CSE 4094 Special Topics in Computer Engineering
Advanced Data Structures
Project 1
Using Cartesian Tree to Solve All Nearest Smaller Values Problem
Due: 02.12.2020 20:00

In this project, you will solve all nearest smaller values (ANSV) problem using Cartesian tree data structure.

ANSV problem is defined as; in a given sequence of elements, finding the nearest smaller value for each element to its right and left. To solve this problem, an algorithm that uses Cartesian tree is proposed in [1].

In this project, your task is to implement the proposed algorithm to solve the ANSV problem. You can find the details of the algorithm below. Also, you can check the study given in [1].

First of all, you should construct the binary Cartesian tree for a given sequence of integer numbers. On the resulting Cartesian tree, you will use the following algorithm to find the left nearest smaller neighbor for each node in the Cartesian tree. To find the right nearest smaller neighbor for each node, you will use a symmetric algorithm.

Algorithm 1. Algorithm to find the left nearest smaller neighbor for each node in the Cartesian tree [1]

(1) For every node, maintain two variables,
   a. `node.index` which is set to the node's index in the sequence corresponding to the in-order traversal of the Cartesian tree and never changed, and
   b. `node.inherited`, which is initialized to `null`.

(2) For each level *i* of the tree from 1 to *d* (*d* is the depth of tree, root depth is 1)
   a. For all nodes at level *i*, pass
      i. `node.inherited` to `inherited` variable of its left child and
      ii. `node.index` to `inherited` variable of its right child.

(3) For all nodes
   a. if `node.inherited` $\neq$ `null`, then `node.inherited` denotes the index of the node's nearest smaller neighbor on the left.
   b. Otherwise, it does not have smaller neighbor on the left.

Your program will take a sequence of numbers as input. The input should be taken from a text file. The output of your program will be the corresponding nearest left and right smaller values for each element in this sequence. If, for an element, there is no smaller value to its right/left, it will be represented by a dash (—).

Example:
Suppose that the input sequence is: 0, 8, 4, 12, 2, 10, 6, 14, 1, 9, 5, 13, 3, 11, 7, 15
Your program should give the following output (if there is no smaller value, put a dash):

- nearest left smaller values: —, 0, 0, 4, 0, 2, 2, 6, 0, 1, 1, 5, 1, 3, 3, 7
- nearest right smaller values: —, 4, 2, 2, 1, 6, 1, 1, —, 5, 3, 3, —, 7, —, —

For the implementation, you can use C, Java, or Python programming language.

You will submit the source code of your program using the Canvas service.

Note that the deadline is strict and will not change. Your possible schedule for exams and project deadlines is already considered.

Note: You can work in groups of 3 members at most. **I will schedule an online demo session on December 3rd and 4th for each group.** The schedule will be announced later. Each group member must be ready in the demo. Note that the grading may be different for the group members, so make sure that you work active on the project.

References
[1] J. Shun and G. E. Blelloch, "A simple parallel cartesian tree algorithm and its application to parallel suffix tree construction", *ACM Transactions on Parallel Computing (TOPC)*, 2014.