CSE 311 Foundations of Computing I

Lecture 16
Recursively Defined Sets
and Structural Induction
Spring 2013

Announcements

- Reading assignments
  - Today:
    - 5.3 7th Edition
    - 5.3 6th Edition
- Midterm Friday, May 10, MGH 389
  - Closed book, closed notes
  - Tables of inference rules and equivalences will be included on test
  - Sample questions from old midterms are now posted

Induction

- Mathematical Induction

$$\frac{\begin{array}{l} P(0) \\ \forall\, k\geq 0\ (P(k) \rightarrow P(k+1)) \end{array}}{\therefore\ \forall\, n\geq 0\ \ P(n)}$$

- Induction proof layout:
  1. Let P(n) be " ". By induction we will show that P(n) is true for every n≥0
  2. Base Case: Prove P(0)
  3. Inductive Hypothesis: Assume that P(k) is true for some arbitrary integer k ≥ 0
  4. Inductive Step: Prove that P(k+1) is true using Inductive Hypothesis that P(k) is true
  5. Conclusion: Result follows by induction

Strong Induction

$$\frac{\begin{array}{l} P(0) \\ \forall\, k\ (\forall\, j\ ((0\leq j \leq k) \rightarrow P(j)) \rightarrow P(k+1)) \end{array}}{\therefore\ \forall\, n\ P(n)}$$

- Strong Induction proof layout:
  1. Let P(n) be "…". By induction we will show that P(n) is true for every n≥0
  2. Base Case: Prove P(0)
  3. Inductive Hypothesis: Assume that for some arbitrary integer k ≥ 0 we have P(j) true for every integer j with 0 ≤ j ≤ k.
  4. Inductive Step: Prove that P(k+1) is true using Inductive Hypothesis that P(0),…,P(k) are true
  5. Conclusion: Result follows by induction

Bounding the Fibonacci Numbers

How we did it last time

$f_0 = 0$; $f_1 = 1$; $f_n = f_{n-1} + f_{n-2}$ for all $n \geq 2$

Theorem: $2^{n/2-1} \leq f_n < 2^n$ for all $n \geq 2$

Proof:

1. Let P(n) be "$2^{n/2-1} \leq f_n < 2^n$. By (strong) induction we prove P(n) for all $n \geq 2$.
2. **Base Case:** … P(2) is true, … P(3) is true
3. **Ind.Hyp:** Assume $2^{j/2-1} \leq f_j < 2^j$ for all integers j with $2 \leq j \leq k$ for … $k \geq 3$.
4. **Ind. Step:** $\boxed{\text{Goal: Show } 2^{(k+1)/2-1} \leq f_{k+1} < 2^{k+1}}$

$$f_{k+1} = f_k + f_{k-1} \geq 2^{k/2-1} + 2^{(k-1)/2-1} \text{ by I.H. since } k-1 \geq 2$$
$$> 2^{(k-1)/2-1} + 2^{(k-1)/2-1} = 2 \cdot 2^{(k-1)/2-1} = 2^{(k+1)/2-1}$$
$$f_{k+1} = f_k + f_{k-1} < 2^k + 2^{(k-1)} \text{ by I.H. since } k-1 \geq 2$$
$$< 2^k + 2^k = 2 \cdot 2^k = 2^{k+1}$$

Bounding the Fibonacci Numbers

Alternative Layout

$f_0 = 0$; $f_1 = 1$; $f_n = f_{n-1} + f_{n-2}$ for all $n \geq 2$

Theorem: $2^{n/2-1} \leq f_n < 2^n$ for all $n \geq 2$

Proof:

1. Let P(n) be "$2^{n/2-1} \leq f_n < 2^n$. By (strong) induction we prove P(n) for all $n \geq 2$.
2. **Base Case:** **…** P(2) is true
3. **Ind.Hyp:** Assume $2^{j/2-1} \leq f_j < 2^j$ for all integers j with $2 \leq j \leq k$ for … $k \geq 2$.
4. **Ind. Step:** $\boxed{\text{Goal: Show } 2^{(k+1)/2-1} \leq f_{k+1} < 2^{k+1}}$

<u>Case k=2</u>: **…** P(3) is true

<u>Case k≥3:</u> $f_{k+1} = f_k + f_{k-1} \geq 2^{k/2-1} + 2^{(k-1)/2-1}$ by I.H. since $k-1 \geq 2$

$$> 2^{(k-1)/2-1} + 2^{(k-1)/2-1} = 2 \cdot 2^{(k-1)/2-1} = 2^{(k+1)/2-1}$$
$$f_{k+1} = f_k + f_{k-1} < 2^k + 2^{(k-1)} \text{ by I.H. since } k-1 \geq 2$$
$$< 2^k + 2^k = 2 \cdot 2^k = 2^{k+1}$$

Recursive Definitions of Sets

- Recursive definition
  - Basis step: $0 \in S$
  - Recursive step: if $x \in S$, then $x + 2 \in S$
  - Exclusion rule: Every element in S follows from basis steps and a finite number of recursive steps

Recursive definitions of sets

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

Basis: $[1, 1, 0] \in S$, $[0, 1, 1] \in S$;
Recursive:
if $[x, y, z] \in S$, α in $\mathbb{R}$, then $[\alpha x, \alpha y, \alpha z] \in S$
if $[x_1, y_1, z_1], [x_2, y_2, z_2] \in S$
then $[x_1 + x_2, y_1 + y_2, z_1 + z_2] \in S$

Powers of 3

Recursive Definitions of Sets: General Form

- Recursive definition
  - *Basis step:* Some specific elements are in S
  - *Recursive step:* Given some existing named elements in S some new objects constructed from these named elements are also in S.
  - Exclusion rule: Every element in S follows from basis steps and a finite number of recursive steps

Strings

- An *alphabet* Σ is any finite set of characters.
- The set Σ^* of *strings* over the alphabet Σ is defined by
  - **Basis:** $\lambda \in \Sigma^*$ (λ is the empty string)
  - **Recursive:** if $w \in \Sigma^*$, $a \in \Sigma$, then $wa \in \Sigma^*$

Palindromes

- Palindromes are strings that are the same backwards and forwards
- **Basis:** λ is a palindrome and any $a \in \Sigma$ is a palindrome
- **Recursive step:** If *p* is a palindrome then a*p*a is a palindrome for every $a \in \Sigma$

All binary strings with no 1's before 0's

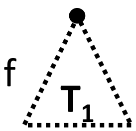
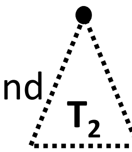
Function definitions on recursively defined sets

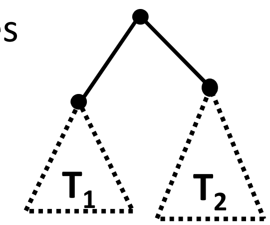
len (λ) = 0;
len (wa) = 1 + len(w); for $w \in \Sigma^*$, $a \in \Sigma$

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

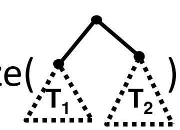
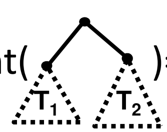
Concatenation:
$x \bullet \lambda = x$ for $x \in \Sigma^*$
$x \bullet wa = (x \bullet w)a$ for $x, w \in \Sigma^*$, $a \in \Sigma$

Rooted Binary trees

- **Basis:** • is a rooted binary tree
- **Recursive Step:** If  and  are rooted binary trees then so is:



Functions defined 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)}

Structural Induction: proving properties of recursively defined sets

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

•**Base Case:** Show that P is true for all specific elements 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 holds for each of the new elements constructed in the *Recursive step* using the named elements mentioned in the Inductive Hypothesis

•Conclude that $\forall x \in S.\ P(x)$

Structural Induction versus 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 $k+1 \in \mathbb{N}$
- Structural induction follows from ordinary induction
  - Let Q(n) be true iff for all $x \in S$ that take n Recursive steps to be constructed, P(x) is true.

Using Structural Induction

- Let S be given by
  - **Basis:** $6 \in S$; $15 \in S$;
  - **Recursive:** if $x, y \in S$, then $x + y \in S$.
- **Claim:** Every element of S is divisible by 3

Structural Induction for strings

- Let S be a set of strings over {a,b} defined as follows
  - Basis: $a \in S$
  - Recursive:
    - If $w \in S$ then $aw \in S$ and $baw \in S$
    - If $u \in S$ and $v \in S$ then $uv \in S$
- Claim: if $w \in S$ then w has more a's than b's

len(x•y)=len(x)+len(y) for all strings x and y

Let P(w) be "len(x•w)=len(x)+len(w)"