Introduction to Algorithms

Mong-Jen Kao (高孟駿)

Tuesday 10:10 – 12:00

Thursday 15:30 – 16:20

Program Assignment - IV

Segment Tree with Lazy Propagation

- We have seen that segment trees can be used to process & answer queries related to “segments” via divide-and-conquer.

- *Upon updates*, we have to update the “information” stored in some nodes of the tree.
  - In lazy propagation scheme, we ***“queue” the updates*** (in the top-most node possible) and *perform them* ***only when necessary***.

i.e., when the deadline comes...

Segment Tree with Lazy Propagation

- For example,
  consider *the union of segments problem*.
  - Suppose that we have inserted an interval $I_1 = [2,6]$ into the set A.
  - Then, provided that we still have I_1 in A, inserting $I_2 = [3,4]$ or $I_3 = [2,5]$ will have no effect on any subsequent queries.
    - The queries can be queued until I_1 is removed.

Segment Tree with Lazy Propagation

- In lazy propagation scheme, we **“queue” the updates** (in the top-most node possible) and perform them **only when necessary**.

- Two levels of “laziness”

a bit lazy

1. Queue the “unnecessary updates” and only execute them when necessary.

2. Queue “all the updates” and only execute them upon query.

so lazy

A – Substring Cut & Paste

Substring Cut & Paste

- In this problem, we need to deal with split & merge operations on an ordered sequence.
  - Hence, treap would be an ideal data structure.

- Note that, we cannot store the indexes of the characters, as they change after each operation.
  - We can store “***<u>the number of nodes</u>***” ***in the left-*** and ***in the right subtrees***, respectively, for each node, and it suffices.
  - Implement the operations for treap accordingly.