Induction

CS 152 (Spring 2021)

Harvard University

Tuesday, February 2, 2021

Today, we learn to

- define an inductive set
- derive the induction principle of an inductive set
- prove properties of programs by induction
- use Coq to check our proofs
- believe in induction!

Expressing Program Properties

Progress

$$\forall e \in \mathbf{Exp}.\ \forall \sigma \in \mathbf{Store}.$$
$$\text{either } e \in \mathbf{Int} \text{ or } \exists e', \sigma'.\ \ < e, \sigma > \longrightarrow < e', \sigma' >$$

Termination

$$\forall e \in \textbf{Exp}.\ \forall \sigma_0 \in \textbf{Store}.\ \exists \sigma \in \textbf{Store}.\ \exists n \in \textbf{Int}.$$
$$< e, \sigma_0 > \longrightarrow^{*} < n, \sigma >$$

Deterministic Result

$$\begin{aligned} \forall e \in \textbf{Exp}.\ \forall \sigma_0, \sigma, \sigma' \in \textbf{Store}.\ \forall n, n' \in \textbf{Int}. \\ \text{if } \ < e, \sigma_0 > \longrightarrow^* < n, \sigma > \ \text{ and} \\ < e, \sigma_0 > \longrightarrow^* < n', \sigma' > \ \text{ then} \\ n = n' \text{ and } \sigma = \sigma'. \end{aligned}$$

Inductive Sets

Inductive Set: Definition

Axiom:

$$\frac{}{a \in A}$$

Inductive Rule:

$$\frac{a_1 \in A \qquad \dots \qquad a_n \in A}{a \in A}$$

Grammar for **Exp**

$$e ::= x \mid n \mid e_1 + e_2 \mid e_1 \times e_2 \mid x := e_1; e_2$$

Inductive Set $\mathbf{Exp}$

$$\textsc{Var}\ \frac{}{x \in \mathbf{Exp}}\ x \in \mathbf{Var} \qquad \textsc{Int}\ \frac{}{n \in \mathbf{Exp}}\ n \in \mathbf{Int}$$

$$\textsc{Add}\ \frac{e_1 \in \mathbf{Exp} \quad e_2 \in \mathbf{Exp}}{e_1 + e_2 \in \mathbf{Exp}}$$

$$\textsc{Mul}\ \frac{e_1 \in \mathbf{Exp} \quad e_2 \in \mathbf{Exp}}{e_1 \times e_2 \in \mathbf{Exp}}$$

$$\textsc{Asg}\ \frac{e_1 \in \mathbf{Exp} \quad e_2 \in \mathbf{Exp}}{x := e_1; e_2 \in \mathbf{Exp}}\ x \in \mathbf{Var}$$

Grammar Equivalent to Inductive Set

$$e ::= x \mid n \mid e_1 + e_2 \mid e_1 \times e_2 \mid x := e_1; e_2$$

$$\textsc{Var}\ \frac{}{x \in \mathbf{Exp}}\ x \in \mathbf{Var} \qquad \textsc{Int}\ \frac{}{n \in \mathbf{Exp}}\ n \in \mathbf{Int}$$

$$\textsc{Add}\ \frac{e_1 \in \mathbf{Exp} \quad e_2 \in \mathbf{Exp}}{e_1 + e_2 \in \mathbf{Exp}}$$

$$\textsc{Mul}\ \frac{e_1 \in \mathbf{Exp} \quad e_2 \in \mathbf{Exp}}{e_1 \times e_2 \in \mathbf{Exp}}$$

$$\textsc{Asg}\ \frac{e_1 \in \mathbf{Exp} \quad e_2 \in \mathbf{Exp}}{x := e_1; e_2 \in \mathbf{Exp}}\ x \in \mathbf{Var}$$

Inductive Set **Exp**: Example Derivation

$$\textsc{Mul}\ \dfrac{\textsc{Add}\ \dfrac{\textsc{Var}\ \dfrac{}{\mathsf{foo} \in \mathbf{Exp}} \qquad \textsc{Int}\ \dfrac{}{3 \in \mathbf{Exp}}}{(\mathsf{foo} + 3) \in \mathbf{Exp}} \qquad \textsc{Var}\ \dfrac{}{\mathsf{bar} \in \mathbf{Exp}}}{(\mathsf{foo} + 3) \times \mathsf{bar} \in \mathbf{Exp}}$$

Inductive Set $\mathbb{N}$ (Natural Numbers)

The natural numbers can be inductively defined:

$$\frac{}{0 \in \mathbb{N}} \qquad \frac{n \in \mathbb{N}}{succ(n) \in \mathbb{N}}$$

where $succ(n)$ is the successor of n.

Inductive Set $\longrightarrow$ (Step Relation)

The small-step evaluation relation $\longrightarrow$ is an inductively defined set. The definition of this set is given by the semantic rules.

Inductive Set $\longrightarrow^*$ (Multi-Step Rel.)

$$\frac{}{< e, \sigma > \longrightarrow^* < e, \sigma >}$$

$$\frac{< e, \sigma > \longrightarrow < e', \sigma' > \qquad < e', \sigma' > \longrightarrow^* < e'', \sigma'' >}{< e, \sigma > \longrightarrow^* < e'', \sigma'' >}$$

Inductive proofs

Mathematical induction

Mathematical induction

For any property P,
If

- $P(0)$ holds
- For all natural numbers n, if $P(n)$ holds then $P(n+1)$ holds

then for all natural numbers k, $P(k)$ holds.

Mathematical induction

$$\frac{}{0 \in \mathbb{N}} \qquad \frac{n \in \mathbb{N}}{succ(n) \in \mathbb{N}}$$

For any property P,
If

- $P(0)$ holds
- For all natural numbers n, if $P(n)$ holds then $P(n+1)$ holds

then for all natural numbers k, $P(k)$ holds.

Induction on inductively-defined sets

Induction on inductively-defined sets

For any property P,

If

- **Base cases:** For each axiom

$$\frac{}{a \in A},$$

  $P(a)$ holds.

- **Inductive cases:** For each inference rule

$$\frac{a_1 \in A \quad \dots \quad a_n \in A}{a \in A},$$

  if $P(a_1)$ and ... and $P(a_n)$ then $P(a)$.

then for all $a \in A$, $P(a)$ holds.

Inductive reasoning principle for set **Exp**

For any property P,
If

- For all variables x, $P(x)$ holds.
- For all integers n, $P(n)$ holds.
- For all $e_1 \in$ **Exp** and $e_2 \in$ **Exp**, if $P(e_1)$ and $P(e_2)$ then $P(e_1 + e_2)$ holds.
- For all $e_1 \in$ **Exp** and $e_2 \in$ **Exp**, if $P(e_1)$ and $P(e_2)$ then $P(e_1 \times e_2)$ holds.
- For all variables x and $e_1 \in$ **Exp** and $e_2 \in$ **Exp**, if $P(e_1)$ and $P(e_2)$ then $P(x := e_1; e_2)$ holds.

then for all $e \in$ **Exp**, $P(e)$ holds.

Case Int

$$\textsc{Int}\ \frac{}{n \in \mathbf{Exp}}\ n \in \mathbf{Int}$$

For all integers n,
$P(n)$ holds

Case $\textsc{Add}$

$$\textsc{Add}\ \frac{e_1 \in \textbf{Exp} \quad e_2 \in \textbf{Exp}}{e_1 + e_2 \in \textbf{Exp}}$$

For all $e_1 \in \textbf{Exp}$ and $e_2 \in \textbf{Exp}$,
if $P(e_1)$ and $P(e_2)$
then $P(e_1 + e_2)$ holds.

Inductive reasoning principle for set $\longrightarrow$

For any property P, **If**

- VAR: For all variables x, stores σ and integers n such that $\sigma(x) = n$, $P(< x, \sigma > \longrightarrow < n, \sigma >)$ holds.
- ADD: For all integers n, m, p such that $p = n + m$, and stores σ, $P(< n + m, \sigma > \longrightarrow < p, \sigma >)$ holds.
- MUL: For all integers n, m, p such that $p = n \times m$, and stores σ, $P(< n \times m, \sigma > \longrightarrow < p, \sigma >)$ holds.
- ASG: For all variables x, integers n and expressions $e \in$ **Exp**, $P(< x := n; e, \sigma > \longrightarrow < e, \sigma[x \mapsto n] >)$ holds.
- LADD: For all expressions $e_1, e_2, e_1' \in$ **Exp** and stores σ and σ', if $P(< e_1, \sigma > \longrightarrow < e_1', \sigma' >)$ holds then $P(< e_1 + e_2, \sigma > \longrightarrow < e_1' + e_2, \sigma' >)$ holds.
- RADD: For all integers n, expressions $e_2, e_2' \in$ **Exp** and stores σ and σ', if $P(< e_2, \sigma > \longrightarrow < e_2', \sigma' >)$ holds then $P(< n + e_2, \sigma > \longrightarrow < n + e_2', \sigma' >)$ holds.
- LMUL: For all expressions $e_1, e_2, e_1' \in$ **Exp** and stores σ and σ', if $P(< e_1, \sigma > \longrightarrow < e_1', \sigma' >)$ holds then $P(< e_1 \times e_2, \sigma > \longrightarrow < e_1' \times e_2, \sigma' >)$ holds.
- RMUL: For all integers n, expressions $e_2, e_2' \in$ **Exp** and stores σ and σ', if $P(< e_2, \sigma > \longrightarrow < e_2', \sigma' >)$ holds then $P(< n \times e_2, \sigma > \longrightarrow < n \times e_2', \sigma' >)$ holds.
- ASG1: For all variables x, expressions $e_1, e_2, e_1' \in$ **Exp** and stores σ and σ', if $P(< e_1, \sigma > \longrightarrow < e_1', \sigma' >)$ holds then $P(< x := e_1; e_2, \sigma > \longrightarrow < x := e_1'; e_2, \sigma' >)$ holds.

then for all $< e, \sigma > \longrightarrow < e', \sigma' >$,
$P(< e, \sigma > \longrightarrow < e', \sigma' >)$ holds.

Proving progress

Progress (Statement)

Progress: For each store σ and expression e that is not an integer, there exists a possible transition for $< e, \sigma >$:

$$\forall e \in \mathbf{Exp}.\ \forall \sigma \in \mathbf{Store}.$$
$$\text{either } e \in \mathbf{Int} \text{ or } \exists e', \sigma'.\ < e, \sigma > \longrightarrow < e', \sigma' >$$

Progress (Rephrased)

$$P(e) = \forall \sigma. \, (e \in \mathbf{Int}) \vee (\exists e', \sigma'. \; < e, \sigma > \longrightarrow < e', \sigma' >)$$

Progress (Rephrased)

$$\forall e \in \textbf{Exp}.\ \forall \sigma \in \textbf{Store}.$$
$$\text{either } e \in \textbf{Int} \text{ or } \exists e', \sigma'.\ < e, \sigma > \longrightarrow < e', \sigma' >$$

$$P(e) = \forall \sigma. \left(e \in \textbf{Int}\right) \vee \left(\exists e', \sigma'.\ < e, \sigma > \longrightarrow < e', \sigma' >\right)$$

Example: Proving progress

by "structural induction on the expressions e"

We will prove by structural induction on expressions **Exp** that for all expressions $e \in \mathbf{Exp}$ we have

$$P(e) = \forall \sigma. \, (e \in \mathbf{Int}) \vee (\exists e', \sigma'. \ < e, \sigma > \longrightarrow < e', \sigma' >).$$

Consider the possible cases for e.

Proving progress: Case $e = x$

By the $\textsc{Var}$ axiom, we can evaluate $< x, \sigma >$ in any state: $< x, \sigma > \longrightarrow < n, \sigma >$, where $n = \sigma(x)$. So $e' = n$ is a witness that there exists e' such that $< x, \sigma > \longrightarrow < e', \sigma >$, and $P(x)$ holds.

Proving progress: Case $e = x$

$$\textsc{Var}\ \frac{}{< x, \sigma > \longrightarrow < n, \sigma >}\ \text{where } n = \sigma(x)$$

By the $\textsc{Var}$ axiom, we can evaluate $< x, \sigma >$ in any state: $< x, \sigma > \longrightarrow < n, \sigma >$, where $n = \sigma(x)$. So $e' = n$ is a witness that there exists e' such that $< x, \sigma > \longrightarrow < e', \sigma >$, and $P(x)$ holds.

Proving progress: Case $e = n$

Then $e \in \textbf{Int}$, so $P(n)$ trivially holds.

Proving progress: Case $e = e_1 + e_2$

This is an inductive step. The inductive hypothesis is that P holds for subexpressions e_1 and e_2. We need to show that P holds for e. In other words, we want to show that $P(e_1)$ and $P(e_2)$ implies $P(e)$. Let's expand these properties. We know that the following hold:

$$P(e_1) = \forall \sigma.\ (e_1 \in \textbf{Int}) \vee (\exists e', \sigma'.\ < e_1, \sigma > \longrightarrow < e', \sigma' >)$$
$$P(e_2) = \forall \sigma.\ (e_2 \in \textbf{Int}) \vee (\exists e', \sigma'.\ < e_2, \sigma > \longrightarrow < e', \sigma' >)$$

and we want to show:

$$P(e) = \forall \sigma. (e \in \textbf{Int}) \vee (\exists e', \sigma'.\ < e, \sigma > \longrightarrow < e', \sigma' >)$$

We must inspect several subcases.

Proving progress: Case $e = e_1 + e_2$, $e_1, e_2 \in \textbf{Int}$

First, if both e_1 and e_2 are integer constants, say $e_1 = n_1$ and $e_2 = n_2$, then by rule ADD we know that the transition $< n_1 + n_2, \sigma > \longrightarrow < n, \sigma >$ is valid, where n is the sum of n_1 and n_2. Hence, $P(e) = P(n_1 + n_2)$ holds (with witness $e' = n$).

Proving progress: Case $e = e_1 + e_2$, $e_1 \notin \textbf{Int}$

Second, if e_1 is not an integer constant, then by the inductive hypothesis $P(e_1)$ we know that $< e_1, \sigma > \longrightarrow < e', \sigma' >$ for some e' and σ'. We can then use rule $\textsc{LAdd}$ to conclude $< e_1 + e_2, \sigma > \longrightarrow < e' + e_2, \sigma' >$, so $P(e) = P(e_1 + e_2)$ holds.

Proving progress: Case $e = e_1 + e_2$, $e_1 \in \mathbf{Int}$, $e_2 \notin \mathbf{Int}$

Third, if e_1 is an integer constant, say $e_1 = n_1$, but e_2 is not, then by the inductive hypothesis $P(e_2)$ we know that $< e_2, \sigma > \longrightarrow < e', \sigma' >$ for some e' and σ'. We can then use rule $\textsc{RAdd}$ to conclude $< n_1 + e_2, \sigma > \longrightarrow < n_1 + e', \sigma' >$, so $P(e) = P(n_1 + e_2)$ holds.

Proving progress: Remaining cases

Case $e = e_1 \times e_2$ and case $e = x := e_1; e_2$. These are also inductive cases, and their proofs are similar to the previous case. [Note that if you were writing this proof out for a homework, you should write these cases out in full.]

Incremental update

For all expressions e and stores σ, if
$< e, \sigma > \longrightarrow < e', \sigma' >$ then
either $\sigma = \sigma'$ or
there is some variable x and integer n such that
$\sigma' = \sigma[x \mapsto n]$.

Proving incremental update

We proceed by induction on the derivation of $< e, \sigma > \longrightarrow < e', \sigma' >$. Suppose we have e, σ, e' and σ' such that $< e, \sigma > \longrightarrow < e', \sigma' >$. The property P that we will prove of e, σ, e' and σ', which we will write as $P(< e, \sigma > \longrightarrow < e', \sigma' >)$, is that either $\sigma = \sigma'$ or there is some variable x and integer n such that $\sigma' = \sigma[x \mapsto n]$:

$$P(< e, \sigma > \longrightarrow < e', \sigma' >) \triangleq$$
$$\sigma = \sigma' \vee (\exists x \in \mathbf{Var}, n \in \mathbf{Int}.\ \sigma' = \sigma[x \mapsto n]).$$

Consider the cases for the derivation of $< e, \sigma > \longrightarrow < e', \sigma' >$.

Proving incremental update: Case $\textsc{Add}$

This is an axiom. Here, $e \equiv n + m$ and $e' = p$ where p is the sum of m and n, and $\sigma' = \sigma$. The result holds immediately.

Proving incremental update: Case $\textsc{LAdd}$

This is an inductive case. Here, $e \equiv e_1 + e_2$ and $e' \equiv e_1' + e_2$ and $< e_1, \sigma > \longrightarrow < e_1', \sigma' >$. By the inductive hypothesis, applied to $< e_1, \sigma > \longrightarrow < e_1', \sigma' >$, we have that either $\sigma = \sigma'$ or there is some variable x and integer n such that $\sigma' = \sigma[x \mapsto n]$, as required.

Proving incremental update: Case $\textsc{Asg}$

This is an axiom. Here $e \equiv x := n; e_2$ and $e' \equiv e_2$ and $\sigma' = \sigma[x \mapsto n]$. The result holds immediately.

Proving incremental update: remaining cases

We leave the other cases (Var, RAdd, LMul, RMul, Mul, and Asg1) as exercises. Seriously, try them. Make sure you can do them. Go on.

Break

Incremental update:
For all expressions e and stores σ, if
$< e, \sigma > \longrightarrow < e', \sigma' >$ then
either $\sigma = \sigma'$ or
there is some variable x and integer n such that
$\sigma' = \sigma[x \mapsto n]$.

Can you prove incremental update by structural induction on the expression e
instead of by induction on the derivation
$< e, \sigma > \longrightarrow < e', \sigma' >$ (as we just did)?

Interlude: What if induction weren't true?

Peano Axioms

$$0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow \ldots$$

1. zero is a number.
2. If a is a number, the successor of a is a number.
3. zero is not the successor of a number.
4. Two numbers of which the successors are equal are themselves equal.
5. (induction axiom.) If a set S of numbers contains zero and also the successor of every number in S, then every number is in S.

Monster Chains

$$
\begin{array}{c}
0 \;\rightarrow 1 \;\rightarrow 2 \;\rightarrow 3 \;\rightarrow \dots \\
\dots \rightarrow -a1 \rightarrow a0 \rightarrow a1 \rightarrow a2' \rightarrow a3' \rightarrow \dots \\
\dots \rightarrow -b1 \rightarrow b0 \rightarrow b1' \rightarrow b2' \rightarrow b3' \rightarrow \dots
\end{array}
$$