Midterm

- Midterm in class Monday
- Covers material up to ordinary induction (HW5)
- Closed book, closed notes
  - will provide reference sheets
- No calculators
  - arithmetic is intended to be straightforward
  - (only a small point deduction anyway)

Midterm

- 5 problems covering:
  - Logic / English translation
  - Circuits / Boolean algebra / normal forms
  - Modular equations
  - Induction
  - Set theory
  - (any English proofs would have templates)

- 10 minutes per problem
  - write quickly
  - focus on the overall structure of the solution

CSE 311: Foundations of Computing

Lecture 19: Structural Induction

Last time: Recursive Definitions of Sets

Even numbers

Basis: $0 \in S$

Recursive: If $x \in S$, then $x+2 \in S$

Powers of 3

Basis: $1 \in S$

Recursive: If $x \in S$, then $3x \in S$.

Fibonacci numbers

Basis: $(0, 0) \in S$, $(1, 1) \in S$

Recursive: If $(n-1, x) \in S$ and $(n, y) \in S$, then $(n+1, x + y) \in S$.

Last time: Structural Induction

How to prove $\forall\, x \in S, P(x)$ is true:

Base Case: Show that $P(u)$ is true for all specific elements u of S mentioned in the ***Basis step***

Inductive Hypothesis: Assume that P is true for some arbitrary values of *each* of the existing named elements mentioned in the ***Recursive step***

Inductive Step: Prove that $P(w)$ holds for each of the new elements w constructed in the ***Recursive step*** using the named elements mentioned in the Inductive Hypothesis

Conclude that $\forall\, x \in S, P(x)$

Last time: Structural vs. Ordinary Induction

Ordinary induction is a special case of structural induction:

Recursive definition of $\mathbb{N}$

Basis: $0 \in \mathbb{N}$

Recursive step: If $k \in \mathbb{N}$, then $k + 1 \in \mathbb{N}$

Last time: Recursive Definitions

- Recursively defined functions and sets are our mathematical models of code and the data it uses
  - any recursively defined set can be translated into a Java
  - any recursively defined function can be translated into a Java function

    some (but not all) can be written more cleanly as loops

- Can now do proofs about CS-specific objects

Lists of Integers

- **Basis:** $\text{nil} \in \textbf{List}$
- **Recursive step:**

  if $L \in \textbf{List}$ **and** $a \in \mathbb{Z}$,

  then $a :: L \in \textbf{List}$

Examples:

– nil	[]
– $1 :: \text{nil}$	[1]
– $1 :: 2 :: \text{nil}$	[1, 2]
– $1 :: 2 :: 3 :: \text{nil}$	[1, 2, 3]

Functions on Lists

Length:

$$\text{len}(\text{nil}) := 0$$
$$\text{len}(a :: L) := \text{len}(L) + 1 \qquad \textbf{for any } L \in \textbf{List and } a \in \mathbb{Z}$$

Concatenation:

$$\text{concat}(\text{nil}, R) := R \qquad \textbf{for any } R \in \textbf{List}$$
$$\text{concat}(a :: L, R) := a :: \text{concat}(L, R) \qquad \textbf{for any } L, R \in \textbf{List and any } a \in \mathbb{Z}$$

Last time: Structural Induction

Basis: nil ∈ List

Recursive step:
if L ∈ List and a ∈ ℤ,
then a :: L ∈ List

How to prove $\forall\, x \in S, P(x)$ is true:

Base Case: Show that $P(u)$ is true for all specific elements u of S mentioned in the *Basis step*

Inductive Hypothesis: Assume that P is true for some arbitrary values of *each* of the existing named elements mentioned in the *Recursive step*

Inductive Step: Prove that $P(w)$ holds for each of the new elements w constructed in the *Recursive step* using the named elements mentioned in the Inductive Hypothesis

Conclude that $\forall\, x \in S, P(x)$

Claim: $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $L \in$ **List**

Claim: $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $L \in$ **List**

Let $P(L)$ **be** "$\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$".

We prove $P(L)$ **for all** $L \in$ **List by structural induction.**

Claim: $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $L \in$ **List**

Let P(L) **be** "$\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$".
We prove P(L) **for all** $L \in$ **List by structural induction.**

Base Case (nil):

Length:

$\text{len}(\text{nil}) := 0$
$\text{len}(a :: L) := \text{len}(L) + 1$

Concatenation:

$\text{concat}(\text{nil}, R) := R$
$\text{concat}(a :: L, R) := a :: \text{concat}(L, R)$

Claim: $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $L \in$ **List**

Let P(L) **be** "$\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$" .
We prove P(L) **for all** $L \in$ **List by structural induction.**

Base Case (nil):

$$\begin{aligned} \text{len}(\text{concat}(\text{nil}, R)) &= \text{len}(R) && \textbf{def of concat} \\ &= 0 + \text{len}(R) \\ &= \text{len}(\text{nil}) + \text{len}(R) && \textbf{def of len} \end{aligned}$$

Claim: $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $L \in$ **List**

Let P(L) **be** "$\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$".
We prove P(L) **for all** $L \in$ **List by structural induction.**

Base Case (nil): **We have** $\text{len}(\text{concat}(\text{nil}, R)) = \text{len}(R) = 0 + \text{len}(R) = \text{len}(\text{nil}) + \text{len}(R)$, **showing** P(nil).

Inductive Hypothesis: Assume that P(L) **is true for some arbitrary** $L \in$ List, **i.e.,** $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$.

Claim: $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $L \in$ **List**

Let P(L) **be** "$\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$".
We prove P(L) **for all** $L \in$ **List by structural induction.**

Base Case (nil): **We have** $\text{len}(\text{concat}(\text{nil}, R)) = \text{len}(R) = 0 + \text{len}(R) = \text{len}(\text{nil}) + \text{len}(R)$, **showing** P(nil).

Inductive Hypothesis: Assume that P(L) **is true for some arbitrary** $L \in \text{List}$, **i.e.,** $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$.

Inductive Step: **Goal: Show that** P(a :: L) **is true**

Claim: $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $L \in$ **List**

Let P(L) **be** "$\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$".
We prove P(L) **for all** $L \in$ **List by structural induction.**

Base Case (nil): **We have** $\text{len}(\text{concat}(\text{nil}, R)) = \text{len}(R) = 0 + \text{len}(R) = \text{len}(\text{nil}) + \text{len}(R)$, **showing** P(nil).

Inductive Hypothesis: Assume that P(L) **is true for some arbitrary** $L \in$ List, **i.e.,** $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$.

Inductive Step: **Goal: Show that** P(a :: L) **is true**

Length:

$\text{len}(\text{nil}) := 0$
$\text{len}(a :: L) := \text{len}(L) + 1$

Concatenation:

$\text{concat}(\text{nil}, R) := R$
$\text{concat}(a :: L, R) := a :: \text{concat}(L, R)$

Claim: $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $L \in$ **List**

Let P(L) **be** "$\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$".
We prove P(L) **for all** $L \in$ **List by structural induction.**

Base Case (nil): **We have** $\text{len}(\text{concat}(\text{nil}, R)) = \text{len}(R) = 0 + \text{len}(R) = \text{len}(\text{nil}) + \text{len}(R)$, **showing** P(nil).

Inductive Hypothesis: Assume that P(L) **is true for some arbitrary** $L \in \text{List}$, **i.e.,** $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$.

Inductive Step: **Goal: Show that** P(a :: L) **is true**

We can calculate

$$\begin{aligned}
\text{len}(\text{concat}(a :: L, R)) &= \text{len}(a :: \text{concat}(L, R)) && \textbf{def of } \text{concat} \\
&= 1 + \text{len}(\text{concat}(L, R)) && \textbf{def of } \text{len} \\
&= 1 + \text{len}(L) + \text{len}(R) && \textbf{IH} \\
&= \text{len}(a :: L) + \text{len}(R) && \textbf{def of } \text{len}
\end{aligned}$$

which is P(a :: L).

Claim: $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $L \in$ **List**

Let P(L) **be** "$\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$".
We prove P(L) **for all** $L \in$ **List by structural induction.**

Base Case (nil): **We have** $\text{len}(\text{concat}(\text{nil}, R)) = \text{len}(R) = 0 + \text{len}(R) = \text{len}(\text{nil}) + \text{len}(R)$, **showing** P(nil).

Inductive Hypothesis: Assume that P(L) **is true for some arbitrary** $L \in \text{List}$, **i.e.,** $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$.

Inductive Step: **Goal: Show that** P(a :: L) **is true**

We can calculate

$$\begin{aligned}
\text{len}(\text{concat}(a :: L, R)) &= \text{len}(a :: \text{concat}(L, R)) && \textbf{def of } \text{concat} \\
&= 1 + \text{len}(\text{concat}(L, R)) && \textbf{def of } \text{len} \\
&= 1 + \text{len}(L) + \text{len}(R) && \textbf{IH} \\
&= \text{len}(a :: L) + \text{len}(R) && \textbf{def of } \text{len}
\end{aligned}$$

which is P(a :: L).

By induction, we have shown the claim holds for all $L \in$ **List**.

Claim: $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $L, R \in$ **List**

Claim: $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $L, R \in$ **List**

Let $P(L)$ **be** "$\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $R \in$ **List**".
We prove $P(L)$ **for all** $L \in$ **List by structural induction.**

Claim: $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $L, R \in$ **List**

Let $P(L)$ **be** "$\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $R \in$ **List**".
We prove $P(L)$ **for all** $L \in$ **List by structural induction.**

Base Case (nil): **Let** $R \in$ **List be arbitrary. Then,**

Claim: $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $L, R \in$ **List**

Let $P(L)$ **be** "$\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $R \in$ **List**".
We prove $P(L)$ **for all** $L \in$ **List by structural induction.**

Base Case (nil): **Let** $R \in$ **List be arbitrary. Then,**

$$\begin{aligned} \text{len}(\text{concat}(\text{nil}, R)) &= \text{len}(R) && \textbf{def of concat} \\ &= 0 + \text{len}(R) \\ &= \text{len}(\text{nil}) + \text{len}(R) && \textbf{def of len} \end{aligned}$$

Since R **was arbitrary,** $P(\text{nil})$ **holds.**

Claim: $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $L, R \in$ **List**

Let $P(L)$ **be** "$\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $R \in$ **List**".
We prove $P(L)$ **for all** $L \in$ **List by structural induction.**

Base Case (nil): **Let** $R \in$ **List be arbitrary. Then,** $\text{len}(\text{concat}(\text{nil}, R))$ $= \text{len}(R) = 0 + \text{len}(R) = \text{len}(\text{nil}) + \text{len}(R)$, **showing** $P(\text{nil})$.

Inductive Hypothesis: Assume that $P(L)$ **is true for some arbitrary** $L \in \text{List}$, **i.e.,** $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $R \in$ **List**.

Claim: $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $L, R \in$ **List**

Let $P(L)$ **be** "$\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $R \in$ **List**".
We prove $P(L)$ **for all** $L \in$ **List by structural induction.**

Base Case (nil): **Let** $R \in$ **List be arbitrary. Then,** $\text{len}(\text{concat}(\text{nil}, R))$ $= \text{len}(R) = 0 + \text{len}(R) = \text{len}(\text{nil}) + \text{len}(R)$, **showing** $P(\text{nil})$.

Inductive Hypothesis: Assume that $P(L)$ **is true for some arbitrary** $L \in \text{List}$, **i.e.,** $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $R \in$ **List**.

Inductive Step: **Goal: Show that** $P(a :: L)$ **is true**

Claim: $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $L, R \in$ **List**

Let P(L) **be** "$\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $R \in$ **List**".
We prove P(L) **for all** $L \in$ **List by structural induction.**

Base Case (nil): **Let** $R \in$ **List be arbitrary. Then,** $\text{len}(\text{concat}(\text{nil}, R))$ $= \text{len}(R) = 0 + \text{len}(R) = \text{len}(\text{nil}) + \text{len}(R)$, **showing** P(nil).

Inductive Hypothesis: Assume that P(L) **is true for some arbitrary** $L \in$ List, **i.e.,** $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $R \in$ **List.**

Inductive Step: **Goal: Show that** P(a :: L) **is true**

Let $R \in$ **List be arbitrary. Then,**

Claim: $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $L, R \in$ **List**

Let $P(L)$ **be** "$\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $R \in$ **List**".
We prove $P(L)$ **for all** $L \in$ **List by structural induction.**

Base Case (nil): **Let** $R \in$ **List be arbitrary. Then,** $\text{len}(\text{concat}(\text{nil}, R)) = \text{len}(R) = 0 + \text{len}(R) = \text{len}(\text{nil}) + \text{len}(R)$, **showing** $P(\text{nil})$.

Inductive Hypothesis: Assume that $P(L)$ **is true for some arbitrary** $L \in \text{List}$, **i.e.,** $\text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R)$ **for all** $R \in$ **List**.

Inductive Step: **Goal: Show that** $P(a :: L)$ **is true**

Let $R \in$ **List be arbitrary. Then, we can calculate**

$$\begin{aligned}
\text{len}(\text{concat}(a :: L, R)) &= \text{len}(a :: \text{concat}(L, R)) && \textbf{def of } \text{concat} \\
&= 1 + \text{len}(\text{concat}(L, R)) && \textbf{def of } \text{len} \\
&= 1 + \text{len}(L) + \text{len}(R) && \textbf{IH} \\
&= \text{len}(a :: L) + \text{len}(R) && \textbf{def of } \text{len}
\end{aligned}$$

Since R **was arbitrary, we have shown** $P(a :: L)$.

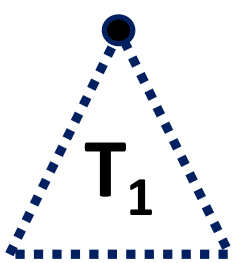
By induction, we have shown the claim holds for all $L \in$ **List**.

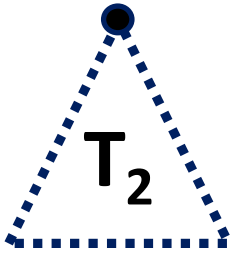
Rooted Binary Trees

- **Basis:**
  - is a rooted binary tree

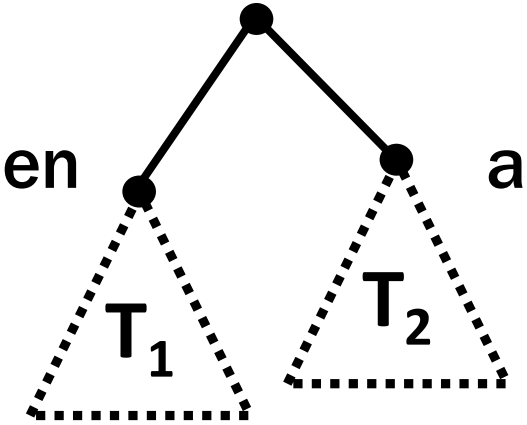
Rooted Binary Trees

- **Basis:** • is a rooted binary tree
- **Recursive step:**

If 

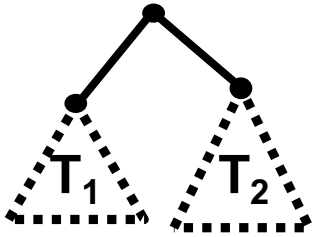
and 

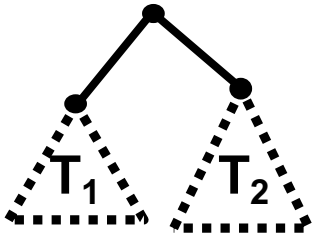
are rooted binary trees,

then 

also is a rooted binary tree.

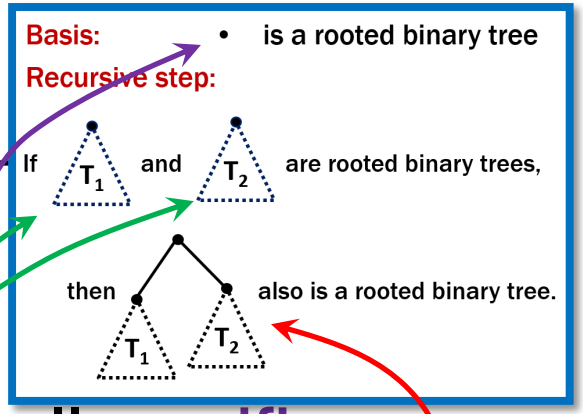
Defining Functions on Rooted Binary Trees

- size(•) ::= 1
- size 

::= 1 + size(T_1) + size(T_2)
- height(•) ::= 0
- height 

::= 1 + max{height(T_1), height(T_2)}

Last time: Structural Induction



How to prove $\forall\, x \in S, P(x)$ **is true:**

Base Case: Show that $P(u)$ is true for all specific elements u of S mentioned in the *Basis step*

Inductive Hypothesis: Assume that P is true for some arbitrary values of *each* of the existing named elements mentioned in the *Recursive step*

Inductive Step: Prove that $P(w)$ holds for each of the new elements w constructed in the *Recursive step* using the named elements mentioned in the Inductive Hypothesis

Conclude that $\forall\, x \in S, P(x)$

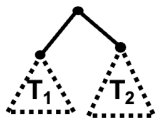
Claim: For every rooted binary tree **T**, $\text{size}(\mathbf{T}) \leq 2^{\text{height}(\mathbf{T}) + 1} - 1$

Claim: For every rooted binary tree **T**, $\text{size}(\mathbf{T}) \leq 2^{\text{height}(\mathbf{T})+1} - 1$

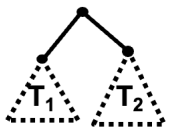
---

1. Let $P(\mathbf{T})$ be "$\text{size}(T) \leq 2^{\text{height}(T)+1}-1$". We prove $P(\mathbf{T})$ for all rooted binary trees **T** by structural induction.

size(•) ::= 1

size () ::= 1 + size($\mathbf{T_1}$) + size($\mathbf{T_2}$)

height(•) ::= 0

height () ::= 1 + max{height($\mathbf{T_1}$), height($\mathbf{T_2}$)}

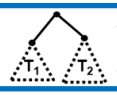
Claim: **For every rooted binary tree T,** $\text{size}(T) \leq 2^{\text{height}(T)+1} - 1$

---

1. **Let** $P(T)$ **be** "$\text{size}(T) \leq 2^{\text{height}(T)+1}-1$". **We prove** $P(T)$ **for all rooted binary trees T by structural induction.**
2. **Base Case:** $\text{size}(\bullet)=1$, $\text{height}(\bullet)=0$, **and** $2^{0+1}-1=2^1-1=1$ **so** $P(\bullet)$ **is true.**

Claim: For every rooted binary tree T, $\text{size}(T) \leq 2^{\text{height}(T)+1} - 1$

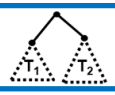
---

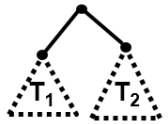
1. Let P(T) be "$\text{size}(T) \leq 2^{\text{height}(T)+1}-1$". We prove P(T) for all rooted binary trees T by structural induction.
2. Base Case: $\text{size}(\bullet)=1$, $\text{height}(\bullet)=0$, and $2^{0+1}-1=2^1-1=1$ so P(•) is true.
3. Inductive Hypothesis: Suppose that P(T_1) and P(T_2) are true for some rooted binary trees T_1 and T_2, i.e., $\text{size}(T_k) \leq 2^{\text{height}(T_k)+1} - 1$ for k=1,2
4. Inductive Step: Goal: Prove P().

Claim: For every rooted binary tree T, $size(T) \leq 2^{height(T)+1} - 1$

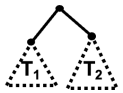
---

1. Let P(T) be "$size(T) \leq 2^{height(T)+1}-1$". We prove P(T) for all rooted binary trees T by structural induction.
2. Base Case: $size(\bullet)=1$, $height(\bullet)=0$, and $2^{0+1}-1=2^1-1=1$ so P(•) is true.
3. Inductive Hypothesis: Suppose that $P(T_1)$ and $P(T_2)$ are true for some rooted binary trees T_1 and T_2, i.e., $size(T_k) \leq 2^{height(T_k)+1} - 1$ for k=1,2
4. Inductive Step:

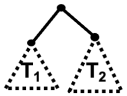
Goal: Prove P().

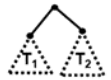
size()

size(•) ::= 1

size () ::= 1 + size(T_1) + size(T_2)

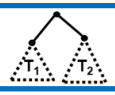
height(•) ::= 0

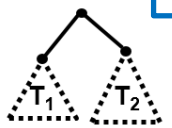
height () ::= 1 + max{height(T_1), height(T_2)}

$\leq 2^{height(}$  $)+1 - 1$

Claim: For every rooted binary tree T, $\text{size}(T) \le 2^{\text{height}(T)+1} - 1$

---

1. Let P(T) be "$\text{size}(T) \le 2^{\text{height}(T)+1}-1$". We prove P(T) for all rooted binary trees T by structural induction.
2. Base Case: $\text{size}(\bullet)=1$, $\text{height}(\bullet)=0$, and $2^{0+1}-1=2^1-1=1$ so P(•) is true.
3. Inductive Hypothesis: Suppose that P(T_1) and P(T_2) are true for some rooted binary trees T_1 and T_2, i.e., $\text{size}(T_k) \le 2^{\text{height}(T_k)+1} - 1$ for k=1,2
4. Inductive Step: **Goal: Prove P().**

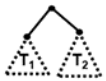
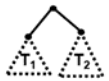
By def, size() $= 1+\text{size}(T_1)+\text{size}(T_2)$

$\le 1+2^{\text{height}(T_1)+1}-1+2^{\text{height}(T_2)+1}-1$

by IH for T_1 and T_2

$\le 2^{\text{height}(T_1)+1}+2^{\text{height}(T_2)+1}-1$

$\le 2(2^{\max(\text{height}(T_1),\text{height}(T_2))+1})-1$

$\le 2(2^{\text{height}($  $)}) - 1 \le 2^{\text{height}($  $)+1} - 1$

which is what we wanted to show.

5. So, the P(T) is true for all rooted binary trees by structural induction.

Strings

- An *alphabet* Σ is any finite set of characters

- The set Σ^* of *strings* over the alphabet Σ
  - example: {0,1}* is the set of *binary strings*

    0, 1, 00, 01, 10, 11, 000, 001, ... and “”

- Σ^* is defined recursively by
  - **Basis:** $\varepsilon \in \Sigma^*$ (ε is the empty string, i.e., “”)
  - **Recursive:** if $w \in \Sigma^*$, $a \in \Sigma$, then $wa \in \Sigma^*$

Last time: Structural Induction

Basis: $\varepsilon \in \Sigma^*$

Recursive Steps:

if $w \in \Sigma^*$ and $a \in \Sigma$, then $wa \in \Sigma^*$

How to prove $\forall\, x \in S, P(x)$ is true:

Base Case: Show that $P(u)$ is true for all specific elements u of S mentioned in the ***Basis step***

Inductive Hypothesis: Assume that P is true for some arbitrary values of *each* of the existing named elements mentioned in the ***Recursive step***

Inductive Step: Prove that $P(w)$ holds for each of the new elements w constructed in the ***Recursive step*** using the named elements mentioned in the Inductive Hypothesis

Conclude that $\forall\, x \in S, P(x)$

Functions on Recursively Defined Sets (on Σ^*)

Length:

$\text{len}(\varepsilon) ::= 0$

$\text{len}(wa) ::= \text{len}(w) + 1$ for $w \in \Sigma^*$, $a \in \Sigma$

Concatenation:

$x \bullet \varepsilon ::= x$ for $x \in \Sigma^*$

$x \bullet wa ::= (x \bullet w)a$ for $x \in \Sigma^*$, $a \in \Sigma$

Reversal:

$\varepsilon^R ::= \varepsilon$

$(wa)^R ::= a \bullet w^R$ for $w \in \Sigma^*$, $a \in \Sigma$

Number of c's in a string:

$\#_c(\varepsilon) ::= 0$

$\#_c(wc) ::= \#_c(w) + 1$ for $w \in \Sigma^*$

$\#_c(wa) ::= \#_c(w)$ for $w \in \Sigma^*$, $a \in \Sigma$, $a \neq c$

separate cases for
c vs $a \neq c$

Claim: $\text{len}(x \bullet y) = \text{len}(x) + \text{len}(y)$ **for all** $x,y \in \Sigma^*$

Let P(y) **be** "$\text{len}(x \bullet y) = \text{len}(x) + \text{len}(y)$ for all $x \in \Sigma^*$" .
We prove P(y) **for all** $y \in \Sigma^*$ **by structural induction.**

Base Case ($y = \varepsilon$): **Let** $x \in \Sigma^*$ **be arbitrary. Then,** $\text{len}(x \bullet \varepsilon) = \text{len}(x) = \text{len}(x) + \text{len}(\varepsilon)$ **since** $\text{len}(\varepsilon)=0$. **Since** x **was arbitrary,** P(ε) **holds.**

Inductive Hypothesis: Assume that P(w) **is true for some arbitrary** $w \in \Sigma^*$, **i.e.,** $\text{len}(x \bullet w) = \text{len}(x) + \text{len}(w)$ for all x

Claim: $\text{len}(x \bullet y) = \text{len}(x) + \text{len}(y)$ **for all** $x,y \in \Sigma^*$

Let $P(y)$ **be** "$\text{len}(x \bullet y) = \text{len}(x) + \text{len}(y)$ for all $x \in$
We prove $P(y)$ **for all** $y \in \Sigma^*$ **by structural indu**

Does this look familiar?

Base Case ($y = \varepsilon$): **Let** $x \in \Sigma^*$ **be arbitrary. Then,** $\text{len}(x \bullet \varepsilon) = \text{len}(x) = \text{len}(x) + \text{len}(\varepsilon)$ **since** $\text{len}(\varepsilon)=0$. **Since** x **was arbitrary,** $P(\varepsilon)$ **holds.**

Inductive Hypothesis: Assume that $P(w)$ **is true for some arbitrary** $w \in \Sigma^*$, **i.e.,** $\text{len}(x \bullet w) = \text{len}(x) + \text{len}(w)$ for all x

Inductive Step: **Goal: Show that** $P(wa)$ **is true for every** $a \in \Sigma$

Let $a \in \Sigma$ **and** $x \in \Sigma^*$. **Then**

$$\begin{aligned} \text{len}(x \bullet wa) &= \text{len}((x \bullet w)a) && \textbf{by def of } \bullet \\ &= \text{len}(x \bullet w)+1 && \textbf{by def of len} \\ &= \text{len}(x)+\text{len}(w)+1 && \textbf{by I.H.} \\ &= \text{len}(x)+\text{len}(wa) && \textbf{by def of len} \end{aligned}$$

Therefore, $\text{len}(x \bullet wa)= \text{len}(x)+\text{len}(wa)$ **for all** $x \in \Sigma^*$, **so** $P(wa)$ **is true.**

So, by induction $\text{len}(x \bullet y) = \text{len}(x) + \text{len}(y)$ **for all** $x,y \in \Sigma^*$

Claim: $\text{len}(x^R) = \text{len}(x)$ **for all** $x \in \Sigma^*$

Let $P(x)$ **be** "$\text{len}(x^R) = \text{len}(x)$".

We will prove $P(x)$ **for all** $x \in \Sigma^*$ **by structural induction.**

Length:

$\text{len}(\varepsilon) ::= 0$

$\text{len}(wa) ::= \text{len}(w) + 1$ for $w \in \Sigma^*$, $a \in \Sigma$

Reversal:

$\varepsilon^R ::= \varepsilon$

$(wa)^R ::= a \bullet w^R$ for $w \in \Sigma^*$, $a \in \Sigma$

Claim: $\text{len}(x^R) = \text{len}(x)$ **for all** $x \in \Sigma^*$

Let $P(x)$ **be** "$\text{len}(x^R) = \text{len}(x)$".

We will prove $P(x)$ **for all** $x \in \Sigma^*$ **by structural induction.**

Base Case ($x = \varepsilon$): **Then,** $\text{len}(\varepsilon^R) = \text{len}(\varepsilon)$ **by def of string reverse.**

Claim: $\text{len}(x^R) = \text{len}(x)$ **for all** $x \in \Sigma^*$

Let P(x) **be** "$\text{len}(x^R) = \text{len}(x)$".

We will prove P(x) **for all** $x \in \Sigma^*$ **by structural induction.**

Base Case ($x = \varepsilon$): **Then,** $\text{len}(\varepsilon^R) = \text{len}(\varepsilon)$ **by def of string reverse.**

Inductive Hypothesis: Assume that P(w) **is true for some arbitrary** $w \in \Sigma^*$, **i.e.,** $\text{len}(w^R) = \text{len}(w)$.

Inductive Step: **Goal: Show that** $\text{len}((wa)^R) = \text{len}(wa)$ **for every** a

Length:

$\text{len}(\varepsilon) ::= 0$

$\text{len}(wa) ::= \text{len}(w) + 1$ for $w \in \Sigma^*$, $a \in \Sigma$

Reversal:

$\varepsilon^R ::= \varepsilon$

$(wa)^R ::= a \bullet w^R$ for $w \in \Sigma^*$, $a \in \Sigma$

Claim: $\text{len}(x^R) = \text{len}(x)$ **for all** $x \in \Sigma^*$

Let P(x) **be** "$\text{len}(x^R) = \text{len}(x)$".

We will prove P(x) **for all** $x \in \Sigma^*$ **by structural induction.**

Base Case ($x = \varepsilon$): **Then,** $\text{len}(\varepsilon^R) = \text{len}(\varepsilon)$ **by def of string reverse.**

Inductive Hypothesis: Assume that P(w) **is true for some arbitrary** $w \in \Sigma^*$, **i.e.,** $\text{len}(w^R) = \text{len}(w)$.

Inductive Step: **Goal: Show that** $\text{len}((wa)^R) = \text{len}(wa)$ **for every** a

Let $a \in \Sigma$. **Then,**

$\text{len}((wa)^R) = \text{len}(a \bullet w^R)$	**def of reverse**
$= \text{len}(a) + \text{len}(w^R)$	**by previous result**
$= \text{len}(a) + \text{len}(w)$	**IH**
$= \text{len}(1) + \text{len}(w)$	**def of len**
$= \text{len}(wa)$	**def of len**

Therefore, $\text{len}((wa)^R) = \text{len}(wa)$, **so** P(wa) **is true for every** $a \in \Sigma$.

So, we have shown $\text{len}(x^R) = \text{len}(x)$ **for all** $x \in \Sigma^*$ **by induction.**

More Theorems

Structural induction is the tool used to prove many more interesting theorems

- General associativity follows from our one rule
  - likewise for generalized De Morgan's laws

- Okay to substitute y for x everywhere in a modular equation when we know that $x \equiv_m y$

- More coming shortly…