CS 221 **Programming Languages** **Handout 1**
Fall 2005 **Oct 25, 2005**

Rule Induction Example

This handout provides a detailed examination of Rule Induction for the particular rule set defining the small-step dynamic semantics for the simple language Arith+Let (see Section 1). It gives a full presentation of the Induction Principle for this rule set, and then a detailed (even pedantic) sample inductive proof based on the Induction Principle.

1 Small-step dynamic semantics for Arith with Let

$$\frac{(p = m + n)}{\texttt{plus}(\texttt{num}[m], \texttt{num}[n]) \mapsto \texttt{num}[p]} \quad (AD_1) \qquad \frac{(p = m \times n)}{\texttt{times}(\texttt{num}[m], \texttt{num}[n]) \mapsto \texttt{num}[p]} \quad (AD_2)$$

$$\frac{}{\texttt{let}(\texttt{num}[n],\, x.e_2) \mapsto \{\texttt{num}[p]/x\}e_2} \quad (AD_3)$$

$$\frac{e_1 \mapsto e_1'}{\texttt{plus}(e_1, e_2) \mapsto \texttt{plus}(e_1', e_2)} \quad (AD_4) \qquad \frac{e_2 \mapsto e_2'}{\texttt{plus}(\texttt{num}[n], e_2) \mapsto \texttt{plus}(\texttt{num}[n], e_2')} \quad (AD_5)$$

$$\frac{e_1 \mapsto e_1'}{\texttt{times}(e_1, e_2) \mapsto \texttt{times}(e_1', e_2)} \quad (AD_6) \qquad \frac{e_2 \mapsto e_2'}{\texttt{times}(\texttt{num}[n], e_2) \mapsto \texttt{times}(\texttt{num}[n], e_2')} \quad (AD_7)$$

$$\frac{e_1 \mapsto e_1'}{\texttt{let}(e_1,\, x.e_2) \mapsto \texttt{let}(e_1',\, x.e_2)} \quad (AD_8)$$

2 Induction Principle for Dynamic Sematics Rules

The Induction Principle for the transition relation $\mapsto$ defined by rules AD_1 through AD_8 allows us to prove general statements of the form

$$IConc: \quad \forall e, e'.\, e \mapsto e' \Rightarrow P(e, e') \tag{1}$$

where P is some binary property on expressions. For instance, to prove the following proposition stating that the transition relation is deterministic:

$$\forall e_1, e_2, e_3.\, e_1 \mapsto e_2 \,\&\, e_1 \mapsto e_3 \;\Rightarrow\; e_2 = e_3 \tag{2}$$

we could define the property P as follows:

$$P(e, e') \;\Leftrightarrow\; \forall e''.\, e \mapsto e'' \;\Rightarrow\; e'' = e' \tag{3}$$

The Induction Principle is the implication

$$IC_1 \,\&\, IC_2 \,\&\, \ldots \,\&\, IC_8 \;\Rightarrow\; IConc \tag{4}$$

where each IC_i is an *induction clause* for the corresponding inference rule AD_i, as described below.

Instruction Rules: The induction clauses for the three *instruction* rules are simple because they do not introduce induction hypotheses.

Rule AD_1

$$IC_1: \quad \forall e, e'.\, \text{if } e \mapsto e' \text{ by } AD_1, \text{ then } P(e, e') \tag{5}$$

or equivalently

$$IC_1: \quad \forall m, n.\, P(\texttt{plus}(\texttt{num}[m], \texttt{num}[n]), \texttt{num}[m + n]) \tag{6}$$

Rule AD_2

$$IC_2: \quad \forall e, e'.\, \text{if } e \mapsto e' \text{ by } AD_2, \text{ then } P(e, e') \tag{7}$$

or equivalently

$$IC_2: \quad \forall m, n.\, P(\texttt{times}(\texttt{num}[m], \texttt{num}[n]), \texttt{num}[m \times n]) \tag{8}$$

Rule AD_3

$$IC_3: \quad \forall e, e'.\, \text{if } e \mapsto e' \text{ by } AD_2, \text{ then } P(e, e') \tag{9}$$

or equivalently

$$IC_3: \quad \forall n, e.\, P(\texttt{let}(\texttt{num}[n],\, x.e), \{\texttt{num}[n]/x\}e) \tag{10}$$

Search Rules: The induction clauses for the *search* rules are more complicated, and involve induction hypotheses (which are underlined here).

Rule AD_4

$$\begin{aligned} IC_4: \quad & \forall e, e'. \text{ if } e \mapsto e' \text{ by } AD_4, \text{ so that for some } e_1, e_2, \text{ and } e_1', \\ & \quad e = \texttt{plus}(e_1, e_2) \text{ and } e' = \texttt{plus}(e_1', e_2) \text{ and } e_1 \mapsto e_1', \\ & \quad \text{then } \underline{P(e_1, e_1')} \Rightarrow P(e, e') \end{aligned} \tag{11}$$

or equivalently

$$IC_4: \quad \forall e_1, e_1', e_2.\ e_1 \mapsto e_1' \ \&\ \underline{P(e_1, e_1')} \Rightarrow P(\texttt{plus}(e_1, e_2), \texttt{plus}(e_1', e_2)) \tag{12}$$

Rule AD_5

$$\begin{aligned} IC_5: \quad & \forall e, e'. \text{ if } e \mapsto e' \text{ by } AD_5, \text{ so that for some } n, e_2, \text{ and } e_2', \\ & \quad e = \texttt{plus}(\texttt{num}[n], e_2) \text{ and } e' = \texttt{plus}(\texttt{num}[n], e_2') \text{ and } e_2 \mapsto e_2', \\ & \quad \text{then } \underline{P(e_2, e_2')} \Rightarrow P(e, e') \end{aligned} \tag{13}$$

or equivalently

$$IC_5: \quad \forall n, e_2, e_2'.\ e_2 \mapsto e_2' \ \&\ \underline{P(e_2, e_2')} \Rightarrow P(\texttt{plus}(\texttt{num}[n], e_2), \texttt{plus}(\texttt{num}[n], e_2')) \tag{14}$$

Rule AD_6 IC_6 is similar to IC_4 with `times` substituted for `plus`.

Rule AD_7 IC_7 is similar to IC_5 with `times` substituted for `plus`.

Rule AD_8

$$\begin{aligned} IC_8: \quad & \forall e, e'. \text{ if } e \mapsto e' \text{ by } AD_8, \text{ so that for some } x, e_1, e_2, \text{ and } e_1', \\ & \quad e = \texttt{let}(e_1, x.e_2) \text{ and } e' = \texttt{let}(e_1', x.e_2) \text{ and } e_1 \mapsto e_1', \\ & \quad \text{then } \underline{P(e_1, e_1')} \Rightarrow P(e, e') \end{aligned} \tag{15}$$

or equivalently

$$IC_8: \quad \forall x, e_1, e_1', e_2.\ e_1 \mapsto e_1' \ \&\ \underline{P(e_1, e_1')} \Rightarrow P(\texttt{let}(e_1, x.e_2), \texttt{let}(e_1', x.e_2)) \tag{16}$$

3 Example Proof

Now let's use this induction principle for Arith+Let to do a representative proof by Rule Induction over the rule set AD_1 through AD_8 defining the small step transition relation $\mapsto$.

Proposition: $\forall e_1, e_2, e_3.\ e_1 \mapsto e_2 \ \&\ e_1 \mapsto e_3 \Rightarrow e_2 = e_3$

Proof: We recast the statement of the proposition so that it has the form of the conclusion of our Induction Principle (1), using the property P defined in (3). So our goal is to use the induction principle to prove

$$\forall e, e'.\, e \mapsto e' \Rightarrow (\forall e''.\, e \mapsto e'' \;\Rightarrow\; e'' = e') \tag{17}$$

The proof proceeds by cases, one case for each rule, or more precisely for each inductive clause IC_1 through IC_8. Having proved each inductive clause, we can apply the Inductive Principle (4) to prove the conclusion.

Case IC_1. Assume that e is such that

$$\texttt{plus}(\texttt{num}[m], \texttt{num}[n]) \mapsto e \tag{18}$$

Only rules AD_1 or AD_4 or AD_5 could possibly derive (19) because only their conclusions match the form of the source expression $\texttt{plus}(\texttt{num}[m], \texttt{num}[n])$. But AD_4 cannot be used to derive (19) since $\texttt{num}[m]$ is final, and similarly AD_5 cannot be used because $\texttt{num}[n]$ is final.

Thus (19) can only be derived by rule AD_1. It follows that e must be $\texttt{num}[m+n]$.

Case IC_2. The proof is similar to that of IC_1, replacing `plus` with `times` and $+$ with $\times$.

Case IC_3. Assume that

$$\texttt{let}(\texttt{num}[n], x.e_2) \mapsto e \tag{19}$$

Only rules AD_3 or AD_8 could possibly apply because only their conclusions match the form of the source expression $\texttt{let}(\texttt{num}[n], x.e_2)$. But AD_8 cannot be used to derive (19) since $\texttt{num}[n]$ is final.

Thus (19) can only be derived by rule AD_3. It follows that e must be $\{\texttt{num}n/x\}e_2$.

Case IC_4. Assume that

$$\texttt{plus}(e_1, e_2) \mapsto \texttt{plus}(e_1', e_2) \tag{20}$$

by rule AD_4, implying that $e_1 \mapsto e_1'$. We can assume the Induction Hypothesis $P(e_1, e_1')$, *i.e.*

$$\forall e''.\, e_1 \mapsto e'' \;\Rightarrow\; e'' = e_1' \tag{21}$$

Now suppose that e' is such that

$$\texttt{plus}(e_1, e_2) \mapsto e' \tag{22}$$

Then (22) must be derived using either AD_1, AD_4, or AD_5. But AD_1 and AD_5 are not possible because e_1 is not final ($e_1 \mapsto e_1'$) and so e_1 cannot be of the form $\texttt{num}[n]$. Hence (22) must be derived using AD_4, implying that there is an expression e_1'' such that $e' = \texttt{plus}(e_1'', e_2)$ and $e_1 \mapsto e_1''$. But then the Induction Hypothesis (21) implies that $e_1'' = e_1'$, which in turn implies that $e' = \texttt{plus}(e_1', e_2)$.

Case IC_5. Assume that

$$\texttt{plus}(\texttt{num}[n], e_2) \mapsto \texttt{plus}(\texttt{num}[n], e_2') \tag{23}$$

by rule AD_5, implying that $e_2 \mapsto e_2'$. We can assume the Induction Hypothesis $P(e_2, e_2')$, *i.e.*

$$\forall e''.\, e_2 \mapsto e'' \;\Rightarrow\; e'' = e_2' \tag{24}$$

Now suppose that e' is such that

$$\texttt{plus}(\texttt{num}[n], e_2) \mapsto e' \tag{25}$$

Then (25) must be derived using either AD_1, AD_4, or AD_5. But AD_1 is not possible because e_2 is not final ($e_2 \mapsto e_2'$) and so is not of the form $\texttt{num}[n]$, and AD_4 is not possible because $\texttt{num}[n]$ *is* final. Hence (25) must be derived using AD_5, implying that there is an expression e_2'' such that $e' = \texttt{plus}(\texttt{num}[n], e_2'')$ and $e_2 \mapsto e_2''$. But then the Induction Hypothesis (24) implies that $e_2'' = e_2'$, which in turn implies that $e' = \texttt{plus}(\texttt{num}[n], e_2')$.

Case IC_6. The proof is similar to that of IC_4, replacing `plus` with `times` and $+$ with $\times$.

Case IC_7. The proof is similar to that of IC_5, replacing `plus` with `times` and $+$ with $\times$.

Case IC_8. Assume that

$$\texttt{let}(e_1, x.e_2) \mapsto \texttt{let}(e_1', x.e_2) \tag{26}$$

by rule AD_8, implying that $e_1 \mapsto e_1'$. We can assume the Induction Hypothesis $P(e_1, e_1')$, *i.e.*

$$\forall e''.\, e_1 \mapsto e'' \;\Rightarrow\; e'' = e_1' \tag{27}$$

Now suppose that e' is such that

$$\texttt{let}(e_1, x.e_2) \mapsto e' \tag{28}$$

Then (28) must be derived using either AD_3 or AD_8. But AD_1 is not possible because e_1 is not final ($e_1 \mapsto e_1'$) and so is not of the form $\texttt{num}[n]$. Hence (28) must be derived using AD_8, implying that there is an expression e_1'' such that $e' = \texttt{let}(e_1'', x.e_2)$ and $e_1 \mapsto e_1''$. But then the Induction Hypothesis (27) implies that $e_1'' = e_1'$, which in turn implies that $e' = \texttt{plus}(e_1', e_2)$.

QED

4 A More Informal Proof

Lets now look at a more conventional, semi-formal way of presenting the same proof (roughly comparable to the style of proof found in Harper's text and the research literature). In such a proof, the the Induction Principle is assumed but not made explicit, and sometimes the induction hypothesis for an inductive case is also not stated explicitly, but instead is refered to by the phrase "by induction". We'll just give a couple representative cases for this proof.

Proof: We proceed by rule induction on the hypothesis $e_1 \mapsto e_2$.

Case: $e_1 \mapsto e_2$ by Rule AD_1. Then

$$e_1 = \texttt{plus}(\texttt{num}[m], \texttt{num}[n]) \quad \text{and} \tag{29}$$

$$e_2 = \texttt{num}[m+n] \tag{30}$$

If $e_1 \mapsto e'$ for some expression e', this must be derived using a Rule matching e_1. But the only such rule is AD_1, because the fact that $\texttt{num}[m]$ and $\texttt{num}[n]$ are final rules out AD_4 and AD_5. But if $e_1 \mapsto e'$ by rule AD_1, it is clear that $e' = \texttt{num}[m+n]$, and hence $e' = e_2$.

Case: $e_1 \mapsto e_2$ by Rule AD_4. Then

$$e_1 = \texttt{plus}(e_{1,1}, e_{1,2}) \quad \text{and} \tag{31}$$
$$e_2 = \texttt{plus}(e'_{1,1}, e_{1,2}) \tag{32}$$

where

$$e_{1,1} \mapsto e'_{1,1} \tag{33}$$

Suppose that for some e',

$$e_1 \mapsto e' \tag{34}$$

Since $e_{1,1}$ cannot be a number expression because of (33), the only rule that could derive (34) is AD_4, which implies that $e' = \texttt{plus}(e'_1, e_{1,2})$ for some e'_1 such that $e_{1,1} \mapsto e'_1$. But *by induction*, $e'_{1,1}$ is the unique expression such that (33) holds, and therefore we must have $e'_1 = e'_{1,1}$, and hence $e' = e_2$.

5 Appendix: Static Semantics for Arith with Let

Just for completeness, we include the static semantics for Arith with Let.

The judgement $\Gamma \vdash e$ ok expresses the fact that e is well-formed with free variables contained in the set Γ.

$$\frac{(x \in \Gamma)}{\Gamma \vdash x \text{ ok}} \quad (AS_1) \qquad \frac{(n \geq 0)}{\Gamma \vdash \texttt{num}[n] \text{ ok}} \quad (AS_2)$$

$$\frac{\Gamma \vdash e_1 \text{ ok} \qquad \Gamma \vdash e_2 \text{ ok}}{\Gamma \vdash \texttt{plus}(e_1, e_2) \text{ ok}} \quad (AS_3) \qquad \frac{\Gamma \vdash e_1 \text{ ok} \qquad \Gamma \vdash e_2 \text{ ok}}{\Gamma \vdash \times(e_1, e_2) \text{ ok}} \quad (AS_4)$$

$$\frac{\Gamma \vdash e_1 \text{ ok} \qquad \Gamma \cup \{x\} \vdash e_2 \text{ ok} \qquad (x \notin \Gamma)}{\Gamma \vdash \texttt{let}(e_1,\, x.e_2) \text{ ok}} \quad (AS_5)$$