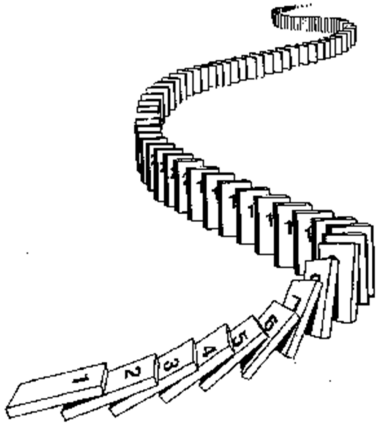
CSE 311: Foundations of Computing

Fall 2013

Lecture 16: Recursively defined sets and structural induction



announcements

Reading assignment

Induction

5.3, 7th edition

4.3, 6th edition

Homework 5 due today. No new homework now.

Midterm

Monday, November 4th, IN CLASS

Topics: Everything up to ordinary induction and recursive definitions of functions.

Sample questions from old midterms now posted

Solutions will be posted later

review: induction

$$\frac{P(0) \quad \forall\, k\,(P(k) \;\rightarrow\; P(k+1))}{\therefore\ \forall\, n\, P(n)}$$

1. "By induction we will show that P(n) is true for every n ≥ 0."
2. "Base Case:" Prove P(0)
3. "Inductive Hypothesis:"
   Assume P(k) is true for some arbitrary integer k ≥ 0"
4. "Inductive Step:" Want to prove that P(k+1) is true:
   Use the goal to figure out what you need.
   Make sure you are using I.H. and point out where you are using it. (Don't assume P(k+1) !!)
5. "Conclusion: Result follows by induction"

review: strong induction

$$\frac{P(0) \quad \forall k\,\Big(\big(P(0) \wedge P(1) \wedge P(2) \wedge \cdots \wedge P(k)\big) \rightarrow P(k+1)\Big)}{\therefore\ \forall n\, P(n)}$$

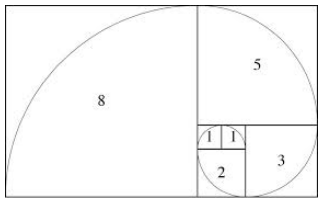
1. By induction we will show that $P(n)$ is true for every $n \geq 0$
2. Base Case: Prove $P(0)$
3. Inductive Hypothesis:
   Assume that for some arbitrary integer $k \geq 0$, $P(j)$ is true for every j from 0 to k
4. Inductive Step:
   Prove that $P(k+1)$ is true using the Inductive Hypothesis (that $P(j)$ is true for all values $\leq k$)
5. Conclusion: Result follows by induction

review: recursive definitions of functions

- $F(0) = 0$; $F(n+1) = F(n) + 1$ **for all** $n \geq 0$
- $G(0) = 1$; $G(n+1) = 2 \times G(n)$ **for all** $n \geq 0$
- $0! = 1$; $(n+1)! = (n+1) \times n!$ **for all** $n \geq 0$
- $H(0) = 1$; $H(n+1) = 2^{H(n)}$ **for all** $n \geq 0$

fibonacci numbers

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



bounding the fibonacci numbers

$f_0 = 0$; $f_1 = 1$; $f_n = f_{n-1} + f_{n-2}$ **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 ≥ 2.
2. **Base Case:** P(2) is true: $f_2=1$, $2^{2/2-1}=2^0=1 \leq f_2$, $2^2=4>f_2$
3. **Ind.Hyp:** Assume $2^{j/2-1} \leq f_j < 2^j$ for all integers j with $2 \leq j \leq k$ for some arbitrary integer k ≥ 2.
4. **Ind. Step:** Goal: Show $2^{(k+1)/2-1} \leq f_{k+1} < 2^{k+1}$

$f_0 = 0$; $f_1 = 1$; $f_n = f_{n-1} + f_{n-2}$ **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 ≥ 2.
2. **Base Case:** P(2) is true: $f_2=1$, $2^{2/2-1}=2^0=1 \leq f_2$, $2^2=4>f_2$
3. **Ind.Hyp:** Assume $2^{j/2-1} \leq f_j < 2^j$ for all integers j with $2 \leq j \leq k$ for some arbitrary integer k ≥ 2.
4. **Ind. Step:** Goal: Show $2^{(k+1)/2-1} \leq f_{k+1} < 2^{k+1}$

Case k=2: P(3) is true: $f_3=f_2+f_1=1+1=2$, $2^{3/2-1}=2^{1/2} \leq 2 = f_3$, $2^3=8 > f_3$

Case k≥3:

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

$> 2^{(k-1)/2-1} + 2^{(k-1)/2-1} = 2 \cdot 2^{(k-1)/2-1} = 2^{(k+1)/2-1}$

$f_{k+1} = f_k + f_{k-1} < 2^k + 2^{(k-1)}$ by I.H. since k-1 ≥ 2

$< 2^k + 2^k = 2 \cdot 2^k = 2^{k+1}$

running time of euclid's algorithm

Theorem: Suppose that Euclid's algorithm takes n steps for $\gcd(a, b)$ with $a > b$, then $a \geq f_{n+1}$

Set $r_{n+1} = a, r_n = b$ then Euclid's algorithm computes

$$
\begin{aligned}
r_{n+1} &= q_n r_n + r_{n-1} \\
r_n &= q_{n-1} r_{n-1} + r_{n-2} \\
&\vdots \\
r_3 &= q_2 r_2 + 1 \\
r_2 &= q_1 r_1
\end{aligned}
$$

each quotient $q_i \geq 1$

$r_1 \geq 1$

recursive definition 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 $[x, y, z] \in S$, $\alpha \in \mathbb{R}$, 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 apa 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 $w \in \Sigma^*, a \in \Sigma$

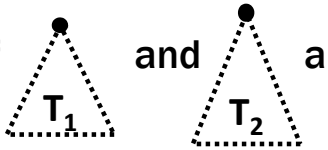
Reversal:

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

Concatenation:

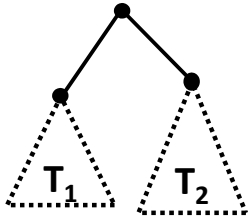
$x \bullet \lambda = x$ for $x \in \Sigma^*$
$x \bullet wa = (x \bullet w)a$ for $x, w \in \Sigma^*, a \in \Sigma$

rooted binary trees

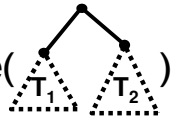
- **Basis:** • is a rooted binary tree
- **Recursive step:** If 

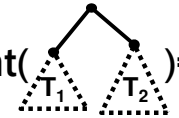
and are rooted binary trees

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

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 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 $k+1 \in \mathbb{N}$

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**

Claim: If $w \in S$ then w has more a's than b's

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 $uv \in S$

$\text{len}(x \bullet y) = \text{len}(x) + \text{len}(y)$ for all strings x and y

Let $P(w)$ be "$\text{len}(x \bullet w) = \text{len}(x) + \text{len}(w)$ for all strings x"

$\text{len}(x \bullet y) = \text{len}(x) + \text{len}(y)$ **for all strings** x **and** y

---

Let $P(w)$ **be** "$\text{len}(x \bullet w) = \text{len}(x) + \text{len}(w)$ **for all strings** x"