CS 624: Notes 06

Ryan Culpepper

September 26, 2022

These notes contain an outline of what I said in lecture (but only an outline), and they also contain interactive questions and exercises. The corresponding slides are in `slides04.pdf`.

1 Administrative

Homework 02 will come out soon.

2 Homework 01 Review

Problem 6. Based on a quick sampling:

- some good answers
- some that claimed to use induction but didn't
- some that did induction but skipped important steps
- some that "proved" it for one example or for a special case

2.1 Example

(slides)

2.2 What exactly is the goal?

Prove:

> for all $n \in \mathbb{N}$, for all binary trees t:
> if $\text{mindepth}(t) \geq n$, then $\text{countnil}(t) \geq 2^n$

That is:

$$\forall n \in \mathbb{N},\ \forall t \in \text{BinaryTree},\ \ \text{mindepth}(t) \geq n \implies \text{countnil}(t) \geq 2^n$$

2.3 Proof by Induction

By induction on n.

Base case: n = 0

Let t be any binary tree.

$$\text{mindepth}(t) \geq 0 \qquad \text{always}$$

Since every binary tree must eventually end in NIL:

$$\text{countnil}(t) \geq 1 = 2^0$$

Done with base case.

Inductive case: n = k+1

Inductive hypothesis (k):

for all t, if $\text{mindepth}(t) \geq k$, then $\text{countnil}(t) \geq 2^k$

Goal:

for all t, if $\text{mindepth}(t) \geq k+1$, then $\text{countnil}(t) \geq 2^{k+1}$

Strategy: What do I need to show? What do I have available?

- The goal has the form $\forall x(A(x) \to B(x))$.
  For a particular x, assume $A(x)$ (that gives info!) and prove $B(x)$.
- The IH has the form $\forall x(A(x) \to B(x))$.
  I can apply it to an x (can even apply multiple times to different xs), but I must have a proof of $A(x)$ in order to get the information I want ($B(x)$).
- How can I use what I have to prove the goal?
  - I probably need to apply the IH to the left and right children.
  - But I need appropriate facts about their mindepths to do so. What do I have available that might tell me something about the mindepths of a node's children?

Let t be an arbitrary binary tree.
Goal: if $\text{mindepth}(t) \geq k+1$, then $\text{countnil}(t) \geq 2^{k+1}$

Assume $\text{mindepth}(t) \geq k+1$
Goal: $\text{countnil}(t) \geq 2^{k+1}$

Since $\text{mindepth}(t) \geq k+1$, then t can't be NIL.
So the second cases of each function apply.

$$\begin{aligned} \text{mindepth}(t) &\geq k+1 \\ 1 + \min(\text{mindepth}(\text{left}(t)), \text{mindepth}(\text{right}(t))) &\geq k+1 \\ \min(\text{mindepth}(\text{left}(t)), \text{mindepth}(\text{right}(t))) &\geq k \end{aligned}$$

Facts about min:

$$\min(a,b) \leq a \qquad \min(a,b) \leq b$$

So

$$\begin{aligned} \text{mindepth}(\text{left}(t)) &\geq \min(\dots) \geq k \\ \text{mindepth}(\text{right}(t)) &\geq \min(\dots) \geq k \end{aligned}$$

Now I can finally apply the IH to left(t) and right(t)!

$$\begin{aligned} \text{countnil}(\text{left}(t)) &\geq 2^k \\ \text{countnil}(\text{right}(t)) &\geq 2^k \end{aligned}$$

Reminder: the goal is countnil(t) $\geq 2^{k+1}$

$$\begin{aligned} \text{countnil}(t) &= \text{countnil}(\text{left}(t)) + \text{countnil}(\text{right}(t)) \\ &\geq 2^k + 2^k = 2^{k+1} \end{aligned}$$

Done with inductive case.

That completes the proof.

3 Quicksort

3.1 Partition

```
| S | S | S | L | L | L | ? | ? | x |
  p       i  i+1     j-1  j       r
```

where

`x` is the *pivot*
`S` is smaller ($\leq$) than the pivot — `A[p..i]`
`L` is larger ($>$) than the pivot — `A[i+1..j-1]`
`?` has not yet been examined — `A[j..r-1]`

How can we place `A[j]` if *larger* than pivot, minimizing moves?

```
j ← j + 1
```

How can we place `A[j]` if *smaller*, minimizing moves?

```
swap A[j] with A[i+1]
i ← i + 1   // small zone gets bigger!
j ← j + 1
```

3.2 Slide 4: Partition

```
Partition(A,p,r) :=

x ← A[r]
i ← p-1
for j ← p to r-1 do
  if A[j] ≤ x then
    i ← i+1
    exchange A[i] ↔ A[j]
  end if
end for
exchange A[i+1] ↔ A[r]
return i+1
```

3.3 Slide 8: Correctness

Another loop invariant.

Why is this true initially?

3.4 Slide 13: Running time (best case)

`Partition(A,p,r)` is clearly $\Theta(n)$ where $n = r - p + 1$.

`Quicksort(A,p,r)` is ???.

3.5 Slide 15: Running time (average case)

Empirically, in many cases, $O(n \log n)$.

Thought experiment:

> What if we don't get a 50%/50% split (the best case),
> but we get a 90%/10% split at each step instead?

$$T(n) = T(0.9n) + T(0.1n) + cn$$

How can we calculate bounds for this T(n)? What technique?

recursion tree

The master method does not apply. We can use a recursion tree, though.