Lecture 17: PCF Interpreters

CSC 131
Fall, 2014

Kim Bruce

PCF Syntax & Semantics

```
e ::= x | n | true | false | succ | pred | iszero |
      if e then e else e | (fn x => e) | (e e) |
      rec x => e | let x = e1 in e2 end
```

(1) n ⇒ n *for n an integer.*

(2) true ⇒ true, false ⇒ false

(3) error ⇒ error

(4) succ ⇒ succ, *and similarly for the other initial functions*

$$(5)\ \frac{b \Rightarrow \text{true} \qquad e1 \Rightarrow v}{\text{if } b \text{ then } e1 \text{ else } e2 \Rightarrow v}$$

More PCF Semantics

$$(6)\ \frac{b \Rightarrow \text{false} \qquad e2 \Rightarrow v}{\text{if } b \text{ then } e1 \text{ else } e2 \Rightarrow v}$$

$$(7)\ \frac{e1 \Rightarrow \text{succ} \qquad e2 \Rightarrow n}{(e1\ e2) \Rightarrow (n+1)}$$

$$(8)\ \frac{e1 \Rightarrow \text{pred} \quad e2 \Rightarrow 0}{(e1\ e2) \Rightarrow 0} \qquad \frac{e1 \Rightarrow \text{pred} \quad e2 \Rightarrow (n+1)}{(e1\ e2) \Rightarrow n}$$

$$(9)\ \frac{e1 \Rightarrow \text{iszero} \quad e2 \Rightarrow 0}{(e1\ e2) \Rightarrow \text{true}} \qquad \frac{e1 \Rightarrow \text{iszero} \quad e2 \Rightarrow (n+1)}{(e1\ e2) \Rightarrow \text{false}}$$

More PCF Semantics

$$(10)\ \frac{}{(\text{fn } x \Rightarrow e) \Rightarrow (\text{fn } x \Rightarrow e)}$$

$$(11)\ \frac{e1 \Rightarrow (\text{fn } x \Rightarrow e3) \qquad e2 \Rightarrow v1 \qquad e3[x:=v1] \Rightarrow v}{(e1\ e2) \Rightarrow v}$$

Call by value!

$$(12)\ \frac{e[x:=\text{rec } x \Rightarrow e] \Rightarrow v}{(\text{rec } x \Rightarrow e) \Rightarrow v}$$

Like Y combinator!

Recursion

```
f n  = if (n == 0) then 1 else n*(f(n-1)
```

is written in PCF (assuming have already defined `mult`) as

```
rec f => fn n => if (isZero n) then 1
                                else mult n (f (pred n))
```

which is equivalent to

```
Y(λf. λn. cond (isZero n) 1 (mult n (f (pred n))))
```

Computed via unwinding.

Substitution-based Interpreter

```
data Term = AST_ID String | AST_NUM Int | AST_BOOL Bool
          | AST_SUCC | AST_PRED | AST_ISZERO
          | AST_IF (Term, Term, Term) | AST_ERROR String
          | AST_FUN (String, Term) | AST_APP (Term, Term)
          | AST_REC (String, Term)
```

- Key is to get right definition of substitution that matches static scope
- Interpreter code matches semantic rules
  - PCFSubstInterpreter.hs

PCF Semantics w/Environments

- Substitution slow & space consuming
- Can't handle terms w/free variables
- Environment allows to evaluate once.
- Meaning now separate set of values -- not just rewriting
- Meaning of function is closure, which carries around its environment of definition.

PCF Syntax & Semantics with Environments

```
env:: string -> value
```

(0) (id, env) ⇒ env(id)

(1) (n, env) ⇒ n *for n an integer.*

(2) (true, env) ⇒ true, (false, e) ⇒ false

(3) (error, env) ⇒ error

(4) (succ, env) ⇒ succ, *similarly for other initial functions*

$$(5)\ \frac{(b, env) \Rightarrow true \qquad (e1, env) \Rightarrow v}{(\text{if } b \text{ then } e1 \text{ else } e2, env) \Rightarrow v}$$

More PCF Semantics

$$\text{(6)}\quad \frac{(b, env) \Rightarrow \text{false} \qquad (e2, env) \Rightarrow v}{(\text{if } b \text{ then } e1 \text{ else } e2, env) \Rightarrow v}$$

$$\text{(7)}\quad \frac{(e1, env) \Rightarrow \text{succ} \qquad (e2, env) \Rightarrow n}{((e1\ e2), env) \Rightarrow (n+1)}$$

(8) ...

(9) ...

Revised PCF Semantics

Closure(x,e,env)

$$\text{(10)}\quad ((\text{fn } x \Rightarrow e), env) \Rightarrow \langle \text{fn } x \Rightarrow e, env \rangle$$

$$\text{(11)}\quad \frac{(e1,env) \Rightarrow \langle \text{fn } x \Rightarrow e3, env' \rangle \qquad (e2, env) \Rightarrow v1 \qquad (e3, env'[v1/x]) \Rightarrow v}{((e1\ e2), env) \Rightarrow v}$$

Thunk(rec x => e, env)

$$\text{(12)}\quad \frac{(e, env[(\text{rec } x \Rightarrow e)/x]) \Rightarrow v}{((\text{rec } x \Rightarrow e), env) \Rightarrow v}$$