CSCI 241

Scott Wehrwein

Balance Factor

Goals

Know why we want our BSTs to be balanced.

Be able to calculate the balance factor of a binary tree.

Back to BSTs

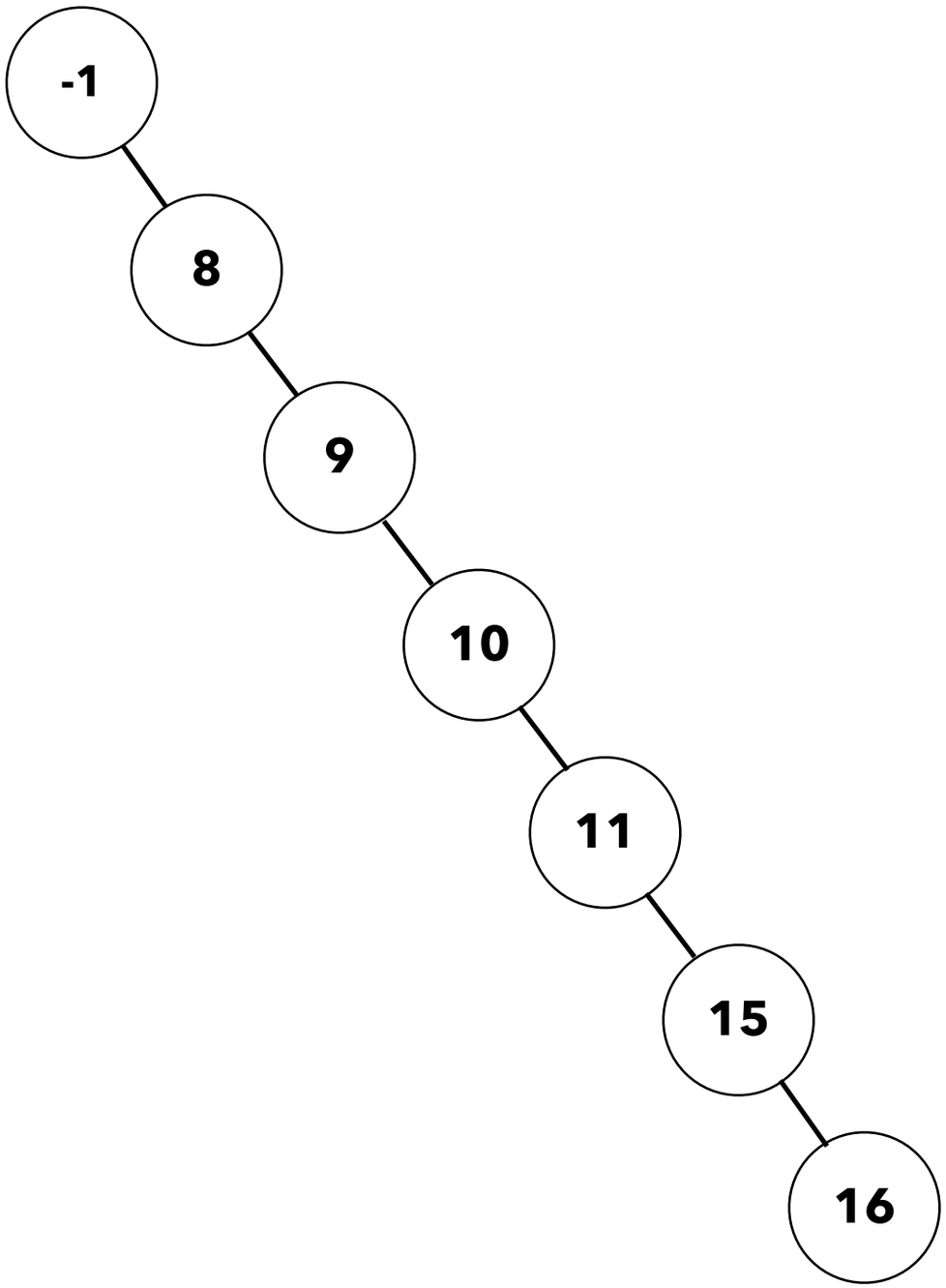
Long ago, we built some trees:

```
t = new BST();
t.insert(-1)
t.insert(8)
t.insert(9)
t.insert(10)
t.insert(11)
t.insert(15)
t.insert(16)
t.insert(16)
```

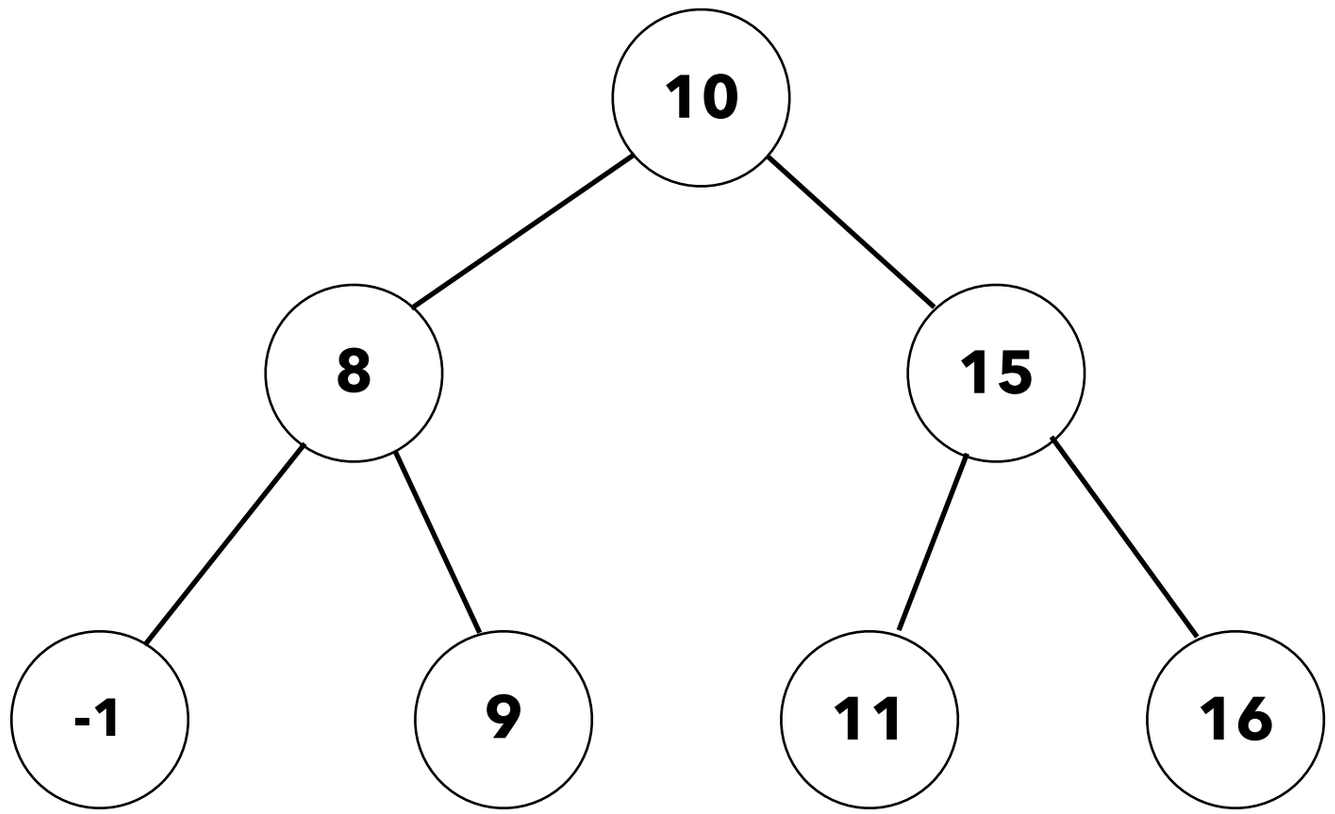
```
t = new BST();
t.insert(10)
t.insert(15)
t.insert(16)
t.insert(8)
t.insert(16)
t.insert(9)
t.insert(11)
t.insert(-1)
```

Same values, different trees

Bad tree =(



Good tree =)



Quantifying Badness: Height

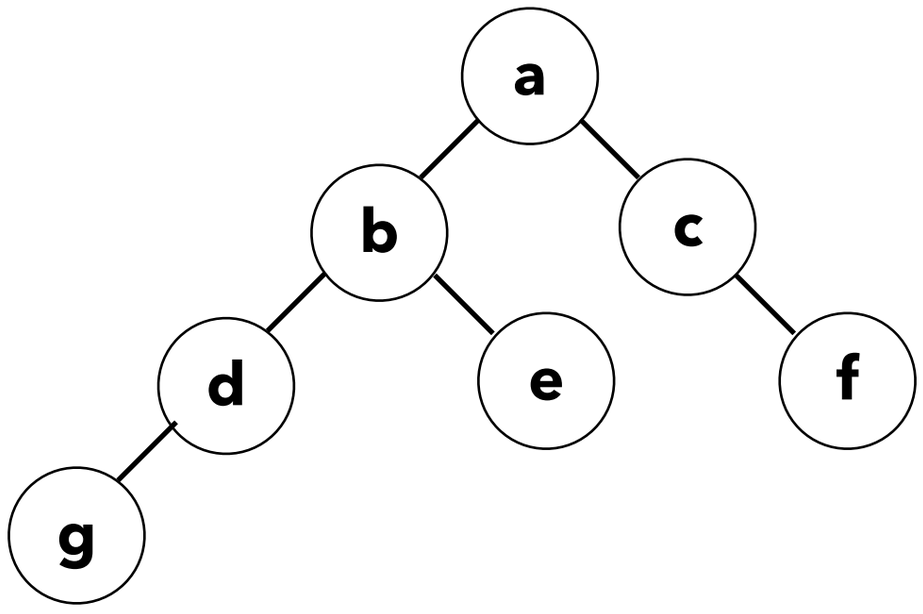
Height(t): path length from t's deepest descendant (leaf) to t's root. ← *our original definition*

Height(empty tree): -1

Height(n): height of the subtree rooted at n

for convenience

Height(n): 1 + max(Height(n.left) Height(n.right)) ← *consequence*



Quantifying Badness: Height

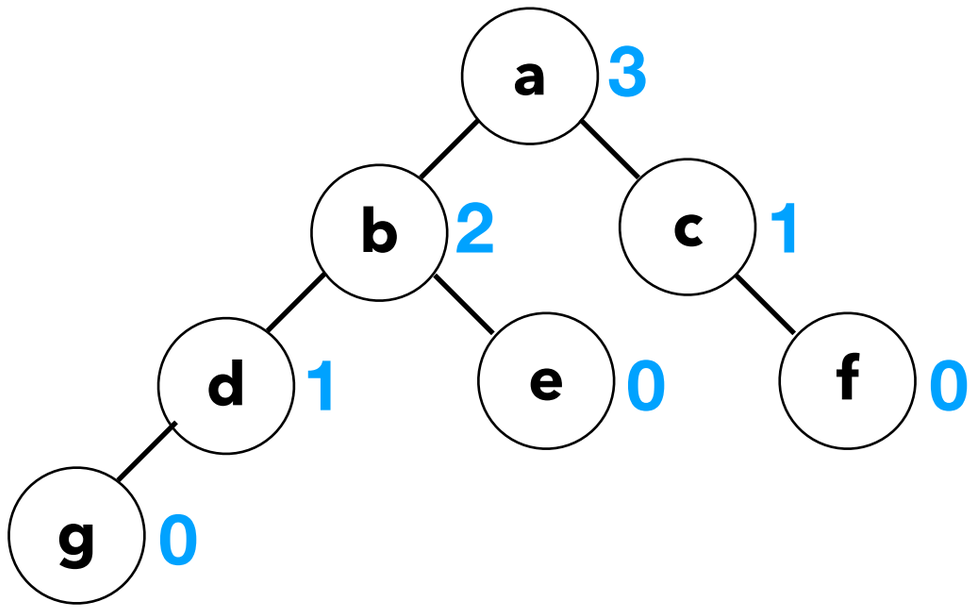
Height(t): path length from t's deepest descendant (leaf) to t's root. ← *our original definition*

Height(empty tree): -1

Height(n): height of the subtree rooted at n

for convenience

Height(n): 1 + max(Height(n.left) Height(n.right)) ← *consequence*



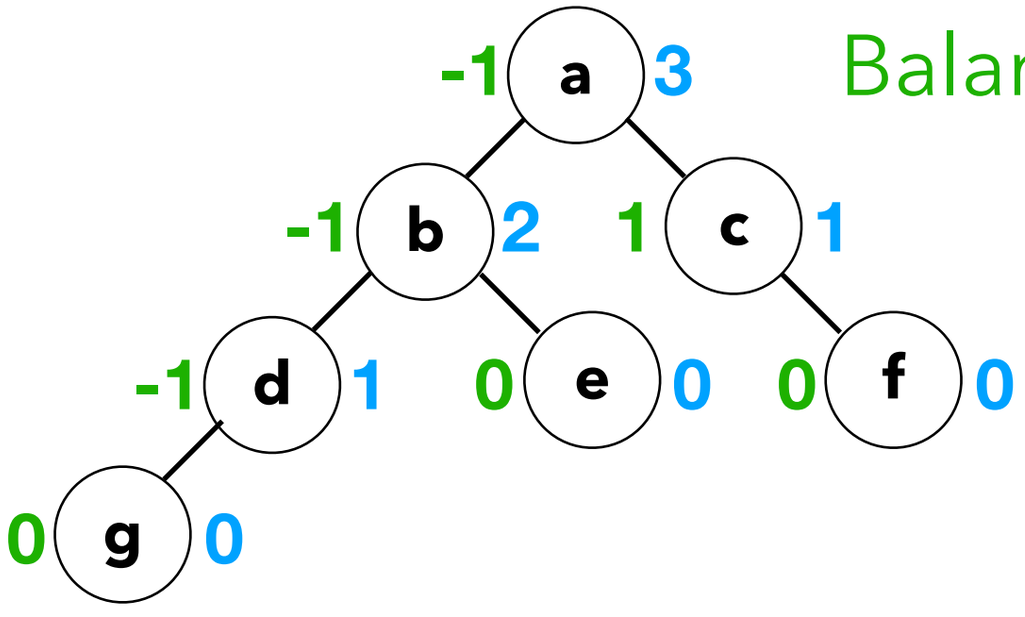
Quantifying Badness: Balance Factor

Height(t): path length from t's deepest descendant (leaf) to t's root.

Height(empty tree): -1

Height(n): height of the subtree rooted at n

Height(n): 1 + max(Height(n.left) Height(n.right))



Balance(n): height(n.right) - height(n.left)