Lecture 18:  Recursively Defined Sets & Structural Induction
Midterm

- Friday in class

- Covers material up to end of ordinary induction

- Closed book, closed notes
  - will provide reference sheets

- No calculators
  - arithmetic is intended to be straightforward
  - (only a small point deduction anyway)
Midterm

• 5 problems covering:
  – Logic / English translation
  – Circuits / Boolean algebra / normal forms
  – Solving modular equations
  – Induction
  – Set theory
  – (all English proofs)

• 10 minutes per problem
  – write quickly
  – focus on the overall structure of the solution

Review Session
Thu at 1:30
in ECE 125
Last time: Recursive definitions of functions

• $0! = 1$; $(n + 1)! = (n + 1) \cdot n!$ for all $n \geq 0$.

• $F(0) = 0$; $F(n + 1) = F(n) + 1$ for all $n \geq 0$.

• $G(0) = 1$; $G(n + 1) = 2 \cdot G(n)$ for all $n \geq 0$.

• $H(0) = 1$; $H(n + 1) = 2^{H(n)}$ for all $n \geq 0$. 
Last time: Recursive definitions of functions

• Recursive functions allow general computation
  – saw examples not expressible with simple expressions

• So far, we have considered only simple data
  – inputs and outputs were just integers

• We need general data as well...
  – these will also be described recursively
  – will allow us to describe data of real programs
Recursive Definitions of Sets (Data)

Natural numbers
- Basis: $0 \in S$
- Recursive: If $x \in S$, then $x+1 \in S$

Even numbers
- Basis: $0 \in S$
- Recursive: If $x \in S$, then $x+2 \in S$
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.

We need the exclusion rule because otherwise $S=\mathbb{N}$ would satisfy the other two parts. However, we won’t always write it down on these slides.
Recursive Definitions of Sets

Natural numbers
Basis: \( 0 \in S \)
Recursive: If \( x \in S \), then \( x+1 \in S \)

Even numbers
Basis: \( 0 \in S \)
Recursive: If \( x \in S \), then \( x+2 \in S \)

Powers of 3:
Basis: \( 1 \in S \)
Recursive: If \( x \in S \), then \( 3x \in S \).

Basis: \( (0, 0) \in S, (1, 1) \in S \)
Recursive: If \( (n-1, x) \in S \) and \( (n, y) \in S \), then \( (n+1, x + y) \in S \).
Recursive Definitions of Sets

Natural numbers
Basis: \(0 \in S\)
Recursive: If \(x \in S\), then \(x + 1 \in S\)

Even numbers
Basis: \(0 \in S\)
Recursive: If \(x \in S\), then \(x + 2 \in S\)

Powers of 3:
Basis: \(1 \in S\)
Recursive: If \(x \in S\), then \(3x \in S\).

Basis: \((0, 0) \in S, (1, 1) \in S\)
Recursive: If \((n-1, x) \in S\) and \((n, y) \in S\), then \((n+1, x + y) \in S\).
Strings

• An *alphabet* $\Sigma$ is any finite set of characters

• The set $\Sigma^*$ of *strings* over the alphabet $\Sigma$
  
  – example: $\{0,1\}^*$ is the set of *binary strings*
    
    0, 1, 00, 01, 10, 11, 000, 001, ... and “”

• $\Sigma^*$ is defined recursively by
  
  – **Basis**: $\varepsilon \in \Sigma^*$ (\(\varepsilon\) is the empty string, i.e., “”)
  
  – **Recursive**: if $w \in \Sigma^*$, $a \in \Sigma$, then $wa \in \Sigma^*$
Palindromes

Palindromes are strings that are the same when read backwards and forwards

**Basis:**

\[ \varepsilon \text{ is a palindrome} \]
\[ \text{any } a \in \Sigma \text{ is a palindrome} \]

**Recursive step:**

If \( p \) is a palindrome,
then \( apa \) is a palindrome for every \( a \in \Sigma \)
Functions on Recursively Defined Sets (on $\Sigma^*$)

**Length:**
\[
\text{len}(\varepsilon) = 0
\]
\[
\text{len}(wa) = \text{len}(w) + 1 \text{ for } w \in \Sigma^*, \ a \in \Sigma
\]

**Concatenation:**
\[
x \cdot \varepsilon = x \text{ for } x \in \Sigma^*
\]
\[
x \cdot wa = (x \cdot w)a \text{ for } x \in \Sigma^*, \ a \in \Sigma
\]

**Reversal:**
\[
\varepsilon^R = \varepsilon
\]
\[
(wa)^R = a \cdot w^R \text{ for } w \in \Sigma^*, \ a \in \Sigma
\]

**Number of $c$'s in a string:**
\[
\#_c(\varepsilon) = 0
\]
\[
\#_c(wc) = \#_c(w) + 1 \text{ for } w \in \Sigma^*
\]
\[
\#_c(wa) = \#_c(w) \text{ for } w \in \Sigma^*, \ a \in \Sigma, \ a \neq c
\]
All Binary Strings with no 1’s before 0’s

Basis:
ε ∈ S

Recursive:
If x ∈ S, then 0 • x ∈ S
If x ∈ S, then x1 ∈ S

Those have no 1s before 0s. But is that every such string?
Rooted Binary Trees

• Basis: • is a rooted binary tree
Rooted Binary Trees

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

  If $T_1$ and $T_2$ are rooted binary trees, then $T_1 \cup T_2$ also is a rooted binary tree.

```
  T_1
     /
    /
   /
 T_2
```

then

```
  /
 T_1
```

also is a rooted binary tree.
Rooted Binary Trees in Java

```java
public static class BinaryTree {
    static BinaryTree LEAF = ...;
    public BinaryTree(
        BinaryTree T1, BinaryTree T2) {
        ...
    }
}

Create a binary tree with
    BinaryTree.LEAF or
    new BinaryTree(T1, T2)
```
Defining Functions on Rooted Binary Trees

- \( \text{size}(\bullet) = 1 \)

- \( \text{size}(T_1, T_2) = 1 + \text{size}(T_1) + \text{size}(T_2) \)

- \( \text{height}(\bullet) = 0 \)

- \( \text{height}(T_1, T_2) = 1 + \max\{\text{height}(T_1), \text{height}(T_2)\} \)
Functions on Rooted Binary Trees in Java

• \( \text{size}(\bullet) = 1 \)

• \( \text{size}(T) = 1 + \text{size}(T_1) + \text{size}(T_2) \)

public int size(BinaryTree T) {
    if (T == BinaryTree.LEAF) {
        return 1;
    } else {
        return 1 + size(T.left()) + size(T.right());
    }
}
Last time: Recursive definitions of functions

• Before, we considered only simple data
  – inputs and outputs were just integers

• Proved facts about those functions with induction
  – $n! \leq n^n$
  – $f_n < 2^n$ and $f_n \geq 2^{n/2-1}$

• How do we prove facts about functions that work with more complex (recursively defined) data?
  – we need a more sophisticated form of induction
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 $\forall x \in S, P(x)$
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 $\forall x \in S, P(x)$
Structural Induction vs. Ordinary Induction

Structural induction follows from ordinary induction:
Define $Q(n)$ to be “for all $x \in S$ that can be constructed in at most $n$ recursive steps, $P(x)$ is true.”

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 $k + 1 \in \mathbb{N}$
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.
**Claim:** Every element of $S$ is divisible by 3.

1. Let $P(x)$ be “$3 \mid x$”. We prove that $P(x)$ is true for all $x \in S$ by structural induction.

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.

1. Let $P(x)$ be “3|$x$”. We prove that $P(x)$ is true for all $x \in S$ by structural induction.
2. Base Case: $3|6$ and $3|15$ so $P(6)$ and $P(15)$ are true

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.

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

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.

1. Let \( P(x) \) be “3 | \( x \)”. We prove that \( P(x) \) is true for all \( x \in S \) 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 \( x,y \in S \)

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 \( x \in S \).

Basis: 6 \( \in \) \( S \); 15 \( \in \) \( S \);

Recursive: if \( x,y \in S \) then \( x+y \in S \)