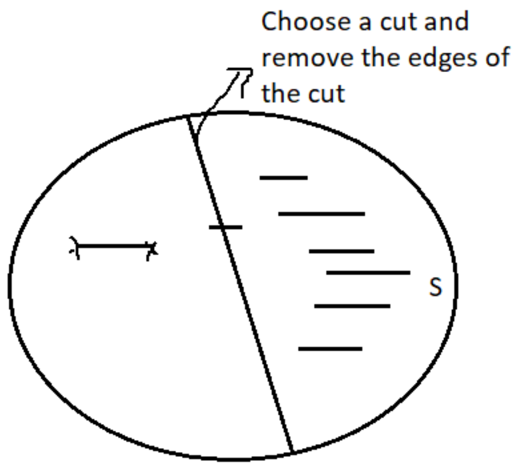

Figure 1:

4 Reduce Graph

Definition : The number of edges and vertices become smaller to keep a track and conclude with connectivity.

Approach :
Step 1: Start with the original Graph G.
Step 2: Draw the components called expander.Expanders within same component should be connected if connected in original G.Refer Figure 2
Step 3: Use a single vertex to represent an expander. See the figure components A,B,C .. becomes single vertices respectively.

Therefore, the size is significantly reduced

5 Expander Decomposition - To find connectivity using Reduced Graph

- **Mapping** : P : V -> V_H, for any vector x $\in V$,
  P(x) denotes the vertex in H that corresponds to the expander containing vertex x.
  Note : $x, y \in V$
  x and y are connected iff P(x) and P(y) are connected in H.
  Efficient Cost and Running Time :

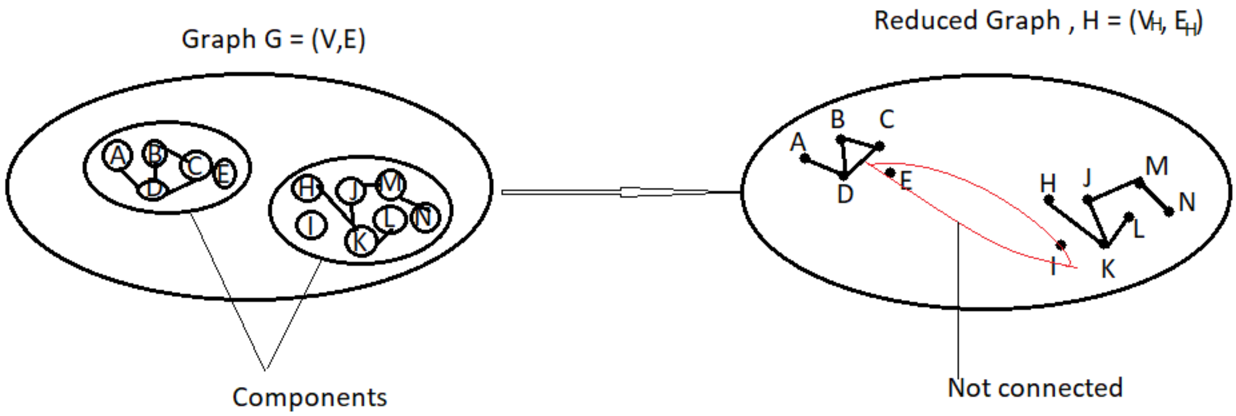

Figure 2:

Assume graph G that has bounded degree ≤ 3.
Create a set of vertices for the original vertex. refer to the Figure 3.

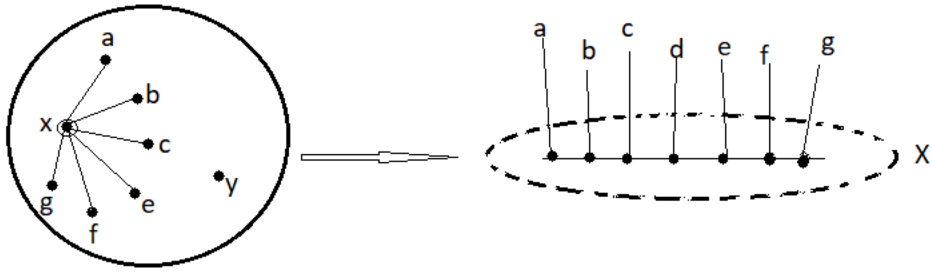

Figure 3:

- In figure 3, if you want to know the connect between X and Y , it is equivalent to any any vertex in X to y.

6 Handle a Single Update

Assume Original Graph , $\phi = 1/\sqrt{m}$, $(\phi, \phi.n^{O(1)})$ is reduced to $m^{1/2+O(1)}$ vertices and edged in the reduced graph.
Goal : Handle the update in time $m^{1/2+O(1)}$

- **Insert an Edge(x,y)**
- **Delete an Edge(x,y)** : There are 2 different cases
  - **(x,y) is a remote and crossing edge**
    If (x,y) is a crossing edge and remote
    No change is seen since the edge is remote
    So expander does not change
  - **(x,y) belongs to some expander**
    Then update the expander also
    Perform expander pruning to expander piece that has been updated

Goal : To Update Expander decomposition and reduced graph as well as Mapping (P)
Step 1: Apply expander prunning to deleted remote Edge.
Step 2: After Pruning, we have a set S .
Step 3: Now, we update the decomposition such that $\overline{S}$ becomes expander piece
Step 4: Move all vertices of S to the reduced graph
Note: If there is an edge $\overline{S}$ to the outer edge remove that, and use the edge from S.Refer Figure 4

7 Update Mapping (P)

P(x) = x , x ∈ S –> O(Vol(s)) ∼ $O(\sqrt{m})$
Running time to make the above changes

- 1: Add vertices in S to H -O(Vol(S))
- 2: Add edges incident to some vertices in S to H[O(Vol(S)) + number of vertices in S]
- 3: Remove some edges from $\overline{S}$ in H

Therefore, Total cost to update reduce graph is ∼ O(Vol(S)) ∼ $O(\sqrt{m})$

8 Handle Update for Insert edge(x,y)

- **(x,y) is not a crossing edge - then NO change on decomposition, only one insertion in reduced graph**

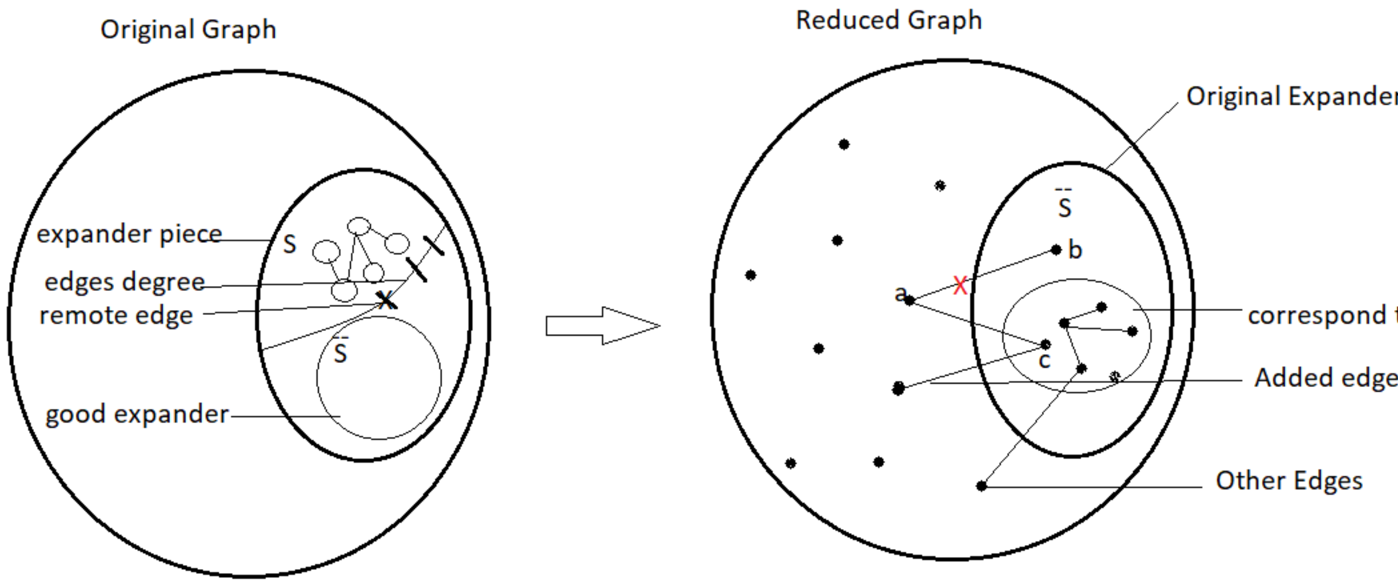

Figure 4:

- **Approach : Reduce to edge deletion and proceed** Refer figure 5
  - 1 : Look at all existing edges x to Y
  - 2 : Delete (x,a)(x,b)(x,c)(y,d)(y,e)(y,f)
  - 3 : Use the edge deletion algorithm to delete each edge
  - 4 : After handling deletions x,y $\in$ H
  - 5 : (x,a)(x,b)(x,c)(y,d)(y,e)(y,f) become crossing edges. Add (x,a)(x,b)(x,c)(y,d)(y,e)(y,f) back
  - 6: At the end, insert edge (x,y) to G gives inserting a crossing edge
- **Edge Expansion** : min(S) E(S,$\overline{S}$)/min(Vol(S),Vol($\overline{S}$))
- **Cost** : Cost for all the above steps is $O(\sqrt{m})$

Question : What happens if I want to handle multiple Updates Answer :
We can generalize this to,
Lemma : One can handle a sequence of updates in time $O(t.\sqrt{m})$.Here, t = number of updates in the sequence.
if before updated it is ϕ expander composition
Once you have the lemma, you can turn it into Dynamic Algorithm with time $O(m^{1/2+O(1)})$

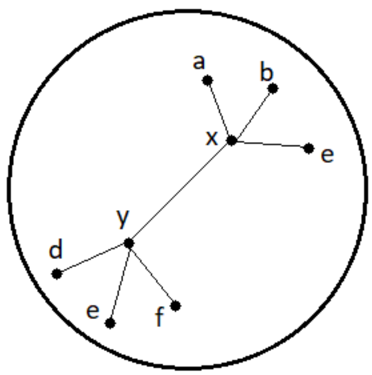

Figure 5: