CS 330 Lecture 15

- More operational Semantics

Control Flow

IFTRUE:

$$\frac{\langle e_1,g,f,p\rangle \Rightarrow \langle v_1,g',f,p'\rangle \quad v_1 \mathrel{!=} 0 \quad \langle e_2,g',f,p'\rangle \Rightarrow \langle v_2, g'', f, p''\rangle}{\langle IF(e_1,e_2,e_3),g,f,p\rangle \Rightarrow \langle v_2, g'', f, p''\rangle}$$

IFFALSE:

$$\frac{\langle e_1,g,f,p\rangle \Rightarrow \langle v_1,g',f,p'\rangle \quad v_1 = 0 \quad \langle e_3,g',f,p'\rangle \Rightarrow \langle v_3, g'', f, p''\rangle}{\langle IF(e_1,e_2,e_3),g,f,p\rangle \Rightarrow \langle v_3, g'', f, p''\rangle}$$

Loops

WHILE-ITERATE

$$\frac{\begin{array}{l}\langle e_1,g,f,p\rangle \Rightarrow \langle v_1, g', f, p'\rangle \quad v_1 \mathrel{!=} 0 \\ \langle e_2,g', f, p'\rangle \Rightarrow \langle v_2, g'', f, p''\rangle \quad \langle WHILE(e_1,e_2, g'', f, p')'\rangle \Rightarrow \langle v_3,g''',f, p'''\rangle\end{array}}{\langle WHILE(e_1,e_2,g,f,p)\rangle \Rightarrow \langle v_3, g''', f, p'''\rangle}$$

WHILEEND

$$\frac{\langle e_1, g, f, p \rangle \Rightarrow \langle v_1, g', f, p'\rangle \quad v_1 = 0}{WHILE(e_1,e_2), g, f, p\rangle \Rightarrow \langle 0,g', f, p'\rangle}$$

e_2 evaluated only for side effects

Sequential Execution

$$\frac{}{\langle BEGIN(), g, f, p\rangle \Rightarrow \langle 0, g, f, p\rangle}$$

order of expressions matters
order of premises doesn't

$$\frac{\begin{array}{l}\langle e_1, g_0, f, p_0\rangle \Rightarrow \langle v_1, g_1, f, p_1\rangle \\ \langle e_2, g_1, f, p_1\rangle \Rightarrow \langle v_2, g_2, f, p_2\rangle \\ .... \\ .... \\ \langle e_n, g_{n-1}, f, p_{n-1}\rangle \Rightarrow \langle v_n, g_n, f, p_n\rangle\end{array}}{\langle BEGIN(e_1,e_2,....e_n), g_0, f, p_0)\rangle \Rightarrow \langle v_n, g_n, f, p_n\rangle}$$

Function Application

$f(foo) = USER(<x_1,.....x_n>, e)$

$x_1,...., x_n$ all distinct

$<e_1, g_0, f, p_0> => <v_1, g_1, f, p_1>$

...

$<e_n, g_{n-1},f, p_{n-1}> => <v_n, g_n, f, p_n>$

$<e, g_n, f, \{x_1->v_1, ... x_n->v_n\}> => <v, g', f, p'>$

-- APPLY USER

$<APPLY(foo, e_1, ..., e_n), g_0, f, p_0> => <v, g', f, p_n>$

behavior of function doesn't depend on function name, only definition
body of a function can't get the formal parameters of its caller
functions assigns to formal parameters changes are not visible outside
p' can be thrown away after evaluation => VERY IMPORTANT

Primitive Functions

f(foo) = PRIMITIVE(+)

$<e_1, g_0, f, p_0> => <v_1,g_1,f, p_1>$

$<e_2, g_1, f, p_1> => <v_2,g_2,f, p_2>$

$<APPLY(foo, e_1, e_2), g_0, f, p_0> => <v_1+v_2, g_2, f, p_2>$

APPLYADD

Top-level items

<e, g, f, { }> => <v, g', f, p'>

---------------------------------- EVALEXP

<EXP(e), g, f> -> <g', f>

<e, g, f, { }> => <v, g', f, p'>

--- DEFINEGLOBAL

<VAL(x,e),g,f> -> <g'{x->v}, f>

x1, x2,... xn all distinct

---------------------------- DEFINE FUNCTION

$<DEFINE(foo, <x_1,..., x_n>, e>, e), g, f> -> \{g, f->USER(<x_1,..x_n>, e)\}>$