Lecture 19: Structural induction and regular expressions
• An *alphabet* \( \Sigma \) is any finite set of characters.

  e.g. \( \Sigma = \{0, 1\} \) or \( \Sigma = \{A, B, C, \ldots X, Y, Z\} \) or

\[
\Sigma = \text{\texttt{\textasciitilde}}
\]

• The set \( \Sigma^* \) of *strings* over the alphabet \( \Sigma \) is defined by
  – **Basis:** \( \varepsilon \in \Sigma^* \) (\( \varepsilon \) is the empty string)
  – **Recursive:** if \( w \in \Sigma^* \), \( a \in \Sigma \), then \( wa \in \Sigma^* \)

\[
\Sigma^* = \Sigma^* \setminus \varepsilon \quad \text{Basis: } \forall a \in \Sigma, \ a \in \Sigma^* \\
\text{Recur: } \text{if } w \in \Sigma^* \text{ and } a \in \Sigma \Rightarrow wa \in \Sigma^*
\]
function definitions on recursively defined sets

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

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

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

• Basis:
  • is a rooted binary tree

• Recursive step:
  If \( T_1 \) and \( T_2 \) are rooted binary trees, then so is:

\[
\begin{align*}
T_1 & \quad \text{and} \quad T_2 \\
\text{then so is:} & \quad T_1 \quad \text{and} \quad T_2
\end{align*}
\]
defining a function on rooted binary trees

• size(•) = 1

• size \((T_1, T_2)\) = 1 + size\((T_1)\) + size\((T_2)\)

• height(•) = 0

• height \((T_1, T_2)\) = 1 + max\{height\((T_1)\), height\((T_2)\)\}
How to prove ∀ 𝑥 ∈ 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 ∀ 𝑥 ∈ 𝑆, 𝑃(𝑥)
Let $S$ be a set of strings over $\Sigma = \{a, b\}$ defined by

**Basis:** $a \in S$

**Recursive:**

- If $w \in S$ then $wa \in S$ and $wba \in S$
- If $u, v \in S$ then $uv \in S$

**Claim:** If $w \in S$ then $w$ has more $a$’s than $b$’s.

$p(w) = "$w has more a’s than b’s."

**Base Case:** Show $p(a)$ holds.

$\#_a(a) = 1 > \#_b(a)$

**IH:** For some $w, u, v \in S$, $p(w), p(u), p(v)$ hold.

**IS:**

$\#_a(wa) = \#_a(w) + 1 > \#_b(w) + 1 > \#_b(wa) = \#_b(wa) => p(wa)$ holds.

$\#_a(wba) = \#_a(w) + 1 > \#_b(w) + 1 = \#_b(wba) => p(wba)$ holds.
proof continued?

\[ \#_a(uv) = \#_a(u) + \#_a(v) > \#_b(u) + \#_b(v) \]

\[ \uparrow_{\text{IH}} \quad = \#_b(uv) \]

\[ \Rightarrow P(uv) \text{ holds.} \]
**prove:** \( \text{len}(x \cdot y) = \text{len}(x) + \text{len}(y) \) for all \( x, y \in \Sigma^* \)

Let \( P(y) \) be “\( \text{len}(x \cdot y) = \text{len}(x) + \text{len}(y) \) for all \( x \in \Sigma^* \)

**Base Case:** \( P(\varepsilon) \) holds.

\[
\text{len}(x \cdot \varepsilon) = \text{len}(x) = \text{len}(x) + \text{len}(\varepsilon)
\]

\( \text{def of } \text{len} \)

**IH:** for some \( w \in \Sigma^* \), \( P(w) \) holds.

**IS:** Goal: \( P(wa) \) holds for any \( a \in \Sigma \).

Fix \( x \in \Sigma^* \)

\[
\text{len}(x \cdot wa) = \text{len}((x \cdot w)a) = \text{len}(x \cdot w) + 1
\]

\( \text{def of } \text{len} \)

\[\text{IH} \rightarrow \]

\[
= \text{len}(x) + \text{len}(w) + 1
\]

\( \text{def of } \text{len} \)

\[
= \text{len}(x) + \text{len}(wa)
\]

\( \text{IH} \rightarrow \)

\[
\Rightarrow P(wa) \text{ holds}
\]

**Conclusion:** \( P(y) \) holds \( \forall y \in \Sigma^* \)

**Length:**

\[
\text{len}(\varepsilon) = 0;
\]

\[
\text{len}(wa) = 1 + \text{len}(w); \text{ for } w \in \Sigma^*, a \in \Sigma
\]
defining a function on rooted binary trees

- size(•) = 1

- size \( \left( \begin{array}{c} \mathcal{T}_1 \\ \mathcal{T}_2 \end{array} \right) \) = 1 + size(\mathcal{T}_1) + size(\mathcal{T}_2)

- height(•) = 0

- height \( \left( \begin{array}{c} \mathcal{T}_1 \\ \mathcal{T}_2 \end{array} \right) \) = 1 + \( \max\{ \text{height}(\mathcal{T}_1), \text{height}(\mathcal{T}_2) \} \)
Claim: For every rooted binary tree $T$, $\text{size}(T) \leq 2^{\text{height}(T)+1} - 1$

Let $P(T) = "\text{size}(T) \leq 2^{\text{height}(T)+1} - 1"$

Base Case: $P(\cdot)$ holds

\[ \text{size}(\cdot) = 1 \leq 2^{0+1} - 1 = 2 - 1 = 1 \]

IH: $P(T_1), P(T_2)$ hold for some $T_1, T_2$

IS: Goal: $P(T_3)$ holds

\[ \text{size}(T_3) = 1 + \text{size}(T_1) + \text{size}(T_2) \leq 1 + 2^{\text{height}(T_1) + 1} - 1 + 2^{\text{height}(T_2) + 1} - 1 \\
= 2 \cdot 2 \cdot 2^{\max\{\text{height}(T_1), \text{height}(T_2)\} + 1} - 1 \\
\leq 2 \cdot 2 \cdot 2 \cdot 2^{\text{height}(T_3) + 1} - 1 = 2 \cdot 2^{\text{height}(T_3) + 1} - 1 = 2^{\text{height}(T_3) + 2} - 1 \]

\[ \Rightarrow P(T_3) \]
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