COSE212: Programming Languages

Lecture 2 — Inductive Definitions (2)

Hakjoo Oh
2022 Fall

Contents

- More examples of inductive definitions
  - natural numbers, strings, booleans
  - lists, binary trees
  - arithmetic expressions, propositional logic
- Structural induction
  - three example proofs

Natural Numbers

The set of natural numbers:

$$\mathbb{N} = \{0, 1, 2, 3, \ldots\}$$

is inductively defined:

$$\frac{}{0} \qquad \frac{n}{n+1}$$

The inference rules can be expressed by a grammar:

$$n \rightarrow 0 \mid n + 1$$

Interpretation:

- 0 is a natural number.
- If n is a natural number then so is $n + 1$.

Strings

The set of strings over alphabet $\{\mathtt{a}, \ldots, \mathtt{z}\}$, e.g., ϵ, a, b, ..., z, aa, ab, ..., az, ba, ... az, aaa, ..., zzz, and so on. Inference rules:

$$\frac{}{\epsilon} \qquad \frac{\alpha}{\mathtt{a}\alpha} \qquad \frac{\alpha}{\mathtt{b}\alpha} \quad \cdots \qquad \frac{\alpha}{\mathtt{z}\alpha}$$

or simply,

$$\frac{}{\epsilon} \qquad \frac{\alpha}{x\alpha}\ x \in \{\mathtt{a}, \ldots, \mathtt{z}\}$$

In grammar:

$$\begin{array}{rcl} \alpha & \rightarrow & \epsilon \\ & \mid & x\alpha \quad (x \in \{\mathtt{a}, \ldots, \mathtt{z}\}) \end{array}$$

Boolean Values

The set of boolean values:

$$\mathbb{B} = \{\boldsymbol{true}, \boldsymbol{false}\}.$$

If a set is finite, just enumerate all of its elements by axioms:

$$\frac{}{\boldsymbol{true}} \qquad \frac{}{\boldsymbol{false}}$$

In grammar:

$$\boldsymbol{b} \rightarrow \boldsymbol{true} \mid \boldsymbol{false}$$

Lists

Examples of lists of integers:

1. **nil**
2. $\mathbf{14} \cdot$ **nil**
3. $\mathbf{3} \cdot \mathbf{14} \cdot$ **nil**
4. $\mathbf{-7} \cdot \mathbf{3} \cdot \mathbf{14} \cdot$ **nil**

Inference rules:

$$\frac{}{\textbf{nil}} \qquad \frac{l}{n \cdot l}\; n \in \mathbb{Z}$$

In grammar:

$$\begin{array}{rcll} l & \rightarrow & \textbf{nil} & \\ & \mid & n \cdot l & (n \in \mathbb{Z}) \end{array}$$

Lists

A proof that $-\mathbf{7} \cdot \mathbf{3} \cdot \mathbf{14} \cdot \textbf{nil}$ is a list of integers:

$$\frac{\dfrac{\dfrac{\dfrac{}{\textbf{nil}} \quad \mathbf{14} \in \mathbb{Z}}{\mathbf{14} \cdot \textbf{nil}} \quad \mathbf{3} \in \mathbb{Z}}{\mathbf{3} \cdot \mathbf{14} \cdot \textbf{nil}} \quad -\mathbf{7} \in \mathbb{Z}}{-\mathbf{7} \cdot \mathbf{3} \cdot \mathbf{14} \cdot \textbf{nil}}$$

The proof tree is also called *derivation tree* or *deduction tree*.

Binary Trees

Examples of binary trees:

1. $\textbf{leaf}$
2. $(2, \textbf{leaf}, \textbf{leaf})$
3. $(1, (2, \textbf{leaf}, \textbf{leaf}), \textbf{leaf})$
4. $(1, (2, \textbf{leaf}, \textbf{leaf}), (3, (4, \textbf{leaf}, \textbf{leaf}), \textbf{leaf}))$

Inference rules:

$$\frac{}{\textbf{leaf}} \qquad \frac{t_1 \quad t_2}{(n, t_1, t_2)}\; n \in \mathbb{Z}$$

In grammar:

$$\begin{array}{rcl} t & \rightarrow & \textbf{leaf} \\ & \mid & (n, t, t) \quad (n \in \mathbb{Z}) \end{array}$$

Binary Trees

A proof that

$$(1, (2, \textsf{leaf}, \textsf{leaf}), (3, (4, \textsf{leaf}, \textsf{leaf}), \textsf{leaf}))$$

is a binary tree:

$$\frac{\dfrac{\overline{\textsf{leaf}}}{(2, \textsf{leaf}, \textsf{leaf})}\ 2 \in \mathbb{Z} \quad \dfrac{\dfrac{\overline{\textsf{leaf}}}{(4, \textsf{leaf}, \textsf{leaf})}\ 4 \in \mathbb{Z}}{(3, (4, \textsf{leaf}, \textsf{leaf}), \textsf{leaf})}\ 3 \in \mathbb{Z}}{(1, (2, \textsf{leaf}, \textsf{leaf}), (3, (4, \textsf{leaf}, \textsf{leaf}), \textsf{leaf}))}\ 1 \in \mathbb{Z}$$

Binary Trees: a different version

Binary tree examples: $\mathbf{1}$, $(\mathbf{1}, \textbf{nil})$, $(\mathbf{1}, \mathbf{2})$, $((\mathbf{1}, \mathbf{2}), \textbf{nil})$, $((\mathbf{1}, \mathbf{2}), (\mathbf{3}, \mathbf{4}))$.
Inference rules:

$$\frac{}{\boldsymbol{n}}\ \boldsymbol{n} \in \mathbb{Z} \qquad \frac{\boldsymbol{t}}{(\boldsymbol{t}, \textbf{nil})} \qquad \frac{\boldsymbol{t}}{(\textbf{nil}, \boldsymbol{t})} \qquad \frac{\boldsymbol{t_1} \quad \boldsymbol{t_2}}{(\boldsymbol{t_1}, \boldsymbol{t_2})}$$

In grammar:

$$\begin{array}{rcl} \boldsymbol{t} & \rightarrow & \boldsymbol{n} \quad (\boldsymbol{n} \in \mathbb{Z}) \\ & | & (\boldsymbol{t}, \textbf{nil}) \\ & | & (\textbf{nil}, \boldsymbol{t}) \\ & | & (\boldsymbol{t}, \boldsymbol{t}) \end{array}$$

A proof that $((\mathbf{1}, \mathbf{2}), (\mathbf{3}, \textbf{nil}))$ is a binary tree:

$$\frac{\dfrac{\overline{\mathbf{1}} \quad \overline{\mathbf{2}}}{(\mathbf{1}, \mathbf{2})} \qquad \dfrac{\overline{\mathbf{3}}}{(\mathbf{3}, \textbf{nil})}}{((\mathbf{1}, \mathbf{2}), (\mathbf{3}, \textbf{nil}))}$$

Expressions

Expression examples: $\mathbf{2}$, $-\mathbf{2}$, $\mathbf{1}+\mathbf{2}$, $\mathbf{1}+(\mathbf{2}*(-\mathbf{3}))$, etc.
Inference rules:

$$\frac{}{n}\; n \in \mathbb{Z} \qquad \frac{e}{-e} \qquad \frac{e_1 \quad e_2}{e_1 + e_2} \qquad \frac{e_1 \quad e_2}{e_1 * e_2} \qquad \frac{e}{(e)}$$

In grammar:

$$\begin{array}{rcll} e & \to & n & (n \in \mathbb{Z}) \\ & \mid & -e & \\ & \mid & e + e & \\ & \mid & e * e & \\ & \mid & (e) & \end{array}$$

Example:

$$\cfrac{\cfrac{}{\mathbf{1}} \quad \cfrac{\cfrac{\cfrac{}{\mathbf{2}} \quad \cfrac{\cfrac{\cfrac{}{\mathbf{3}}}{-\mathbf{3}}}{(-\mathbf{3})}}{\mathbf{2}*(-\mathbf{3})}}{(\mathbf{2}*(-\mathbf{3}))}}{\mathbf{1}+(\mathbf{2}*(-\mathbf{3}))}$$

Propositional Logic

Examples:

- $\boldsymbol{T}$, $\boldsymbol{F}$
- $\boldsymbol{T} \land \boldsymbol{F}$
- $\boldsymbol{T} \lor \boldsymbol{F}$
- $(\boldsymbol{T} \land \boldsymbol{F}) \land (\boldsymbol{T} \lor \boldsymbol{F})$
- $\boldsymbol{T} \Rightarrow (\boldsymbol{F} \Rightarrow \boldsymbol{T})$

Propositional Logic

Syntax:

$$\begin{array}{rcl} f & \rightarrow & T \mid F \\ & \mid & \neg f \\ & \mid & f \wedge f \\ & \mid & f \vee f \\ & \mid & f \Rightarrow f \end{array}$$

Semantics ($[\![f]\!]$):

$$\begin{array}{rcl} [\![T]\!] & = & \mathit{true} \\ [\![F]\!] & = & \mathit{false} \\ [\![\neg f]\!] & = & \text{not } [\![f]\!] \\ [\![f_1 \wedge f_2]\!] & = & [\![f_1]\!] \text{ andalso } [\![f_2]\!] \\ [\![f_1 \vee f_2]\!] & = & [\![f_1]\!] \text{ orelse } [\![f_2]\!] \\ [\![f_1 \Rightarrow f_2]\!] & = & [\![f_1]\!] \text{ implies } [\![f_2]\!] \end{array}$$

Propositional Logic

$$\begin{aligned}
[\![(T \wedge (T \vee F)) \Rightarrow F]\!] &= [\![T \wedge (T \vee F)]\!] \text{ implies } [\![F]\!] \\
&= ([\![T]\!] \text{ andalso } [\![T \vee F]\!]) \text{ implies } \mathit{false} \\
&= (\mathit{true} \text{ andalso } ([\![T]\!] \text{ orelse } [\![F]\!])) \text{ implies } \mathit{false} \\
&= (\mathit{true} \text{ andalso } (\mathit{true} \text{ orelse } \mathit{false})) \text{ implies } \mathit{false} \\
&= \mathit{false}
\end{aligned}$$

Structural Induction

A technique for proving properties about inductively defined sets.

To prove that a proposition $P(s)$ is true for all structures s, prove the following:

1. (Base case) P is true on simple structures (those without substructures)
2. (Inductive case) If P is true on the substructures of s, then it is true on s itself. The assumption is called *induction hypothesis (I.H.)*.

Example 1

Let S be the set defined by the following inference rules:

$$\frac{}{3} \qquad \frac{x \quad y}{x+y}$$

Prove that for all $x \in S$, x is divisible by 3.
Proof. By structural induction.

- (Base case) The base case is when x is 3. Obviously, x is divisible by 3.
- (Inductive case) The induction hypothesis (I.H.) is

  $$x \text{ is divisible by } 3, \qquad y \text{ is divisible by } 3.$$

  Let $x = 3k_1$ and $y = 3k_2$. Using I.H., we derive

  $$x + y \text{ is divisible by } 3$$

  as follows:

  $$\begin{array}{rcll} x + y & = & 3k_1 + 3k_2 & \cdots \text{by I.H.} \\ & = & 3(k_1 + k_2) & \end{array}$$

□

Example 2

Let S be the set defined by the following inference rules:

$$\frac{}{()} \qquad \frac{x}{(x)} \qquad \frac{x \quad y}{xy}$$

Prove that every element of the set has the same number of ('s and)'s.
Proof Restate the claim formally:

$$\text{If } x \in S \text{ then } l(x) = r(x)$$

where $l(x)$ and $r(x)$ denote the number of ('s and)'s, respectively:

$$\begin{array}{rcl} l(()) & = & 1 \\ l((x)) & = & l(x) + 1 \\ l(xy) & = & l(x) + l(y) \end{array} \qquad \begin{array}{rcl} r(()) & = & 1 \\ r((x)) & = & r(x) + 1 \\ r(xy) & = & r(x) + r(y) \end{array}$$

We prove it by structural induction:

- (Base case): The base case is when $x = ()$. Then $l(x) = 1 = r(x)$.

Example 2

- (Inductive case): There are two inductive cases:

$$\frac{x}{(x)} \qquad \frac{x \quad y}{xy}$$

Induction hypotheses (I.H.):

$$l(x) = r(x), \qquad l(y) = r(y).$$

  - The first case. We prove $l((x)) = r((x))$:

$$\begin{array}{rcll} l((x)) & = & l(x) + 1 & \cdots \text{by definition of } l((x)) \\ & = & r(x) + 1 & \cdots \text{by I.H.} \\ & = & r((x)) & \cdots \text{by definition of } r((x)) \end{array}$$

  - The second case. We prove $l(xy) = r(xy)$:

$$\begin{array}{rcll} l(xy) & = & l(x) + l(y) & \cdots \text{by definition of } l(xy) \\ & = & r(x) + r(y) & \cdots \text{by I.H.} \\ & = & r(xy) & \cdots \text{by definition of } r(xy) \end{array}$$

□

Example 3

Let T be the set of binary trees:

$$\frac{}{\textbf{leaf}} \qquad \frac{t_1 \quad t_2}{(n, t_1, t_2)}\ n \in \mathbb{Z}$$

Prove that for all such trees, the number of leaves is always one more than the number of internal nodes.

Proof. Restate the claim more formally:

$$\text{If } t \in T \text{ then } l(t) = i(t) + 1$$

where $l(t)$ and $i(t)$ denote the number of leaves and internal nodes, respectively:

$$\begin{array}{rcl} l(\textbf{leaf}) & = & 1 \\ l(n, t_1, t_2) & = & l(t_1) + l(t_2) \end{array} \qquad \begin{array}{rcl} i(\textbf{leaf}) & = & 0 \\ i(n, t_1, t_2) & = & i(t_1) + i(t_2) + 1 \end{array}$$

We prove it by structural induction:

- (Base case): The base case is when $t = \textbf{leaf}$, where $l(t) = 1$ and $i(t) = 0$.
- (Inductive case): The induction hypothesis:

$$l(t_1) = i(t_1) + 1, \qquad l(t_2) = i(t_2) + 1$$

Using I.H., we prove $l((n, t_1, t_2)) = i((n, t_1, t_2)) + 1$:

$$\begin{array}{rcll} l((n, t_1, t_2)) & = & l(t_1) + l(t_2) & \text{definition of of } l \\ & = & i(t_1) + 1 + i(t_2) + 1 & \text{by induction hypothesis} \\ & = & i(n, t_1, t_2) + 1 & \text{definition of } i \end{array}$$

Summary

- Computer science is full of inductive definitions.
  - primitive values: booleans, characters, integers, strings, etc
  - compound values: lists, trees, graphs, etc
  - language syntax and semantics
- Structural induction
  - a general technique for reasoning about inductively defined sets