Constructive Logic (15-317), Spring 2023
Recitation 5

Clogic Staff

February 22, 2023

1 Rule Induction and Programming Languages Dynamics

In lecture on Thursday, we discussed the dynamic of programming languages and how there is some difference between logic and these dynamics. Several theorems were brought up that are important to prove about programming languages. Here, we look at progress. The statement of the theorem is as follows:

If $M : A$ then either $M \longrightarrow M'$ or $M\ \mathit{value}$ (but not both).

While doing the entire proof would take a long time, lets look at the cases of the progress proof involving implication as well as disjunction to get practice with rule induction. As a reminder, the dynamics for terms involving implication are the following:

$$\frac{M \longrightarrow M'}{M\ N \longrightarrow M'\ N} \qquad \frac{M\ \mathit{value}}{M\ N \longrightarrow M\ N'}$$

$$\frac{N\ \mathit{value}}{(\lambda x.M)N \longrightarrow [N/x]M} \qquad \frac{}{(\lambda x.M)\ \mathit{value}}$$

While the typing rules (statics) involving implication are:

$$\frac{M : A \supset B \quad N : A}{M\ N : B} \supset\! E$$

$$\begin{array}{c}\overline{x : A}^{\ x} \\ \vdots \\ \dfrac{M : B}{\lambda x.M : A \supset B} \supset\! I^x\end{array}$$

The dynamics involving disjunction are the following:

$$\frac{M\ \mathit{value}}{\textbf{inl}\ M\ \mathit{value}}\ 1 \qquad \frac{M\ \mathit{value}}{\textbf{inr}\ M\ \mathit{value}}\ 2$$

$$\frac{M \longrightarrow M'}{\textbf{inl}\ M \longrightarrow \textbf{inl}\ M'}\ 3 \qquad \frac{M \longrightarrow M'}{\textbf{inr}\ M \longrightarrow \textbf{inr}\ M'}\ 4$$

$$\frac{M \longrightarrow M'}{\textbf{case}(M, x.N, y.P) \longrightarrow \textbf{case}(M', x.N, y.P)}\ 5$$

$$\frac{M\ value}{\textbf{case}(\textbf{inl}\ M, x.N, y.P) \longrightarrow [M/x]N}\ 6 \qquad \frac{M\ value}{\textbf{case}(\textbf{inr}\ M, x.N, y.P) \longrightarrow [M/y]P}\ 7$$

The typing rules (statics) involving disjunction are the following:

$$\frac{M : A}{\textbf{inl}\ M : A \vee B}\ \vee I_1 \qquad \frac{M : B}{\textbf{inr}\ M : A \vee B}\ \vee I_1$$

$$\frac{M : A \vee B \quad \begin{array}{c}\overline{x : A}^{\ x} \\ \vdots \\ N : C\end{array} \quad \begin{array}{c}\overline{y : B}^{\ y} \\ \vdots \\ P : C\end{array}}{\textbf{case}(M, x.N, y.P)}\ \vee E^{x,y}$$

Solution:

$\supset E$

We prove this by induction. IH: the statement of the theorem.
By assumption we have $M\ N : B$
By inversion we have $\exists A\ s.t.$

$$\frac{\begin{array}{c}\mathcal{D}_1 \\ M : A \supset B\end{array} \quad \begin{array}{c}\mathcal{D}_2 \\ N : A\end{array}}{M\ N : B}$$

By IH on $\mathcal{D}_1$ we have 2 cases.
Case 1. M is a value of the form $\lambda x.W$

By IH on $\mathcal{D}_2$ we again have 2 cases.

Case 1a. N is a value. Then exactly 1 one dynamics rule applies, and we get that

$$M\ N = \lambda x.W\ N \longrightarrow [N/x]\ W$$

Case 1b. $N \longrightarrow N'$. Then exactly 1 one dynamics rule applies, and we get that

$$M\ N = \lambda x.W\ N \longrightarrow \lambda x.W\ N'$$

Case 2. M steps to some M'. Only one dynamics rule applies and we get that

$$M\ N \longrightarrow M'\ N$$

$\supset I^x$

Only 1 rule applies when we have a term of the form $\lambda x.M$ which states that $\lambda x.M$ is a value.

$\vee I_1$ ($\vee I_2$ is analogous)

We prove this by induction. IH: the statement of the theorem.
By assumption we have **inl** $M : A \vee B$
By inversion we have:

$$\frac{\begin{matrix}\mathcal{D}\\ M : A\end{matrix}}{\textbf{inl}\ M : A \vee B}$$

By IH on $\mathcal{D}$ we get two cases.
Case 1. M is a value. Then applying rule 1 yields **inl** M is a value
Case 2. $M \longrightarrow M'$. Then apply rule 3, and we get **inl** $M \longrightarrow$ **inl** M'

$\vee E^{x,y}$

We prove this by induction. IH: the statement of the theorem.
By assumption we have **case** $(M, x.N, y.P) : C$
By inversion we have

$$\frac{\begin{matrix}\\ \\ \mathcal{D}\\ M : A \vee B\end{matrix} \quad \begin{matrix}\overline{x : A}\\ \mathcal{E}_1\\ N : C\end{matrix} \quad \begin{matrix}\overline{y : B}\\ \mathcal{E}_2\\ P : C\end{matrix}}{\textbf{case}(M, x.N, y.P) : C}\ \vee E^{x,y}$$

By IH on $\mathcal{D}$ We have 2 cases.
Case 1. M is a value. Two options

M is **inl** M_1 or **inr** M_1. These cases are analogous so just doing the **inl** case.

Then by inversion, we get M_1 is a value.

Applying rule 6, we get **case** (**inl** $M_1, x.N, y.P) \longrightarrow [M_1/x]N$

Case 2, $M \longrightarrow M'$. Applying rule 5 gives us **case**$(M, x.N, y.P) \longrightarrow$ **case**$(M', x.N, y.P)$.

2 Primitive Recursion

In lecture, several recursive definitions were given. For example, pred was given as

$$pred(n) \triangleq R(n, 0, x.r.x)$$

plus was given as

$$plus(n) \triangleq R(n, \lambda k.k, x.r.\lambda k.s(r\ k))$$

mult was given as

$$mult(n) \triangleq R(n, \lambda k.0, x.r.\lambda k.plus(r\ k)\ k)$$

Here we aim to construct a few more.
Task 2. Define a recursive definition for factorial being allowed to use any of the definitions above.
Solution: First, we see we want $fact(0) = 1$ and $fact(n+1) = mult(n+1, fact(n))$ converting this into a recursive definition:

$$fact(n) \triangleq R(n, s(0), x.r.mult\ s(x)\ r)$$

Task 3. Define a recursive definition for subtraction being allowed to use any of the definitions above.
Solution: First, we have $n - 0 = n$ and $n - (k+1) = (n-k) - 1$. So we have $sub(n, 0) = n$ and $sub(n, (s(k)) = pred(sub(n, k))$ Note that unlike the previous cases, the recurance is on the second argument not on the first.
converting this into a recursive definition:

$$sub(k) \triangleq R(k, \lambda n.n, x.r.\lambda n.pred(r\ n))$$