COL751 - Lecture 16

In this lecture we will study applications of Gomory-Hu tree.

1 All Pairs Min-Cut Oracle

Let $G = (V, E, c)$ be an undirected graph with integer edge capacities in range $[1,\ poly(n)]$. We will show construction of an $O(n)$ sized data-structure that given any two vertices $x, y \in V$, reports the (x, y)-min-cut value $\lambda_{x,y,G}$ in constant time.

Our data structure first constructs a Gomory-Hu tree $\mathcal{T}$ for G. Next it computes a rooted binary tree, denoted by T_{LCA}, that satisfies the following properties:

1. Internal nodes of T_{LCA} correspond to edges of Gomory-Hu tree $\mathcal{T}$.
2. Leaf nodes of T_{LCA} correspond to vertices of Gomory-Hu tree $\mathcal{T}$.
3. For any two leaf nodes x, y in T_{LCA}, the lowest common ancestor (LCA) of x, y in T_{LCA} is an edge of least weight on the unique path connecting x, y in tree $\mathcal{T}$.

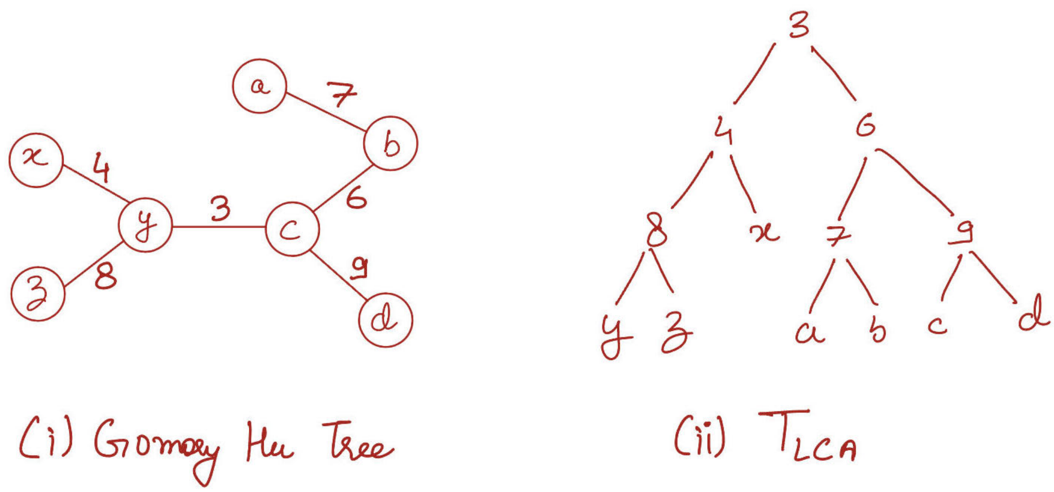

Figure 1: Depiction of Gomory-Hu tree and T_{LCA}.

Note that for answering the LCA queries on N node rooted trees there exists an $O(N)$ sized data-structure with constant query time. Therefore, we get the following theorem.

Theorem 1 *Any n-vertex undirected graph $G = (V, E, c)$ with integer edge capacities in range $[1,\ poly(n)]$ can be processed in polynomial time to compute a data-structure of $O(n)$ size that given any two vertices x, y reports $\lambda_{x,y,G}$ in $O(1)$ time.*

2 An $O(mnk)$ time algorithm for computing k-edge connected components

The k-edge-connectivity relation in an undirected multigraph $G = (V, E)$ is an equivalence relation. Indeed if vertex-pairs (x, y) and (y, z) are k-edge-connected (that is, are not separated by a cut of size $k - 1$), then so is the pair (x, z). (Proof: Homework).

Let $\mathcal{W} = \{W_1, \ldots, W_r\}$ denote the k-edge-connected-components of G, and let $G_{\mathcal{W}}$ be quotient graph of G induced by k-edge-connectivity relation.

A threshold-k Gomory-Hu tree (k-GHT) of G is defined as Gomory-Hu tree of graph $G_{\mathcal{W}}$. Equivalently, it is a partial Gomory-Hu tree $\mathcal{T}$ for G whose vertex-set is precisely set $\mathcal{W}$. We will present in this section an $O(mnk)$ time algorithm to compute k-GHT.

Algorithm Recall that in our algorithm to compute GHT (see Lec 15) we recursively split super-nodes in $\mathcal{T}$ of size at least 2. Our algorithm for k-GHT differs from GHT algorithm as follows:

After choosing a pair of vertices s, t in a super-node $X \in V(\mathcal{T})$, we first check in $O(mk)$ time if $\lambda_{s,t,G}$ is less than k. In case $\lambda_{s,t,G} < k$ then we proceed with usual split process, and if $\lambda_{s,t,G} \geqslant k$ then we merge vertices s and t into a single node in G, thereby, reducing size of X by 1.

$\mathcal{T} = (\{V\}, \emptyset)$.
for $i = 2$ *to* n **do**
    Let $X \in V(\mathcal{T})$ be a set of size at least two.
    Take any two vertices s, t in X.
    if *Size of* (s, t)*-min-cut in* $G \geqslant k$ **then**
        Merge s, t in G into a single node.
    else
        Let $C_1, \ldots, C_k$ be connected-components in $\mathcal{T} - X$.
        Let H be a graph obtained from G by contracting $C_1, \ldots, C_k$ into k super-nodes.
        Compute an (s, t)-min-cut, say (S_H, T_H), in H and let (S, T) be an (s, t)-cut in G obtained from (S_H, T_H) on uncontracting $C_1, \ldots, C_k$.
        Split X into two nodes $X_S = S \cap X$ and $X_T = T \cap X$, and for $j \in [1, k]$, connect C_j to X_S if $V(C_j) \subseteq S$ and X_T otherwise.
        Set $c^*(X_S, X_T) = \lambda_{s,t,G}$.
Return $\mathcal{T}$.

As each iteration takes at most $O(mk)$ time, the total running time of the algorithm is $O(mnk)$.

Proposition 1 *For any n vertex, m edge undirected multigraph $G = (V, E)$ and any integer $k \geqslant 1$, we can compute a threshold-k Gomory-Hu tree in $O\big(mnk\big)$ time.*

Corollary 1 *For any n vertex, m edge undirected multigraph $G = (V, E)$ and any integer $k \geqslant 1$, we can compute k-edge-connected components of G in $O\big(mnk\big)$ time.*

3 An $O(m + n^2k^2)$ time algorithm for computing k-edge connected components

We first show how to compute a threshold-k Gomory-Hu tree $\mathcal{T}$ for G using a sparse k-edge-connectivity-preserver.

Lemma 1 *Let $G = (V, E)$ be an n-vertex multigraph. Given a k-edge-connectivity-preserver H for G having $O(nk)$ edges, one can compute a threshold-k Gomory-Hu tree $\mathcal{T}$ for G in $O(n^2k^2)$ time.*

Proof: Let H be a k-edge-connectivity-preserver of G having $O(nk)$ edges. One can use Proposition 1 to compute a threshold-k Gomory-Hu tree $\mathcal{T}$ for H in $O(|E_H|nk) = O(n^2k^2)$ time. Since H preserves edge-connectivity up to value k, tree $\mathcal{T}$ will also be a threshold-k Gomory-Hu tree for G. This proves the claim. □

We will discuss in next lecture how to compute for a multigraph $G = (V, E)$ a k-edge connectivity preserver $H = (V, E_H \subseteq E)$ in $O(m + n)$ time such that $|E_H| \leqslant nk$. Using this as black box, we get the following result.

Proposition 2 *For any n vertex, m edge undirected multigraph $G = (V, E)$ and any integer $k \geqslant 1$, we can compute a threshold-k Gomory-Hu tree in $O\big(m + n^2k^2\big)$ time.*

Corollary 2 *For any n vertex, m edge undirected multigraph $G = (V, E)$ and any integer $k \geqslant 1$, we can compute k-edge-connected components of G in $O\big(m + n^2k^2\big)$ time.*