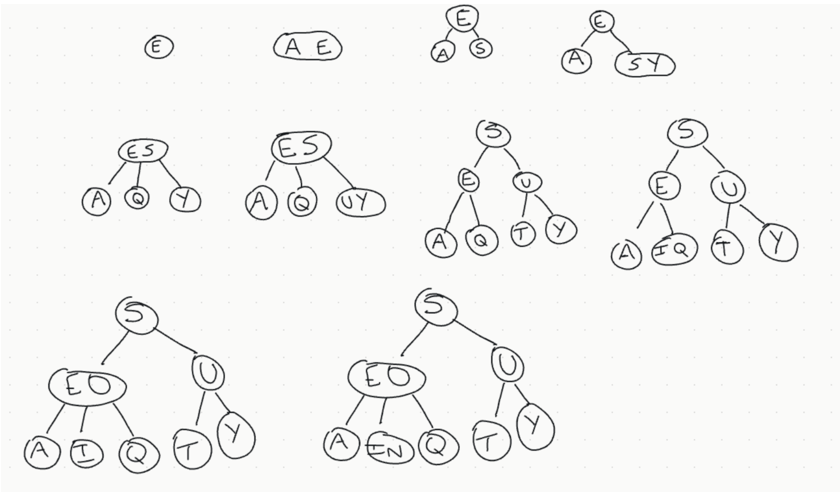

Figure 1: Sequence of 2-3 trees when inserting the keys in Exercise 1.

Sergio E. Garcia Tapia
Algorithms by Sedgewick and Wayne (4th edition) [SW11]
December 31st, 2024

3.3: Balanced Search Trees

Exercise 1. Draw the 2-3 tree that results when you insert the keys `E A S Y Q U T I O N` in that order into an initially empty tree.

Solution. See Figure 1

Exercise 2. Draw the 2-3 tree that results when you insert the keys `Y L P M X H C R A E S` in that order into an initially empty tree.

Solution. See Figure 2

Exercise 3. Find an insertion order for the keys `S E A R C H X M` that leads to a 2-3 tree of height 1.

Solution. See Figure 3.

First, consider the the sort of the keys is `A C E H M R S X`. Since there are 8 keys, we can obtain a tree of height 1 if we have four 3-nodes. In particular, `E` and `R` are at the root, and `A` and `X` are in the leftmost and rightmost leaves, respectively. We can begin by inserting `AEX`, which places `E` at the root as desired. We can then insert `R` and `H`. Both fall in the rightmost leaf, and the split of the 4-node that is created causes `R` to move to the root node, and the creation of the new leaf `H`. Now inserting `C` yields the 3-node with `A` and `C`, inserting `M` yields the node with `H` and `M`, and inserting `S` yields the node with `S` and `X`.

Exercise 4. Prove that the height of a 2-3 tree with n keys is between $\lfloor \log_3 n \rfloor \approx 0.63 \lg n$ (for a tree that is all 3-nodes) and $\lfloor \lg_n \rfloor$ (for a tree that is all 2-nodes).

Solution.

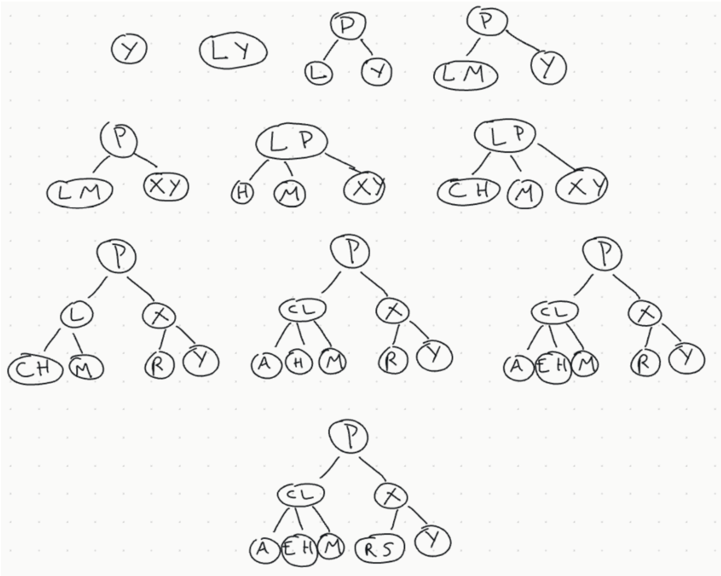

Figure 2: Sequence of 2-3 trees when inserting the keys in Exercise 2.

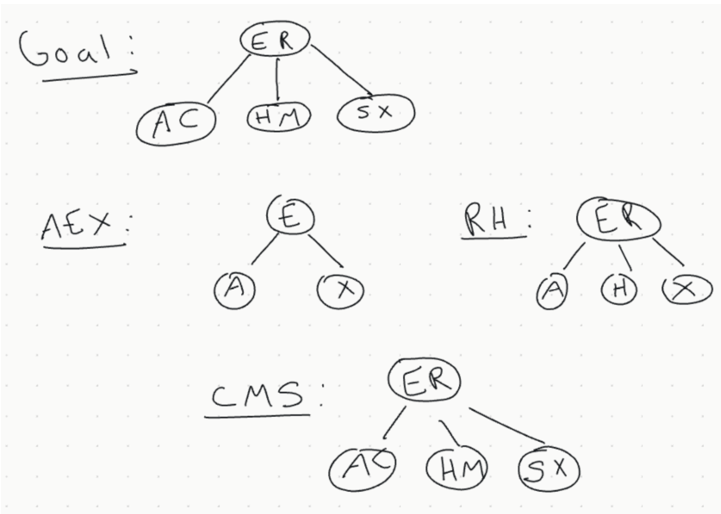

Figure 3: Tree of height 1 for keys in Exercise 3.

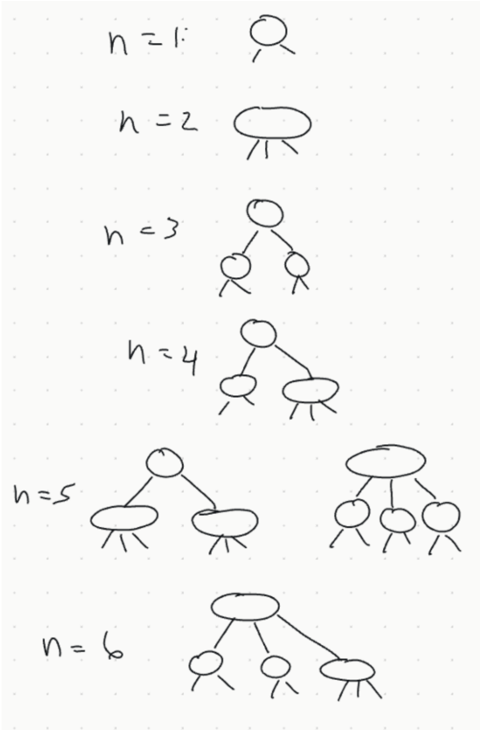

Figure 4: All of the structurally different 2-3 trees with n keys, for n from 1 through 6.

Proof. Suppose T is a 2-3 search tree. By construction, a 2-3 search tree is perfectly balanced, so T is perfectly balanced also. In the worst case, if T consists of only 2-nodes, then it will have 2^k nodes at all depths (except possibly the last one) because it is balanced. Similarly, in the best case, if T consists only of 3-nodes then it will have 3^k nodes at all depths (except possibly the last one). The former tree has a height of $\lfloor \lg_n \rfloor$, and the latter has a height of $\lfloor \log_3 n \rfloor$. We conclude that T's height lies in between the two. $\square$

Exercise 5. Figure 4 shows all the *structurally different* 2-3 trees with n keys, for n from 1 up to 6 (ignore the order of the subtrees). Draw all the structurally different trees for $n = 7$, 8, 9, and 10.

Solution. See Figure 5.

Exercise 6. Find the probability that each of the 2-3 trees in Exercise 3.3.5 is the result of the insertion of n random distinct keys into an initially empty tree.

Solution. See Figure 6 and Figure 7, both of which I have annotated. Note that when a given tree can lead to more than one structurally different tree, I have annotated the arrows indicating the probability with which it can give rise to said structure. This "transitional probability" then plays a role in the computation of the probabilities of the resulting tree.

Exercise 8. Show all possible ways that one might represent a 4-node with three 2-nodes bound together with red links (not necessarily left-leaning).

Solution. See Figure 8.

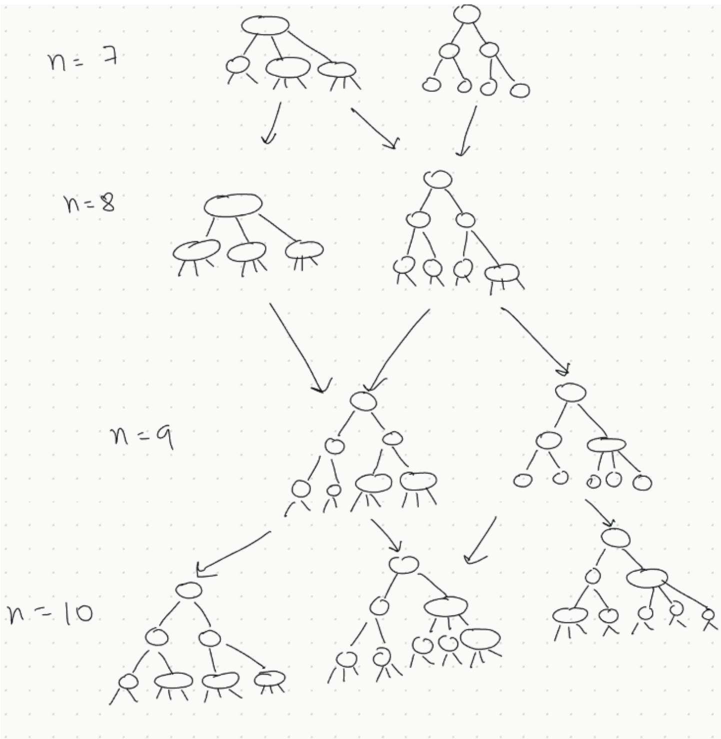

Figure 5: All of the structurally different 2-3 trees with n keys, for n from 7 through 10.

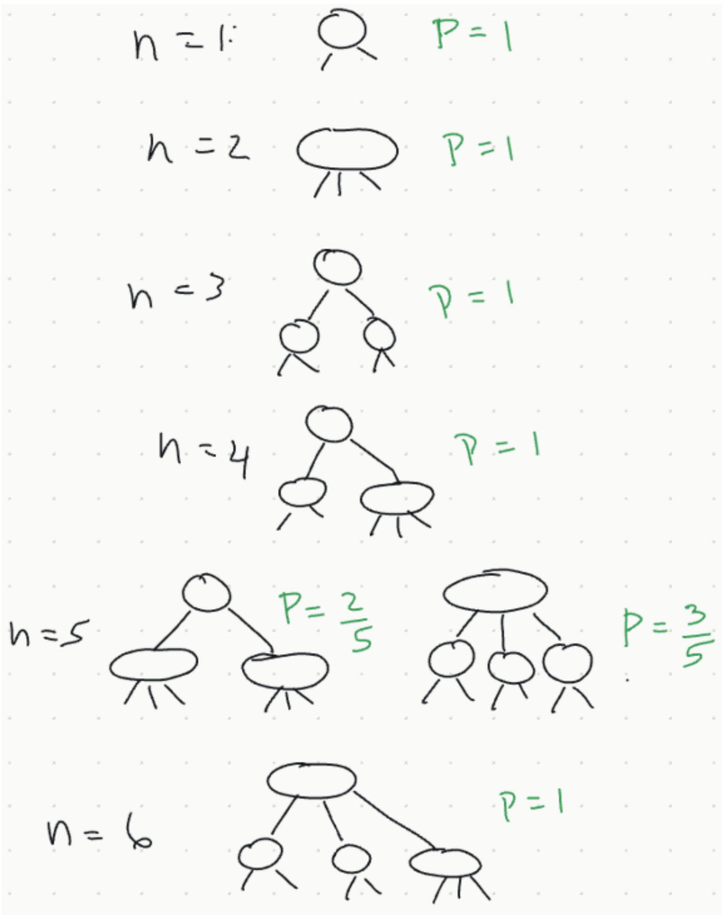

Figure 6: All of the structurally different 2-3 trees with n keys, for n from 1 through 6 and their probabilities.

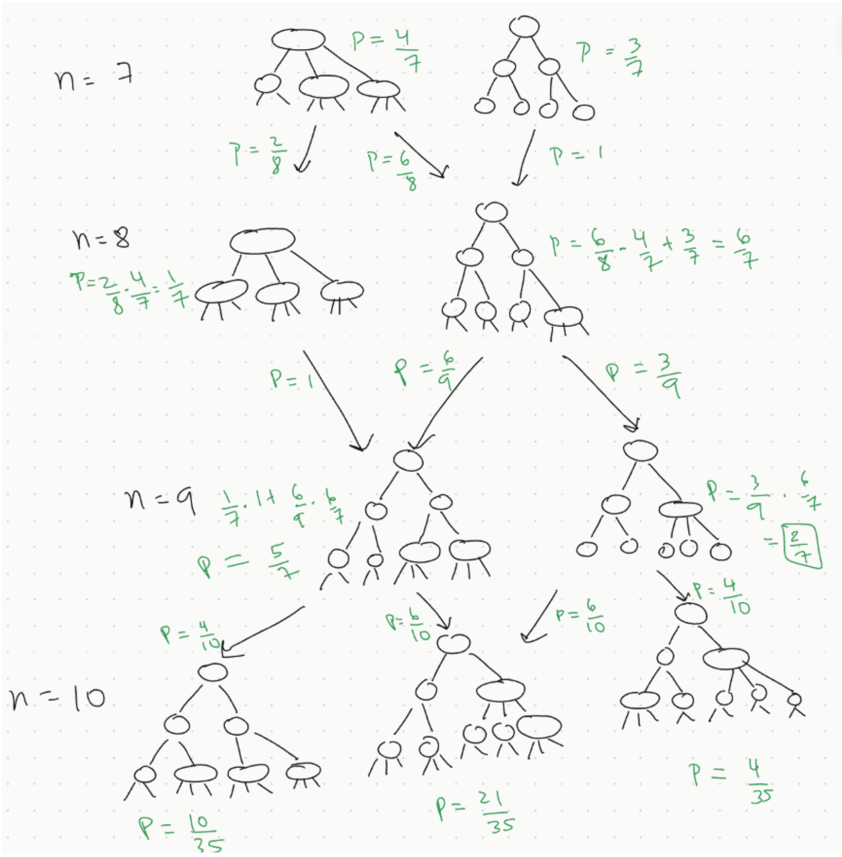

Figure 7: All of the structurally different 2-3 trees with n keys, for n from 7 through 10 and their probabilities.

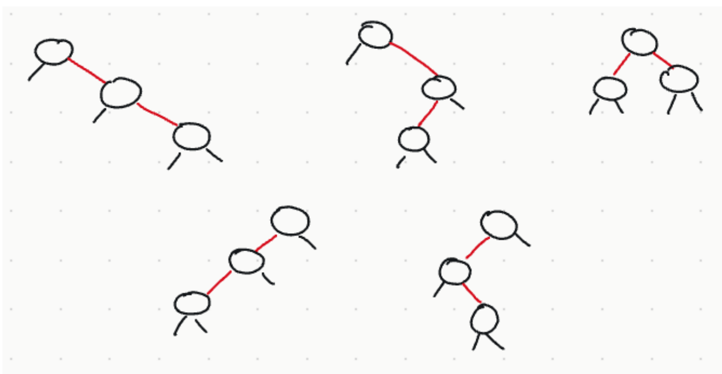

Figure 8: All ways to represent a 4-node with three 2-nodes bound by red links.

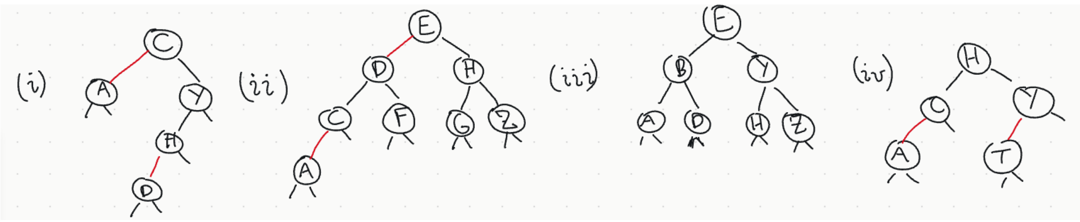

Figure 9: Options for Exercise 9.

Exercise 9. Which of the trees in Figure 9 are red-black BSTs?

Solution. Tree (i) is not a red-black BST because it does not have perfect black balance. Tree (ii) is not a red-black BST because it is not ordered. In particular, `F` is on the left subtree of `E` instead of right. The remaining trees are red-black BSTs.

Exercise 10. Draw the red-black BST that results when you insert items with the keys `E A S Y Q U T I O N` in that order into an initially empty tree.

Solution. See Figure 10.

Exercise 11. Draw the red-black BST that results when you insert the keys `Y L P M X H C R A E S` in that order into an initially empty tree.

Solution. By using the 1-1 corresponding of 2-3 trees and red-black BSTs, we can use the 2-3 tree in Exercise 3.3.2 to create the corresponding red-black BST, depicted in Figure 11.

Exercise 13. True or false: If you insert keys in increasing order into a red-black BST, the tree height is monotonically increasing.

Solution. Depends on what we mean by the tree height. The black three height monotonically increases, but the tree height strictly increases.

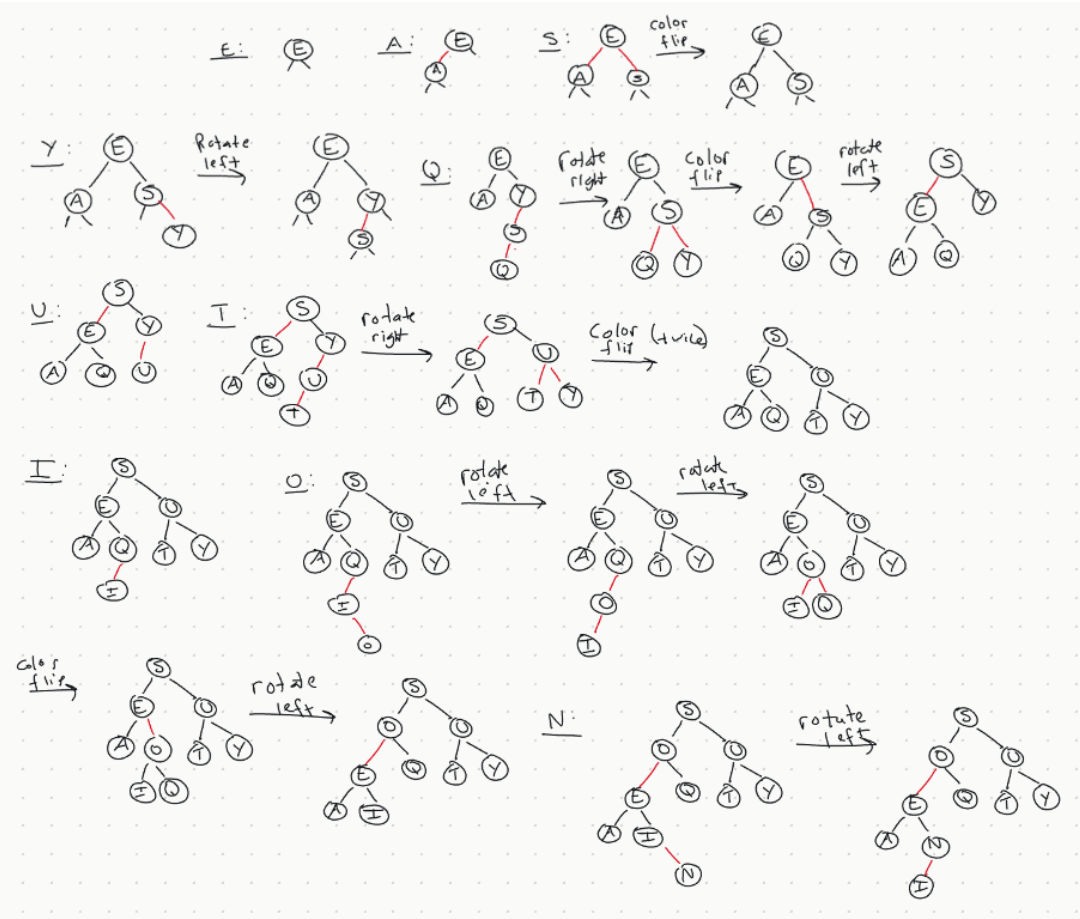

Figure 10: Sequence of red-black BSTs generated by the key sequence in Exercise 10.

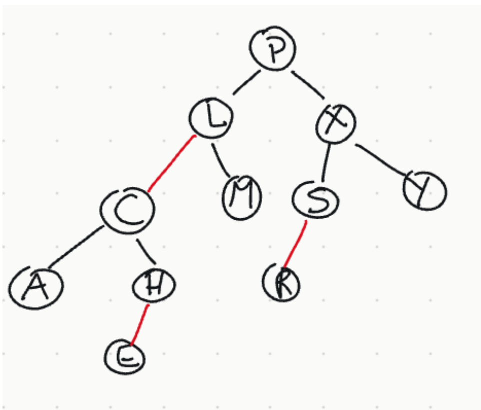

Figure 11: Red-black BST resulting from the key sequence in Exercise 11.

References

[SW11] Robert Sedgewick and Kevin Wayne. *Algorithms*. 4th ed. Addison-Wesley, 2011. ISBN: 9780321573513.