Homework 1 (CS-594)

Q1) A Randomized Algorithm for Dynamic Graph connectivity Giving Nodes a bit representation, and the power of XOR. Suppose we delete edge $e = u - v$, causing the tree T to split into L and R.

Assume to each node v a $\log_2 n$ -bit label $l(v)$. Assume some total ordering on the nodes, so the edge uv (where $v \prec V$ in this ordering) has a $2\log_2 n$ - bit label that concatenates the names of its two vertices in that order.

If there are multiple possible replacement edges going from L to R, the result of the calculation will be the XOR of the labels of all these edges.

Let us make an weaker assumption. Suppose there are K replacement edges, then we know this value K, we keep a subsampled Graph G' obtained with probability $1/K$

$$P_r(\text{exists unique } L \text{ to } R \text{ edge in } G') = K \cdot 1/K \, (1 - 1/K)^{K-1} \geq 1/e$$

So with constant probability, there is a unique edge, and run the fingerprint algorithm moreover, by repeating this process $O(\log n)$ times independently we can ensure that one of these repetitions will give a unique crossing edge with probability $1 - 1/\text{poly}(n)$.

Finally, we can try subsampling at rates $1/2, 1/4, \ldots 1/n$ (i e) all powers of 2 between $1/2$ and $1/n$. There are $\log_2 n$ such values, and one of them will be the correct one, upto a factor of 2.

Hence, it is proved that if we assign to each node r a $(\log_2 n)$ - bit label $l(v)$ will get a uniqve edge with probability $1 - 1/\text{poly}(n)$

Therefore $2(\log_2 n)$ bits of binary names for edges are needed for $1 - 1/n$ probability.

Q2) Let a graph $G = (V, E)$

We start with preprocessing phase. For the preprocessing phase, we pick an arbitrary node $v = V(G_i)$ and apply following algorithm with parameter $C \leftarrow V(G_i)$

Algorithm : Find Weighted Ball (u, C, i) where C is a cluster in C_i

1. $B_0 \leftarrow \{u\}$;
2. for $r = 0, 1, \ldots, R$ do
   if $|E_j \cap ((B_{G_i(C)}(u, r+1) \setminus B_{G_i(C)}(u, r)) \times B_{G_i(C)}(u, r)| \leq \beta \cdot |E_j \cap E(G_i)[B_{G_i(C)}(u, r)]|$, $\forall 1 \leq j \leq i+1$ then
   return $\bigcup_{i=0}^{r} B_i$ as $B_{G_i(C)}(u, r)$

Then we randomly select a node in $B_{G_i}(u, r)$ rooted at which an Even-Shiloach tree is initialised ; repeat this process on the rest of the graph $C \in C / B_{G_i}(u, r)$. Summarize the preprocessing algorithm as pseudocode 2 and

Algorithm 2: Init clustering (G_i)

$C_i \leftarrow \phi$, $C \leftarrow V(G_i)$;
while $C \neq \phi$ do
  Pick an arbitrary node $u \in C$;
  $B_{G_i(C)}(u, r) \leftarrow$ Find Weighted Ball (u, C, i);
  run New Weighted Cluster $(B_{G_i(C)}(u, r))$;
  $C \leftarrow C \setminus B_{G_i(C)}(u, r)$;
return G_i;

Algorithm 3: New Weighted Cluster (c)

1. Let E_c be a snapshot of $|E(c)|$;
2. uniformly sample an edge from $E(c)$ and pick one of its endpoint $c_* \in V$ arbitrarily;
3. build an Even-Shiloach tree of $G_i(c)$ maintaining distances upto $3R$, rooted at the node that contains c_*;
4. $C_i \leftarrow C_i \cup \{c\}$;

After initializing C_i, select all Even-Shiloach tree edges as tree edges for the final low-stretch tree, and $V(G_{i+1})$ is defined by contracting every cluster in C_i to a single node.

Q3) In the procedure for finding a Low Diameter Randomized Decomposition for a graph $G=(V, E)$ and a parameter $\delta > 0$ and output a partition V_i of V we have:

- $\text{diam}(G[V_i]) \leq \delta$ for all i
- $P_\pi[\text{edge } \{i,j\} \text{ not in any } E(G[V_K])] \leq (d(i,j) \times \log n)/\delta$.

→ In the proof for Bastal's algorithm on a distribution D of trees, where $\Delta = \text{diam}(G)$ we have the following claims:

- For all $u,v \in V$, $d_T(u,v) \geq d_G(u,v)$.
- For all $x \in V(T)$, $d_T(r,x) \leq 2\Delta$. For all $u,v \in V(T)$, $d_T(u,v) \leq 4\Delta$

we write

$$E[d_T(u,v)] = E[d_T(u,v) \mid u,v \text{ don't lie in same } T_i] \times P_\pi(u,v \text{ don't lie in same } T_i) + E[d_T(u,v) \mid u,v \text{ lie in same } T_i] \times P_\pi[u,v \text{ lie in same } T_i]$$

we have,

$$E[d_i(u,v) \mid u,v \text{ don't lie in same } T_i] < 4\Delta$$

$$P_\pi[u,v \text{ don't lie in same } T_i] \leq \frac{4 d_G(u,v)}{\Delta} \log n$$

$$E[d_T(u,v) \mid u,v \text{ lie in same } T_i] \leq 8 \log \text{diam}(G(V(T_i))) \log n \times d_G(u,v)$$

$$P_\pi[u,v \text{ lie in same } T_i] \leq 1$$

$$\Rightarrow E[d_i(u,v)] \leq \frac{d_G(u,v)}{\Delta/2} \log n \times 4\Delta + 8 \log(\Delta/2) \log n \times d_G(u,v)$$

$$\Rightarrow E[d_i(u,v)] \leq 4\Delta \times \frac{d_G(u,v) \log n \log \Delta}{\Delta/2}$$

Q 4) Laplacian matrix for a graph is defined as:

$$L_G(i,j) = \begin{cases} \deg(i) & \text{if } i=j \\ -1 & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$

for graph $G=(V,E)$ with $|V|=n$ and $L_G = D-A$, where D is the degree matrix for G - a diagonal matrix where $D(i,i)$ is the degree of the i^{th} node in G and A is the adjacency matrix with $A(i,j)=1$ if and only if $(i,j) \in E$.

We know if λ is an eigenvalue of L_G then

$$L_G \cdot V = \lambda \cdot V \quad \text{where } v \text{ is an eigenvector}$$

$$L_G \cdot V = \deg(i)\, v(i) - \sum_{j:(i,j)\in E} v_j = \sum_{j:(i,j)\in E} (v(i) - v(j)) \quad \text{— (1)}$$

i^{th} element of $L_G \cdot V$ is the sum of differences between $v(i)$ and the indices of v corresponding to the neighbors of i in the graph G.

1. Suppose an eigen vector $V = (1/n, 1/n, \ldots, 1/n)$

From (1) we have the i^{th} entry of $L_G \cdot V = \sum_{j:(i,j)\in E} (v(i) - v(j))$

$$\Rightarrow \sum (1/n - 1/n) = 0$$

$$= 0 \cdot v(i)$$

From above we can write $L_G \cdot V = 0 \cdot V$. Therefore the Laplacian matrix of a graph has atleast one eigenvalue which is **0**.

2. Consider

$$v^t \cdot L_G \, v = \sum_i v(i) \sum_{j:(i,j)\in E} (v(i) - v(j))$$

$$= \sum_{i<j:(i,j)\in E} (v(i) - v(j))^2 \quad \text{— (2)}$$

Let G has k connected components, partitioning $|V|$ into disjoint sets $S_1, \ldots, S_k$. Define k vectors $v_1, \ldots, v_k$ s.t. $v_i(j) = 1/\sqrt{|S_i|}$ if $j \in S_i$ and 0 otherwise. For $i = 1 \ldots k$, we have $\|v_i\| = 1$. For $i \neq j$ sets S_i, S_j are disjoint, $\langle v_i, v_j \rangle = 0$, and we have $L_G \cdot v_i = 0$. Hence, we have a set of atleast k orthonormal vectors that are eigenvectors of L_1 with eigen value 0.

From (2) we have,

$$v^t \cdot L_G \cdot v = \sum_{i<j:(i,j)\in E} (v(i) - v(j))^2$$

which can be zero if v is constant on every connected component.

If we suppose there is $k+1^{st}$ vector v that is a zero eigenvector, orthogonal to $v_1, \ldots, v_k$, then v must be non-zero in some coordinate in set S_i and hence is non-zero and constant on all indices in set S_i, in which case v cannot be orthogonal to v_i, and there can be no $k+1^{st}$ eigenvector with eigenvalue 0.

Therefore, the number of 0 eigenvalues of L_G is eqaul to the number of connected components of graph G.

Q5) Run the below steps for cut size $S = 1$ to C :

1. Run BFS on vertex v for s/ϕ edges.
2. Let T be the set of all visited edges.
3. Cut edge $\in T$
4. Check which of the edges in T is a cut edge
   (a) When found then throw away to see if another cut edge is found
   (b) Enumerate $e \in T$, remove e, run BFS from v for s/ϕ steps.

Above algorithm can be used to find all cuts of size atmost c in time $O(n \cdot (c/\phi)^{O(c)})$.