Untyped Arithmetic Expressions

Harley Eades

Syntax +
Inductive Definitions +
Induction Principles

Syntax

t ::=
| true
| false
| if t then t else t
| 0
| succ t
| pred t
| iszero t

Syntax

$t ::=$

$\mid$ true	
$\mid$ false	
$\mid$ if t then t else t	
$\mid$ 0	zero
$\mid$ succ t	successor
$\mid$ pred t	predecessor
$\mid$ iszero t	zero test

Unary Natural Numbers

n	p
0	0
1	succ 0
2	succ succ 0
3	succ succ succ 0
$\vdots$	$\vdots$
n	succ^n 0

Unary Natural Numbers

Predecessor

$$\mathsf{pred}\, 0 = 0$$
$$\mathsf{pred}\, (\mathsf{succ}\, n) = n$$

Unary Natural Numbers

Zero Test

$$\mathsf{iszero}\,0 = \mathsf{true}$$
$$\mathsf{iszero}\,(\mathsf{succ}\,n) = \mathsf{false}$$

Exercise

How do we define addition?

$$\text{add}\, p_1\, p_2 = \, ?$$

Exercise

How do we define addition?

$$\text{add}\,0\,p_2 = p_2$$
$$\text{add}\,(\text{succ}\,p_1)\,p_2 = \text{succ}\,(\text{add}\,p_1\,p_2)$$

Exercise

How do we define subtraction?

$$\mathsf{sub}\, p_1\, p_2 = 0$$

Exercise

How do we define subtraction?

$$\mathsf{sub}\, p_1\, 0 = p_1$$
$$\mathsf{sub}\, p_1\, (\mathsf{succ}\, p_2) = \mathsf{pred}\, p_1$$

Exercise

How do we define multiplication?

$$\text{mult}\, p_1\, p_2 = ?$$

Exercise

How do we define multiplication?

$$\text{mult}\,0\,p_2 = 0$$
$$\text{mult}\,(\text{succ}\,0)\,p_2 = p_2$$
$$\text{mult}\,(\text{succ}\,p_1)\,p_2 = \text{add}\,(\text{mult}\,p_1\,p_2)\,p_2$$

Terms

t ::=

| true

| false

| if t then t else t

| 0

| succ t

| pred t

| iszero t

We call the programs generated by t <u>terms</u>.

Terms

Definition: The set of <u>terms</u> is the smallest set $\mathcal{T}$ such that:

1. $\{\text{true}, \text{false}\} \subseteq \mathcal{T}$;
2. if $t \in \mathcal{T}$, then $\{\text{succ}\ t, \text{pred}\ t\ \text{iszero}\ t\} \subseteq \mathcal{T}$;
3. if $\{t_1, t_2, t_3\} \subseteq \mathcal{T}$, then if t_1 then t_2 else $t_3 \in \mathcal{T}$

Terms

$\boxed{\text{An inductive definition}}$

Definition: The set of terms is the smallest set $\mathcal{T}$ such that:

1. $\{\text{true}, \text{false}\} \subseteq \mathcal{T}$;
2. if $t \in \mathcal{T}$, then $\{\text{succ}\ t, \text{pred}\ t\ \text{iszero}\ t\} \subseteq \mathcal{T}$;
3. if $\{t_1, t_2, t_3\} \subseteq \mathcal{T}$, then $\text{if}\ t_1\ \text{then}\ t_2\ \text{else}\ t_3 \in \mathcal{T}$

Terms

<u>Smallest Set</u>: For every other set, $\mathcal{T}'$, satisfying conditions 1-3, it is the case that $\mathcal{T} \subseteq \mathcal{T}'$.

Definition: The set of <u>terms</u> is the smallest set $\mathcal{T}$ such that:

1. $\{\mathsf{true}, \mathsf{false}\} \subseteq \mathcal{T}$;
2. if $t \in \mathcal{T}$, then $\{\mathsf{succ}\ t, \mathsf{pred}\ t\ \mathsf{iszero}\ t\} \subseteq \mathcal{T}$;
3. if $\{t_1, t_2, t_3\} \subseteq \mathcal{T}$, then $\mathsf{if}\ t_1\ \mathsf{then}\ t_2\ \mathsf{else}\ t_3 \in \mathcal{T}$

Inductive Definitions

- A <u>judgment</u> is some predicate.
- <u>Inference rules</u> define when a judgment $\mathscr{J}$ is true.
- Given a judgment $\mathscr{J}'$, we show that $\mathscr{J}'$ holds by giving a <u>derivation tree</u> of the inference rules that begins with $\mathscr{J}'$ and all branches end with axioms.

Inference Rules

Axiom

$$\frac{}{\mathscr{J}}\ \text{Name}$$

Compound

$$\frac{\mathscr{J}_1 \quad \cdots \quad \mathscr{J}_i}{\mathscr{J}}\ \text{Name}$$

Inference Rules

Axiom

$$\frac{}{\mathscr{J}}\ \text{Name}$$

Compound

$$\frac{\mathscr{J}_1 \quad \cdots \quad \mathscr{J}_i}{\mathscr{J}}\ \text{Name}$$

if $\mathscr{J}_1 \wedge \cdots \wedge \mathscr{J}_i$, then $\mathscr{J}$

Inductive Definition of Terms

$\underline{t \in \mathscr{T}}$ is our judgment defined by:

$$\frac{}{\text{true} \in \mathscr{T}}\;\text{T} \qquad \frac{t_1 \in \mathscr{T}}{\text{succ}\, t_1 \in \mathscr{T}}\;\text{succ}$$

$$\frac{}{\text{false} \in \mathscr{T}}\;\text{F} \qquad \frac{t_1 \in \mathscr{T}}{\text{pred}\, t_1 \in \mathscr{T}}\;\text{pred} \qquad \frac{t_1 \in \mathscr{T} \quad t_2 \in \mathscr{T} \quad t_3 \in \mathscr{T}}{\text{if}\; t_1 \;\text{then}\; t_2 \;\text{else}\; t_3 \in \mathscr{T}}\;\text{if}$$

$$\frac{}{0 \in \mathscr{T}}\;0 \qquad \frac{t_1 \in \mathscr{T}}{\text{iszero}\, t_1 \in \mathscr{T}}\;\text{iszero}$$

Inductive Definition: Derivations

Goal Directed Proof

Start with what you're trying to prove, the goal, and then reduce it into one or more subgoals; repeat this process on each subgoal until all subgoals reduce to axioms.

Inductive Definition: Derivations

All our goals will be judgments. Suppose $\mathscr{J}$ is some judgment. Then we <u>derive</u> $\mathscr{J}$ using the inference rules defining when judgments of the form of $\mathscr{J}$ hold using a <u>derivation tree</u>.

Inductive Definition: Derivations

We call $\mathscr{D}$ as <u>derivation tree</u> of judgment $\mathscr{J}$ if:

1. $\mathscr{D}$ is an axiom whose conclusion matches $\mathscr{J}$;
2. $\mathscr{D}$ has the form:

$$\frac{\mathscr{D}_1 \quad \cdots \quad \mathscr{D}_i}{\mathscr{J}}\,\text{Name}$$

where $\mathscr{D}_1, \ldots, \mathscr{D}_i$ are derivation trees and "Name" is one of the inference rules of $\mathscr{J}$ where the premises match the conclusions of $\mathscr{D}_1, \ldots, \mathscr{D}_i$ respectively.

Inductive Definition of Terms

$\underline{t \in \mathscr{T}}$ is our judgment defined by:

$$\frac{}{\text{true} \in \mathscr{T}}\;\text{T} \qquad \frac{t_1 \in \mathscr{T}}{\text{succ}\; t_1 \in \mathscr{T}}\;\text{succ}$$

$$\frac{}{\text{false} \in \mathscr{T}}\;\text{F} \qquad \frac{t_1 \in \mathscr{T}}{\text{pred}\; t_1 \in \mathscr{T}}\;\text{pred} \qquad \frac{t_1 \in \mathscr{T} \quad t_2 \in \mathscr{T} \quad t_3 \in \mathscr{T}}{\text{if}\; t_1 \;\text{then}\; t_2 \;\text{else}\; t_3 \in \mathscr{T}}\;\text{if}$$

$$\frac{}{0 \in \mathscr{T}}\;\text{0} \qquad \frac{t_1 \in \mathscr{T}}{\text{iszero}\; t_1 \in \mathscr{T}}\;\text{iszero}$$

Inductive Definition of Terms

$$\text{if iszero } 0 \text{ then true else false} \in \mathscr{T}$$

Inductive Definition of Terms

$$\dfrac{\text{iszero} \in \mathscr{T} \qquad \text{true} \in \mathscr{T} \qquad \text{false} \in \mathscr{T}}{\text{if iszero } 0 \text{ then true else false} \in \mathscr{T}}\ \text{if}$$

Inductive Definition of Terms

$$\dfrac{\dfrac{0 \in \mathscr{T}}{\text{iszero}\, 0 \in \mathscr{T}}\;\text{iszero} \qquad \text{true} \in \mathscr{T} \qquad \text{false} \in \mathscr{T}}{\text{if iszero}\, 0 \text{ then true else false} \in \mathscr{T}}\;\text{if}$$

Inductive Definition of Terms

$$\dfrac{\dfrac{\dfrac{}{0 \in \mathscr{T}}\;\textsf{0}}{\text{iszero}\,0 \in \mathscr{T}}\;\textsf{iszero} \qquad \text{true} \in \mathscr{T} \qquad \text{false} \in \mathscr{T}}{\text{if iszero}\,0\text{ then true else false} \in \mathscr{T}}\;\textsf{if}$$

Inductive Definition of Terms

$$\dfrac{\dfrac{\dfrac{}{0 \in \mathscr{T}}\,\textsf{0}}{\textsf{iszero}\ 0 \in \mathscr{T}}\,\textsf{iszero} \qquad \dfrac{}{\textsf{true} \in \mathscr{T}}\,\textsf{true} \qquad \textsf{false} \in \mathscr{T}}{\textsf{if iszero}\ 0\ \textsf{then true else false} \in \mathscr{T}}\,\textsf{if}$$

Derivations using Terms

$$\cfrac{\cfrac{\cfrac{}{0 \in \mathscr{T}}\;\textsf{0}}{\textsf{iszero}\; 0 \in \mathscr{T}}\;\textsf{iszero} \qquad \cfrac{}{\textsf{true} \in \mathscr{T}}\;\textsf{true} \qquad \cfrac{}{\textsf{false} \in \mathscr{T}}\;\textsf{false}}{\textsf{if iszero}\; 0\; \textsf{then true else false} \in \mathscr{T}}\;\textsf{if}$$

Exercise

Suppose our judgment is

$$t \in \mathscr{T}_{\mathbb{N}}$$

where $\mathscr{T}_{\mathbb{N}}$ is the subset of terms corresponding to the natural numbers.

i. Define inference rules for when this judgment holds.
ii. Derive: $\text{succ pred succ } 0 \in \mathscr{T}_{\mathbb{N}}$

Induction on Terms

What can we say about t if we know $t \in \mathscr{T}$?

Induction on Terms

What can we say about t if we know $t \in \mathscr{T}$?

1. t is a constant

Induction on Terms

What can we say about t if we know $t \in \mathscr{T}$?

1. t is a constant; or
2. $t \in \{\mathsf{succ}\, t_1, \mathsf{pred}\, t_1, \mathsf{iszero}\, t_1\}$ for some smaller term t_1

Induction on Terms

What can we say about t if we know $t \in \mathcal{T}$?

1. t is a constant; or
2. $t \in \{\text{succ}\, t_1, \text{pred}\, t_1, \text{iszero}\, t_1\}$ for some smaller term t_1; or
3. $t = \text{if}\, t_1, \text{then}\, t_2\, \text{else}\, t_3$ for some smaller terms t_1, t_2, t_3.

Induction on Terms

This is an important reasoning principle!

1. Inductive definitions of functions.
2. Inductive proofs.

What can we say about t if we know $t \in \mathcal{T}$?

1. t is a constant; or
2. $t \in \{\text{succ}\, t_1, \text{pred}\, t_1, \text{iszero}\, t_1\}$ for some smaller term t_1; or
3. $t = \text{if}\ t_1\ \text{then}\ t_2\ \text{else}\ t_3$ for some smaller terms t_1, t_2, t_3.

Inductive Definition of Functions

$$
\begin{aligned}
\mathit{const}(\mathsf{true}) &= \{\mathsf{true}\} \\
\mathit{const}(\mathsf{false}) &= \{\mathsf{false}\} \\
\mathit{const}(0) &= \{0\} \\
\mathit{const}(\mathsf{succ}(t)) &= \mathit{const}(t) \\
\mathit{const}(\mathsf{pred}(t)) &= \mathit{const}(t) \\
\mathit{const}(\mathsf{iszero}(t)) &= \mathit{const}(t) \\
\mathit{const}(\mathsf{if}\ t_1\ \mathsf{then}\ t_2\ \mathsf{else}\ t_3) &= \mathit{const}(t_1) \cup \mathit{const}(t_2) \cup \mathit{const}(t_3)
\end{aligned}
$$

Inductive Definition of Functions

$$
\begin{aligned}
\mathit{size}(\mathsf{true}) &= 1\\
\mathit{size}(\mathsf{false}) &= 1\\
\mathit{size}(0) &= 1\\
\mathit{size}(\mathsf{succ}(t)) &= \mathit{size}(t) + 1\\
\mathit{size}(\mathsf{pred}(t)) &= \mathit{size}(t) + 1\\
\mathit{size}(\mathsf{iszero}(t)) &= \mathit{size}(t) + 1\\
\mathit{size}(\mathsf{if}\ t_1\ \mathsf{then}\ t_2\ \mathsf{else}\ t_3) &= \mathit{size}(t_1) + \mathit{size}(t_2) + \mathit{size}(t_3) + 1
\end{aligned}
$$

Inductive Definition of Functions

$$
\begin{aligned}
depth(\mathsf{true}) &= 1 \\
depth(\mathsf{false}) &= 1 \\
depth(0) &= 1 \\
depth(\mathsf{succ}(t)) &= depth(t) + 1 \\
depth(\mathsf{pred}(t)) &= depth(t) + 1 \\
depth(\mathsf{iszero}(t)) &= depth(t) + 1 \\
depth(\mathsf{if}\ t_1\ \mathsf{then}\ t_2\ \mathsf{else}\ t_3) &= max(depth(t_1), depth(t_2), depth(t_3))
\end{aligned}
$$

Principles of Induction on Terms

Depth gives an induction principle for reasoning about terms:

If, for each term s,
given $P(r)$ for all r such that $depth(r) < depth(s)$
we can show $P(s)$,
then $P(s)$ holds for all s.

Principles of Induction on Terms

Suppose we know for every subterm t' of a term t, $size(t') < size(t)$.

Lemma. For every $t \in \mathcal{T}$, $|const(t)| \leq size(t)$

Proof. By induction on the depth of t. There are three cases to consider.

1. t is a constant.

   By definition, $|const(t)| = |\{t\}| = 1 = size(t)$.

2. $t \in \{\text{succ}(t'), \text{pred}(t'), \text{iszero}(t')\}$.

   By IH, $|const(t')| \leq size(t')$.
   By definition and the given result,
   $|const(t)| = |const(t')| \leq size(t') < size(t)$.

Principles of Induction on Terms

Suppose we know for every subterm t' of a term t, $size(t') < size(t)$.

Lemma. For every $t \in \mathscr{T}$, $|const(t)| \leq size(t)$

Proof. By induction on the depth of t. There are three cases to consider.

3. $t =$ if t_1 then t_2 else t_3.

   By IH:

   i. $|const(t_1)| \leq size(t_1)$

   ii. $|const(t_2)| \leq size(t_2)$

   iii. $|const(t_3)| \leq size(t_3)$

$$
\begin{aligned}
|const(t)| &= |const(\text{if } t_1 \text{ then } t_2 \text{ else } t_3)| \\
&= |const(t_1) \cup const(t_2) \cup const(t_3)| \\
&= |const(t_1)| + |const(t_2)| + |const(t_3)| \\
&\leq size(t_1) + size(t_2) + size(t_3) \\
&< size(t)
\end{aligned}
$$

Principles of Induction on Terms

<u>Size</u> gives an induction principle for reasoning about terms:

If, for each term s,

given $P(r)$ for all r such that $size(r) < size(s)$

we can show $P(s)$,

then $P(s)$ holds for all s.

Principles of Induction on Terms

Structural Induction gives an induction principle for reasoning about terms:

If, for each term s,
given $P(r)$ for all immediate subterms r of s
we can show $P(s)$,
then $P(s)$ holds for all s.

Principles of Induction on Terms

- Induction on depth or size of terms is analogous to complete induction on natural numbers.
- Ordinary structural induction corresponds to the ordinary natural number induction principle where the induction step requires that $P(n+1)$ be established from just the assumption $P(n)$.
- The choice of one term induction principle over another is determined by which one leads to a simpler structure for the proof at hand.
  - They are inter-derivable.

Principles of Induction on Terms

- Use structural induction wherever possible, since it works on terms directly, avoiding the detour via numbers.
- Most proofs using these principles have a similar structure.

Proof: By induction on $\mathtt{t}$.

Case: $\mathtt{t} = \mathtt{true}$

... show $P(\mathtt{true})$ ...

Case: $\mathtt{t} = \mathtt{false}$

... show $P(\mathtt{false})$ ...

Case: $\mathtt{t} = \mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3$

... show $P(\mathtt{if}\ \mathtt{t}_1\ \mathtt{then}\ \mathtt{t}_2\ \mathtt{else}\ \mathtt{t}_3)$, using $P(\mathtt{t}_1)$, $P(\mathtt{t}_2)$, and $P(\mathtt{t}_3)$ ...

(And similarly for the other syntactic forms.) □

Semantics

Operational Semantics

- Specifies the behavior of a programming language by defining a simple abstract machine for it
- For simple languages:
  - Abstract: Terms as machine code
  - Machine Behavior: Specified by a transition function
  - <u>Meaning</u>: Final state of the machine when started with initial state
- Useful: Abstract Machine + Actual Interpreter or Compiler
  - Correctness: Prove both behave the same in some suitable sense

Denotational Semantics

- Meaning of a term is some mathematical object
  - For example, numbers or functions
- Requires finding mathematical domain(s) and then defining an <u>interpretation function</u> from the language to the domain(s)
- Can abstract away tedious details of the language
- Can be used to derive powerful reasoning principles about program behaviors
- Can be used to understand the expressivity of the language

Axiomatic Semantics

- Takes the laws of the behavior or the language as the definition of the language itself
- Meaning: what we can prove about a program
- No operational or denotational semantics is given.
- Elegant because focuses on the behavior of programs
- Lead to the invention of the "invariant"

Evaluation: Booleans

$$\frac{}{\text{if true then } t_2 \text{ else } t_3 \leadsto t_2}\ \text{If-True} \qquad \frac{}{\text{if false then } t_2 \text{ else } t_3 \leadsto t_3}\ \text{If-False}$$

$$\frac{t_1 \leadsto t_1'}{\text{if } t_1 \text{ then } t_2 \text{ else } t_3 \leadsto \text{if } t_1' \text{ then } t_2 \text{ else } t_3}\ \text{If}$$

Evaluation: Booleans

The <u>one-step evaluation</u> relation $\rightsquigarrow$ is the <u>smallest</u> binary relation on terms satisfying the three rules:

$$\frac{}{\text{if true then } t_2 \text{ else } t_3 \rightsquigarrow t_2}\ \text{If-True} \qquad \frac{}{\text{if false then } t_2 \text{ else } t_3 \rightsquigarrow t_3}\ \text{If-False}$$

$$\frac{t_1 \rightsquigarrow t_1'}{\text{if } t_1 \text{ then } t_2 \text{ else } t_3 \rightsquigarrow \text{if } t_1' \text{ then } t_2 \text{ else } t_3}\ \text{If}$$

Evaluation: Booleans

An <u>instance</u> of an inference rule is obtained by consistently replacing each metavariable ($t_1, t_2, \ldots$) by the <u>same term</u> in the rule's conclusion and all its premises (if any).

Evaluation: Booleans

Example:

$$\frac{t_1 \rightsquigarrow t_1'}{\text{if } t_1 \text{ then } t_2 \text{ else } t_3 \rightsquigarrow \text{if } t_1' \text{ then } t_2 \text{ else } t_3}\ \text{If}$$

$$\frac{\text{if true then true else false} \rightsquigarrow \text{true}}{\text{if (if true then true else false) then false else true} \rightsquigarrow \text{if true then false else true}}\ \text{If}$$

Evaluation: Booleans

A rule is satisfied by a relation if, for each instance of the rule, either the conclusion is in the relation or one of the premises is not.

Evaluation: Booleans

Theorem [Determinacy of one-step evaluation]: If $t \rightsquigarrow t'$ and $t \rightsquigarrow t''$, then $t' = t''$

Theorem [Determinacy of one-step evaluation]: If $t \rightsquigarrow t'$ and $t \rightsquigarrow t''$, then $t' = t''$

Proof. By induction on the derivation of $t \rightsquigarrow t'$. We case split over the rule used in the conclusion of the former derivation.

Case (If-True). Then $t = \text{if true then } t_2 \text{ else } t_3$ and $t' = t_2$. We must show $t' = t'' = t_2$, but this must be the case or else $t \rightsquigarrow t''$ is not derivable; which was assumed.

Case (If-False). Similar to If-True above.

Theorem [Determinacy of one-step evaluation]: If $t \rightsquigarrow t'$ and $t \rightsquigarrow t''$, then $t' = t''$

Proof. ...

Case (If). Then $t = \text{if}\, t_1 \,\text{then}\, t_2 \,\text{else}\, t_3$ and $t' = \text{if}\, t_1' \,\text{then}\, t_2 \,\text{else}\, t_3$. It is also the case that $t_1 \rightsquigarrow t_1'$ by assumption. By IH: if $t_1 \rightsquigarrow t_1''$ and $t_1 \rightsquigarrow t_1'''$, then $t_1'' \rightsquigarrow t_1'''$. By assumption we know that $t \rightsquigarrow t'$ which implies that $t' = \text{if}\, t_1'' \,\text{then}\, t_2 \,\text{else}\, t_3$ for some new t_1'' and $t_1 \rightsquigarrow t_1''$ (by the assumption of derivability), but the IH implies that $t_1' = t_1''$; and thus, $t = t'$.

Evaluation: Booleans

The <u>states of evaluation</u> are each term reached after a single step of evaluation. The final state of this chain of reduction steps are called <u>normal forms</u>.

A term t is a <u>normal form</u> iff no evaluation rule applies to it; e.g., there are no instances of any rule where $t \rightsquigarrow t'$ occurs as a conclusion of a rule.

Evaluation: Booleans

For example: true and false are normal forms.

In fact, every value is in normal form.

It should always be the case that values are normal forms. If not, the language is broken.

Evaluation: Booleans

The multi-step evaluation relation $\rightsquigarrow^*$ is the reflexive, transitive closure of one-step evaluation ($\rightsquigarrow$). That is, it is the smallest relation satisfying the following rules:

$$\frac{}{t \rightsquigarrow^* t}\ \text{Refl} \qquad \frac{t_1 \rightsquigarrow t_2 \qquad t_2 \rightsquigarrow^* t_3}{t_1 \rightsquigarrow^* t_3}\ \text{Mult}$$

*Transitivity is a provable property of the two rules above.