Tutorial 4

Data Structures 1, 14. 3. 2025 `https://iuuk.mff.cuni.cz/~chmel/2425/ds1en/`

Exercise 1 (Splay trees, potentially)
Ensure that you understand the definition of the potential in splay trees and that you understand the main ideas of the amortized analysis of the splay operation.

a) What is the definition of the potential of a splay tree?

b) What is the amortized cost of a rotation (a single and a double)?

c) What is the amortized cost of the full splay operation and how does it follow from the amortized costs of rotations?

d) What is the *real* cost of the splay operation (and what units do we measure it in)?

e) What is the potential of a perfectly balanced BST? (A reasonable upper and lower bound are sufficient, assume the tree has $n = 2^k - 1$ nodes.)

f) What is the potential of a path? (Again, reasonable bounds are sufficient.)

Exercise 2 (Subset theorem)
Recall that the working set theorem of Sleator and Tarjan states the following: Let $x_1, \dots, x_m$ be a sequence of elements in a BST on n nodes. We set z_i to denote the number of different elements that were accessed after the last access to x_i. Then, the total cost of all the Find(x_i) operations in this order is $\mathcal{O}(n \log n + m + \sum_{i=1}^{m} \log(1+z_i))$.
Using this theorem, prove the "subset theorem": Let T be a tree on the elements $[n]$ and consider a subset $S \subseteq [n]$, where we denote $s := |S|$. Then the queries for elements $x_1, \dots, x_m \in S$ have total cost $\mathcal{O}(n \log n + m \cdot \log s)$.

Exercise 3 (Semisplay)
In the zig-zig steps, we could be lazier and instead of performing a double rotation, we could just rotate a single rotation of the top edge. To be precise, in the case LL, let us have three nodes x, y, z such that x is a left child of y and y is a left child of z. Then, when splaying x, we perform a single rotation of the edge zy and we continue by splaying the node y (instead of x). In particular, the originally splayed element might not become the root in the end. The zig-zag step and the final single rotation work in the same way. We call this operation semisplay. Analyze the amortized cost of semisplay (try using a similar analysis as for splay).

Exercise 4 (Extremely lazy splay trees)
Motivated by our laziness that led to the successful operation semisplay, we would like to perform even fewer rotations. What would happen if, when splaying an element x, we would rotate only every second edge on the path from x to the root and gradually change the splayed element as we move up the tree?

Exercise 5 (Split and join)
For a splay tree T and a value k, design the operation Split that splits T into two trees T', T'', such that T' contains all nodes with values less than or equal to k, and the tree T'' contains all nodes with values higher than k. If possible, try to make sure that the amortized complexity remains logarithmic.
Next, for splay trees T', T'' (where all values in T' are smaller than all values in T''), design the operation Join(T', T'') that joins the two trees into one (again, while preserving logarithmic amortized complexity).

Bonus exercises

Exercise 6 (Analysing the successor using a potential)
During the first exercise, we showed that using $n-1$ successor operations on any BST, when starting in the node with minimum key, has total cost $\mathcal{O}(n)$. Prove the same result using a potential.