Harvard School of Engineering and Applied Sciences — CS 152: Programming Languages

Definitional translations; References and continuations (Lectures 9–10) Section and Practice Problems

Week 6: Tue Mar 9–Fri Mar 13, 2020

---

1 Definitional translations

Consider an applied lambda calculus with booleans, conjunction, a few constant natural numbers, and addition, whose syntax is defined as follows.

$$e ::= x \mid \lambda x.\, e \mid e_1\ e_2 \mid \mathsf{true} \mid \mathsf{false} \mid e_1 \text{ and } e_2 \mid 0 \mid 1 \mid 2 \mid e_1 + e_2 \mid \text{if } e_1 \text{ then } e_2 \text{ else } e_3$$

Give a translation to the pure lambda calculus. Use the encodings of booleans and natural numbers that we considered in class. (You can assume that both the source and target languages have full-beta reduction semantics.)

Answer:

$$\begin{aligned}
\mathcal{T}[\![x]\!] &= x \\
\mathcal{T}[\![\lambda x.\, e]\!] &= \lambda x.\, \mathcal{T}[\![e]\!] \\
\mathcal{T}[\![e_1\ e_2]\!] &= \mathcal{T}[\![e_1]\!]\ \mathcal{T}[\![e_2]\!] \\
\mathcal{T}[\![\mathsf{true}]\!] &= \lambda x.\, \lambda y.\, x \\
\mathcal{T}[\![\mathsf{false}]\!] &= \lambda x.\, \lambda y.\, y \\
\mathcal{T}[\![e_1 \text{ and } e_2]\!] &= (\lambda b_1.\, \lambda b_2.\, b_1\ b_2\ \mathcal{T}[\![\mathsf{false}]\!])\ \mathcal{T}[\![e_1]\!]\ \mathcal{T}[\![e_2]\!] \\
\mathcal{T}[\![0]\!] &= \lambda f.\, \lambda x.\, x \\
\mathcal{T}[\![1]\!] &= \lambda f.\, \lambda x.\, f\ x \\
\mathcal{T}[\![2]\!] &= \lambda f.\, \lambda x.\, f\ (f\ x) \\
\mathcal{T}[\![e_1 + e_2]\!] &= (\lambda n_1.\, \lambda n_2.\, n_1\ \mathcal{T}[\![SUCC]\!]\ n_2)\ \mathcal{T}[\![e_1]\!]\ \mathcal{T}[\![e_2]\!] \\
\mathcal{T}[\![\text{if } e_1 \text{ then } e_2 \text{ else } e_3]\!] &= \mathcal{T}[\![e_1]\!]\ \mathcal{T}[\![e_2]\!]\ \mathcal{T}[\![e_3]\!] \\
\\
\mathcal{T}[\![SUCC]\!] &= \lambda n.\, \lambda f.\, \lambda x.\, f\ (n\ f\ x)
\end{aligned}$$

Note that if our target language had, say, CBV semantics, the translation of if e_1 then e_2 else e_3 *would evaluate both branches i.e., would evaluate both* $\mathcal{T}[\![e_2]\!]$ *and* $\mathcal{T}[\![e_3]\!]$*. Why would this be undesirable? (Hint: think about* **if** false **then** Ω **else** 0*...) How could you change the translation of conditionals to avoid this issue?*

2 Evaluation context

Consider the lambda calculus with let expressions and pairs (§1.3 of Lecture 9), and a semantics defined using evaluation contexts. For each of the following expressions, show one step of evaluation. Be clear about what the evaluation context is.

(a) $(\lambda x.\, x)\ (\lambda y.\, y)\ (\lambda z.\, z)$

Answer:

$$E = [\cdot](\lambda z.\, z)$$

$$E[(\lambda x.\, x)\ (\lambda y.\, y)] \longrightarrow (\lambda y.\, y)(\lambda z.\, z)$$

(b) let $x = 5$ in $(\lambda y.\, y + x)\ 9$

Answer:

$$E = [\cdot]$$

$$E[\textit{let}\, x = 5\ \textit{in}\ (\lambda y.\, y + x)\ 9] \longrightarrow (\lambda y.\, y + 5)\ 9$$

(c) $(4, ((\lambda x.\, x)\ 8, 9))$

Answer:

$$E = (4, ([\cdot], 9))$$

$$E[(\lambda x.\, x)\ 8] \longrightarrow (4, (8, 9))$$

(d) let $x = \#1\ ((\lambda y.\, y)\ (3, 4))$ in $x + 2$

Answer:

$$E = \textit{let}\, x = \#1\ [\cdot]\ \textit{in}\ x + 2)$$

$$E[(\lambda y.\, y)\ (3, 4)] \longrightarrow \textit{let}\, x = \#1\ (3, 4)\ \textit{in}\ x + 2)$$

3 References

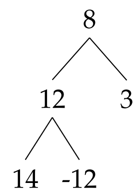
(a) Evaluate the following program. (That is, show the sequence of configurations that the small-step evaluation of the program will take. The initial store should by $\emptyset$, i.e., the partial function with an empty domain.)

$$\text{let } a = \text{ref } 17 \text{ in let } b = \text{ref } !a \text{ in } !b + (b := 8)$$

Answer:

$$\begin{aligned}
\langle \textit{let}\, a = \textit{ref}\, 17\ \textit{in}\ \textit{let}\, b = \textit{ref}\, !a\ \textit{in}\ !b + (b := 8), \emptyset\rangle &\longrightarrow \langle \textit{let}\, b = \textit{ref}\, !a\ \textit{in}\ !b + (b := 8), \{(a, 17)\}\rangle \\
&\longrightarrow \langle \textit{let}\, b = \textit{ref}\, 17\ \textit{in}\ !b + (b := 8), \{(a, 17)\}\rangle \\
&\longrightarrow \langle !b + (b := 8), \{(a, 17), (b, 17)\}\rangle \\
&\longrightarrow \langle 17 + (b := 8), \{(a, 17), (b, 17)\}\rangle \\
&\longrightarrow \langle 17 + 8, \{(a, 17), (b, 8)\}\rangle \\
&\longrightarrow \langle 25, \{(a, 17), (b, 8)\}\rangle
\end{aligned}$$

(b) Construct a program that represents the following binary tree, where an interior node of the binary tree is represented by a value of the form $(v, (\ell_{left}, \ell_{right}))$, where v is the value of the node, ℓ_{left} is a location that contains the left child, and ℓ_{right} is a location that contains the right child.



(It may be useful to define a function that creates internal nodes. Feel free to use let expressions to make your program easier to read and write.)

Answer: *Following the hint, we define a function that creates internal nodes:*

$$\textit{let}\, makeNode = \lambda r.\, \lambda t_l.\, \lambda t_r.\, (r, (\textit{ref}\, t_l, \textit{ref}\, t_r))\ \textit{in}\ \cdots$$

and now the binary tree above can be constructed by filling in the $\cdots$ with:

$$\textit{let}\, t' = makeNode\ 12\ 14\ (-12)\ \textit{in}\ makeNode\ 8\ t'\ 3$$

4 Continuations

(a) Suppose we add let expressions to our CBV lambda-calculus. How would you define $\mathcal{CPS}[\![\mathsf{let}\ x = e_1\ \mathsf{in}\ e_2]\!]$? (Note, even though $\mathsf{let}\ x = e_1\ \mathsf{in}\ e_2$ is equivalent to $(\lambda x.\, e_2)\ e_1$, don't use $\mathcal{CPS}[\![(\lambda x.\, e_2)\ e_1]\!]$, as there is a better CPS translation of $\mathsf{let}\ x = e_1\ \mathsf{in}\ e_2$. Why is that?)

Answer:

$$\mathcal{CPS}[\![\textit{let}\, x = e_1\ \textit{in}\, e_2]\!]k = \mathcal{CPS}[\![e_1]\!]\ (\lambda x.\, \mathcal{CPS}[\![e_2]\!]\ k)$$

(b) Translate the expression $\mathsf{let}\ f = \lambda x.\, x + 1\ \mathsf{in}\ (f\ 19) + (f\ 21)$ into continuation-passing style. That is, what is $\mathcal{CPS}[\![\mathsf{let}\ f = \lambda x.\, x + 1\ \mathsf{in}\ (f\ 19) + (f\ 21)]\!]$?

(Use your definition of $\mathcal{CPS}[\![\mathsf{let}\ x = e_1\ \mathsf{in}\ e_2]\!]$ from above.)

Answer: *Apologies for the small font. You can use the zoom feature in your PDF for better clarity.*

$$
\begin{aligned}
&\mathcal{CPS}[\![\textit{let } f = \lambda x.\, x + 1 \textit{ in } (f\ 19) + (f\ 21)]\!]k \\
&= \mathcal{CPS}[\![\lambda x.\, x + 1]\!]\ (\lambda f.\, \mathcal{CPS}[\![(f\ 19) + (f\ 21)]\!]k) \\
&= \mathcal{CPS}[\![\lambda x.\, x + 1]\!]\ (\lambda f.\, \mathcal{CPS}[\![(f\ 19) + (f\ 21)]\!]k) \\
&= (\lambda f.\, \mathcal{CPS}[\![(f\ 19) + (f\ 21)]\!]k)\ (\lambda x, k'.\, \mathcal{CPS}[\![x + 1]\!]k') \\
&= (\lambda f.\, \mathcal{CPS}[\![(f\ 19)]\!]\ (\lambda v.\, \mathcal{CPS}[\![(f\ 21)]\!]\ (\lambda w.\, k\ (v + w))))\ (\lambda x, k'.\, \mathcal{CPS}[\![x + 1]\!]k') \\
&= (\lambda f.\, \mathcal{CPS}[\![(f\ 19)]\!]\ (\lambda v.\, \mathcal{CPS}[\![(f\ 21)]\!]\ (\lambda w.\, k\ (v + w))))\ (\lambda x, k'.\, \mathcal{CPS}[\![x]\!]\ (\lambda v.\, \mathcal{CPS}[\![1]\!]\ (\lambda w.\, k'(v + w)))) \\
&= (\lambda f.\, \mathcal{CPS}[\![(f\ 19)]\!]\ (\lambda v.\, \mathcal{CPS}[\![(f\ 21)]\!]\ (\lambda w.\, k\ (v + w))))\ (\lambda x, k'.\, (\lambda v.\, (\lambda w.\, k'(v + w))\ 1)\ x) \\
&= (\lambda f.\, \mathcal{CPS}[\![f]\!]\ (\lambda f'.\, \mathcal{CPS}[\![19]\!]\ (\lambda v.\, f'\ v\ (\lambda v.\, \mathcal{CPS}[\![(f\ 21)]\!]\ (\lambda w.\, k\ (v + w))))))\ (\lambda x, k'.\, (\lambda v.\, (\lambda w.\, k'(v + w))\ 1)\ x) \\
&= (\lambda f.\, (\lambda f'.\, (\lambda v.\, f'\ v\ (\lambda v'.\, \mathcal{CPS}[\![(f\ 21)]\!]\ (\lambda w'.\, k\ (v' + w'))))\ 19)\ f)\ (\lambda x, k'.\, (\lambda v.\, (\lambda w.\, k'\ (v + w))\ 1)\ x) \\
&= (\lambda f.\, (\lambda f'.\, (\lambda v.\, f'\ v\ (\lambda v'.\, \lambda f''.\, \lambda v''.\, f''\ v''\ (\lambda w'.\, k\ (v' + w'))\ 21\ f'))\ 19)\ f)\ (\lambda x, k'.\, (\lambda v.\, (\lambda w.\, k'\ (v + w))\ 1)\ x) \\
&\longrightarrow (\lambda f'.\, (\lambda v.\, f'\ v\ (\lambda v'.\, \lambda f''.\, \lambda v''.\, f''\ v''\ (\lambda w'.\, k\ (v' + w'))\ 21\ f'))\ 19)\ (\lambda x, k'.\, (\lambda v.\, (\lambda w.\, k'\ (v + w))\ 1)\ x) \\
&\longrightarrow (\lambda v.\, (\lambda x, k'.\, (\lambda v.\, (\lambda w.\, k'\ (v + w))\ 1)\ x)\ v\ (\lambda v'.\, \lambda f''.\, \lambda v''.\, f''\ v''\ (\lambda w'.\, k\ (v' + w'))\ 21\ (\lambda x, k'.\, (\lambda v.\, (\lambda w.\, k'\ (v + w))\ 1)\ x)))\ 19 \\
&\longrightarrow (\lambda x, k'.\, (\lambda v.\, (\lambda w.\, k'\ (v + w))\ 1)\ x)\ 19\ (\lambda v'.\, \lambda f''.\, \lambda v''.\, f''\ v''\ (\lambda w'.\, k\ (v' + w'))\ 21\ (\lambda x, k'.\, (\lambda v.\, (\lambda w.\, k'\ (v + w))\ 1)\ x)) \\
&\longrightarrow (\lambda v.\, (\lambda w.\, (\lambda v'.\, \lambda f''.\, \lambda v''.\, f''\ v''\ (\lambda w'.\, k\ (v' + w'))\ 21\ (\lambda x, k'.\, (\lambda v.\, (\lambda w.\, k'\ (v + w))\ 1)\ x))\ (v + w))\ 1)\ 19 \\
&\longrightarrow (\lambda w.\, (\lambda v'.\, \lambda f''.\, \lambda v''.\, f''\ v''\ (\lambda w'.\, k\ (v' + w'))\ 21\ (\lambda x, k'.\, (\lambda v.\, (\lambda w.\, k'\ (v + w))\ 1)\ x))\ (19 + w))\ 1 \\
&\longrightarrow (\lambda v'.\, \lambda f''.\, \lambda v''.\, f''\ v''\ (\lambda w'.\, k\ (v' + w'))\ 21\ (\lambda x, k'.\, (\lambda v.\, (\lambda w.\, k'\ (v + w))\ 1)\ x))\ (19 + 1) \\
&\longrightarrow (\lambda v'.\, \lambda f''.\, \lambda v''.\, f''\ v''\ (\lambda w'.\, k\ (v' + w'))\ 21\ (\lambda x, k'.\, (\lambda v.\, (\lambda w.\, k'\ (v + w))\ 1)\ x))\ 20 \\
&\longrightarrow (\lambda f''.\, \lambda v''.\, f''\ v''\ (\lambda w'.\, k\ (20 + w'))\ 21\ (\lambda x, k'.\, (\lambda v.\, (\lambda w.\, k'\ (v + w))\ 1)\ x) \\
&\longrightarrow \lambda v''.\, (\lambda x, k'.\, (\lambda v.\, (\lambda w.\, k'\ (v + w))\ 1)\ x)\ v''\ (\lambda w'.\, k\ (20 + w'))\ 21 \\
&\longrightarrow (\lambda x, k'.\, (\lambda v.\, (\lambda w.\, k'\ (v + w))\ 1)\ x)\ 21\ (\lambda w'.\, k\ (20 + w')) \\
&\longrightarrow (\lambda v.\, (\lambda w.\, (\lambda w'.\, k\ (20 + w'))\ (v + w))\ 1)\ 21 \\
&\longrightarrow (\lambda w.\, (\lambda w'.\, k\ (20 + w'))\ (21 + w))\ 1 \\
&\longrightarrow (\lambda w'.\, k\ (20 + w'))\ (21 + 1) \\
&\longrightarrow (\lambda w'.\, k\ (20 + w'))\ 22 \\
&\longrightarrow k\ (20 + 22) \\
&\longrightarrow k\ 42
\end{aligned}
$$