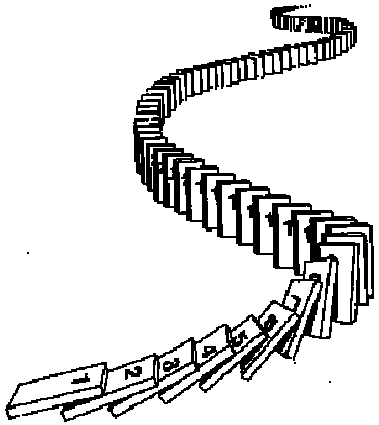
CSE 311: Foundations of Computing

Fall 2014

Lecture 16: Recursively Defined Sets



Strong Induction

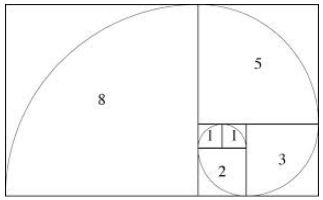
$$P(0)$$
$$\forall k \left(\left(P(0) \wedge P(1) \wedge P(2) \wedge \cdots \wedge P(k)\right) \rightarrow P(k+1)\right)$$

$$\therefore \forall n\, P(n)$$

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

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 $k \geq 2$.
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 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

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

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:** $\varepsilon \in \Sigma^*$ (ε 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:

ε 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

Length:

len (ε) = 0;

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

Reversal:

$\varepsilon^R = \varepsilon$

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

Concatenation:

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

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