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

References and continuations; Simply-typed lambda calculus; Type soundness (Lectures 10-11) Section and Practice Problems

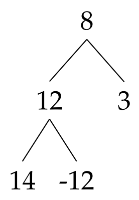
Feb 27-Mar 2, 2018

1 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.)

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

(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.)

2 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?)

(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.)

3 Simply-typed lambda calculus

(a) Add appropriate type annotations to the following expressions, and state the type of the expression.

(i) $\lambda a.\, a + 4$

(ii) $\lambda f.\, 3 + f\ ()$

(iii) $(\lambda x.\, x)\ (\lambda f.\, f\ (f\ 42))$

(b) For each of the following expressions, give a derivation showing that the expression is well typed.

(i) $(\lambda f\!:\!\mathbf{int} \to \mathbf{int}.\, f\ 38)\ (\lambda a\!:\!\mathbf{int}.\, a + 4)$

(ii) $\lambda g\!:\!(\mathbf{int} \to \mathbf{int}) \to (\mathbf{int} \to \mathbf{int}).\, g\ (\lambda c\!:\!\mathbf{int}.\, c + 1)\ 7$

(iii) $\lambda f\!:\!\mathbf{int} \to \mathbf{int}.\, \lambda g\!:\!\mathbf{int} \to \mathbf{int}.\, \lambda x\!:\!\mathbf{int}.\, g\ (f\ x)$

4 Type soundness

(a) Recall the substitution lemma that we used in the proof of type soundness.

Lemma (Substitution)**.** *If* $x : \tau' \vdash e : \tau$ *and* $\vdash v : \tau'$ *then* $\vdash e\{v/x\} : \tau$.

Using the definition of substitution given in Assignment 2, prove this lemma. You may assume that v does not have any free variables (i.e., $FV(v) = \emptyset$).

Remember to state what set you are performing induction on and what the property is that you are proving for every element in that set. If you are not sure what cases you need to consider, or what you are able to assume in each case of the inductive proof, we strongly suggest that you write down the inductive reasoning principle for the inductively defined set.

(b) Recall the context lemma that we used in the proof of type soundness.

Lemma (Context)**.** *If* $\vdash E[e_0] : \tau$ *and* $\vdash e_0 : \tau'$ *and* $\vdash e_1 : \tau'$ *then* $\vdash E[e_1] : \tau$.

Prove this lemma.

Remember to state what set you are performing induction on and what the property is that you are proving for every element in that set. If you are not sure what cases you need to consider, or what you are able to assume in each case of the inductive proof, we strongly suggest that you write down the inductive reasoning principle for the inductively defined set.