

Announcements

Thursday Extras,

None next week

Next week: Likely Thursday

The following week: Likely Tuesday and Thursday

Office Hours

Today: 2:30-4:00, as usual

Next Week (tentative)

Monday-Thursday: 2:30-4:00

Friday: canceled

Suppose a tree is based on the following TreeNode definition

```
String name;  
TreeNode left;  
TreeNode right;
```

Nodes are ordered by name

(if last names are the same, searching goes right)

We want to determine if a tree is height balanced
i.e., the heights of all leaves are within 1

Two approaches are proposed

1. Traverse the tree once to find the minimum height of a leaf and a second time to find the maximum height of the leaf.
Then compute if the $\text{max} - \text{min} \leq 1$
2. In a single traversal of the tree, keep track of both the minimum and maximum height of trees below

Which of these approaches would work?

- A. Both 1 and 2
- B. 1, not 2

- C. 2, not 1
- D. Neither 1 nor 2

Suppose a tree is based on the following TreeNode definition

```
String name;  
TreeNode left;  
TreeNode right;
```

Nodes are ordered by name

(if last names are the same, searching goes right)

We want to determine if a tree is height balanced
i.e., the heights of all leaves are within 1

Two approaches are proposed

1. Traverse the tree once to find the minimum height of a leaf and a second time to find the maximum height of the leaf.
Then compute if the $\text{max} - \text{min} \leq 1$
2. In a single traversal of the tree, keep track of both the minimum and maximum height of trees below

Which of these approaches would be more efficient?

- | | |
|---|----------------------------|
| A. 1 and 2 have about the same efficiency | B. 1 more efficient than 2 |
| | C. 2 more efficient than 1 |

We want to determine if a tree is height balanced
i.e., the heights of all leaves are within 1

We want to answer the question with a single traversal of the tree, keeping track of both the minimum and maximum height of trees below

The user would want a method
boolean isBalanced ()
that simply returns true or false.

What is a reasonable way to organize processing, so a traversal keeps track of minimums and maximums, but the method returns a boolean?

- A. Can't be done
- B. Global variables
- C. Husk and kernel methods, where kernel may have parameters and may return something non-boolean

Given

boolean isBalanced ()

returns true or false and calls a kernel method,
how should isBalancedKernel organize processing (e.g.,
parameters and return)?

- A. Have a parameter that keeps track of node levels, as one goes down the tree, and make decisions about balance at the bottom of the tree.
- B. Pass leaf information for both minimums and maximums up the tree, checking balance going upward.
- C. Another approach

Given

boolean isBalanced ()

returns true or false and calls a kernel method,

How could we pass multiple data elements up a tree with isBalancedKernel?

1. Have separate min, max, balanced parameters down the tree, and update them when each method is called.
2. Pass an array of data as a parameter down the tree, and update the array when each method is called.
3. Pass an object with several fields as a parameter, and update object when each method is called.
4. Designate that isBalancedKernel returns an object with several fields.

A. Both 1,2 B. 1, not 2 C. 2, not 1 D. not 1,2

A. Both 3,4 B. 3, not 4 C. 4, not 2 D. not 3,4

In processing a height-balanced tree, what information do we need at each node?

Height of node from bottom,
whether node and its subtrees are balanced

Can this be done in with a single int return?

Ideas?