Trouble in Induction Paradise

Consider the grammar rule for commands in our IMP language

$$\mathrm{C} ::= \textbf{skip} \mid \mathrm{V} := \mathrm{A} \mid \mathrm{C} \,;\, \mathrm{C} \mid \textbf{if}\ \mathrm{B}\ \textbf{then}\ \mathrm{C}\ \textbf{else}\ \mathrm{C}\ \textbf{end} \mid \textbf{while}\ \mathrm{B}\ \textbf{do}\ \mathrm{C}\ \textbf{end}$$

Then the set **Com** contains all possible programs we can write in this language. We can give an inductive definition for this set,

1. $\textbf{skip} \in \textbf{Com}$
2. $x := a \in \textbf{Com}$ if $x \in \textbf{Loc}, a \in \textbf{Aexp}$
3. $c_0 \,;\, c_1 \in \textbf{Com}$ if $c_0, c_1 \in \textbf{Com}$
4. $\textbf{if}\ b\ \textbf{then}\ c_0\ \textbf{else}\ c_1\ \textbf{end} \in \textbf{Com}$ if $b \in \textbf{Bexp}, c_0, c_1 \in \textbf{Com}$
5. $\textbf{while}\ b\ \textbf{do}\ c\ \textbf{end} \in \textbf{Com}$ if $b \in \textbf{Bexp}, c \in \textbf{Com}$

The first two rules are the least elements in **Com** and the remaining rules define terms inductively. As in the case with the arithmetic expressions there is a clear ordering of the terms.

Trouble in Induction Paradise

Proposition: All programs terminate. Formally,[1]

$$\forall c \in \textbf{Com}, \forall \sigma \in \Sigma, \exists \sigma' \in \Sigma.\ (c, \sigma) \mapsto \sigma'.$$

Proof: By induction on the structure of **Com**.

Case **skip**. Let $\sigma \in \Sigma$, then $(\textbf{skip}, \sigma) \mapsto \sigma$. Clearly terminates.

Case $x := a$. We have already established that arithmetic expressions terminate with $(a, \sigma) \mapsto k$ where $\sigma \in \Sigma$ and $k \in \mathbb{I}$. It follows from the semantic rules for assignment,

$$(x := a, \sigma) \mapsto \sigma[k/x].$$

This also terminates.

[1] This is of course silly, since we know that not all programs terminate, e.g., **while true do skip end**, but it is interesting to see where exactly the proof fails.

Trouble in Induction Paradise

Case $c_0 \,;\, c_1$. As our induction hypothesis we assume that commands c_0 and c_1 terminate in all states. This means that they will also terminate for the following state configuration:

$$\begin{aligned} (c_0, \sigma) &\mapsto \sigma' \\ (c_1, \sigma') &\mapsto \sigma'' \end{aligned}$$

Our inductive step then is,

$$\frac{(c_0, \sigma) \mapsto \sigma' \qquad (c_1, \sigma') \mapsto \sigma''}{(c_0 \,;\, c_1, \sigma) \mapsto \sigma''}$$

Statement composition clearly terminates.

Trouble in Induction Paradise

Case **if** b **then** c_0 **else** c_1 **end**. It can be shown that all boolean expressions terminate (similar proof as for the arithmetic expressions). As our induction hypothesis we assume that commands c_0 and c_1 terminate in all states,

$$\begin{aligned} (c_0, \sigma) &\mapsto \sigma' \\ (c_1, \sigma) &\mapsto \sigma' \end{aligned}$$

Our inductive step then is, (a) for the case that the boolean evaluates to true,

$$\frac{(b, \sigma) \mapsto \mathit{true} \qquad (c_0, \sigma) \mapsto \sigma'}{(\textbf{if } b \textbf{ then } c_0 \textbf{ else } c_1 \textbf{ end}, \sigma) \mapsto \sigma'}$$

and (b) for the case that the boolean evaluates to false,

$$\frac{(b, \sigma) \mapsto \mathit{false} \qquad (c_1, \sigma) \mapsto \sigma'}{(\textbf{if } b \textbf{ then } c_0 \textbf{ else } c_1 \textbf{ end}, \sigma) \mapsto \sigma'}$$

Conditional statements terminate.

Trouble in Induction Paradise

Case **while** b **do** c **end**. It can be shown that all boolean expressions terminate. As our induction hypothesis we assume that command c terminates in all states,

$$(c, \sigma) \mapsto \sigma'$$

Our inductive step then is, (a) for the case that the boolean evaluates to false,

$$\frac{(b, \sigma) \mapsto \textbf{false}}{(\textbf{while } b \textbf{ do } c \textbf{ end}, \sigma) \mapsto \sigma}$$

and (b) for the case that the boolean evaluates to true,

$$\frac{(b, \sigma) \mapsto \textbf{true} \qquad (c, \sigma) \mapsto \sigma' \qquad (\textbf{while } b \textbf{ do } c \textbf{ end}, \sigma') \mapsto \sigma''}{(\textbf{while } b \textbf{ do } c \textbf{ end}, \sigma) \mapsto \sigma''}$$

THE PROOF DOES NOT WORK! There are two different ways at looking at why it does not work:

- If we were to assume that the premise given in red is true, then there is nothing to prove: the loop terminates because the loop terminates.
- From a structural induction point of view this argument does not work because the premise given in red is not a strict subterm of the while loop.

Trouble in Induction Paradise

Another way of interpreting the results from our proof; since the proof only failed for loops and was successful for the rest of the commands we can say that the language of commands without loops would terminate for every state.

Trouble in Induction Paradise

All this points to the observation that termination of loops *cannot* be determined by simply looking at the syntax. Consider the following two loops,

$$\textbf{while } x \geq 1 \textbf{ do } x := x - 1 \textbf{ end}$$

and

$$\textbf{while } x \geq 1 \textbf{ do } x := x + 1 \textbf{ end}$$

The first loop terminates for all possible states and the second loop only terminates for states σ such that $\sigma(x) < 1$. It is clear that we can determine loop termination only by looking at the computation/states more carefully.

Trouble in Induction Paradise

The good news is that structural induction on commands only fails when we consider *semantic issues*, we can still use structural induction on commands when considering *syntactic issues*. Consider the following.

Proposition: All terms in **Com** have the same number of **if**/**while** and **end** keywords.

Proof: Proof by induction over the structure of **Com**. We let $K(c)$ be the number of occurrences of keywords **if** and **while** and we let $E(c)$ be number of occurrences of the keyword **end** in $c \in \mathbf{Com}$. We need to show that $K(c) = E(c)$ for all $c \in \mathbf{Com}$.

Case **skip**. Clearly, $K(\mathbf{skip}) = E(\mathbf{skip}) = 0$.

Case $x := a$. Arithmetic expressions cannot have keywords, therefore it is easy to see that $K(x := a) = E(x := a) = 0$.

Case $c_0 \ ; \ c_1$. As our inductive hypothesis we assume that $K(c_0) = E(c_0)$ and $K(c_1) = E(c_1)$. Observe that the following identities hold,

$$\begin{aligned} K(c_0 \ ; \ c_1) &= K(c_0) + K(c_1) \\ E(c_0 \ ; \ c_1) &= E(c_0) + E(c_1) \end{aligned}$$

We now show that $K(c_0 \ ; \ c_1) = E(c_0 \ ; \ c_1)$ holds,

$$\begin{aligned} K(c_0 \ ; \ c_1) &= K(c_0) + K(c_1) \\ &= E(c_0) + E(c_1) \\ &= E(c_0 \ ; \ c_1) \end{aligned}$$

Trouble in Induction Paradise

Case **if** b **then** c_0 **else** c_1 **end**. Boolean expressions do not contain keywords. As our inductive hypothesis we assume that $K(c_0) = E(c_0)$ and $K(c_1) = E(c_1)$. Observe that the following identities hold,

$$\begin{aligned} K(\textbf{if } b \textbf{ then } c_0 \textbf{ else } c_1 \textbf{ end}) &= K(c_0) + K(c_1) + 1 \\ E(\textbf{if } b \textbf{ then } c_0 \textbf{ else } c_1 \textbf{ end}) &= E(c_0) + E(c_1) + 1 \end{aligned}$$

We now show that

$$K(\textbf{if } b \textbf{ then } c_0 \textbf{ else } c_1 \textbf{ end}) = E(\textbf{if } b \textbf{ then } c_0 \textbf{ else } c_1 \textbf{ end})$$

holds,

$$\begin{aligned} K(\textbf{if } b \textbf{ then } c_0 \textbf{ else } c_1 \textbf{ end}) &= K(c_0) + K(c_1) + 1 \\ &= E(c_0) + E(c_1) + 1 \\ &= E(\textbf{if } b \textbf{ then } c_0 \textbf{ else } c_1 \textbf{ end}) \end{aligned}$$

Trouble in Induction Paradise

Case **while** b **do** c **end**. Boolean expressions do not contain keywords. As our inductive hypothesis we assume that $K(c) = E(c)$. Observe that the following identities hold,

$$\begin{aligned} K(\textbf{while } b \textbf{ do } c \textbf{ end}) &= K(c) + 1 \\ E(\textbf{while } b \textbf{ do } c \textbf{ end}) &= E(c) + 1 \end{aligned}$$

We now show that

$$K(\textbf{while } b \textbf{ do } c \textbf{ end}) = E(\textbf{while } b \textbf{ do } c \textbf{ end})$$

holds,

$$\begin{aligned} K(\textbf{while } b \textbf{ do } c \textbf{ end}) &= K(c) + 1 \\ &= E(c) + 1 \\ &= E(\textbf{while } b \textbf{ do } c \textbf{ end}) \end{aligned}$$

This concludes the proof. □

Assignment

HW#3 – see webpage.

Well-Founded Relations

The essential feature of sets that allows for induction is that the ordering relation between elements in these sets do not give rise to infinite descending chains – there has to be a minimal element!

Clearly, inductively defined sets satisfy this requirement.

Definition: A *well-founded relation* is a binary relation $\prec$ on a set A such that there are no infinite descending chains $\cdots \prec a_i \prec \cdots a_1 \prec a_0$. Here, $a \prec b$ means a is the *predecessor* of b.

Well-Founded Relations

We can express well-founded relations in terms of *minimal elements*.

Proposition: Let $\prec$ be a binary relation on a set A. The relation $\prec$ is well-founded iff any nonempty subset Q of A has a minimal element, i.e., an element m such that

$$m \in Q \wedge \forall b \prec m.b \notin Q.$$

Proof: "only-if" direction: Assume that $\prec$ is well-founded. Let Q be a nonempty subset of A. We can construct a chain of elements in Q where a_0 is any element in Q and $a_n \prec \cdots \prec a_0$ is a chain where $a_i \in Q$ with $i = 0, \ldots, n$. Observe that there is some $b \in Q$ such that $b \prec a_n$ or there is not. If not, then $a_n \prec \cdots \prec a_0$ is the largest chain in Q and we stop the construction. Otherwise, take $a_{n+1} = b$ where $a_{n+1} \prec a_n \prec \cdots \prec a_0$. Since $\prec$ is well-founded, the chain cannot be infinite and will be of the form $a_m \prec \cdots \prec a_0$ where a_m is the minimal element as required.

"if" direction: By contradiction.[2] Assume that every nonempty subset Q of A has a minimal element. Now assume that $\prec$ is not well-founded, that is, $\cdots \prec a_i \prec \cdots \prec a_1 \prec a_0$ is an infinite descending chain in Q. This implies the set $Q = \{a_i \mid i \in \mathbb{N}\}$ would be nonempty without a minimal element. A contradiction, therefore $\prec$ is well-founded. $\square$

[2] **Note**: In proofs by contradiction we want to prove *if A then B*.In order to do so we assume A and $\neg B$ and show that this leads to a contradiction. Therefore, we conclude B.

Well-Founded Induction

Proposition: (Well-Founded Induction Principle) Let $\prec$ be a well-founded relation on a set A, let P be a predicate over the elements of A, then

$$\forall a.\ P(a) \text{ iff } \forall a, \forall b.\ b \prec a \wedge P(b) \Rightarrow P(a),$$

with $a, b \in A$.

Well-Founded Induction

In this formulation the condition

$$\forall a, \forall b.\ b \prec a \wedge P(b) \Rightarrow P(a)$$

is always false for minimal elements in A which makes the base case proof vacuously true for these elements (Why?). It is therefore customary to write the well-founded induction principle in two steps:

Proposition: Let $\prec$ be a well-founded relation on a set A, let P be a predicate over elements of A, then

$$\forall e.\ P(e) \text{ iff } \forall m.\ P(m) \wedge \forall a, \forall b.\ b \prec a \wedge P(b) \Rightarrow P(a),$$

here $e \in A$, $m \in A$ denotes the $\prec$-minimal elements in A, and $a, b \in A$ are non-minimal elements of A.

Well-Founded Induction

Proof: "only-if" direction: Assume that $P(e), \forall e \in A$, this clearly implies that $\forall m.\ P(m) \wedge \forall a, \forall b.\ b \prec a \wedge P(b) \Rightarrow P(a)$ holds.

"if" direction: By contradiction. Assume that $\forall m.\ P(m) \wedge \forall a, \forall b.\ b \prec a \wedge P(b) \Rightarrow P(a)$ holds. Suppose P is not true for every $a \in A$. Since $\prec$ is a well-founded relation, the set $Q = \{a \in A \,|\, \neg P(a)\}$ has a $\prec$-minimal element n with $\neg P(n)$. If this element is an $\prec$-minimal element of A itself, then the condition $\forall m \in A.P(m)$ gives rise to $P(n)$, a contradiction. If the element n has $\prec$-predecessors say $b \prec n$ and $b \notin Q$, then by assumption we have $P(b)$ for every b and by condition $\forall a, \forall b.\ b \prec a \wedge P(b) \Rightarrow P(a)$ we have $P(n)$, a contradiction. Therefore, $P(a)$ has to hold for all $a \in A$.[3] $\square$

This shows that the "domino principle" has to hold for *all* elements in an ordered set.

[3] Reference: PlanetMath.org

Well-Founded Induction on States

Proposition: Prove that $p \equiv$ **while** $x \geq 1$ **do** $x := x - 1$ **end** terminates, i.e.,

$$\forall \sigma \in \Sigma,\ \exists \sigma' \in \Sigma.(p, \sigma) \mapsto \sigma'.$$

Proof Idea: The key to loop termination is the loop condition. The proof approach here is to group the states into two sets: (a) The set of states where the loop terminates trivially, that is, all states σ where $\sigma(x) \leq -1$. (b) The set of states where we need to explicitly show that the loop terminates, all states σ where $\sigma(x) \geq 0$. In the latter set we create an ordering based on the loop index x and then perform well-founded induction on the states using states with the loop index of 0 as the basis and any state with loop index value k for the inductive step.

Well-Founded Induction on States

Proof: It is easy to see that for all states $\sigma \in \Sigma$ where $\sigma(x) \leq -1$ we have $(p, \sigma) \mapsto \sigma$. It remains to show that p terminates for states σ where $\sigma(x) \geq 0$. Let $S = \{\sigma \in \Sigma \,|\, \sigma(x) \geq 0\}$. Define the predicate P over S as

$$P(\sigma) \equiv \exists \sigma'.(p, \sigma) \mapsto \sigma'.$$

Define the relation $\prec$ over S as $\sigma' \prec \sigma$ iff $\sigma(x) = \sigma'(x) + 1$. Clearly, the relation $\prec$ is well-founded because no infinite chains $\cdots \prec \sigma_n \prec \cdots \prec \sigma_k$ such that $\sigma_i(x) \geq 0$ can exists. We also observe that $\sigma_m \in S$ such that $\sigma_m(x) = 0$ are minimal elements in S. We apply the principle of well-founded induction

$$\forall \sigma.\ P(\sigma) \text{ if } \forall \sigma_m.\ P(\sigma_m) \wedge \forall \sigma, \forall \sigma_b.\ \sigma_b \prec \sigma \wedge P(\sigma_b) \Rightarrow P(\sigma).$$

Well-Founded Induction on States

Base case: For minimal elements in S we have $\sigma_m(x) = 0$, it follows that $(p, \sigma_m) \mapsto \sigma_m$. Clearly, $P(\sigma_m)$ holds.

Inductive step: Assume that we have some state σ such that $\sigma(x) = k$ with $k \geq 1$. As the inductive hypothesis we assume that $P(\sigma)$ holds. Now let $\sigma \prec \sigma_{++}$, such that $\sigma_{++}(x) = \sigma(x) + 1$ and $\sigma_{++}(y) = \sigma(y)$ with $y \neq x$, then we have

$$\dfrac{\dfrac{\vdots}{(x \geq 1, \sigma_{++}) \mapsto \textbf{true}} \qquad \dfrac{\vdots}{(x := x - 1, \sigma_{++}) \mapsto \sigma} \qquad (p, \sigma) \mapsto \sigma'}{(p, \sigma_{++}) \mapsto \sigma'}$$

The rightmost premise holds due to our inductive hypothesis. To complete the proof we need to show that the states σ_{++} and σ are related to each other via the assignment $x := x - 1$, specifically,

$$\sigma_{++}[(\sigma_{++}(x) - 1)/x] = \sigma_{++}[(\sigma(x) + 1 - 1)/x] = \sigma_{++}[\sigma(x)/x] = \sigma$$

The rightmost identity follows from the extensionality property of functions and the identities above.

Therefore, the program p terminates for all $\sigma \in \Sigma$. $\square$

Structural Induction

Read David Schmidt, Section 1.2