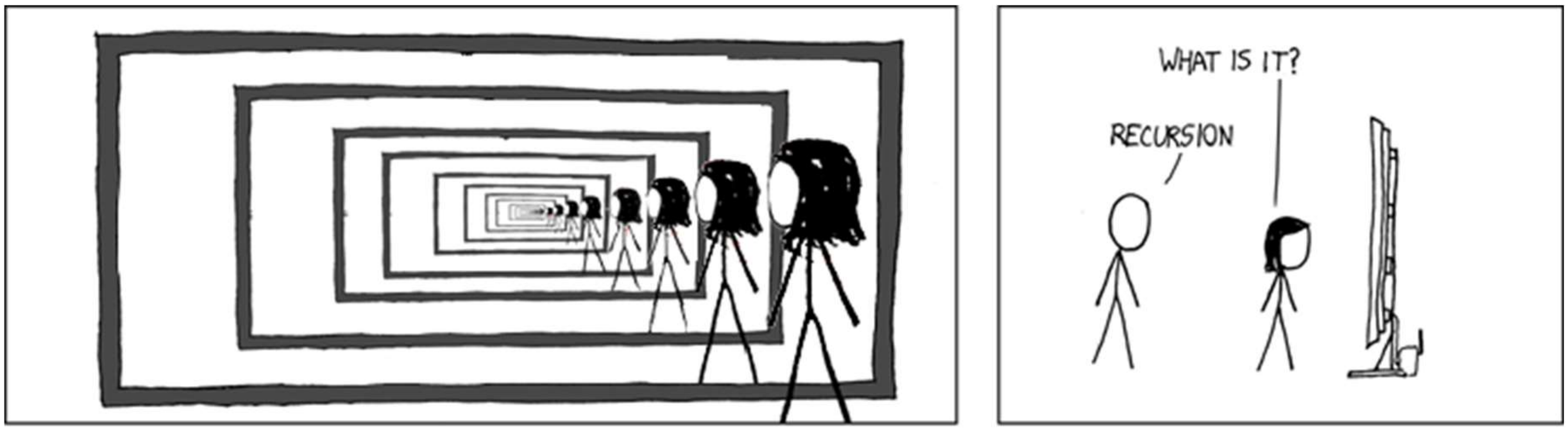


Structural Induction

CSE 311: Foundations of Computing I
Lecture 14

Announcements

- HW4 due tonight at 11:59 pm. Turn it in with no late days to receive feedback by tomorrow for induction

Find the Bug

Find the Bug

Claim: For every odd integer n, $n^2 \equiv_4 1$.

Proof: Let n be an arbitrary odd integer. Then by definition of odd, $n = 2k + 1$ for some integer k. Then consider $n^2 \equiv_4 1$. Plugging in $n = 2k + 1$ for n^2:

$$n^2 \equiv_4 1$$
$$(2k+1)^2 \equiv_4 1$$
$$4k^2 + 4k + 1 \equiv_4 1$$

Then by definition of congruence, $4 \mid 4k^2 + 4k + 1 - 1$, so $4 \mid 4k^2 + 4k$. Since this is true, the claim holds.

Find the Bug

Claim: For every odd integer n, $n^2 \equiv_4 1$.

Proof: Let n be an arbitrary odd integer. Then by definition of odd, $n = 2k + 1$ for some integer k. Then consider $n^2 \equiv_4 1$. Plugging in $n = 2k + 1$ for n^2:

$$n^2 \equiv_4 1$$
$$(2k+1)^2 \equiv_4 1$$
$$4k^2 + 4k + 1 \equiv_4 1$$

Backwards Reasoning: **Assumes** the statement we're trying to prove is true.

Then by definition of congruence, $4 \mid 4k^2 + 4k + 1 - 1$, so $4 \mid 4k^2 + 4k$. Since this is true, the claim holds.

Fixed Proof

Claim: For every odd integer n, $n^2 \equiv_4 1$.

Proof: Let n be an arbitrary odd integer. Then by definition of odd, $n = 2k + 1$ for some integer k. Then consider n^2:

$$n^2 =$$

Since k is an integer, $k^2 + k$ is an integer. So by definition of divides, $4 \mid n^2 - 1$. So by definition of congruence, $n^2 \equiv_4 1$. Since n was arbitrary, the claim holds.

Fixed Proof

Claim: For every odd integer n, $n^2 \equiv_4 1$.

Proof: Let n be an arbitrary odd integer. Then by definition of odd, $n = 2k + 1$ for some integer k. Then consider n^2:

$$n^2 = (2k+1)^2 = 4k^2 + 4k + 1$$
$$n^2 - 1 = 4k^2 + 4k$$
$$n^2 - 1 = 4(k^2 + k)$$

Since k is an integer, $k^2 + k$ is an integer. So by definition of divides, $4 \mid n^2 - 1$. So by definition of congruence, $n^2 \equiv_4 1$. Since n was arbitrary, the claim holds.

Backwards Reasoning

Backwards reasoning is the incorrect proof technique of *assuming* the goal is true, and then deriving some other true statement.

This reasoning can be used to incorrectly prove false statements.

Claim: For all integer x, if $x^2 = 25$, then $x = 5$.

Backwards Proof: Let x be an arbitrary integer. Suppose $x^2 = 25$. Plugging in $x = 5$, we have $5^2 = 25$. Since this is true, the claim holds.

False! What if $x = -5$?

Find the 4 Bugs

Claim: For all integers $n \geq 1$, $1 + \cdots + n = \frac{n(n+1)}{2}$.

Proof: Let $\mathrm{P}(n)$ be "$1 + \cdots + n = \frac{n(n+1)}{2}$ for all integers $n \geq 1$". We prove by induction.

Base Case: Plugging in $n = 1$, we have $1 = \frac{1(1+1)}{2}$. So $1 = \frac{2}{2}$. So $1 = 1$. Since this is true, the base case holds.

IH: Suppose $1 + \cdots + k = \frac{k(k+1)}{2}$ for an arbitrary integer k.

IS: We aim to show $\mathrm{P}(k+1)$. Observe that:

$$1 + \cdots + (k+1) = 1 + \cdots + k + (k+1) = \frac{k(k+1)}{2} + (k+1) = \frac{k(k+1)+2(k+1)}{2} = \frac{(k+1)(k+2)}{2}$$

So $\mathrm{P}(k+1)$ holds.

Conclusion: Thus $P(n)$ holds for all integers $n \geq 1$ by induction.

Find the 4 Bugs

Claim: For all integers $n \geq 1$, $1 + \cdots + n = \frac{n(n+1)}{2}$.

Proof: Let $\mathrm{P}(n)$ be "$1 + \cdots + n = \frac{n(n+1)}{2}$ for all integers $n \geq 1$". We prove by induction.

Base Case: Plugging in $n = 1$, we have $1 = \frac{1(1+1)}{2}$. So $1 = \frac{2}{2}$. So $1 = 1$. Since this is true, the base case holds.

IH: Suppose $1 + \cdots + k = \frac{k(k+1)}{2}$ for an arbitrary integer k.

IS: We aim to show $\mathrm{P}(k+1)$. Observe that:

$$1 + \cdots + (k+1) = 1 + \cdots + k + (k+1) = \frac{k(k+1)}{2} + (k+1) = \frac{k(k+1)+2(k+1)}{2} = \frac{(k+1)(k+2)}{2}$$

So $\mathrm{P}(k+1)$ holds.

Conclusion: Thus $P(n)$ holds for all integers $n \geq 1$ by induction.

Definition of $\mathrm{P}(n)$:
Including the "for all n" **inside** the definition of P.

Backwards Reasoning:
Assumes the base case holds

Should be $k \geq 1$

Didn't cite where we used the IH

Avoiding Backwards Reasoning in the Base Case

Incorrect Technique: Backwards reasoning

Plugging in $n = 1$, we have $1 = \frac{1(1+1)}{2}$. So $1 = \frac{2}{2}$. So $1 = 1$. Since this is true, the base case holds.

Valid Technique 1: Separating LHS and RHS

The LHS evaluates to 1. The RHS evaluates to $\frac{1(1+1)}{2} = \frac{2}{2} = 1$. Since $1 = 1$, the base case holds.

Valid Technique 2: Start from Left, convert to the Right

Observe that $1 = \frac{2}{2} = \frac{1 \cdot 2}{2} = \frac{1(1+1)}{2}$. So the base case holds.

Induction Big Picture

Weak and Strong Induction: Prove statements over the natural numbers.

“Prove that P(n) holds for all natural numbers n.”

Structural Induction: In CS, we deal with Strings, Lists, Trees, and other objects. Now we prove statements about these objects.

“Prove that P(T) holds for all trees T.”

“Prove that P(x) holds for all strings x.”

Recursively Defined Sets

Recursively Defined Sets

- In order to prove a fact about all trees or all lists, we need rigorous mathematical definitions for these sets.
- We will define these sets *recursively*. A recursively defined set has 3 components:
  - Basis Step
  - Recursive Step
  - Exclusion Rule

Recursively Defined Sets

For example, define a set S as follows:

Basis Step: $0 \in S$

Recursive Step: If $x \in S$ then $x + 2 \in S$.

Exclusion Rule: Every element of S follows from the basis step or a finite number of recursive steps.

What is S? The set of all non-negative even integers. {0, 2, 4,…}

Why do we need the exclusion rule? To clarify that there aren't any *other* elements in the set. In practice this isn't usually written.

Recursively Defined Sets

Natural Numbers ($\mathbb{N}$)

Integers ($\mathbb{Z}$)

Integer coordinates in the line $y = x$

Recursively Defined Sets

Natural Numbers ($\mathbb{N}$)

Basis Step: $0 \in S$

Recursive Step: If $x \in S$ then $x + 1 \in S$.

Integers ($\mathbb{Z}$)

Basis Step: $0 \in S$

Recursive Step: If $x \in S$ then $x + 1 \in S$ and $x - 1 \in S$.

Integer coordinates in the line $y = x$

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

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

Recursively Defined Sets

Q1: Write a recursive definition for the set of positive even integers

Basis Step:

Recursive Step:

Q2: Write a recursive definition for the set of powers of 3 $\{1,3,9,27,\ldots\}$

Basis Step:

Recursive Step:

Recursively Defined Sets

Q1: Write a recursive definition for the set of positive even integers

Basis Step: $2 \in S$

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

Q2: Write a recursive definition for the set of powers of 3 $\{1,3,9,27, \ldots\}$

Basis Step: $1 \in S$

Recursive Step: If $n \in S$, then $3n \in S$

Structural Induction

On Sets of Numbers

Claim about a Recursively Defined Set

Let S be the set defined:

Basis Step: $6 \in S, 15 \in S$

Recursive Step: if $x, y \in S$ then $x + y \in S$.

Claim: Every element of S is divisible by 3.

How would we prove this?

Structural Induction Idea

Basis: $6 \in S, 15 \in S$
Recursive: if $x, y \in S$ then $x + y \in S$.

To show $\mathrm{P}(s)$ for all $s \in S$…

Base Case: Show $\mathrm{P}(b)$ for all elements b in the basis step.

Inductive Hypothesis: Assume $\mathrm{P}()$ holds for arbitrary element(s) that we've already constructed.

Inductive Step: Prove that $\mathrm{P}()$ holds for a new element constructed using the recursive step.

Structural Induction Idea

Basis: $6 \in \mathrm{S}, 15 \in S$
Recursive: if $x, y \in S$ then $x + y \in S$.

To show $\mathrm{P}(s)$ for all $s \in S$…

- Here, $\mathrm{P}(s)$ is "$3 \mid s$".

Base Case: Show $\mathrm{P}(b)$ for all elements b in the basis step.

- Show $\mathrm{P}(6)$ and $\mathrm{P}(15)$ hold.

Inductive Hypothesis: Assume $\mathrm{P}()$ holds for arbitrary element(s) that we've already constructed.

- Assume $\mathrm{P}(x)$ and $\mathrm{P}(y)$ for arbitrary $x, y \in S$.

Inductive Step: Prove that $\mathrm{P}()$ holds for a new element constructed using the recursive step.

- Show $\mathrm{P}(x + y)$ holds.

Structural Induction

Basis: $6 \in S, 15 \in S$
Recursive: if $x, y \in S$ then $x + y \in S$.

1. Let $\mathrm{P}(s)$ be "s is divisible by 3". We show $P(s)$ holds for all $s \in S$ by structural induction.

2. Base Case(s): $6 = 2 \cdot 3$ so $3|6$, and $\mathrm{P}(6)$ holds. $15 = 5 \cdot 3$, so $3|15$ and $\mathrm{P}(15)$ holds.

3. Inductive Hypothesis: Suppose $\mathrm{P}(x)$ and $\mathrm{P}(y)$ for arbitrary $x, y \in S$.

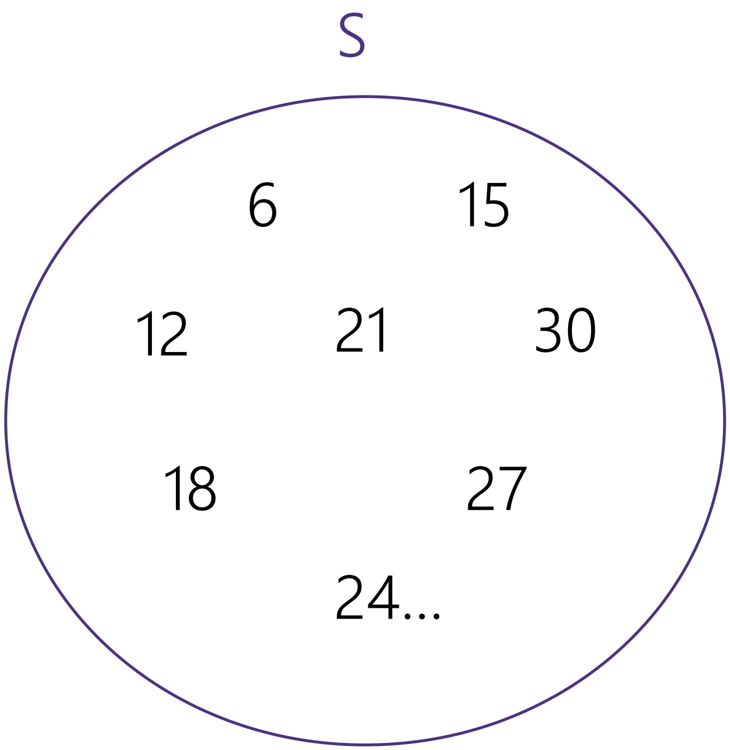
4. Inductive Step: Goal: $\mathrm{P}(x + y)$ holds

By IH $3 \mid x$ and $3 \mid y$. So by definition of divides, $x = 3n$ and $y = 3m$ for integers m, n.

Adding the equations: $x + y = 3(n + m)$. Since n, m are integers $n + m$ is an integer. Thus by definition of divides, $3 \mid (x + y)$. So $\mathrm{P}(x + y)$ holds.

5. Conclusion: Thus $\mathrm{P}(s)$ for all $s \in S$ by structural induction.

How does this work?



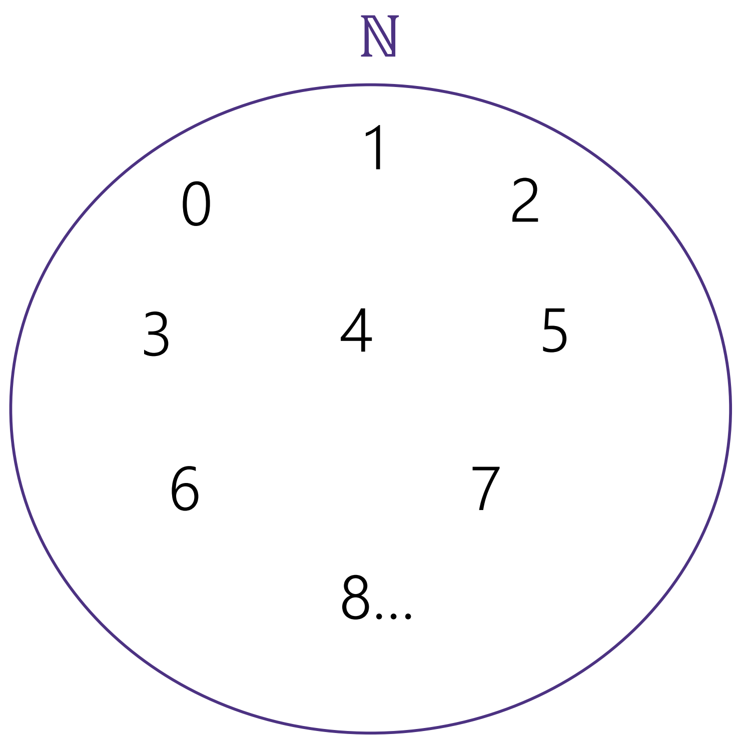
Basis: $6 \in S, 15 \in S$
Recursive: if $x, y \in S$ then $x + y \in S$.

We proved:

Base Case: P(6) and P(15)

IH → IS: If P(x) and P(y), then P(x+y)

Weak Induction is a special case of Structural



Basis: $\mathbf{0} \in \mathbb{N}$
Recursive: if $k \in \mathbb{N}$ then $k + \mathbf{1} \in \mathbb{N}$.

We proved:

Base Case: P(0)

IH → IS: If P(k), then P(k+1)

Wait a minute! Why can we do this?

Think of each element of S as requiring k “applications of a rule” to get in

$P(base\ cases)$ is true

$P(base\ cases) \rightarrow P(one\ application)$ so $P(one\ application)$

$P(one\ application) \rightarrow P(two\ applications)$ so $P(two\ applications)$

…

It’s the same principle as regular induction. You’re just inducting on “how many steps did we need to get this element?”

You’re still only assuming the IH about a domino you’ve knocked over.

Wait a minute! Why can we do this?

Imagine building S “step-by-step”

$$S_0 = \{6,15\}$$
$$S_1 = \{12,21,30\}$$
$$S_2 = \{18,24,27,36,42,45,60\}$$

IS can always of the form “suppose $P(x) \forall x \in (S_0 \cup \cdots \cup S_k)$” and show $P(y)$ for some $y \in S_{k+1}$

We use the structural induction phrasing assuming our reader knows how induction works and so don’t phrase it explicitly in this form.

Structural Induction Template

1. Define $\mathrm{P}()$. Claim that $\mathrm{P}(s)$ holds for all $s \in S$. State your proof is by structural induction.

2. Base Case: Show $\mathrm{P}(b_1), \dots, \mathrm{P}(b_n)$ holds for each basis step $b_1, \dots, b_n$ in S.

3. Inductive Hypothesis: Suppose $\mathrm{P}(x_1), \dots, \mathrm{P}(x_m)$ for all values listed in the recursive rules.

4. Inductive Step: Show $\mathrm{P}()$ holds for the “new element” given by the recursive step. You will need a separate step for every rule.

5. Conclusion: Conclude that $\mathrm{P}(s)$ holds for all $s \in S$ by structural induction.

Structural Induction

On Strings

String Terminology

Σ is the alphabet, i.e. the set of all letters you can use in strings.

For example: $\Sigma = \{0,1\}$ or $\Sigma = \{a, b, c, \ldots., z, \_\}$

Σ^* is the set of all strings you can build from the letters in the alphabet.

For example: If $\Sigma = \{0,1\}$ then $01001 \in \Sigma^*$. If $\Sigma = \{a, b, c, \ldots., z, \_\}$, then $\text{i\_love\_induction} \in \Sigma^*$

- ε is the empty string

Analogous to “” in Java

Recursive definition of Strings

Σ is the alphabet
Σ^* is the set of all strings
ε is the empty string

The set of all strings Σ^* can be defined recursively (using Σ, ε):

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

Recursive Step: If $w \in \Sigma^*$ and $a \in \Sigma$, then $wa \in \Sigma^*$

wa here means the string w with the character a appended on to it

Functions on Strings

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

Recursive: If $w \in \Sigma^*$ and $a \in \Sigma$, then $wa \in \Sigma^*$

To prove interesting facts about strings, we need functions on strings.

Length:

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

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

Reversal:

$\varepsilon^R = \varepsilon$

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

Claim about Strings

Claim: For any string $s \in \Sigma^*$, $\text{len}(s^R) = \text{len}(s)$

Proof

$\text{len}(\varepsilon) = 0$
$\text{len}(wa) = \text{len}(w) + 1$

$\varepsilon^R = \varepsilon$
$(wa)^R = aw^R$

Basis: $\varepsilon \in \Sigma^*$
Recursive: If $w \in \Sigma^*$ and $a \in \Sigma$, then $wa \in \Sigma^*$

1. Let $P(s)$ be   We prove $P(s)$ for all strings s by structural induction.

2. Base Case(s):

3. Inductive Hypothesis:

4. Inductive Step:

5. Conclusion:

Proof

$\text{len}(\varepsilon) = 0$
$\text{len}(wa) = \text{len}(w) + 1$

$\varepsilon^R = \varepsilon$
$(wa)^R = aw^R$

Basis: $\varepsilon \in \Sigma^*$
Recursive: If $w \in \Sigma^*$ and $a \in \Sigma$, then $wa \in \Sigma^*$

1. Let $P(s)$ be $\text{len}(s^R) = \text{len}(s)$. We prove $P(s)$ for all strings s by structural induction.

2. Base Case(s):

3. Inductive Hypothesis:

4. Inductive Step:

5. Conclusion:

Proof

$\text{len}(\varepsilon) = 0$
$\text{len}(wa) = \text{len}(w) + 1$

$\varepsilon^R = \varepsilon$
$(wa)^R = aw^R$

Basis: $\varepsilon \in \Sigma^*$
Recursive: If $w \in \Sigma^*$ and $a \in \Sigma$, then $wa \in \Sigma^*$

1. Let $P(s)$ be $\text{len}(s^R) = \text{len}(s)$. We prove $P(s)$ for all strings s by structural induction.

2. Base Case(s): $(s = \varepsilon)$. LHS: Since $\varepsilon^R = \varepsilon$, $\text{len}(\varepsilon^R) = \text{len}(\varepsilon) = 0$. RHS: $\text{len}(\varepsilon) = 0$. Since $0 = 0$, the base case holds.

3. Inductive Hypothesis:

4. Inductive Step:

5. Conclusion:

Proof

$\text{len}(\varepsilon) = 0$
$\text{len}(wa) = \text{len}(w) + 1$

$\varepsilon^R = \varepsilon$
$(wa)^R = aw^R$

Basis: $\varepsilon \in \Sigma^*$
Recursive: If $w \in \Sigma^*$ and $a \in \Sigma$, then $wa \in \Sigma^*$

1. Let $P(s)$ be $\text{len}(s^R) = \text{len}(s)$. We prove $P(s)$ for all strings s by structural induction.

2. Base Case(s): $(s = \varepsilon)$. LHS: Since $\varepsilon^R = \varepsilon$, $\text{len}(\varepsilon^R) = \text{len}(\varepsilon) = 0$. RHS: $\text{len}(\varepsilon) = 0$. Since $0 = 0$, the base case holds.

3. Inductive Hypothesis: Suppose $P(w)$ for some arbitrary string w. Then $\text{len}(w^R) = \text{len}(w)$

4. Inductive Step:

Goal: $\text{len}\big((wa)^R\big) = \text{len}(wa)$

5. Conclusion:

Proof

$\text{len}(\varepsilon) = 0$
$\text{len}(wa) = \text{len}(w) + 1$

$\varepsilon^R = \varepsilon$
$(wa)^R = aw^R$

Basis: $\varepsilon \in \Sigma^*$
Recursive: If $w \in \Sigma^*$ and $a \in \Sigma$, then $wa \in \Sigma^*$

1. Let $P(s)$ be $\text{len}(s^R) = \text{len}(s)$. We prove $P(s)$ for all strings s by structural induction.

2. Base Case(s): $(s = \varepsilon)$. LHS: Since $\varepsilon^R = \varepsilon$, $\text{len}(\varepsilon^R) = \text{len}(\varepsilon) = 0$. RHS: $\text{len}(\varepsilon) = 0$. Since $0 = 0$, the base case holds.

3. Inductive Hypothesis: Suppose $P(w)$ for some arbitrary string w. Then $\text{len}(w^R) = \text{len}(w)$

4. Inductive Step:

Goal: $\text{len}((wa)^R) = \text{len}(wa)$

Let a be an arbitrary character. Observe:

$\text{len}((wa)^R) = \text{len}(aw^R)$ By definition of reverse

5. Conclusion:

Proof

$\text{len}(\varepsilon) = 0$
$\text{len}(wa) = \text{len}(w) + 1$

$\varepsilon^R = \varepsilon$
$(wa)^R = aw^R$

Basis: $\varepsilon \in \Sigma^*$
Recursive: If $w \in \Sigma^*$ and $a \in \Sigma$, then $wa \in \Sigma^*$

1. Let $P(s)$ be $\text{len}(s^R) = \text{len}(s)$. We prove $P(s)$ for all strings s by structural induction.

2. Base Case(s): $(s = \varepsilon)$. LHS: Since $\varepsilon^R = \varepsilon$, $\text{len}(\varepsilon^R) = \text{len}(\varepsilon) = 0$. RHS: $\text{len}(\varepsilon) = 0$. Since $0 = 0$, the base case holds.

3. Inductive Hypothesis: Suppose $P(w)$ for some arbitrary string w. Then $\text{len}(w^R) = \text{len}(w)$

4. Inductive Step:

Goal: $\text{len}\big((wa)^R\big) = \text{len}(wa)$

Let a be an arbitrary character. Observe:

$\text{len}\big((wa)^R\big) = \text{len}(aw^R)$	By definition of reverse
$= \text{len}(w^R) + 1$	By definition of length
$= \text{len}(w) + 1$	By IH
$= \text{len}(wa)$	By definition of length

5. Conclusion: Thus $\text{P}(s)$ holds for all strings s by structural induction.

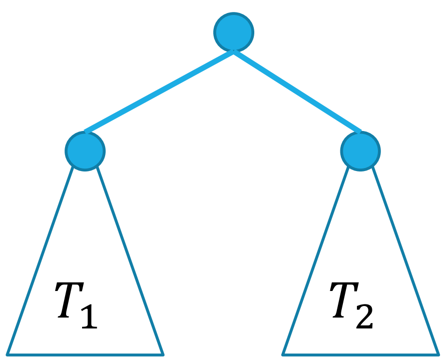
Trees!

More Structural Sets

Binary Trees are another common source of structural induction.

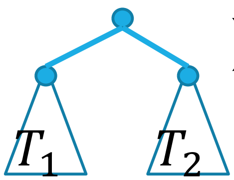
Basis: A single node is a rooted binary tree. 

Recursive Step: If T_1 and T_2 are rooted binary trees with roots r_1 and r_2, then a tree rooted at a new node, with children r_1, r_2 is a binary tree.

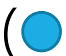


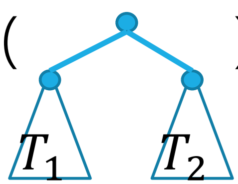
Functions on Binary Trees

size()=1

size(

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

height() = 0

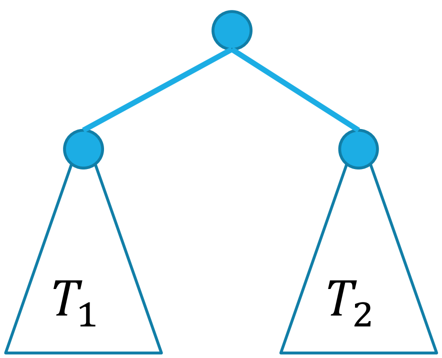
height(

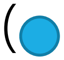
) = 1+$\max$(height(T_1),height(T_2))

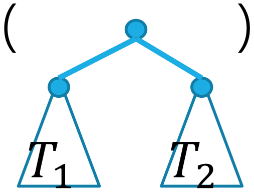
Binary Trees

Basis: A single node is a rooted binary tree.

Recursive Step: If T_1 and T_2 are rooted binary trees with roots r_1 and r_2, then a tree rooted at a new node, with children r_1, r_2 is a binary tree.

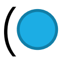


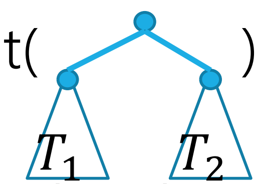
size()=1

size(

) =

size(T_1) + size(T_2) + 1

height() = 0

height(

) =

1+max(height(T_1),height(T_2))

Claim

We want to show that trees of a certain height can't have too many nodes. Specifically our claim is this:

For all trees T, size(T) $\leq 2^{height(T)+1} - 1$

Take a moment to absorb this formula, then we'll do induction!

Structural Induction on Binary Trees

Let $P(T)$ be “. We show $P(T)$ for all binary trees T by structural induction.

Base Case:

Inductive Hypothesis:

Structural Induction on Binary Trees

Let $P(T)$ be “size(T) $\leq 2^{height(T)+1} - 1$“. We show $P(T)$ for all binary trees T by structural induction.

Base Case: Let $T =$ . size(T)=1 and height(T) = 0, so size(T)=1$\leq 2 - 1 = 2^{0+1} - 1 = 2^{height(T)+1} - 1$.

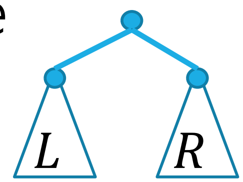
Inductive Hypothesis:

Structural Induction on Binary Trees

Let $P(T)$ be "size(T) $\le 2^{height(T)+1} - 1$". We show $P(T)$ for all binary trees T by structural induction.

Base Case: Let $T =$ . size(T)=1 and height(T) = 0, so size(T)=1$\le 2 - 1 = 2^{0+1} - 1 = 2^{height(T)+1} - 1$.

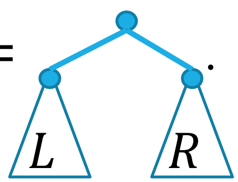
Inductive Hypothesis: Suppose $\mathrm{P}(L)$ and $\mathrm{P}(R)$ hold for arbitrary trees L, R. Let T be the tree



Inductive step: Figure out, (1) what we must show (2) a formula for height and a formula for size of T.

Structural Induction on Binary Trees (cont.)

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

$T =$  .

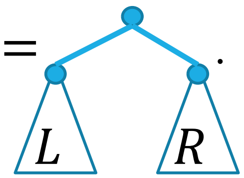
height(T)=$1 + \max\{height(L), height(R)\}$

size(T)= 1 +size(L)+size(R)

So $P(T)$ holds, and we have $P(T)$ for all binary trees T by the principle of induction.

Structural Induction on Binary Trees (cont.)

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

$T =$  .

height(T)=$1 + \max\{height(L), height(R)\}$

size(T)= 1 +size(L)+size(R)

size(T)=1+size(L)+size(R) $\leq 1 + 2^{height(L)+1} - 1 + 2^{height(R)+1} - 1$ (by IH)

$\leq 2^{height(L)+1} + 2^{height(R)+1} - 1$ (cancel 1's)

$\leq 2^{height(T)} + 2^{height(T)} - 1 = 2^{height(T)+1} - 1$ (T taller than subtrees)

So $P(T)$ holds, and we have $P(T)$ for all binary trees T by the principle of induction.