CS345 Theoretical Assignment 5

Ayush Agarwal, 13180

M.Arunothia, 13378

Contents

1 Binary search and predecessor/successor queries under deletions

1.1 Data Structure Overview

2 Extension of the problem of the mid-semester exam

2.1 Pseudo Code

```
procedure findSubgraph(V,E)
    E_s ← φ
    while V != φ do
        Pick any vertex v from G
        if degree(v) <= n^{1/k} then
            Add all edges incident on v to E_s
            Remove v and all edges incident on v from G
        else
            Do a BFS from v in graph G till a depth of k − 1.
            Remove all the non-tree edges (in the formed k − 1 levels) from G
            Call this formed tree of depth k − 1 as cluster.
            For all v ∉ cluster retain just one edge with the cluster and remove all other edges from G
            Add all the tree(cluster) edges along with the edges incident on the cluster to E_s
            Remove cluster and all edges incident on cluster from G
        end if
    end while
    return Es
end procedure
```

2.2 Justifications

2.2.1 Retaining just one edge with the cluster for any $v \notin cluster$ is sufficient

Let v be the root of the *cluster* being discussed. Let $u \notin cluster$ be the vertex outside cluster who has edges to both $x \in cluster$ and $y \in cluster$. Let us consider what happens when we remove say the edge (u, y). We know v being the root of the BFS tree, is connected to both x and y. As the depth of the BFS tree being considered is $k-1$, the maximum path length between v and x(or y) is $k-1$. This means there is a path between x and y via v that has a maximum length of $2(k-1)$. Though we have removed the edge (u, y), u and y are still connected by the path (u, x), then x to y via v. The maximum length of this path between u and y is hence, $2k-1$, satisfying the requirement asked for in the question. This explains why retaining just one edge with the cluster for any $v \notin cluster$ is sufficient.

2.2.2 Non-Tree edges need not be retained

As within the tree any two vertices are always connected via a path whose length $\leq 2k-1$, there is no need to retain non-tree edges.

2.3 Proving $|E_s| = O(n^{1+1/k})$

- If $degree(v) <= n^{1/k}$ then, we add atmost $n^{1/k}$ edges to E_s for the single vertex v.
- If $degree(v) > n^{1/k}$ then, we add atmost $O($ *vertices being removed* + *vertices not being removed* $)$ edges to E_s for a *cluster* of atleast $n^{1/k}$ vertices.
  - The number of edges that belong to the *cluster* is of $O($ *vertices being removed* $)$ as it is a tree.
  - The number of edges that are incident on the cluster is $O($ *vertices not being removed* $)$ as proved from justification (1).
  - Hence, the given order.
- Worst case $|E_s| = n * n^{1/k} + \sum_{i=0}^{k} n^{1-i/k} = O(n^{1+1/k})$

2.4 Time Complexity

The overall algorithm accesses an edge exactly for O(1) times, because in any iteration of the while-loop the edge getting accessed is being removed off from G and hence, it is guaranteed that no edge is being accessed in two different iterations. Hence, the time complexity is $O(m+n)$.