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

More types
Section and Practice Problems

Mar 10-11, 2016

---

1 Products and Sums

For these questions, use the lambda calculus with products and sums (Lecture 13§1.1).

(a) Write a program that constructs two values of type $\textbf{int} + (\textbf{int} \to \textbf{int})$, one using left injection, and one using right injection.

Answer:

$$\begin{array}{l} \mathit{let}\, a : \textbf{\textit{int}} + (\textbf{\textit{int}} \to \textbf{\textit{int}}) = \mathit{inl}_{\textbf{\textit{int}}+(\textbf{\textit{int}}\to\textbf{\textit{int}})}\ 3\ \mathit{in} \\ \mathit{inr}_{\textbf{\textit{int}}+(\textbf{\textit{int}}\to\textbf{\textit{int}})}\ \lambda x : \textbf{\textit{int}}.\, 3 \end{array}$$

(b) Write a function that takes a value of type $\textbf{int} + (\textbf{int} \to \textbf{int})$ and if the value is an integer, it adds 7 to it, and if the value is a function it applies the function to 42.

Answer:

$$\lambda a : \textbf{\textit{int}} + (\textbf{\textit{int}} \to \textbf{\textit{int}}).\ \mathit{case}\ a\ \mathit{of}\ \lambda y : \textbf{\textit{int}}.\, y + 7 \mid \lambda f : \textbf{\textit{int}} \to \textbf{\textit{int}}.\, f\ 42$$

(c) Give a typing derivation for the following program.

$$\lambda p : (\textbf{unit} \to \textbf{int}) \times (\textbf{int} \to \textbf{int}).\ \lambda x : \textbf{unit} + \textbf{int}.\ \mathsf{case}\ x\ \mathsf{of}\ \#1\ p \mid \#2\ p$$

Answer: *For brevity, let* $e_1 \equiv \lambda x : \textbf{\textit{unit}} + \textbf{\textit{int}}.\ \mathit{case}\ x\ \mathit{of}\ \#1\ p \mid \#2\ p$ *and let* $\Gamma = \{p : (\textbf{\textit{unit}} \to \textbf{\textit{int}}) \times (\textbf{\textit{int}} \to \textbf{\textit{int}}), x : \textbf{\textit{unit}} + \textbf{\textit{int}}\}$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash x : \textbf{\textit{unit}} + \textbf{\textit{int}}}\,\text{T-Var} \quad \dfrac{\dfrac{}{\Gamma \vdash p : (\textbf{\textit{unit}} \to \textbf{\textit{int}} \times \textbf{\textit{int}} \to \textbf{\textit{int}})}\,\text{T-Var}}{\Gamma \vdash \#1\ p : \textbf{\textit{unit}} \to \textbf{\textit{int}}}\,\text{T-LProj} \quad \dfrac{\dfrac{}{\Gamma \vdash p : (\textbf{\textit{unit}} \to \textbf{\textit{int}} \times \textbf{\textit{int}} \to \textbf{\textit{int}})}\,\text{T-Var}}{\Gamma \vdash \#2\ p : \textbf{\textit{int}} \to \textbf{\textit{int}}}\,\text{T-RProj}}{\Gamma \vdash \mathit{case}\ x\ \mathit{of}\ \#1\ p \mid \#2\ p : \textbf{\textit{int}}}\,\text{T-Case}}{p : (\textbf{\textit{unit}} \to \textbf{\textit{int}}) \times (\textbf{\textit{int}} \to \textbf{\textit{int}}) \vdash \lambda x : \textbf{\textit{unit}} + \textbf{\textit{int}}.\ \mathit{case}\ x\ \mathit{of}\ \#1\ p \mid \#2\ p : (\textbf{\textit{unit}} + \textbf{\textit{int}}) \to \textbf{\textit{int}}}\,\text{T-Abs}}{\vdash \lambda p : (\textbf{\textit{unit}} \to \textbf{\textit{int}}) \times (\textbf{\textit{int}} \to \textbf{\textit{int}}).\ e_1 : ((\textbf{\textit{unit}} \to \textbf{\textit{int}}) \times (\textbf{\textit{int}} \to \textbf{\textit{int}})) \to \textbf{\textit{int}}}\,\text{T-Abs}$$

(d) Write a program that uses the term in part (c) above to produce the value 42.

Answer: *We refer to the term in part (c) above as* f.

$$f\ (\lambda x : \textbf{\textit{unit}} \to \textbf{\textit{int}}.\, 42, \lambda x : \textbf{\textit{int}}.\, 41)\ \mathit{inl}_{\textbf{\textit{unit}}+\textbf{\textit{int}}}\ ()$$

2 Recursion

(a) Use the $\mu x.\ e$ expression to write a function that takes a natural number n and returns the sum of all even natural numbers less than or equal to n. (You can assume you have appropriate integer comparison operators, and also a modulus operator.)

Answer:

$$\mu f.\ \lambda n.\ \textbf{if}\ n \leq 0\ \textbf{then}\ 0\ \textbf{else if}\ (n\ \textit{mod}\ 2) = 0\ \textbf{then}\ n + f\ (n-2)\ \textbf{else}\ f\ (n-1)$$

(b) Try executing your program by applying it to the number 5.

Answer: *The program executes correctly and returns 6. For brevity, we will refer to the expression from the answer above as F.*

$$
\begin{aligned}
& F\ 5 \\
\longrightarrow & (\lambda n.\ \textbf{if}\ n \leq 0\ \textbf{then}\ 0\ \textbf{else if}\ (n\ \textit{mod}\ 2) = 0\ \textbf{then}\ n + F\ (n-2)\ \textbf{else}\ F\ (n-1))\ 5 \\
\longrightarrow & \textbf{if}\ 5 \leq 0\ \textbf{then}\ 0\ \textbf{else if}\ (5\ \textit{mod}\ 2) = 0\ \textbf{then}\ 5 + F\ (5-2)\ \textbf{else}\ F\ (5-1) \\
\longrightarrow & \textbf{if}\ \textit{false}\ \textbf{then}\ 0\ \textbf{else if}\ (5\ \textit{mod}\ 2) = 0\ \textbf{then}\ 5 + F\ (5-2)\ \textbf{else}\ F\ (5-1) \\
\longrightarrow & \textbf{if}\ (5\ \textit{mod}\ 2) = 0\ \textbf{then}\ 5 + F\ (5-2)\ \textbf{else}\ F\ (5-1) \\
\longrightarrow & \textbf{if}\ 1 = 0\ \textbf{then}\ 5 + F\ (5-2)\ \textbf{else}\ F\ (5-1) \\
\longrightarrow & \textbf{if}\ \textit{false}\ \textbf{then}\ 5 + F\ (5-2)\ \textbf{else}\ F\ (5-1) \\
\longrightarrow & F\ (5-1) \\
\longrightarrow & (\lambda n.\ \textbf{if}\ n \leq 0\ \textbf{then}\ 0\ \textbf{else if}\ (n\ \textit{mod}\ 2) = 0\ \textbf{then}\ n + F\ (n-2)\ \textbf{else}\ F\ (n-1))\ (5-1) \\
\longrightarrow & (\lambda n.\ \textbf{if}\ n \leq 0\ \textbf{then}\ 0\ \textbf{else if}\ (n\ \textit{mod}\ 2) = 0\ \textbf{then}\ n + F\ (n-2)\ \textbf{else}\ F\ (n-1))\ 4 \\
\longrightarrow & \textbf{if}\ 4 \leq 0\ \textbf{then}\ 0\ \textbf{else if}\ (4\ \textit{mod}\ 2) = 0\ \textbf{then}\ 4 + F\ (4-2)\ \textbf{else}\ F\ (4-1) \\
\longrightarrow & \textbf{if}\ \textit{false}\ \textbf{then}\ 0\ \textbf{else if}\ (4\ \textit{mod}\ 2) = 0\ \textbf{then}\ 4 + F\ (4-2)\ \textbf{else}\ F\ (4-1) \\
\longrightarrow & \textbf{if}\ (4\ \textit{mod}\ 2) = 0\ \textbf{then}\ 4 + F\ (4-2)\ \textbf{else}\ F\ (4-1) \\
\longrightarrow & \textbf{if}\ 0 = 0\ \textbf{then}\ 4 + F\ (4-2)\ \textbf{else}\ F\ (4-1) \\
\longrightarrow & \textbf{if}\ \textit{true}\ \textbf{then}\ 4 + F\ (4-2)\ \textbf{else}\ F\ (4-1) \\
\longrightarrow & 4 + F\ (4-2) \\
\longrightarrow & 4 + (\lambda n.\ \textbf{if}\ n \leq 0\ \textbf{then}\ 0\ \textbf{else if}\ (n\ \textit{mod}\ 2) = 0\ \textbf{then}\ n + F\ (n-2)\ \textbf{else}\ F\ (n-1))\ (4-2) \\
\longrightarrow & 4 + (\lambda n.\ \textbf{if}\ n \leq 0\ \textbf{then}\ 0\ \textbf{else if}\ (n\ \textit{mod}\ 2) = 0\ \textbf{then}\ n + F\ (n-2)\ \textbf{else}\ F\ (n-1))\ 2 \\
\longrightarrow & 4 + (\textbf{if}\ 2 \leq 0\ \textbf{then}\ 0\ \textbf{else if}\ (2\ \textit{mod}\ 2) = 0\ \textbf{then}\ 2 + F\ (2-2)\ \textbf{else}\ F\ (2-1)) \\
\longrightarrow & 4 + (\textbf{if}\ \textit{false}\ \textbf{then}\ 0\ \textbf{else if}\ (2\ \textit{mod}\ 2) = 0\ \textbf{then}\ 2 + F\ (2-2)\ \textbf{else}\ F\ (2-1)) \\
\longrightarrow & 4 + (\textbf{if}\ (2\ \textit{mod}\ 2) = 0\ \textbf{then}\ 2 + F\ (2-2)\ \textbf{else}\ F\ (2-1)) \\
\longrightarrow & 4 + (\textbf{if}\ 0 = 0\ \textbf{then}\ 2 + F\ (2-2)\ \textbf{else}\ F\ (2-1)) \\
\longrightarrow & 4 + (\textbf{if}\ \textit{true}\ \textbf{then}\ 2 + F\ (2-2)\ \textbf{else}\ F\ (2-1)) \\
\longrightarrow & 4 + (2 + F\ (2-2)) \\
\longrightarrow & 4 + (2 + (\lambda n.\ \textbf{if}\ n \leq 0\ \textbf{then}\ 0\ \textbf{else if}\ (n\ \textit{mod}\ 2) = 0\ \textbf{then}\ n + F\ (n-2)\ \textbf{else}\ F\ (n-1))\ (2-2)) \\
\longrightarrow & 4 + (2 + (\lambda n.\ \textbf{if}\ n \leq 0\ \textbf{then}\ 0\ \textbf{else if}\ (n\ \textit{mod}\ 2) = 0\ \textbf{then}\ n + F\ (n-2)\ \textbf{else}\ F\ (n-1))\ 0) \\
\longrightarrow & 4 + (2 + (\lambda n.\ \textbf{if}\ n \leq 0\ \textbf{then}\ 0\ \textbf{else if}\ (n\ \textit{mod}\ 2) = 0\ \textbf{then}\ n + F\ (n-2)\ \textbf{else}\ F\ (n-1))\ 0) \\
\longrightarrow & 4 + (2 + (\textbf{if}\ 0 \leq 0\ \textbf{then}\ 0\ \textbf{else if}\ (0\ \textit{mod}\ 2) = 0\ \textbf{then}\ 0 + F\ (0-2)\ \textbf{else}\ F\ (0-1))) \\
\longrightarrow & 4 + (2 + (\textbf{if}\ \textit{true}\ \textbf{then}\ 0\ \textbf{else if}\ (0\ \textit{mod}\ 2) = 0\ \textbf{then}\ 0 + F\ (0-2)\ \textbf{else}\ F\ (0-1)))
\end{aligned}
$$

$\longrightarrow 4 + (2 + (0))$
$\longrightarrow^* 6$

(c) Give a typing derivation for the following program. What happens if you execute the program?

$$\mu p\!:\!(\textbf{int} \to \textbf{int}) \times (\textbf{int} \to \textbf{int}).\ (\lambda n\!:\!\textbf{int}.\ n+1, \#1\ p)$$

Answer: *For brevity, we write τ_p for the type $(\textbf{int} \to \textbf{int}) \times (\textbf{int} \to \textbf{int})$.*

$$\text{T-Rec}\ \dfrac{\text{T-Pair}\ \dfrac{\text{T-Abs}\ \dfrac{\text{T-Sum}\ \dfrac{\text{T-Var}\ \dfrac{}{p\!:\!\tau_p, n\!:\!\textbf{int} \vdash n\!:\!\textbf{int}} \quad \text{T-Int}\ \dfrac{}{p\!:\!\tau_p, n\!:\!\textbf{int} \vdash 1\!:\!\textbf{int}}}{p\!:\!\tau_p, n\!:\!\textbf{int} \vdash n+1\!:\!\textbf{int}}}{p\!:\!\tau_p \vdash \lambda n\!:\!\textbf{int}.\ n+1\!:\!\textbf{int} \to \textbf{int}} \quad \text{T-Proj}\ \dfrac{\text{T-Var}\ \dfrac{}{p\!:\!\tau_p \vdash p\!:\!\tau_p}}{p\!:\!\tau_p \vdash \#1\ p\!:\!\textbf{int} \to \textbf{int}}}{p\!:\!\tau_p \vdash (\lambda n\!:\!\textbf{int}.\ n+1, \#1\ p)\!:\!\tau_p}}{\vdash \mu p\!:\!\tau_p.\ (\lambda n\!:\!\textbf{int}.\ n+1, \#1\ p)\!:\!\tau_p}$$

Now, if you actually tried to execute this expression under a Call-By-Name semantics, it would unfold the recursive expression to $(\lambda n\!:\!\textbf{int}.\ n+1, \#1\ P)$, where P is the recursive expression $\mu p\!:\!(\textbf{int} \to \textbf{int}) \times (\textbf{int} \to \textbf{int}).\ (\lambda n\!:\!\textbf{int}.\ n+1, \#1\ p)$. While the first element of the pair is a value, the second $\#2\ P$ is not, and so we would attempt to evaluate that expression. However, that requires evaluating the expression $P \equiv \mu p\!:\!(\textbf{int} \to \textbf{int}) \times (\textbf{int} \to \textbf{int}).\ (\lambda n\!:\!\textbf{int}.\ n+1, \#1\ p)$.

So, under Call-by-Name semantics, the program will not terminate.

3 References

(a) Give a typing derivation for the following program.

$$\begin{array}{l}\mathsf{let}\ a\!:\!\textbf{int ref} = \mathsf{ref}\ 4\ \mathsf{in}\\ \mathsf{let}\ b\!:\!(\textbf{int} \to \textbf{int})\ \textbf{ref} = \mathsf{ref}\ \lambda x\!:\!\textbf{int}.\ x+38\ \mathsf{in}\\ !b\ !a\end{array}$$

Answer: *For brevity, we will write e for the expression above, and e_b for the subexpression $\textbf{let}\ b : (\textbf{int} \to \textbf{int})\ \textbf{ref} = \textit{ref}\ \lambda x\!:\!\textbf{int}.\ x+38\ \textit{in}\ !b\ !a$*

$$\text{T-Let}\ \dfrac{\text{T-Alloc}\ \dfrac{\text{T-Int}\ \dfrac{}{\vdash 4\!:\!\textbf{int}}}{\vdash \textit{ref}\ 4\!:\!\textbf{int ref}} \quad \text{T-Let}\ \dfrac{\text{T-Alloc}\ \dfrac{\text{T-Abs}\ \dfrac{\vdots_1}{a\!:\!\textbf{int ref}, x\!:\!\textbf{int} \vdash x+38\!:\!\textbf{int}}}{a\!:\!\textbf{int ref} \vdash \lambda x\!:\!\textbf{int}.\ x+38\!:\!\textbf{int} \to \textbf{int}}}{a\!:\!\textbf{int ref} \vdash \textit{ref}\ \lambda x\!:\!\textbf{int}.\ x+38\!:\!(\textbf{int} \to \textbf{int})\ \textbf{ref}} \quad \dfrac{\vdots_2}{a\!:\!\textbf{int ref}, b\!:\!(\textbf{int} \to \textbf{int})\ \textbf{ref} \vdash !b\ !a\!:\!\textbf{int}}}{a\!:\!\textbf{int ref} \vdash e_b\!:\!\textbf{int}}}{\vdash e\!:\!\textbf{int}}$$

The subderivation marked $\vdots_1$ is:

$$\text{T-Add}\ \dfrac{\text{T-Var}\ \dfrac{}{a\!:\!\textbf{int ref}, x\!:\!\textbf{int} \vdash x\!:\!\textbf{int}} \quad \text{T-Int}\ \dfrac{}{a\!:\!\textbf{int ref}, x\!:\!\textbf{int} \vdash 38\!:\!\textbf{int}}}{a\!:\!\textbf{int ref}, x\!:\!\textbf{int} \vdash x+38\!:\!\textbf{int}}$$

The subderivation marked $\vdots_2$ is:

$$\text{T-APP}\ \dfrac{\text{T-DEREF}\ \dfrac{\text{T-VAR}\ \dfrac{}{\Gamma_{ab} \vdash b : (\textbf{\textit{int}} \to \textbf{\textit{int}})\ \textbf{\textit{ref}}}}{\Gamma_{ab} \vdash\, !b : \textbf{\textit{int}} \to \textbf{\textit{int}}} \qquad \text{T-DEREF}\ \dfrac{\text{T-VAR}\ \dfrac{}{\Gamma_{ab} \vdash a : \textbf{\textit{int ref}}}}{\Gamma_{ab} \vdash\, !a : \textbf{\textit{int}}}}{\Gamma_{ab} \vdash\, !b\ !a : \textbf{\textit{int}}}$$

where $\Gamma_{ab} = a : \textbf{\textit{int ref}}, b : (\textbf{\textit{int}} \to \textbf{\textit{int}})\ \textbf{\textit{ref}}$.

(b) Execute the program above for 4 small steps, to get configuration $\langle e, \sigma \rangle$. What is an appropriate Σ such that $\emptyset, \Sigma \vdash e : \tau$ and $\Sigma \vdash \sigma$?

Answer:

$$\begin{aligned}
&\langle \textit{let}\ a : \textbf{\textit{int ref}} = \textit{ref}\ 4\ \textit{in let}\ b : (\textbf{\textit{int}} \to \textbf{\textit{int}})\ \textbf{\textit{ref}} = \textit{ref}\ \lambda x : \textbf{\textit{int}}.\ x + 38\ \textit{in}\ !b\ !a, \emptyset \rangle \\
\longrightarrow &\langle \textit{let}\ a : \textbf{\textit{int ref}} = \ell_a\ \textit{in let}\ b : (\textbf{\textit{int}} \to \textbf{\textit{int}})\ \textbf{\textit{ref}} = \textit{ref}\ \lambda x : \textbf{\textit{int}}.\ x + 38\ \textit{in}\ !b\ !a, [\ell_a \mapsto 4] \rangle \\
\longrightarrow &\langle \textit{let}\ b : (\textbf{\textit{int}} \to \textbf{\textit{int}})\ \textbf{\textit{ref}} = \textit{ref}\ \lambda x : \textbf{\textit{int}}.\ x + 38\ \textit{in}\ !b\ !\ell_a, [\ell_a \mapsto 4] \rangle \\
\longrightarrow &\langle \textit{let}\ b : (\textbf{\textit{int}} \to \textbf{\textit{int}})\ \textbf{\textit{ref}} = \ell_b\ \textit{in}\ !b\ !\ell_a, [\ell_a \mapsto 4, \ell_b \mapsto \lambda x : \textbf{\textit{int}}.\ x + 38] \rangle \\
\longrightarrow &\langle !\ell_b\ !\ell_a, [\ell_a \mapsto 4, \ell_b \mapsto \lambda x : \textbf{\textit{int}}.\ x + 38] \rangle
\end{aligned}$$

An appropriate store typing context is $\Sigma = \ell_a \mapsto \textbf{\textit{int}}, \ell_b \mapsto \textbf{\textit{int}} \to \textbf{\textit{int}}$.