Grapevine

Scientific Committee

22 March 2022

Grapevine

- You are given a weighted tree
- At each time step, one of the following things can happen:
  - A grape appears or disappears at some vertex (soak)
  - The weight of an edge changes (anneal)
  - Given a vertex v, find the nearest grape (seek)

Subtask 1

- $n, q \leq 2000$
- Use DFS for every query

Subtask 2

- For each seek operation $v = 1$
- Root the tree at $v = 1$
- If there are no soak operations, we can simply do a subtree add
- To deal with soak operations, we duplicate every vertex
- Original vertex has no grape, duplicate has grape
- To turn on and off the grape, convert this into an 'anneal' operation changing the weight to 0 or ∞ respectively
- For the rest of this presentation, we will not mention soak operations

Subtask 3

- Balanced binary tree
- Take advantage of the small tree depth
- For every vertex, maintain the distance to nearest grape on its subtree, call this $f(x)$
- Then the answer to seek(v) is

$$\min_{w} d(v, w) + f(w)$$

Subtask 4

- There is at most 1 grape at a time
- By using LCA, we can compute path length between any two vertices dynamically
- Simply keep track of where the grape is

Subtask 5

- Grapes will not disappear after seek queries
- Edge weights will change only to 0 (in particular, they do not increase)
- Make a centroid decomposition
- For each vertex, maintain the nearest grape in its centroid subtree $f(v)$
- Then the answer to seek(v) is

  $$\min_{w} d(v, w) + f(w)$$

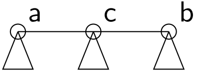
  where w is taken across centroid ancestors
- How to update $f(v)$?

Subtask 5

- Suppose the nodes labelled a, b, c have centroid depths $a < b < c$
- If we change the weight of (b, c), only centroid ancestors of b will have its values changed
- If the nearest grape from 2 does not pass through (b, c), $f(b)$ remains unchanged
- Else, the new value of $f(b)$ is given by

$$\min_{w} d(b, w) + f(w)$$

where w is taken along the centroid path of $c \to b$



Subtask 5

- We need to update $f()$ for ancestors of b also
- If a is an ancestor of b, the new value of $f(a)$ is given by

$$\min(f(a), \min_{w} d(a, w) + f(w))$$

where w is taken along the centroid path of $c \to a$

Subtask 6

- No additional constraints
- Under the constraints of the previous subtask, $f(v)$ never increases
- Now, $f(v)$ can increase, which makes things tricky
- Instead, store $f_1(v), f_2(v), \ldots,$, one for each branch as well as the minimum value across all branches
- A node can have large degree, but we can use priority queue to get an online minimum
  - This was not necessary when values were monotone