CSE 311 Foundations of Computing I

Lecture 15

Strong Induction and Recursive Definitions

Spring 2013

Announcements

- Reading assignments
  - Today:
    - 5.2-5.3 7th Edition
    - 4.2-5.3 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

Highlights from last lecture

- Mathematical Induction

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

- Induction proof layout:
  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 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

$$\begin{array}{l} P(0) \\ \underline{\forall k\ ((P(0) \wedge P(1) \wedge P(2) \wedge \ldots \wedge P(k)) \rightarrow P(k+1))} \\ \therefore \forall n\ P(n) \end{array}$$

Follows from ordinary induction applied to
$Q(n) = P(0) \wedge P(1) \wedge P(2) \wedge \ldots \wedge P(n)$

Strong Induction English Proofs

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 that for some arbitrary integer k ≥ 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 ≤ k)
5. Conclusion: Result follows by induction

Every integer ≥ 2 is the product of primes

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

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

Fibonacci numbers and the running time of Euclid's algorithm

Lamé's Theorem: Suppose that Euclid's algorithm takes n steps for gcd(a,b) with $a>b$, then $a \ge f_{n+1}$ (which we know is $\ge 2^{n/2}$)

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

$r_{n+1}= q_n r_n + r_{n-1}$
$r_n= q_{n-1} r_{n-1} + r_{n-2}$
$\vdots$
$r_3= q_2 r_2 + r_1$
$r_2= q_1 r_1 + 0$

each quotient $q_i \ge 1$
$r_1 \ge 1=f_1$ "r_0"$=0=f_0$

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 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 $w \in \Sigma^*$, $x \in \Sigma$, then $wx \in \Sigma^*$

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