Lecture 20: Regular expressions
Recap: Structural Induction

Consider a recursively defined set $S$:

- **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 $S$. 
Recap: Structural Induction

Consider a *recursively defined set* $S$:

• **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 $S$.

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 $\forall x \in S, P(x)$
Rooted Binary Trees

• **Basis:**
  • is a rooted binary tree

• **Recursive step:**

If $T_1$ and $T_2$ are rooted binary trees, then also is a rooted binary tree.
Defining Functions on Rooted Binary Trees

• size(●) = 1

• size \( \left( \begin{array}{c} T_1 \\ T_2 \end{array} \right) \) = 1 + size(\( T_1 \)) + size(\( T_2 \))

• height(●) = 0

• height \( \left( \begin{array}{c} T_1 \\ T_2 \end{array} \right) \) = 1 + max{height(\( T_1 \), height(\( T_2 \))}
Claim: For every rooted binary tree $T$, $size(T) \leq 2^{\text{height}(T)} + 1 - 1$
**Claim:** For every rooted binary tree $T$, $\text{size}(T) \leq 2^{\text{height}(T)} + 1 - 1$

1. Let $P(T)$ be “$\text{size}(T) \leq 2^{\text{height}(T)+1-1}$”. We prove $P(T)$ for all rooted binary trees $T$ by structural induction.
Claim: For every rooted binary tree $T$, $\text{size}(T) \leq 2^{\text{height}(T)} + 1 - 1$

1. Let $P(T)$ be “$\text{size}(T) \leq 2^{\text{height}(T)}+1−1$”. We prove $P(T)$ for all rooted binary trees $T$ by structural induction.
2. Base Case: $\text{size}(\bullet) = 1$, $\text{height}(\bullet) = 0$, and $2^{0+1}−1=2^1−1=1$ so $P(\bullet)$ is true.
**Claim:** For every rooted binary tree $T$, $\text{size}(T) \leq 2^{\text{height}(T)} + 1 - 1$

1. Let $P(T)$ be “$\text{size}(T) \leq 2^{\text{height}(T)} + 1 - 1$”. We prove $P(T)$ for all rooted binary trees $T$ by structural induction.

2. Base Case: $\text{size}(\bullet) = 1$, $\text{height}(\bullet) = 0$, and $2^{0+1} - 1 = 2^1 - 1 = 1$ so $P(\bullet)$ is true.

3. Inductive Hypothesis: Suppose that $P(T_1)$ and $P(T_2)$ are true for some rooted binary trees $T_1$ and $T_2$.

4. Inductive Step: Goal: Prove $P(\text{a tree})$. 

By defn, $\text{size}(\text{a tree}) = 1 + \text{size}(T_1) + \text{size}(T_2) \leq 1 + 2^{\text{height}(T_1)} + 1 - 1 + 2^{\text{height}(T_2)} + 1 - 1$ by IH for $T_1$ and $T_2$.

$\leq 2(2^{\text{max}(\text{height}(T_1), \text{height}(T_2))} + 1) - 1 \leq 2^{\text{height}(\text{a tree})} + 1 - 1$ which is what we wanted to show.

5. So, the $P(T)$ is true for all rooted bin. trees by structural induction.
Claim: For every rooted binary tree $T$, $\text{size}(T) \leq 2^{\text{height}(T)} + 1 - 1$

1. Let $P(T)$ be “$\text{size}(T) \leq 2^{\text{height}(T)+1} - 1$”. We prove $P(T)$ for all rooted binary trees $T$ by structural induction.

2. Base Case: $\text{size}(\bullet)=1$, $\text{height}(\bullet)=0$, and $2^{0+1} - 1 = 2^1 - 1 = 1$ so $P(\bullet)$ is true.

3. Inductive Hypothesis: Suppose that $P(T_1)$ and $P(T_2)$ are true for some rooted binary trees $T_1$ and $T_2$.

4. Inductive Step: 

   Goal: Prove $P(T)$.

   By defn, 
   
   $$\text{size}(T) = \text{size}(T_1) + \text{size}(T_2)$$

   $$\leq 1 + 2^{\text{height}(T_1)} + 1 + 2^{\text{height}(T_2)} + 1 - 1$$

   by IH for $T_1$ and $T_2$

   $$\leq 2^{\text{height}(T_1)} + 1 + 2^{\text{height}(T_2)} + 1 - 1$$

   $$\leq 2(2^{\max(\text{height}(T_1), \text{height}(T_2)) + 1}) - 1$$

   $$\leq 2(2^{\text{height}(T)}) - 1 \leq 2^{\text{height}(T)} + 1 - 1$$

   which is what we wanted to show.

5. So, the $P(T)$ is true for all rooted bin. trees by structural induction.
Languages: Sets of Strings

• Sets of strings that satisfy special properties are called languages. Examples:
  – English sentences
  – Syntactically correct Java/C/C++ programs
  – $\Sigma^* =$ All strings over alphabet $\Sigma$
  – Palindromes over $\Sigma$
  – Binary strings that don’t have a 0 after a 1
  – Legal variable names, keywords in Java/C/C++
  – Binary strings with an equal # of 0’s and 1’s
Regular Expressions

**Regular expressions over** $\Sigma$

- **Basis:**
  - $\varepsilon$ is a regular expression
  - $\emptyset$ is a regular expression
  - $a$ is a regular expression for any $a \in \Sigma$

- **Recursive step:**
  - If $A$ and $B$ are regular expressions then so are:
    - $A \cup B$
    - $AB$
    - $A^*$
Each Regular Expression is a “pattern”

\( \varepsilon \) matches the **empty string**
\( \emptyset \) does not match any string
\( a \) matches the one character string \( a \)
\( A \cup B \) matches all strings that either \( A \) matches or \( B \) matches (or both)
\( AB \) matches all strings that have a first part that \( A \) matches followed by a second part that \( B \) matches
\( A^* \) matches all strings that have any number of strings (even 0) that \( A \) matches, one after another
Examples

001*
Examples

001*

{00, 001, 0011, 00111, ...}
Examples

\[(0 \cup 1) \ 0 \ (0 \cup 1) \ 0\]

\[(0*1*)*\]
Examples

\[(0 \cup 1) 0 (0 \cup 1) 0\]

\[\{0000, 0010, 1000, 1010\}\]

\[(0*1*)^*\]

All binary strings
Examples

\((0 \cup 1)^* 0110 (0 \cup 1)^*\)

\((00 \cup 11)^* (01010 \cup 10001) (0 \cup 1)^*\)
Examples

\[(0 ∪ 1)^* \ 0110 \ (0 ∪ 1)^*\]

Binary strings that contain “0110”

\[(00 ∪ 11)^* \ (01010 ∪ 10001) \ (0 ∪ 1)^*\]

Binary strings that begin with pairs of characters followed by “01010” or “10001”
Lecture 20 Activity

• You will be assigned to **breakout rooms**. Please:
  • Introduce yourself
  • Choose someone to share screen, showing this PDF

---

Which of these is the language of $0^*1^*$?

- a) The set of all binary strings with any 0s coming before all 1s
- b) The set of all binary strings starting with 0 and ending with 1
- c) The set of all binary strings
- d) The set of all binary strings where every 0 is followed by a 1.

---

Fill out a poll everywhere for **Activity Credit**!
Go to [pollev.com/thomas311](http://pollev.com/thomas311) and login with your UW identity
Regular Expressions in Practice

• Used to define the “tokens”: e.g., legal variable names, keywords in programming languages and compilers

• Legal variable names in Java are

\[(a \cup \cdots \cup z \cup A \cup \cdots \cup Z \cup $ \cup _) (a \cup \cdots \cup z \cup A \cup \cdots \cup Z \cup $ \cup _ \cup 1 \cup \cdots \cup 9)^*\]

• Used in **grep**, a program that does pattern matching searches in UNIX/LINUX

• Pattern matching using regular expressions is an essential feature of PHP

• We can use regular expressions in programs to process strings!
Regular Expressions in Java

• Pattern p = Pattern.compile("a*b");
• Matcher m = p.matcher("aaaaaab");
• boolean b = m.matches();

[01] a 0 or a 1 ^ start of string $ end of string
[0-9] any single digit \ . period \ , comma \ - minus
. any single character
ab a followed by b (AB)
(a | b) a or b (A ∪ B)
a? zero or one of a (A ∪ ε)
a* zero or more of a A*
a+ one or more of a AA*

• e.g. ^\[\-+]?[0-9]* (\ . | \ \ ,)? [0-9]+$ General form of decimal number e.g. 9.12 or -9,8 (Europe)
Limitations of Regular Expressions

• Not all languages can be specified by regular expressions

• Even some easy things like
  – Palindromes
  – Strings with equal number of 0’s and 1’s

• But also more complicated structures in programming languages
  – Matched parentheses
  – Properly formed arithmetic expressions
  – etc.