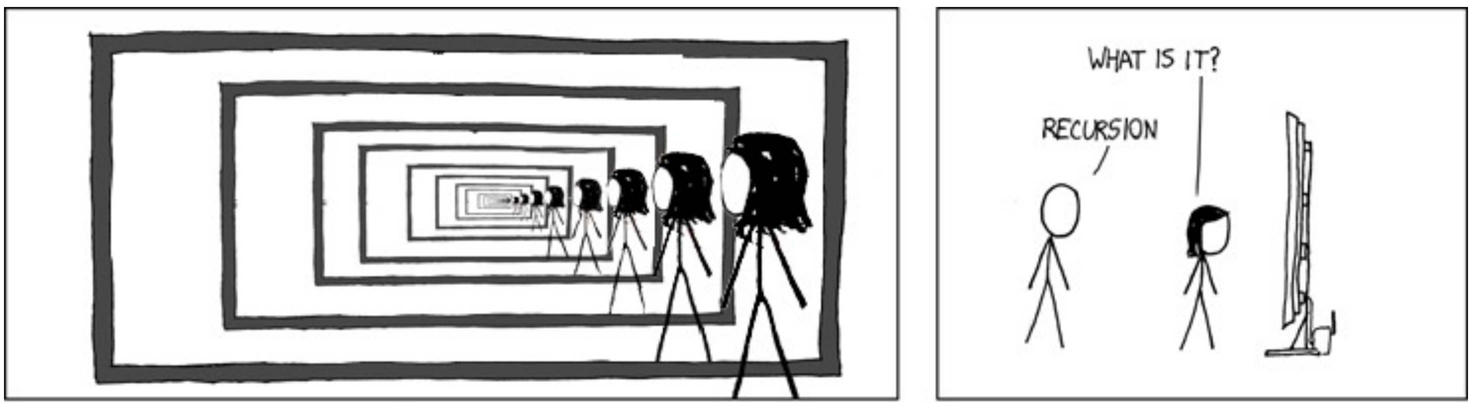
CSE 311: Foundations of Computing

Lecture 19: Recursively Defined Sets & Structural Induction



Administrivia

- Midterm in class Friday
- Midterm review video
  - see Panopto Recordings tab on Canvas
- Midterm review session
  - Thursday at 1:30 in ECE 125
  - come with questions
- HW6 released on Saturday — start early
  - 2 strong induction, 2 structural induction, 2 string problems

Last time: Recursive Definition of Sets

Recursive definition of set S

- **Basis Step:** $0 \in S$
- **Recursive Step: If** $x \in S$**, then** $x + 2 \in S$
- **Exclusion Rule: Every element in** S **follows from the basis step and a finite number of recursive steps.**

Can already build sets using $\{ x \mid P(x) \}$ notation

- these are *constructive* definitions
- translates more naturally into Java etc.

Last Time: Recursive Definitions

- Any recursively defined set can be translated into a Java class
- Any recursively defined function can be translated into a Java function
  - some (but not all) can be written more cleanly as loops
- Recursively defined functions and sets are our mathematical models of code and the data it operates on

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

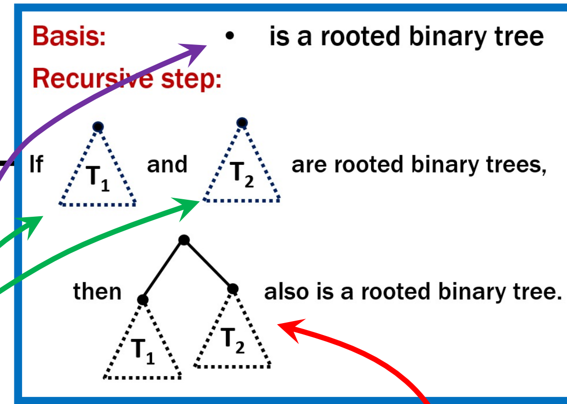
Ordinary induction is a special case of structural induction:

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: 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: Every element of S is divisible by 3.

1. Let P(x) be "3|x". We prove that P(x) is true for all x ∈ S by structural induction.
2. Base Case: 3|6 and 3|15 so P(6) and P(15) are true
3. Inductive Hypothesis: Suppose that P(x) and P(y) are true for some arbitrary x,y ∈ S
4. Inductive Step: **Goal: Show P(x+y)**

   Since P(x) is true, 3|x and so x=3m for some integer m and since P(y) is true, 3|y and so y=3n for some integer n.
   Therefore x+y=3m+3n=3(m+n) and thus 3|(x+y).

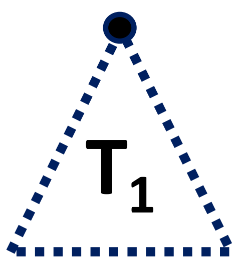
   Hence P(x+y) is true.
5. Therefore by induction 3|x for all x ∈ S.

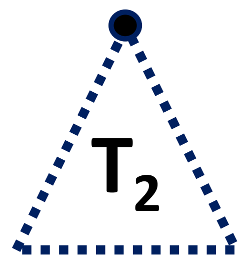
Basis: $6 \in S$; $15 \in S$;

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

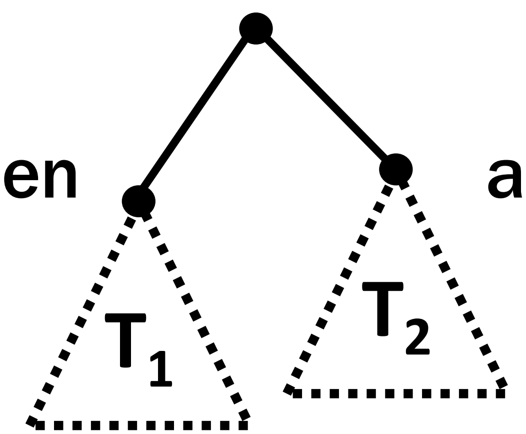
Rooted Binary Trees

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

If 

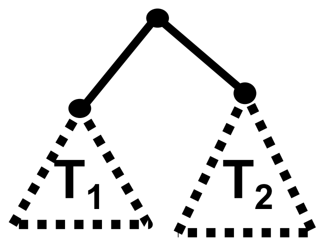
and 

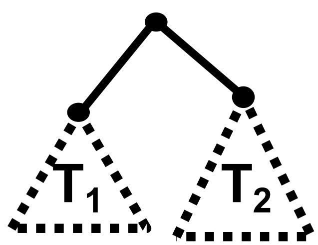
are rooted binary trees,

then 

also is a rooted binary tree.

Defining Functions on Rooted Binary Trees

- size(•) = 1
- size (

) = 1 + size($\mathbf{T_1}$) + size($\mathbf{T_2}$)
- height(•) = 0
- height (

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

Claim: For every rooted binary tree T, size(**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.**

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

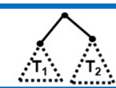
---

1. **Let** P(**T**) **be** "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:** size(•)=1, height(•)=0, **and** $2^{0+1}-1=2^1-1=1$ **so** P(•) **is true.**

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

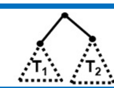
---

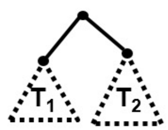
1. **Let** $P(\mathbf{T})$ **be** "$\text{size}(T) \leq 2^{\text{height}(T)+1} - 1$". **We prove** $P(\mathbf{T})$ **for all rooted binary trees T by structural induction.**
2. **Base Case:** $\text{size}(\bullet)=1$, $\text{height}(\bullet)=0$, **and** $2^{0+1}-1=2^1-1=1$ **so** $P(\bullet)$ **is true.**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

Goal: Prove P().

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

1. **Let** P(T) **be** "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:** size(•)=1, height(•)=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) $\leq 2^{\text{height}(T_k)+1} - 1$ **for** k=1,2
4. **Inductive Step:**

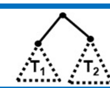
Goal: Prove P().

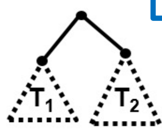
size()

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

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.**
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.**
3. **Inductive Hypothesis: Suppose that** $P(\mathbf{T_1})$ **and** $P(\mathbf{T_2})$ **are true for some rooted binary trees $\mathbf{T_1}$ and $\mathbf{T_2}$, i.e.,** $\text{size}(\mathbf{T_k}) \leq 2^{\text{height}(\mathbf{T_k})+1} - 1$ **for** $k=1,2$
4. **Inductive Step:**

Goal: Prove P().

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

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

by IH for $\mathbf{T_1}$ and $\mathbf{T_2}$

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

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

≤ 2(2^height()) – 1 ≤ 2^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 $wa \in \Sigma^*$

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 $w \in \Sigma^*$, $a \in \Sigma$, $a \neq c$

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

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.**

Claim: len(x•y) = len(x) + len(y) **for all** x,y $\in \Sigma^*$

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

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

Claim: len(x•y) = len(x) + len(y) **for all** x,y ∈ Σ^*

Let P(y) **be** "len(x•y) = len(x) + len(y) for all x ∈ Σ^* " .
We prove P(y) **for all** y ∈ Σ^* **by structural induction.**

Base Case (y = ε): **Let** x ∈ Σ^* **be arbitrary. Then,** len(x • ε) = len(x) = len(x) + len(ε) **since** len(ε)=0. **Since** x **was arbitrary,** P(ε) **holds.**

Inductive Hypothesis: Assume that P(w) **is true for some arbitrary** w ∈ Σ^***, i.e.,** len(x•w) = len(x) + len(w) for all x

Inductive Step: **Goal: Show that** P(wa) **is true for every** a ∈ Σ

Claim: len(x•y) = len(x) + len(y) **for all** x,y ∈ Σ^*

Let P(y) **be** "len(x•y) = len(x) + len(y) for all x ∈ Σ^*" .
We prove P(y) **for all** y ∈ Σ^* **by structural induction.**

Base Case (y = ε): **Let** x ∈ Σ^* **be arbitrary. Then,** len(x • ε) = len(x) = len(x) + len(ε) **since** len(ε)=0. **Since** x **was arbitrary,** P(ε) **holds.**

Inductive Hypothesis: Assume that P(w) **is true for some arbitrary** w ∈ Σ^***, i.e.,** len(x•w) = len(x) + len(w) for all x

Inductive Step: **Goal: Show that** P(wa) **is true for every** a ∈ Σ

Let a ∈ Σ **and** x ∈ Σ^* **be arbitrary. Then,**

len(x•wa)

= len(x)+len(wa)

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

Let $P(y)$ be "$len(x \bullet y) = len(x) + len(y)$ 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, $len(x \bullet \varepsilon) = len(x) = len(x) + len(\varepsilon)$ since $len(\varepsilon)=0$. Since x was arbitrary, $P(\varepsilon)$ holds.

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

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

Let $a \in \Sigma$. Let $x \in \Sigma^*$. Then $len(x \bullet wa) = len((x \bullet w)a)$ by def of $\bullet$
$= len(x \bullet w)+1$ by def of len
$= len(x)+len(w)+1$ by I.H.
$= len(x)+len(wa)$ by def of len

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

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

Parse Trees of Propositions

- **Basis:** Atomic(v) ∈ **Prop for any** v ∈ {p, q, r, …}
- **Recursive step:**

  Neg(A) ∈ **Prop for any** A ∈ **Prop**

  Wedge(A,B) ∈ **Prop for any** A, B ∈ **Prop**

  Vee(A,B) ∈ **Prop for any** A, B ∈ **Prop**

Example

- Wedge(Atomic(p), Neg(Atomic(r)))
  is the parse tree of "$p \wedge \neg r$"

Functions on Propositions

T takes parse tree to corresponding proposition

- **T**(Atomic(v)) = v
- **T**(Wedge(A, B)) = **T**(A) $\wedge$ **T**(B)
- **T**(Vee(A, B)) = **T**(A) $\vee$ **T**(B)
- **T**(Neg(A)) = $\neg$ **T**(A)

Example

T(Wedge(Atomic(p), Neg(Atomic(r))))

= **T**(Atomic(p)) $\wedge$ **T**(Neg(Atomic(r)))

= **T**(Atomic(p)) $\wedge \neg$ **T**(Atomic(r))

= p $\wedge \neg$ r

Functions on Propositions

flip is defined to mirror De Morgan's law

- **flip**(Atomic(v)) = Neg(Atomic(v))
- **flip**(Neg(A)) = A
- **flip**(Wedge(A, B)) = Vee(**flip**(A), **flip**(B))
- **flip**(Vee(A, B)) = Wedge(**flip**(A), **flip**(B))

Example

flip(Wedge(Atomic(p), Neg(Atomic(r))))

= Vee(**flip**(Atomic(p)), **flip**(Neg(Atomic(r))))

= Vee(**flip**(Atomic(p)), Atomic(r))

= Vee(Neg(Atomic(p)), Atomic(r))

Claim: **T**(Neg(A)) ≡ **T**(**flip**(A)) **for all** A ∈ **Prop**

Claim: **T**(Neg(A)) ≡ **T**(**flip**(A)) **for all** A ∈ **Prop**

Let P(A) **be "T**(Neg(A)) ≡ **T**(**flip**(A))**".**

We will prove P(A) **for** A ∈ **Prop by structural induction.**

Claim: **T**(Neg(A)) ≡ **T**(**flip**(A)) for all A ∈ **Prop**

Let P(A) **be "T**(Neg(A)) ≡ **T**(**flip**(A))**".**

We will prove P(A) **for** A ∈ **Prop by structural induction.**

Base Case: **Let** v ∈ {p, q, r, …}**. We want to show** P(Atomic(v))**.**

LHS is T(Neg(Atomic(v)).

RHS is T(**flip**(Atomic(v)) = **T**(Neg(Atomic(v)) **by def of flip.**

So, the two sides are equal.

Claim: **T**(Neg(A)) ≡ **T**(**flip**(A)) **for all** A ∈ **Prop**

Let P(A) **be "T**(Neg(A)) ≡ **T**(**flip**(A))**".**

We will prove P(A) **for** A ∈ **Prop by structural induction.**

Base Case: Let v ∈ {p, q, r, …}**. …** P(Atomic(v)) **holds.**

Inductive Hypothesis: Suppose P(A) **and** P(B) **hold for some arbitrary** A, B ∈ **Prop, i.e., T**(Neg(A)) = **T**(**flip**(A)) **and for** B.

Claim: **T**(Neg(A)) ≡ **T**(**flip**(A)) **for all** A ∈ **Prop**

Let P(A) **be "T**(Neg(A)) ≡ **T**(**flip**(A))**".**

We will prove P(A) **for** A ∈ **Prop by structural induction.**

Base Case: **Let** v ∈ {p, q, r, …}**. …** P(Atomic(v)) **holds.**

Inductive Hypothesis: **Suppose** P(A) **and** P(B) **hold for some arbitrary** A, B ∈ **Prop, i.e., T**(Neg(A)) = **T**(**flip**(A)) **and for** B.

Inductive Step: **Goal: Prove** P(Neg(A)), P(Wedge(A, B)), **and …**

Claim: **T**(Neg(A)) ≡ **T**(**flip**(A)) for all A ∈ **Prop**

Let P(A) **be "T**(Neg(A)) ≡ **T**(**flip**(A))**".**

We will prove P(A) **for** A ∈ **Prop by structural induction.**

Base Case: **Let** v ∈ {p, q, r, …}**. …** P(Atomic(v)) **holds.**

Inductive Hypothesis: **Suppose** P(A) **and** P(B) **hold for some arbitrary** A, B ∈ **Prop, i.e., T**(Neg(A)) = **T**(**flip**(A)) **and for** B.

Inductive Step: **Goal: Prove** P(Neg(A)), P(Wedge(A, B)), **and …**

T(Neg(Neg(A)))

≡ ¬**T**(Neg(A))	Def of **T**
≡ ¬ ¬**T**(A)	Def of **T**
≡ **T**(A)	Double Negation
≡ **T**(**flip**(Neg(A)))	Def of **flip**

Claim: T(Neg(A)) ≡ T(flip(A)) for all A ∈ Prop

Let P(A) **be "T**(Neg(A)) ≡ **T**(**flip**(A))**".**
We will prove P(A) **for** A ∈ **Prop by structural induction.**

Base Case: **Let** v ∈ {p, q, r, …}**. …** P(Atomic(v)) **holds.**

Inductive Hypothesis: **Suppose** P(A) **and** P(B) **hold for some arbitrary** A, B ∈ **Prop, i.e., T**(Neg(A)) = **T**(**flip**(A)) **and for** B.

Inductive Step: **Goal: Prove** P(Neg(A)), P(Wedge(A, B)), **and …**

T(Neg(Wedge(A, B)))

= ¬**T**(Wedge(A, B))	Def of **T**
= ¬(**T**(A) ∧ **T**(B))	Def of **T**
= ¬**T**(A) ∨ ¬**T**(B)	De Morgan
= **T**(Neg(A)) ∨ **T**(Neg(B))	Def of **T**
= **T**(**flip**(A)) ∨ **T**(**flip**(B))	**IH**
= **T**(Vee(**flip**(A), **flip**(B))	Def of **T**
= **T**(**flip**(Wedge(A, B))	Def of **flip**

Claim: $\mathbf{T}(\text{Neg}(A)) \equiv \mathbf{T}(\mathbf{flip}(A))$ **for all** $A \in$ **Prop**

Let P(A) **be "T**(Neg(A)) ≡ **T**(**flip**(A))**".**

We will prove P(A) **for** A ∈ **Prop by structural induction.**

Base Case: **Let** v ∈ {p, q, r, …}**. …** P(Atomic(v)) **holds.**

Inductive Hypothesis: **Suppose** P(A) **and** P(B) **hold for some arbitrary** A, B ∈ **Prop, i.e., T**(Neg(A)) = **T**(**flip**(A)) **and for** B.

Inductive Step: **Goal: Prove** P(Neg(A)), P(Wedge(A, B)), **and …**

T(Neg(Vee(A, B)))

= ¬**T**(Vee(A, B))	Def of **T**
= ¬(**T**(A) ∨ **T**(B))	Def of **T**
= ¬**T**(A) ∧ ¬**T**(B)	De Morgan
= **T**(Neg(A)) ∧ **T**(Neg(B))	Def of **T**
= **T**(**flip**(A)) ∧ **T**(**flip**(B))	**IH**
= **T**(Wedge(**flip**(A), **flip**(B))	Def of **T**
= **T**(**flip**(Vee(A, B))	Def of **flip**

Claim: **T**(Neg(A)) ≡ **T**(**flip**(A)) **for all** A ∈ **Prop**

Let P(A) **be "T**(Neg(A)) ≡ **T**(**flip**(A))**".**
We will prove P(A) **for** A ∈ **Prop by structural induction.**

Base Case: Let v ∈ {p, q, r, …}**. …** P(Atomic(v)) **holds.**

Inductive Hypothesis: Suppose P(A) **and** P(B) **hold for some arbitrary** A, B ∈ **Prop, i.e., T**(Neg(A)) = **T**(**flip**(A)) **and for** B.

Inductive Step: **Goal: Prove** P(Neg(A)), P(Wedge(A, B)), **and …**

… so P(Neg(A)) **holds.**

… so P(Wedge(A, B)) **holds.**

… so P(Vee(A, B)) **holds.**

Thus, we have proven P(A) **for all** A **by structural induction.**

More Theorems

Used structural induction to prove a more general version of De Morgan's law. (Allows any number of ANDs and ORs.)

Structural induction is also the right tool to prove that, if we have $x \equiv_m y$, we can substitute y for x everywhere in an arithmetic expression and the result will be congruent.

(This tool, whose most obvious application is proving facts about programs, also lets us prove important math results.)

Linked Lists of Integers

- **Basis:** null $\in$ **Lists**
- **Recursive step:**

  If L $\in$ **Lists and** v $\in \mathbb{Z}$**, then** Node(v, L) $\in$ **Lists**

Examples:

– null	[]
– Node(1, null)	[1]
– Node(1, Node(2, null))	[1, 2]

Functions on Linked Lists

Set of numbers stored in a list:

- **values**(null) = ∅
- **values**(Node(v, L)) = {v} ∪ **values**(L)

Example:

values(Node(1, Node(2, null))	
= {1} ∪ **values**(Node(2, null)	**Def of values**
= {1} ∪ {2} ∪ **values**(null)	**Def of values**
= {1} ∪ {2} ∪ ∅ = {1, 2}	**Def of values**

Functions on Linked Lists

Remove the numbers that don't satisfy p(v):

- $\textbf{filter}_p$(null) = null
- $\textbf{filter}_p$(Node(v, L)) = Node(v, $\textbf{filter}_p$(L)) **if** p(v)
- $\textbf{filter}_p$(Node(v, L))= $\textbf{filter}_p$(L) **otherwise**

Example: p(v) := v < 2

$\textbf{filter}_p$(Node(1, Node(2, null)))

= Node(1, $\textbf{filter}_p$(Node(2, null))) **Def** $\textbf{filter}_p$

= Node(1, $\textbf{filter}_p$(null)) **Def** $\textbf{filter}_p$

= Node(1, null) **Def** $\textbf{filter}_p$

Claim: $x \in$ **values**(**filter**$_p$(L)) iff $p(x) \wedge x \in$ **values**(L)

Claim: $x \in$ **values**(**filter**$_p$(L)) iff $p(x) \land x \in$ **values**(L)

Q(L) := "$x \in$ **values**(**filter**$_p$(L)) iff $p(x) \land x \in$ **values**(L) for all $x \in \mathbb{Z}$".
We will prove Q(L) for L ∈ **Lists** by structural induction.

Claim: $x \in$ **values**(**filter**$_p$(L)) iff $p(x) \wedge x \in$ **values**(L)

Q(L) := "$x \in$ **values**(**filter**$_p$(L)) iff $p(x) \wedge x \in$ **values**(L) for all $x \in \mathbb{Z}$".
We will prove Q(L) for L ∈ **Lists** by structural induction.

Base Case: Let $x \in \mathbb{Z}$ be arbitrary.

LHS is $x \in$ **values**(**filter**$_p$(null))

$\equiv x \in$ **values**(null)	Def of **filter**$_p$
$\equiv x \in \emptyset$	Def of **values**
$\equiv$ F	Def of $\emptyset$

RHS is $p(x) \wedge x \in$ **values**(null)

$\equiv p(x) \wedge x \in \emptyset$	Def of **values**
$\equiv p(x) \wedge$ F	Def of $\emptyset$
$\equiv$ F	Domination

These are equal as required.
Since x was arbitrary, this shows that Q(null) **holds.**

Claim: $x \in$ **values**(**filter**$_p$(L)) iff $p(x) \land x \in$ **values**(L)

Q(L) := "$x \in$ **values**(**filter**$_p$(L)) iff $p(x) \land x \in$ **values**(L) for all $x \in \mathbb{Z}$".
We will prove Q(L) for L ∈ **Lists** by structural induction.

Base Case: … so Q(null) **holds.**

Inductive Hypothesis: Suppose Q(L) **holds for an arbitrary list L,**
i.e., we have $x \in$ **values**(**filter**$_p$(L)) iff $p(x) \land x \in$ **values**(L)**.**

Claim: x ∈ **values**(**filter**$_p$(L)) iff p(x) ∧ x ∈ **values**(L)

Q(L) := "x ∈ **values**(**filter**$_p$(L)) iff p(x) ∧ x ∈ **values**(L) for all x ∈ ℤ".
We will prove Q(L) for L ∈ **Lists** by structural induction.

Base Case: … so Q(null) **holds.**

Inductive Hypothesis: Suppose Q(L) **holds for an arbitrary list L,**
i.e., we have x ∈ **values**(**filter**$_p$(L)) iff p(x) ∧ x ∈ **values**(L)**.**

Inductive Step: **Goal: Prove** Q(Node(v, L)) **for all** v ∈ ℤ

Claim: v ∈ **values**(**filter**$_p$(L)) iff p(x) ∧ x ∈ **values**(L)

Q(L) := "x ∈ **values**(**filter**$_p$(L)) iff p(x) ∧ x ∈ **values**(L) for all x ∈ ℤ".
We will prove Q(L) for L ∈ **Lists** by structural induction.

Base Case: ... so Q(null) **holds.**

Inductive Hypothesis: Suppose Q(L) **holds for an arbitrary list L,**
i.e., we have x ∈ **values**(**filter**$_p$(L)) iff p(x) ∧ x ∈ **values**(L)**.**

Inductive Step: **Goal: Prove** Q(Node(v, L)) **for all** v ∈ ℤ

Let x, v ∈ ℤ **be arbitrary. We go by cases. Suppose** ¬p(v).

x ∈ **values**(**filter**$_p$(Node(v, L)))	
≡ x ∈ **values**(**filter**$_p$(L))	Def **filter**$_p$
≡ p(x) ∧ x ∈ **values**(L)	IH

Claim: $v \in \textbf{values}(\textbf{filter}_p(L))$ iff $p(x) \land x \in \textbf{values}(L)$

$Q(L) :=$ "$x \in \textbf{values}(\textbf{filter}_p(L))$ iff $p(x) \land x \in \textbf{values}(L)$ for all $x \in \mathbb{Z}$".
We will prove $Q(L)$ for $L \in$ **Lists** by structural induction.

Base Case: ... so $Q(\text{null})$ **holds.**

Inductive Hypothesis: Suppose $Q(L)$ **holds for an arbitrary list L,**
i.e., we have $x \in \textbf{values}(\textbf{filter}_p(L))$ iff $p(x) \land x \in \textbf{values}(L)$**.**

Inductive Step: Goal: Prove $Q(\text{Node}(v, L))$ **for all** $v \in \mathbb{Z}$

Let $x, v \in \mathbb{Z}$ **be arbitrary. We go by cases. Suppose** $\neg p(v)$**.**

$x \in \textbf{values}(\textbf{filter}_p(\text{Node}(v, L)))$	
$\equiv x \in \textbf{values}(\textbf{filter}_p(L))$	Def $\textbf{filter}_p$
$\equiv p(x) \land x \in \textbf{values}(L)$	IH

If $\neg p(x)$**, then this and** $p(x) \land x \in \textbf{values}(\text{Node}(v, L))$ **are equivalent as they are both false. So now suppose** $p(x)$**...**

Claim: $v \in \textbf{values}(\textbf{filter}_p(L))$ iff $p(x) \land x \in \textbf{values}(L)$

$Q(L)$:= "$x \in \textbf{values}(\textbf{filter}_p(L))$ iff $p(x) \land x \in \textbf{values}(L)$ for all $x \in \mathbb{Z}$".
We will prove $Q(L)$ for $L \in$ **Lists** by structural induction.

Base Case: … so $Q(\text{null})$ **holds.**

Inductive Hypothesis: Suppose $Q(L)$ **holds for an arbitrary list L,**
i.e., we have $x \in \textbf{values}(\textbf{filter}_p(L))$ iff $p(x) \land x \in \textbf{values}(L)$.

Inductive Step: Goal: Prove $Q(\text{Node}(v, L))$ **for all** $v \in \mathbb{Z}$

Let $x, v \in \mathbb{Z}$ **be arbitrary. We go by cases. Suppose** $\neg p(v)$.

$x \in \textbf{values}(\textbf{filter}_p(\text{Node}(v, L)))$	
$\equiv x \in \textbf{values}(\textbf{filter}_p(L))$	Def $\textbf{filter}_p$
$\equiv p(x) \land x \in \textbf{values}(L)$	IH
$\equiv p(x) \land (F \lor x \in \textbf{values}(L))$	Identity
$\equiv p(x) \land (x \in \{v\} \lor x \in \textbf{values}(L))$	??
$\equiv p(x) \land (x \in \{v\} \cup \textbf{values}(L))$	Def $\cup$
$\equiv p(x) \land (x \in \textbf{values}(\text{Node}(v, L)))$	Def **values**

Claim: $v \in$ **values**(**filter**$_p$(L)) iff $p(x) \land x \in$ **values**(L)

Q(L) := "$x \in$ **values**(**filter**$_p$(L)) iff $p(x) \land x \in$ **values**(L) for all $x \in \mathbb{Z}$".
We will prove Q(L) for $L \in$ **Lists** by structural induction.

Base Case: … so Q(null) **holds.**

Inductive Hypothesis: Suppose Q(L) **holds for an arbitrary list L, i.e., we have** $x \in$ **values**(**filter**$_p$(L)) iff $p(x) \land x \in$ **values**(L).

Inductive Step: **Goal: Prove** Q(Node(v, L)) **for all** $v \in \mathbb{Z}$

Let $x, v \in \mathbb{Z}$ **be arbitrary. We go by cases. Suppose** $\neg p(v)$.

$x \in$ **values**(**filter**$_p$(Node(v, L)))	
$\equiv x \in$ **values**(**filter**$_p$(L))	Def **filter**$_p$
$\equiv p(x) \land x \in$ **values**(L)	IH
$\equiv p(x) \land (F \lor x \in$ **values**(L))	Identity
$\equiv p(x) \land (x \in \{v\} \lor x \in$ **values**(L))	$x \neq v$ as $p(x)$ but $\neg p(v)$
$\equiv p(x) \land (x \in \{v\} \cup$ **values**(L))	Def $\cup$
$\equiv p(x) \land (x \in$ **values**(Node(v, L)))	Def **values**

Claim: $v \in \textbf{values}(\textbf{filter}_p(L))$ iff $p(x) \land x \in \textbf{values}(L)$

$Q(L)$:= "$x \in \textbf{values}(\textbf{filter}_p(L))$ iff $p(x) \land x \in \textbf{values}(L)$ for all $x \in \mathbb{Z}$".
We will prove $Q(L)$ for $L \in$ **Lists** by structural induction.

Base Case: ... **so** $Q(\text{null})$ **holds.**

Inductive Hypothesis: **Suppose** $Q(L)$ **holds for an arbitrary list L,**
i.e., we have $x \in \textbf{values}(\textbf{filter}_p(L))$ iff $p(x) \land x \in \textbf{values}(L)$.

Inductive Step: Goal: Prove $Q(\text{Node}(v, L))$ **for all** $v \in \mathbb{Z}$

Let $x, v \in \mathbb{Z}$ **be arbitrary. We go by cases. Suppose** $\neg p(v)$.

$x \in \textbf{values}(\textbf{filter}_p(\text{Node}(v, L)))$

$\equiv$ ...

$\equiv p(x) \land (x \in \textbf{values}(\text{Node}(v, L)))$

Thus, the claimed bicondition holds.
Since x **was arbitrary, we have shown** $Q(\text{Node}(v, L))$.

Claim: $v \in \textbf{values}(\textbf{filter}_p(L))$ iff $p(x) \wedge x \in \textbf{values}(L)$

$Q(L)$:= "$x \in \textbf{values}(\textbf{filter}_p(L))$ iff $p(x) \wedge x \in \textbf{values}(L)$ for all $x \in \mathbb{Z}$".
We will prove $Q(L)$ for $L \in$ **Lists** by structural induction.

Base Case: ... so $Q(\text{null})$ **holds.**

Inductive Hypothesis: Suppose $Q(L)$ **holds for an arbitrary list L,**
i.e., we have $x \in \textbf{values}(\textbf{filter}_p(L))$ iff $p(x) \wedge x \in \textbf{values}(L)$**.**

Inductive Step: Goal: Prove $Q(\text{Node}(v, L))$ **for all** $v \in \mathbb{Z}$

Let $x, v \in \mathbb{Z}$ **be arbitrary. We go by cases. Suppose** $p(v)$**.**

$x \in \textbf{values}(\textbf{filter}_p(\text{Node}(v, L)))$	
$\equiv x \in \textbf{values}(\text{Node}(v, \textbf{filter}_p(L)))$	Def $\textbf{filter}_p$
$\equiv x \in \{v\} \cup \textbf{values}(\textbf{filter}_p(L))$	Def **values**
$\equiv x \in \{v\} \vee x \in \textbf{values}(\textbf{filter}_p(L))$	Def $\cup$
$\equiv x \in \{v\} \vee (p(x) \wedge x \in \textbf{values}(L))$	IH
$\equiv (x \in \{v\} \vee p(x)) \wedge (x \in \{v\} \vee x \in \textbf{values}(L))$	Distributivity
$\equiv (x \in \{v\} \vee p(x)) \wedge (x \in \textbf{values}(\text{Node}(v, L)))$	Def $\cup$, **values**

Claim: $v \in$ **values**(**filter**$_p$(L)) iff $p(x) \land x \in$ **values**(L)

Q(L) := "x ∈ **values**(**filter**$_p$(L)) iff p(x) ∧ x ∈ **values**(L) for all x ∈ ℤ".
We will prove Q(L) for L ∈ **Lists** by structural induction.

Base Case: ... so Q(null) **holds.**

Inductive Hypothesis: Suppose Q(L) **holds for an arbitrary list L,**
i.e., we have x ∈ **values**(**filter**$_p$(L)) iff p(x) ∧ x ∈ **values**(L).

Inductive Step: Goal: Prove Q(Node(v, L)) **for all** v ∈ ℤ

Let x, v ∈ ℤ **be arbitrary. We go by cases. Suppose** p(v).

x ∈ **values**(**filter**$_p$(Node(v, L)))

≡ ...

≡ (x ∈ {v} ∨ p(x)) ∧ (x ∈ **values**(Node(v, L)))

If x ∈ {v} **is false, then the first part is** F ∨ p(x) ≡ p(x).

If true, then x = v, **and first part and** p(x) **are both true. Thus,**

≡ p(x) ∧ (x ∈ **values**(Node(v, L)))

Claim: $v \in \textbf{values}(\textbf{filter}_p(L))$ iff $p(x) \land x \in \textbf{values}(L)$

$Q(L)$:= "$x \in \textbf{values}(\textbf{filter}_p(L))$ iff $p(x) \land x \in \textbf{values}(L)$ for all $x \in \mathbb{Z}$".
We will prove $Q(L)$ for $L \in$ **Lists** by structural induction.

Base Case: … so $Q(\text{null})$ **holds.**

Inductive Hypothesis: Suppose $Q(L)$ **holds for an arbitrary list L,**
i.e., we have $x \in \textbf{values}(\textbf{filter}_p(L))$ iff $p(x) \land x \in \textbf{values}(L)$**.**

Inductive Step: Goal: Prove $P(\text{Node}(v, L))$ **for all** $v \in \mathbb{Z}$

Let $x, v \in \mathbb{Z}$ **be arbitrary. We go by cases. Suppose** $p(v)$**.**

$x \in \textbf{values}(\textbf{filter}_p(\text{Node}(v, L)))$

$\equiv$ …

$\equiv p(x) \land (x \in \textbf{values}(\text{Node}(v, L)))$

Thus, the bicondition claimed by $Q(\text{Node}(v, L))$ **holds.**

In both cases, we have $Q(\text{Node}(v, L))$ **since x was arbitrary.**

Hence, we have shown $Q(L)$ **for all lists by structural induction.**