So far in the course :

PMI / WOP

↳ Properties of $\mathbb{N}$,

gcd and modular arithmetic

↳ $\mathbb{Z}_p$, $\mathbb{Z}_n$ etc

↳ some applications

Along the way, we discussed some proof proof techniques (proof by contradiction, proof by taking contrapositive, proof by explicit construction) and studied some discrete structures, and why / how they are different compared to non-discrete structures.

In this lecture, we will go back to induction, and see how to simplify induction proofs in certain settings, using a more general form of induction called STRUCTURAL INDUCTION.

Lecture 13: Structural Induction

Reference: https://cs.carleton.edu/faculty/dlibenno/book/ch05_mathematical-induction_2021_September_08.pdf

Recall the $\alpha\beta\gamma$ puzzle.

Axiom: $\alpha\beta$

Inference Rules: $x\beta \xrightarrow{(1)} x\beta\gamma$ $\quad x\beta\beta\beta y \xrightarrow{(3)} x\gamma y$

$\alpha x \xrightarrow{(2)} \alpha x x$ $\quad x\gamma\gamma y \xrightarrow{(4)} xy$

Claim 13.1: $\alpha\gamma$ cannot be derived using the inference rules (1), (2), (3), (4), starting with $\alpha\beta$.

As discussed earlier, we will prove something stronger.

Claim 13.2: For all $n \in \mathbb{N}$, if string s is derived using the inference rules (1), (2), (3), (4), starting with $\alpha\beta$, then (number of βs in s) mod $3 \neq 0$.

Proof: using regular induction.

we will perform induction on the number of applications of inference rules.

$P(n)$: For any n-sequence $(\theta_1, \theta_2, \ldots, \theta_n)$ where each $\theta_i \in \{1, 2, 3, 4\}$, if $\alpha\beta \xrightarrow{\theta_1} \xrightarrow{\theta_2} \ldots \xrightarrow{\theta_n} s$, then (number of βs in s) mod $3 \neq 0$.

Base case : $n = 0$.

Induction step : $P(n) \Rightarrow P(n+1)$.

check each inference rule separately

$\vdots$ (complete the proof)

$\vdots$

■

Note that performing induction on the number of applications of inference rules is a non-trivial choice. Why did we not perform induction on the length of string produced ?

Using 'STRUCTURAL INDUCTION', we can have a shorter proof. Structural induction is useful when we have recursively defined sets (which is often the case in computer science).

How is $\alpha\beta\gamma$ puzzle a recursively defined set? Suppose we define the set of all 'theorems' that can be proven in this $\alpha\beta\gamma$ proof system.

Set S defined recursively:

Base case: $\alpha\beta \in S$.

Recursive step:

1. If $x\beta \in S$, then $x\beta\gamma \in S$.
2. If $x\beta\beta\beta y \in S$, then $x\gamma y \in S$.
3. If $\alpha x \in S$, then $\alpha xx \in S$.
4. If $x\gamma\gamma y \in S$, then $xy \in S$.

STRUCTURAL INDUCTION FOR RECURSIVELY DEFINED SETS:

universe. This can be same as S, or different.

Suppose $S \subseteq \mathcal{U}$ is recursively defined, and $P: \mathcal{U} \to \{T, F\}$ some predicate. We want to prove that $P(x) = T$ for all $x \in S$.

Step 0: State that you are using structural induction. Define predicate P.

Step 1: Show $P(x) = T$ for every x in the 'base case' definition of S.

Step 2: Show $P(x) = T$ for every 'recursive step'.
Suppose recursive step is of the form
"if $y_1 \in S$, $y_2 \in S$, ..., $y_k \in S$, then
x derived from $y_1, y_2, \ldots, y_k$ is in S"
Then you need to show that
$P(y_1) \wedge P(y_2) \wedge \ldots \wedge P(y_k) \Rightarrow P(x)$.

We will now prove Claim 13.2 using structural induction.

Proof: Proof by structural induction.

Predicate $P(x)$: T if
(number of β in x) mod $3 \neq 0$.

Base case: $\alpha\beta \in S$.
$x = \alpha\beta$
$P(x) = T$.

Induction step: we will examine each of the rules in the recursive def.ⁿ of S.

1. If $x\beta \in S$, then $x\beta\gamma \in S$.
2. If $x\beta\beta\beta y \in S$, then $x\gamma y \in S$.
3. If $\alpha x \in S$, then $\alpha xx \in S$.
4. If $x\gamma\gamma y \in S$, then $xy \in S$.

Rule 1: $P(x\beta) = T$.
Then $P(x\beta\gamma) = T$

number of β in $x\beta$
= number of β in $x\beta\gamma$

Rule 2: $P(x\beta\beta\beta y) = T$. Then $P(x\gamma y) = T$.

number of β in $x\beta\beta\beta y$
= number of β in $x\gamma y$.

Rule 3: $P(\alpha x) = T$. Then $P(\alpha xx) = T$

number of β in αxx
= 2(number of β in αx).

Therefore, if (number of β in αx) mod $3 \neq 0$,
then (number of β in αxx) mod $3 \neq 0$.

Rule 4: $P(x\gamma\gamma y) = T$. Then $P(xy) = T$.

number of β in $x\gamma\gamma y$
= number of β in xy.

Hence, using structural induction,
$P(x) = T$ for all $x \in S$.

■

Regular induction is a special case of structural induction, since $\mathbb{N}$ can be recursively defined.

$\mathbb{N}$:

Base case : $1 \in \mathbb{N}$

Recursive step : If $x \in \mathbb{N}$, then $x+1 \in S$.

Check that structural induction for $\mathbb{N}$ is identical to PMI.

Proving properties of recursively defined data structures :

Several data structures can be defined recursively, and therefore we can use structural induction to prove properties about the data structure. In COL106, you have studied AVL trees. These are binary search trees where, for every node, $|$ the height of left subtree − height of right subtree $| \leq 1$.

We will first define AVL trees recursively. Since we are interested only in the structure of these trees, we will not include the 'data' at each node.

Defining AVL trees recursively :

Base cases :

empty tree : denoted using $\perp$. $\perp \in AVL$.

height $(\perp) = -1$

leaf node : one whose both children are empty. Denoted using $(\perp, \perp)$.

$(\perp, \perp) \in AVL$.

height $((\perp, \perp)) = 0$.

Recursive Step :

1. If $T_1, T_2 \in AVL$, and height (T_1) = height (T_2), then $T = (T_1, T_2) \in AVL$ and $T' = (T_2, T_1) \in AVL$

   height (T) = height (T') = height $(T_1) + 1$

2. If $T_1, T_2 \in AVL$, and height (T_1) = height $(T_2) + 1$, then $T = (T_1, T_2) \in AVL$ and $T' = (T_2, T_1) \in AVL$

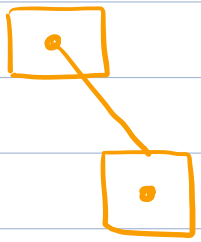
   height (T) = height (T') = height $(T_1) + 1$

3. If $T_1, T_2 \in AVL$, and height $(T_1) + 1$ = height (T_2), then $T = (T_1, T_2) \in AVL$ and $T' = (T_2, T_1) \in AVL$

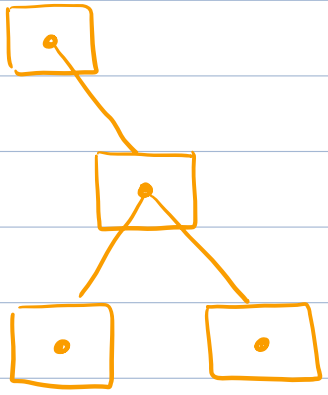
   height (T) = height (T') = height $(T_2) + 1$

Examples of trees using this abstract notation:

T_1 = Leaf node  : $(\perp, \perp)$ height $(T_1) = 0$

T_2 =  : $(\perp, (\perp, \perp))$

height $(T_2) = 1$

T_3 =  : $(\perp, ((\perp, \perp), (\perp, \perp)))$

height $(T_3) = 2$

The property that we want to study:
let $sp(T)$ denote the length of the shortest path from root to any leaf node. Can $sp(T)$ be very short for an AVL tree of height $(T) = h$?

Let us define $sp(T)$ formally.

$sp(\perp) = -1$

$sp((\perp, \perp)) = 0$

If $T = (T_1, T_2) \in AVL$, then

$$sp(T) = \begin{cases} \min(sp(T_1), sp(T_2)) + 1 & \text{if } T_1 \neq \perp, T_2 \neq \perp \\ sp(T_1) + 1 & \text{if } T_2 = \perp, T_1 \neq \perp \\ sp(T_2) + 1 & \text{if } T_1 = \perp, T_2 \neq \perp \\ 0 & \text{if } T_1 = T_2 = \perp \end{cases}$$

Claim: For all $T \in AVL$, $T \neq \perp$,
$sp(T) \geq height(T)/2$.

Proof by structural induction.

Predicate $P(T)$ = if $T \neq \perp$, then
$sp(T) \geq height(T)/2$

Base Case:
$sp((\perp, \perp)) = height((\perp, \perp)) = 0$.
Since we are considering $T \neq \perp$, we dont need to consider the empty tree.

Induction Step:
We need to show that the predicate $P(\cdot)$ holds for each of the three rules.

Rule 1: $T_1, T_2 \in AVL$, $height(T_1) = height(T_2)$
$height(T) = height(T_1) + 1$

If $T_1 = T_2 = \perp$, then $height(T) = sp(T) = 0$.

Else, if $height(T_1) = height(T_2)$,
then $T_1 \neq \perp$, $T_2 \neq \perp$.

Therefore, by definition of $sp(\cdot)$,
$sp(T) = \min(sp(T_1), sp(T_2)) + 1$

Using induction, we know
(i) $2 \cdot sp(T_1) \geq height(T_1)$, $2 \cdot sp(T_2) \geq height(T_2)$.

From (i), $2 \cdot sp(T_1) + 1 \geq height(T_1) + 1$
$= height(T)$

and $2 \cdot sp(T_2) + 1 \geq height(T_2) + 1$
$= height(T)$

?!

Therefore, $2 \min(sp(T_1), sp(T_2)) + 2$
$> 2 \min(sp(T_1), sp(T_2)) + 1$
$\geq height(T)$

Rule 2: $T_1, T_2 \in AVL$, $height(T_1) = height(T_2) + 1$
$height(T) = height(T_1) + 1$

Using induction, we know
(i) $2 \cdot sp(T_1) \geq height(T_1)$, $2 \cdot sp(T_2) \geq height(T_2)$.

From (i), $2 \cdot sp(T_1) + 1 \geq height(T_1) + 1$

$= height(T)$ (*)

and $2 \cdot sp(T_2) + 2 \geq height(T_2) + 2$

$= height(T_1) + 1$ (**)

$= height(T)$

$$sp(T) = \begin{cases} \min(sp(T_1), sp(T_2)) + 1 & \text{if } T_1 \neq \perp,\ T_2 \neq \perp \\ 1 + sp(T_1) & \text{if } T_2 = \perp,\ T_1 \neq \perp \\ 1 + sp(T_2) & \text{if } T_1 = \perp,\ T_2 \neq \perp \\ 0 & \text{if } T_1 = T_2 = \perp. \end{cases}$$

Since $height(T_1) = height(T_2) + 1$, $T_1 \neq \perp$.

$$sp(T) = \begin{cases} \min(sp(T_1), sp(T_2)) + 1 & \text{if } T_2 \neq \perp \\ 1 + sp(T_1) & \text{if } T_2 = \perp \end{cases}$$

If $T_2 \neq \perp$, using (*) and (**), it follows

$2 \cdot sp(T) = 2 \cdot (\min(sp(T_1), sp(T_2)) + 1) \geq height(T)$

If $T_2 = \perp$, $2 \cdot sp(T) = 2 + 2 \cdot sp(T_1) \geq height(T_1) + 2 > height(T)$

Analysis for Rule 3 is similar.

Using structural induction, predicate $P(\cdot)$ is true for all $T \in AVL$. ∎

The above result is tight. One can construct AVL trees where $sp(T) = \lceil height(T)/2 \rceil$.