Date: Sep 11, 2025

Splay tree

- Splay tree is a self balancing binary search tree with a property that recently accessed elements can be re-accessed fast.
- Amortized running time refers to the time required to perform a sequence of (related) operations averaged over all the operations performed.
- Amortized analysis guarantees the average performance of each operation in the worst case.
- It performs the basic operations such as searching, insertion and deletion in $O(\log n)$ amortized time.
- All normal operations of a binary search tree are combined with one basic operation called splaying.

Splaying

- When a node X is accessed, a splay operation is performed on X to move it to the root.
- To perform a splay operation, certain splay steps are performed, each of which moves X closed to the root.
- Splaying a particular node of interest after every access ensures that the recently accessed nodes are kept closer to the root and the tree remains roughly balanced so that the desired amortized time bounds can be achieved.
- Each splay step depends on 3 factors:
  - Whether X is the left or right child of its parent P.
  - Whether P is the root.
  - If not, whether P is the left or right child of its parent G.
- Depending on these factors we have splay steps based on each factor.

Zig step

When P is the root, the tree is rotated on the edge between P and X.
Zig steps exist to deal with the parity issue and will be done only as the last step in the splay operation and only when X has odd depth at the beginning of the operation.

Zig-zig step

When P is not the root and X and P are both right children or both left children.

Zig-zag step

When P is not the root and X is the right child and P is the left child or vice versa (in other words: X, P and G are not in a straight line).

Searching a node into a splay tree

1. Search down the root of the splay tree looking for X.

2. If the search is successful and X is reached, splay the tree at X.
3. Otherwise (if the search is unsuccessful), splay the tree at the last non-null node reached during the search.

Inserting a node into a splay tree

1. Search for X in the splay tree.
2. If the search is successful, splay at the node X.
3. Otherwise, add the new node X in such a way that it replaces the null pointer reached during the search by a pointer a new node X. Then splay the tree at X.

Deletion

1. Search for X that is to be deleted.
2. If search is unsuccessful, splay the tree at the last non-null node encountered during the search.
3. If the search is successful and X is not the root node, then let P be the parent of X, replace X by an appropriate descendant of P and finally splay the tree at P.