Algorithms Example Sheet 3: Further Reading

Red-black trees

Further Reading 1 [Join/split] In one of the exercises, you saw how to access the k-th element in the RBT. Actually it is possible to represent lists as RBTs which also allow for joining and splitting in logarithmic time. You can read about them here.

Further Reading 2 [Range operations] The ability to implement join/split efficiently also implies that we can extend operations on the segment tree so that they work on lists of arbitrary length. *How?*

Further Reading 3 [Cost of restructuring a RBT] Some more tasks involving RBTs:

- Attempt problem 7.2 from here to learn more about the cost of restructuring a RBT.
- Attempt problem 4 from here.

Other BSTs

There are several other BSTs that have been developed. In particular:

- Treaps (see **[CLRS Problem 13.4]** or these lecture notes).
- Skip lists (see here). Together with treaps, these are randomised BSTs that are much simpler to implement and can be used to solve range query problems. However, they have worse worst-case guarantees.
- Splay trees (e.g. these notes). These are BSTs with $\mathcal{O}(\log n)$ amortised guarantees (for range query problems as well), but also tend to be simpler to implement. Their main disadvantage is that the guarantees are amortised, so they will be less efficient for tasks with real-time guarantees (or in a persistent setting).
- AVL trees (see **[CLRS Problem 13.3]**).