OpSem Theory
COMP105 Fall 2015

Phillip Booth

Problem 16

Here are the standard ImpCore inferences rules for `VAR(x)`:

$$\frac{x \in \mathsf{dom}\ \rho}{\langle \mathsf{VAR}(x),\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle \rho(x),\ \xi,\ \phi,\ \rho\rangle}$$

and

$$\frac{x \notin \mathsf{dom}\ \ \rho \qquad x \in \mathsf{dom}\ \xi}{\langle \mathsf{VAR}(x),\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle \xi(x),\ \xi,\ \phi,\ \rho\rangle}$$

(a) Awk-like semantics

If Var(x) is encountered

$$\cfrac{\cfrac{\cfrac{x \notin \mathsf{dom}\ \ \rho \quad x \notin \mathsf{dom}\ \xi}{\langle e,\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle v,\ \xi',\ \phi,\ \rho'\rangle}}{\langle \mathsf{SET}(x,\ 0),\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle v,\ \xi'\ \{x \mapsto 0\},\ \phi,\ \rho'\rangle}}{\langle \mathsf{VAR}(x),\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle \xi'(x),\ \xi',\ \phi,\ \rho'\rangle}$$

If Set(x) is encountered

$$\frac{x \notin \mathsf{dom}\ \ \rho\, x \notin \mathsf{dom}\ \xi\ \langle e,\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle v,\ \xi',\ \phi,\ \rho'\rangle}{\langle \mathsf{SET}(x,\ 0),\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle v,\ \xi' \quad \{x \mapsto 0\},\ \phi,\ \rho'\rangle}$$

(b) Icon-like semantics

If Var(x) is encountered

$$\cfrac{\cfrac{\cfrac{x \notin \mathsf{dom}\ \ \rho \quad x \notin \mathsf{dom}\ \xi}{\langle e,\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle v,\ \xi',\ \phi,\ \rho'\rangle}}{\langle \mathsf{SET}(x,\ 0),\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle v,\ \xi',\ \phi,\ \rho'\ \{x \mapsto 0\}\rangle}}{\langle \mathsf{VAR}(x),\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle \rho'(x),\ \xi',\ \phi,\ \rho'\rangle}$$

If Set(x) is encountered

$$\frac{x \notin \mathsf{dom}\ \ \rho\, x \notin \mathsf{dom}\ \xi\ \langle e,\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle v,\ \xi',\ \phi,\ \rho'\rangle}{\langle \mathsf{SET}(x,\ 0),\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle v,\ \xi',\ \phi,\ \rho' \quad \{x \mapsto 0\}\rangle}$$

(c) Which do you prefer and why?

I prefer the icon change because it has a more classic programming feel. Variables are restricted to the immediate scope.

Problem 13

$$\cfrac{\cfrac{\cfrac{\cfrac{x \in \mathsf{dom}\ \ \rho \qquad \rho(x)\ =\ 99}{\langle \mathsf{VAR}(x),\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle 99,\ \xi,\ \phi,\ \rho\rangle}}{\langle \mathsf{SET}(x,\ 3),\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle v,\ \xi',\ \phi,\ \rho'\ \{x \mapsto v\}\rangle}}{\langle \mathsf{VAR}(x),\ \xi',\ \phi,\ \rho'\rangle \Downarrow \langle \rho'(x),\ \xi',\ \phi,\ \rho'\rangle}}{\langle (begin(set\, x\, 3)\ x),\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle 3,\ \xi,\ \phi,\ \rho\rangle}$$

Problem 14

0.1. Case 1

$$\frac{\langle \mathsf{VAR}(x),\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle v_1,\ \xi,\ \phi,\ \rho\rangle \; v_1 \;=\; 0 \;\; \langle \mathsf{literal}(0),\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle v_1,\ \xi,\ \phi,\ \rho\rangle}{\langle \mathsf{if}(\mathit{var}(x)\ \mathit{var}(x)\ \mathsf{literal}(0)),\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle v_1,\ \xi,\ \phi,\ \rho\rangle}$$

0.2. Case 2

$$\frac{\langle \mathsf{VAR}(x),\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle v_1,\ \xi,\ \phi,\ \rho\rangle \; v_1 \; \textit{not equal to}\; 0 \;\; \langle \mathsf{VAR}(x),\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle v_1,\ \xi,\ \phi,\ \rho\rangle}{\langle \mathsf{if}(\mathit{var}(x)\ \mathit{var}(x)\ \mathsf{literal}(0)),\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle v_1,\ \xi,\ \phi,\ \rho\rangle}$$

In both cases v1 is equivalent to the value of the expression VAR(x), So because v2 is equivalent To the expression $\langle \mathsf{VAR}(x),\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle v_2,\ \xi,\ \phi,\ \rho\rangle$, v1 = v2

Problem 23

0.3. Formal Assign

By the induction hypothesis we can evaluate $\langle e,\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle v_1,\ \xi',\ \phi,\ \rho'\rangle$ using a stack and the evaluation will pop ρ and push ρ' without making a copy of ρ. ρ does not appear anywhere else in the rule, so it is safe to pop it and throw it away. ρ' is used only as part of the result of the rule. We can conslude that we can safely evaluate Set(x, e) on a stack and the evaluation effectively pops ρ which is never used again and pushes ρ'.

0.4. IFFalse

By the induction hypothesis we can evaluate $\langle e_1,\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle v_1,\ \xi',\ \phi,\ \rho'\rangle$ using a stack and the evaluation will pop ρ and push ρ' without making a copy of ρ. ρ does not appear anymwhere else in the rule, so it is safe to pop it and throw it away.

We can then use the induction hypothesis again to shoe that the evaluation of e3 can pop ρ' and push ρ'' and ρ' is not copied. ρ' Is not used anywhere else in the rule after the evaluation of e3. ρ'' is used only as part ofthe result of the rule. We can conslude that we can safely evaluate IF(e1, e2, e3) on a stack and the evaluation effectively pops ρ which is never used again and pushes ρ'.

0.5. While Iterate

By the induction hypothesis we can evaluate $\langle e_1,\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle v_1,\ \xi',\ \phi,\ \rho'\rangle$ using a stack and the evaluation will pop ρ and push ρ' without making a copy of ρ. ρ does not appear anymwhere else in the rule, so it is safe to pop it and throw it away.

We can then use the induction hypothesis again to shoe that the evaluation of e2 can pop ρ' and push ρ'' and ρ' is not copied. ρ' Is not used anywhere else in the rule after the evaluation of e2. Next the while expression is evaluated within the ρ'' environment, However, the induction rules hold, meaning just like ρ' was poped and ρ'' was pushed onto the stack, ρ'' is poped and ρ''' is pushed onto the stack and is used as a result of the rule. We can conslude that we can safely evaluate While(e1, e2) on a stack and the evaluation effectively pops ρ which is never used again and pushes ρ'''.

0.6. While End

By the induction hypothesis we can evaluate $\langle e,\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle v_1,\ \xi',\ \phi,\ \rho'\rangle$ using a stack and the evaluation will pop ρ and push ρ' without making a copy of ρ. ρ does not appear anywhere else in the rule, so it is safe to pop it and throw it away. ρ' is used only as part of the result of the rule. We can conslude that we can safely evaluate While(e1, e2) on a stack and the evaluation effectively pops ρ which is never used again and pushes ρ'.

0.7. Empty Begin

By the induction hypothesis we can evaluate $\langle e,\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle v_1,\ \xi',\ \phi,\ \rho'\rangle$ using a stack and the evaluation will not pop ρ. ρ is used only as part of the result of the rule. We can conslude that we can safely evaluate begin() on a stack and the evaluation does not pop ρ.

0.8. Begin

By the induction hypothesis we can evaluate $\langle e_1,\ \xi,\ \phi,\ \rho\rangle \Downarrow \langle v_1,\ \xi',\ \phi,\ \rho'\rangle$ using a stack and the evaluation will pop ρ and push ρ' without making a copy of ρ. ρ does not appear anymwhere else in the rule, so it is safe to pop it and throw it away.

We can then use the induction hypothesis again to shoe that the evaluation of e2 can pop ρ' and push ρ'' and ρ' is not copied. ρ' Is not used anywhere else in the rule after the evaluation of e3. ρ'' is used only as part ofthe result of the rule. We can conslude that we can safely evaluate Begin(e1, e2) on a stack and the evaluation effectively pops ρ which is never used again and pushes ρ''. If there happened to be more then two e's in the begin exspresion then the induction hypothesis helps to show that the stack and evaluation effectively pops ρ^n-1 which is never used again and pushes ρ^n.