Task Popa - solution description

Author: George Chichirim

First off, an observation: we can check if the node at index *i* can be the root of the continuous subsequence [*x, y*], where *i* belongs to this subsequence, by checking if query(i, i, x, y) is 1. This can be used to solve the problem in the following ways:

- To solve the problem in a quadratic number of queries, try all indices as roots of the sequence, and when finding one valid root, solve what remains recursively.
- To solve the problem in a linearithmic number of queries, try all indices as roots of the sequence, trying elements closer to edges of the sequence before elements further from them, and then solve what remains recursively. Note that to determine the root of a subsequence takes a number of queries proportional to the size of the smaller subtree of the root of the sequence. This implies, using the idea from heavy path decomposition, that the total number of queries is linearithmic.
- Another solution in a linearithmic number of queries is to use binary search to find out a root for a sequence, and then solve what remains recursively. For the binary search use the query(x, y, x, pos) to find out if we should search in [x, pos] or in [pos, y] (the proof that this is correct is left as exercise to the reader).
- To solve the problem in a linear number of queries, construct the tree from left to right, starting from a singleton node which corresponds to the first index in the sequence. To add another node to the tree, note that a new node must be inserted somewhere in the chain from the rightmost node in the current tree to the root of the current tree (proof is left as exercise to the reader). Simply scan this chain from bottom to top, inserting the new node in the highest position where it can be while respecting the restrictions. This yields an amortized linear number of queries.