Supplementary slides for tutorial week 4

BST: 12.2-7

An alternative method of performing an inorder tree walk of an *n*-node binary search tree finds the minimum element in the tree by calling TREE-MINIMUM and then making *n-1* calls to TREE-SUCCESSOR. Prove that this algorithm runs in ϴ(n) time.

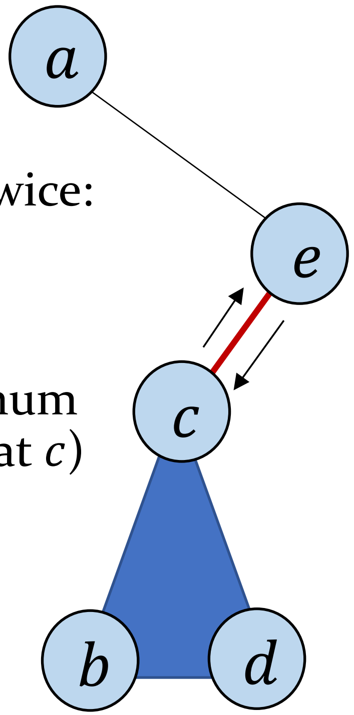
To show this bound on the runtime, we will show that using this procedure we traverse each edge twice. This will suffice because the number of edges in a tree is one less than the number of vertices.

BST: 12.2-7 (cont')

An edge is traversed at most twice.

There are two possibilities for an edge:

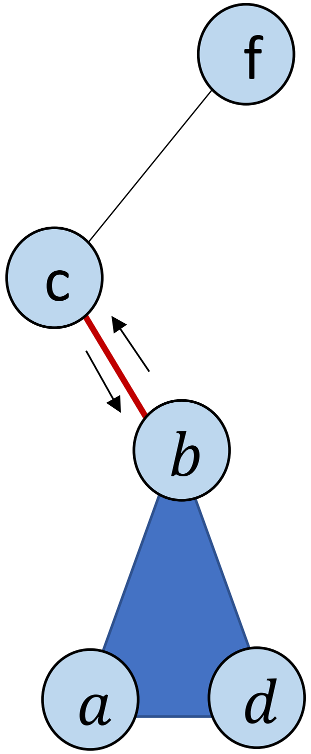
Case 1: the edge connects the node to the left child. (like the edge ec)



ec is traversed at most twice:

- To find successor of a
- To find successor of d (where d is the maximum in the subtree rooted at c)

Case 2: The edge connect the node to the right child. (like the edge cb)



cb is traversed at most twice:

- To find successor of d (where d is the maximum in the subtree rooted at b)
- To find successor of c

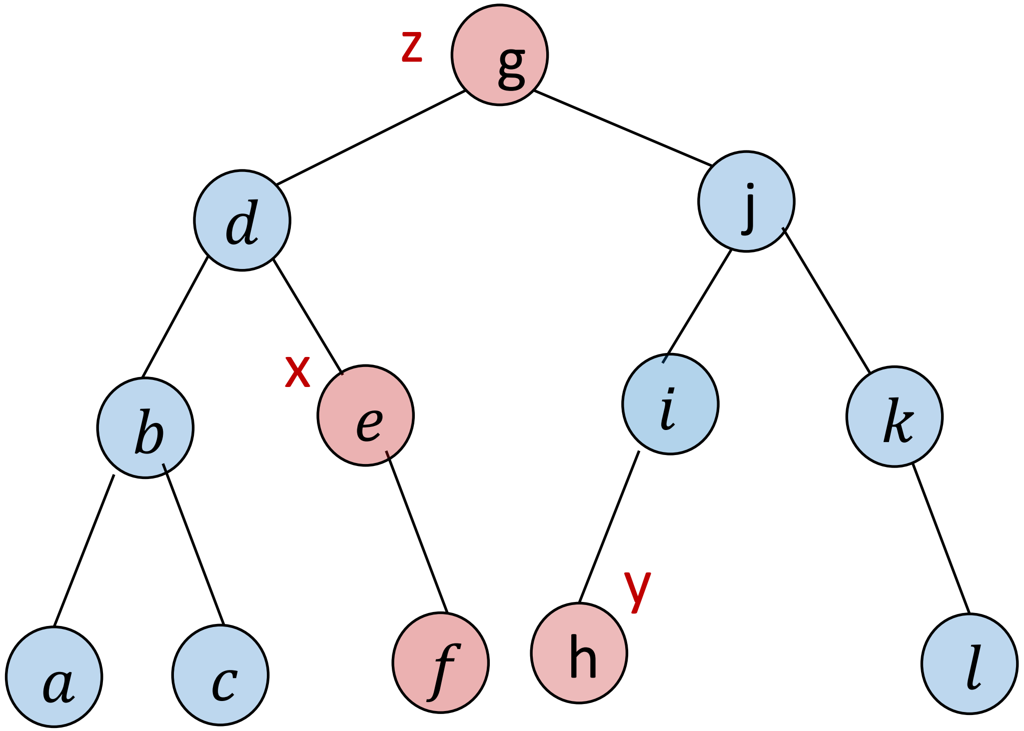
BST: 12.2-7 (cont')

Since these are the only two times that the edge can be used, apart from the initial finding of the tree min. We have that the runtime is $O(n)$. We trivially get the runtime is $\Omega(n)$ because that is the size of the input.

BST: 12.2-8

Prove that no matter what node we start at in a height-*h* binary search tree, *k* successive calls to TREE-SUCCESSOR takes O(*k* + *h*) time.

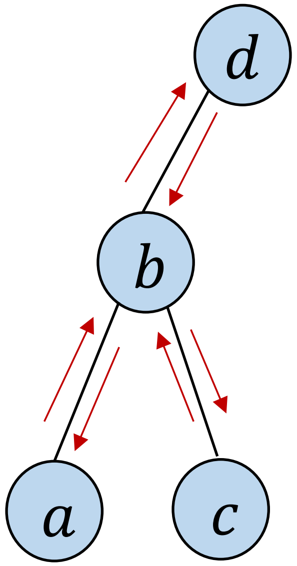
Let x be the node on which we have called TREE-SUCCESSOR and y be the k^{th} successor of x. Let z be the lowest common ancestor of z and y. There are k nodes between x and y. We want to see how many times each vertex is traversed.



BST: 12.2-8

We proved in the previous question that each edge is traversed ay most twice. Since for each vertex there are at most three connected edges (parent, left and right child) we will never examine a single vertex more than three times.

How many times the vertex b can be traversed?



BST: 12.2-8

Moreover, any vertex whose key value isn't between x and y will be examined at most once, and it will occur on a simple path from x to z or y to z. Since the lengths of these paths are bounded by h, the running time can be bounded by $3k + 2h = O(k + h)$