Chapter 05

Newton Ni

July 12, 2020

1 5.3.3

Give a careful proof that $|FV(\mathtt{t})| \leq size(\mathtt{t})$ *for every term* $\mathtt{t}$.

Proof. By structural induction on terms $\mathtt{t}$. Our inductive hypothesis is:

$$H(\mathtt{t}) : |FV(\mathtt{t})| \leq size(\mathtt{t})$$

Case (T-VAR). Here, $|FV(\mathtt{x})| = |\{\mathtt{x}\}| = 1 \leq size(\mathtt{x}) = 1$.
Case (T-ABS). Here, $|FV(\lambda\mathtt{x}.\mathtt{t}_1)| = |FV(\mathtt{t}_1) \setminus \{\mathtt{x}\}|$.

- If $\mathtt{x} \in FV(\mathtt{t}_1)$, then $|FV(\mathtt{t}_1) \setminus \{\mathtt{x}\}| = |FV(\mathtt{t}_1)| - 1$.
- Otherwise $|FV(\mathtt{t}_1) \setminus \{\mathtt{x}\}| = |FV(\mathtt{t}_1)|$.

It follows that $|FV(\mathtt{t}_1) \setminus \{\mathtt{x}\}| \leq |FV(\mathtt{t}_1)|$.

$$\begin{aligned}
|FV(\lambda\mathtt{x}.\mathtt{t}_1)| &= |FV(\mathtt{t}_1) \setminus \{\mathtt{x}\}| \\
&\leq |FV(\mathtt{t}_1)| && \text{(By above)} \\
&\leq size(\mathtt{t}_1) && \text{(By inductive hypothesis)} \\
&\leq size(\mathtt{t}_1) + 1 \\
&\leq size(\lambda\mathtt{x}.\mathtt{t}_1) && \text{(By definition of } size\text{)}
\end{aligned}$$

Case (T-APP).

$$\begin{aligned}
|FV(\mathtt{t}_1\ \mathtt{t}_2)| &= |FV(\mathtt{t}_1) \cup FV(\mathtt{t}_2)| \\
&\leq |FV(\mathtt{t}_1)| + |FV(\mathtt{t}_2)| \\
&\leq size(\mathtt{t}_1) + size(\mathtt{t}_2) && \text{(By inductive hypothesis)} \\
&\leq size(\mathtt{t}_1\ \mathtt{t}_2) && \text{(By definition of } size\text{)}
\end{aligned}$$

□

2 5.3.6

Adapt these rules to describe the other three strategies for evaluation—full beta-reduction, normal-order, and lazy evaluation.

Full Beta-Reduction

$$\frac{\mathtt{t_1} \longrightarrow \mathtt{t_1}'}{\mathtt{t_1}\ \mathtt{t_2} \longrightarrow \mathtt{t_1}'\ \mathtt{t_2}}\ \text{E-App1}$$

$$\frac{\mathtt{t_2} \longrightarrow \mathtt{t_2}'}{\mathtt{t_1}\ \mathtt{t_2} \longrightarrow \mathtt{t_1}\ \mathtt{t_2}'}\ \text{E-App2}$$

$$\frac{\mathtt{t_1} \longrightarrow \mathtt{t_1}'}{\lambda\mathtt{x}.\ \mathtt{t_1} \longrightarrow \lambda\mathtt{x}.\ \mathtt{t_1}'}\ \text{E-Abs}$$

$$\frac{}{(\lambda\mathtt{x}.\ \mathtt{t_1})\ \mathtt{t_2} \longrightarrow [\mathtt{x} \mapsto \mathtt{t_2}]\mathtt{t_1}}\ \text{E-AppAbs}$$

Normal-Order

Lazy

$$\frac{\mathtt{t_1} \longrightarrow \mathtt{t_1}'}{\mathtt{t_1}\ \mathtt{t_2} \longrightarrow \mathtt{t_1}'\ \mathtt{t_2}}\ \text{E-App1}$$

$$\frac{}{(\lambda\mathtt{x}.\ \mathtt{t_1})\ \mathtt{t_2} \longrightarrow [\mathtt{x} \mapsto \mathtt{t_2}]\mathtt{t_1}}\ \text{E-AppAbs}$$

3 3.5.7

Exercise 3.5.16 gave an alternative presentation of the operational semantics of booleans and arithmetic expressions in which stuck terms are defined evaluate to a special constant `wrong`*. Extend this semantics to* λ**NB**.

4 5.3.8

Exercise 4.2.2 introduced a "big-step" style of evaluation for arithmetic expressions, where the basic evaluation relation is "term `t` *evaluates to final result* `v`*". Show how to formulate the evaluation rules for lambda-terms in the big-step style.*

$$\frac{}{\lambda\mathtt{x}.\ \mathtt{t} \Downarrow \lambda\mathtt{x}.\ \mathtt{t}}\ \text{E-Abs}$$

$$\frac{\mathtt{t_1} \Downarrow \lambda\mathtt{x}.\ \mathtt{t} \qquad \mathtt{t_2} \Downarrow \mathtt{v_2} \qquad [\mathtt{x} \mapsto \mathtt{v_2}]\ \mathtt{t} \Downarrow \mathtt{t}'}{\mathtt{t_1}\ \mathtt{t_2} \Downarrow \mathtt{t}'}\ \text{E-App}$$