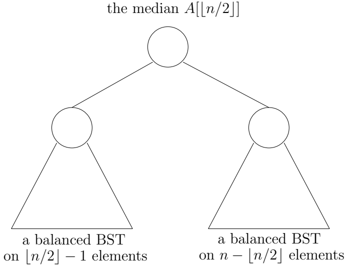
Further Discussions on BSTs

Yufei Tao's Teaching Team

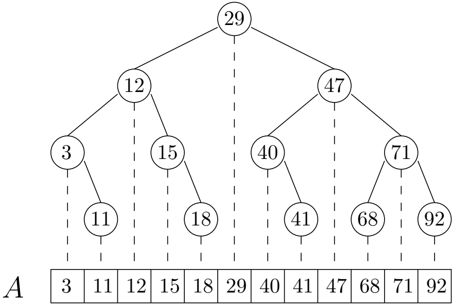
Construction of a Balanced BST from a Sorted Set

We will explain how to construct a balanced BST T on a **sorted** set S of n integers in $O(n)$ time. Assume that S is stored in a sorted array A.



This implies a recursive construction algorithm.

Example



Construction of a Balanced BST from a Sorted Set

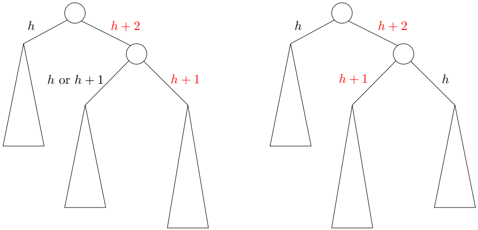
Let $f(n)$ be the maximum running time for constructing a balanced BST from a sorted array of length n. We have:

$$f(1) = O(1)$$
$$f(n) = O(1) + 2 \cdot f(\lceil n/2 \rceil)$$

Solving the recurrence gives $f(n) = O(n)$.

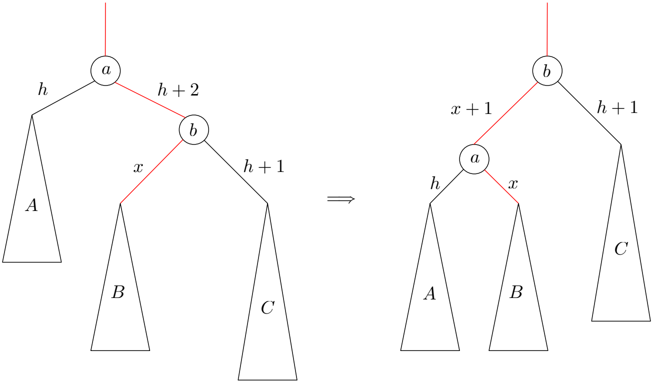
Rebalancing

In the lecture, we discussed the Left-Left and Left-Right cases in detail, so here we will look at Right-Right and Right-Left:



Right-Right

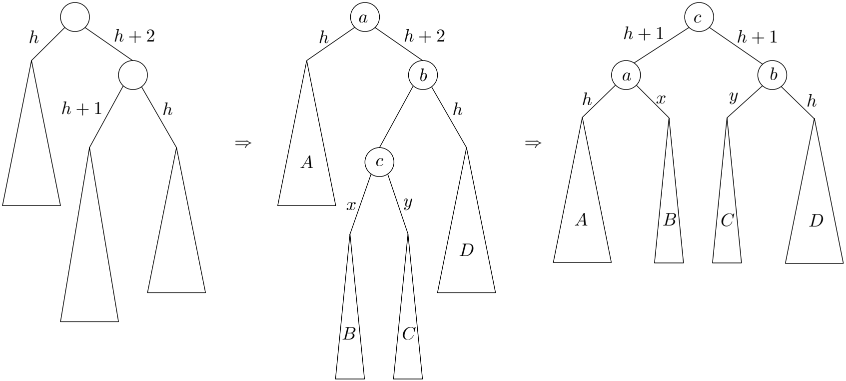
Fix by a **rotation** (symmetric to left-left):



Note that $x = h$ or $h + 1$, and the ordering from left to right of A, a, B, b, C is preserved after rotation.

Right-Left

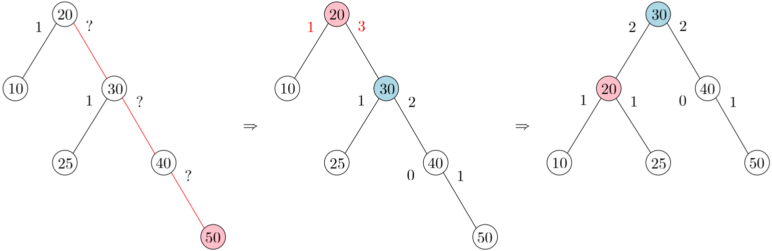
Fix by a **double rotation** (symmetric to left-right):



Note that x and y must be h or $h-1$. Futhermore at least one of them must be h.

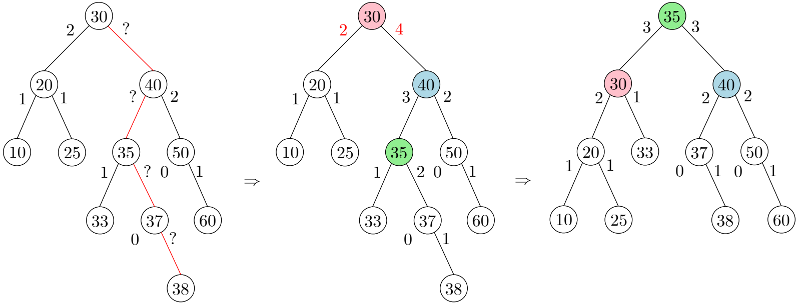
Right-Right Example

Inserting 50:



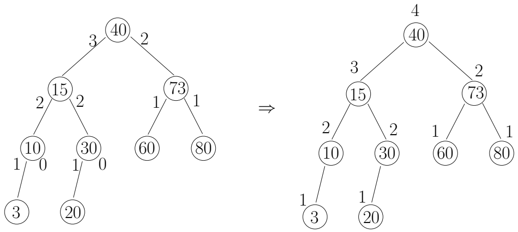
Right-Left Example

Inserting 38:



Storage of an AVL node

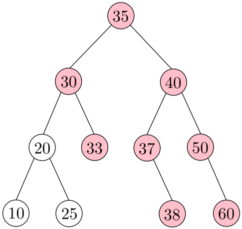
In the lecture, for each node u, we stored two subtree heights at u. Instead, we may store only the height of u at u.



How do we get the left subtree height of 40? How much time does it need?

Range Reporting

Let S be a set of n integers. Given an interval $[q, \infty)$, a **one-sided range query** reports all the integers of S that fall in $[q, \infty)$. We can use a balanced BST to answer a query in $O(\log n + k)$, where k is the number of integers reported.

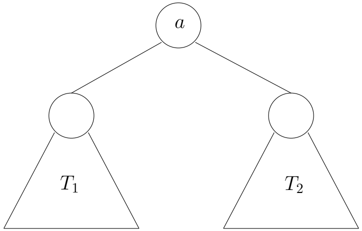


For the query $[27, \infty)$, we need to report the integers in pink.

Range Reporting

To answer a query $[q, \infty)$, we do the following at the root:

- If $a < q$, recursively report the integers in T_2 that fall in $[q, \infty)$.
- If $a = q$, report a and all the integers in T_2.
- If $a > q$, report a and all the integers in T_2. After that, recursively report the integers in T_1 that fall in $[q, \infty)$.



Apply the algorithm to answer the query on the previous slide.

Time Analysis

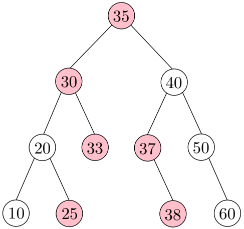
At each level of the recursion, we do the following:

- Comparing q to the integer stored in the root, the cost of which is $O(1)$.
- (If necessary) reporting all the integers in the right subtree, the cost of which is proportional to the number of integers in the right subtree.

As the height of the BST is $O(\log n)$, the first bullet costs $O(\log n)$ in total. The second step reports integers from **disjoint** subtrees and incurs cost $O(k)$ in total. The overall cost is $O(\log n + k)$

Discussion: 2-Sided Range Reporting

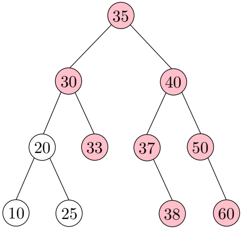
Let S be a set of n integers. Given an interval $[q_1, q_2]$, a **range query** reports all the integers of S that fall in $[q_1, q_2]$. We can use a balanced BST to answer a query in $O(\log n + k)$, where k is the number of integers reported.



For the query $[24, 39]$, we need to report the integers in pink.

Range Counting

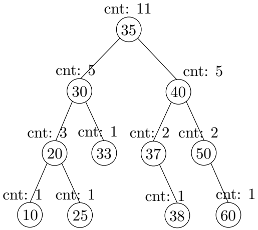
Let S be a set of n integers. Given an interval $[q, \infty)$, a **one-sided range count query** reports the **number** of integers of S that fall in $[q, \infty)$. We can use a balanced BST to answer a query in $O(\log n)$.



For the query $[27, \infty)$, we need to return 8.

Range Counting

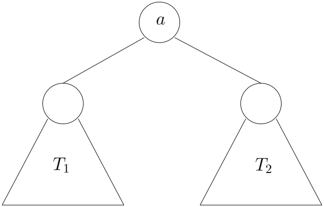
Key idea: at each node u, store a **count**, equal to the number of nodes in the subtree of u.



Range Counting

To answer a query $[q, \infty)$, we do the following at the root:

- If $a < q$, recurse on T_2.
- If $a = q$, add 1 and the right child's count to the answer.
- If $a > q$, add 1 and the right child's count to the answer. After that, recurse on T_1.



Apply the algorithm to answer the $[27, \infty)$ on the previous slide.