Lecture 7-8: Solutions of the Exercises

prof. Marco Bellia, Dip. Informatica, Università di Pisa

March 25-28, 2014

- Let $\mathtt{F} \equiv \lambda\mathtt{f}.\lambda\mathtt{n}.\mathtt{if}(\mathtt{n}=0)\mathtt{then}\ 1\ \mathtt{else}\ \mathtt{n}*\mathtt{f}(\mathtt{n}-1)$ be a functional.
  Notationa remark: $\mathtt{Y\ F}$ is a contraction for: $\mathtt{Yf.F}$
- **To compute $\mathtt{Y}$:**
  Intensional computation. $\forall \mathtt{x} : \mathtt{Y\ F}(\mathtt{x}) = \mathtt{F}\ (\mathtt{Y\ F})(\mathtt{x})$
  Extensional computation. $\mathtt{Y\ F} = \mathtt{Lim}_{\mathtt{n}\in\mathtt{Nat}}(\mathtt{Y\ F})^{\mathtt{n}}$ where:
  $$(\mathtt{Y\ F})^0 = \mathtt{F}(\perp)$$
  $$(\mathtt{Y\ F})^{\mathtt{n}} = \mathtt{F}((\mathtt{Y\ F})^{\mathtt{n}-1})$$

Example

Complete the extensional c. of $\mathtt{Y\ F}\ (2)$ and show how the result has been obtained

Intensional:

$$\mathtt{Y\ F}\ (2) = \mathtt{if}(2=0)\mathtt{then}\ 1\ \mathtt{else}\ 2*\mathtt{Y\ F}\ (1)$$
$$= ... = 2*1*1$$

Extensional:

$\mathtt{Y\ F}^0 = \lambda\mathtt{n}.\mathtt{if}(\mathtt{n}=0)\mathtt{then}\ 1\ \mathtt{else}\ \perp$

$\mathtt{Y\ F}^1 = \lambda\mathtt{n}.\mathtt{if}(\mathtt{n}=0)\mathtt{then}\ 1\ \mathtt{else}\ (\mathtt{if}(\mathtt{n}=1)\mathtt{then}\ 1\ \mathtt{else}\perp)$

$\mathtt{Y\ F}^2 = ...$

Lecture7-8-2014: Slide 12-13

let $\mathcal{D}[\![\texttt{Proc I() C}]\!]_\rho = \texttt{bind}(\texttt{I}, \mathcal{M}[\![\texttt{C}]\!]_\rho, \rho)$.
What is the relation between Static Chain and the ρ?

Complete in the case of Dynamic Scope:
$\mathcal{D}[\![\texttt{Proc I() C}]\!]_\rho = ...$

and answer:

- is Dynamic Chain known at the time of the proc. declaration (i.e. compile time)?
- is Dynamic Chain of a procedure known before the proc. invocation?
- is Dynamic Chain of a procedure the same for all the proc. invocation?

Lecture7-8-2014: Slide 17-18

Example

Exercise 1.
The formula written for parallel declaration is:

$$\mathcal{D}[\![\mathtt{Mut\ D_1\ D_2\ Ally}]\!]_\rho = \mathtt{Y}\mu.\mathcal{D}[\![\mathtt{D_2}]\!](\mathcal{D}[\![\mathtt{D_1}]\!](\rho)(\mu))$$

This writing contains a small bug.
(a) Can you find it?
(b) Do you know how to correct it?
(c) Which consequences in letting the formula unchanged?

Exercise 2.
(a) Do You recognize the language used in the interactive sessions below?
(b)
let rec x = fun u → u + y and y = 5 in x(3);;
- ... what will be printed here?
(c)
let rec x = fun u → y(u) and y = fun u → x(u) in x;;
- ... what will be printed here?
(d)
let rec onetwo = 1::twoone and twoone=2::onetwo in List.nth onetwo 5;;
- ... what will be printed here?
(e)
let rec onetwo = 1::twoone and twoone=2::onetwo in twoone;;
- ... what will be printed here?

Lecture7-8-2014: Slide 19

- Apply the definitions to the declaration below, in the example. To do it:
  - Correct: (a) formula for g and (b) formula for $\mathtt{Y\ H}^0$;
  - Complete the text.

Example

Let A and B two identifiers. Show the bindings of A and B that the following fragment defines:

```
Mut
  Proc A() {Call B();}
  Proc B() {Call A();}
Ally
```

$$g \equiv \mathtt{Y}\mu.\lambda\sigma.\lambda\mu.\mathtt{bind}(\mathtt{B}, \mathcal{M}[\![\{\mathtt{Call\ A();}\}]\!]_\mu, \sigma)(\lambda\sigma.\lambda\mu.\mathtt{bind}(\mathtt{A}, \ldots)(\rho)(\mu))$$

Compute the first 3 approximations to the solution of the functional:

$$\mathtt{H} \equiv \lambda\mu.\mathtt{bind}(\mathtt{B}, \mathcal{M}[\![\{\mathtt{Call\ A();}\}]\!]_\mu, \mathtt{bind}(\mathtt{A}, \mathcal{M}[\![\{\mathtt{Call\ B();}\}]\!]_\mu, \rho))$$

At the starting step:

$$\begin{aligned}
\mathtt{Y\ H}^0 &= \mathtt{H}(\perp) \\
&= \mathtt{bind}(\mathtt{B}, \mathcal{M}[\![\{\mathtt{Call\ A();}\}]\!]_\mu, \mathtt{bind}(\mathtt{A}, \mathcal{M}[\![\{\mathtt{Call\ B();}\}]\!]_\mu, \rho)) \\
\mathtt{Y\ H}^1 &= \mathtt{H}(\mathtt{Y\ H}^0) \\
&= \ldots \\
\mathtt{Y\ H}^2 &= \mathtt{H}(\mathtt{Y\ H}^1) \\
&= \ldots
\end{aligned}$$

Lecture7-8-2014: Slide 20-21

Example

Show the environment Env when the semantics of mutually applies to the fragment

```
...
{...
 Mut
   int x = y;
   int y = 3;
 Ally
...
```

Lecture 9-10: Solutions of the Exercises

prof. Marco Bellia, Dip. Informatica, Università di Pisa

March 28, 2014

Example

The following Haskell expression, `h 3 (f 5)`, when `h` e `f` are:

```
h = \x y -> if (x\=0) then x else y
f n = f(n+1)
```

evaluates to 3.

Can You rephrase it in OCaml, C or Java?

Lecture 9-10: slide 12

Example

The Haskell expression `g[4,(f 5)]`, when `h` and `f` are:

```
g u = if ((head u)==0) then 3 else 7
f n = f(n+1)
```

computes 7.
Can You rephrase it in OCaml, C or in Java?

Lecture 9-10: slide 14

Example

Finitely Approximated, Infinite Values:

```
nat n = n:nat(n+1)
naturali = nat 0
v = take 3 naturali
```

In Haskell, the 3 expressions above, compute one function, the infinite list of naturals, the list of the first 3 naturals.
Can You rephrase it in Caml, C or Java?

Example

Finitary Infinite values:

`data Tree a = T(a,Tree a)` - it defines a polymorphic type of Haskell
`treeM = T(3,treeM)` - a value of Haskell

treeM computes a infinite tree that can be finitely represented (with pointers !?)
Can You rephrase it in Caml, C or Java?

Example

Consider the following C expression:

```
z = x = y
```

The abstract syntax of it, resulting from the compiler or interpreter front-end, in the notation adopted in the provious slides is:

```
Val(Den(z) = Val(Den(x) = Val(y)))
```

Do the same with the following C expression:

```
A[*v+j] = x = y + A[*v+1]
....
```

Lecture9-10: Slide 19

Example

Show the environment Env when the semantics of mutually applies to the fragment

```
Mut
  int x = y;
  final int y = 3;
Ally
```

Check definitions for bugs: (a) fix them; (b) motivate

Example

Semantic Functions

$$\mathcal{D}[\![\mathtt{D}]\!]_\rho : \mathtt{Store} \to (\mathtt{Env} \times \mathtt{Store})_\perp$$

$$\mathcal{D}[\![\mathtt{Var\ I;}\,]\!]_\rho$$
$$= \lambda s.\ \mathtt{Let}\{(l, s_l) = \mathit{allocate}(s)\}\ (\mathtt{bind}(\mathtt{I}, l, \rho), s)$$
$$\mathcal{D}[\![\mathtt{Var\ I} = \mathtt{E;}\,]\!]_\rho(s)$$
$$= \mathtt{Let}\{(v_e, s_e) = \mathcal{E}[\![\mathtt{E}]\!]_\rho(s)\}\ (\mathtt{bind}(\mathtt{I}, v_e, \rho), s_e)$$

Lecture 11: Solutions of the Exercises

prof. Marco Bellia, Dip. Informatica, Università di Pisa

March 28, 2014

1. Complete with the suitable definitions in order to run the following Ocaml codes

Example

```
let x = ref 0 in
 let pippo xr =
     function n -> xr := !xr + n in
  let pippo1 = pippo(x) in
   pippo1(3);
   print(!x);
   (let x = ref 0 in
     pippo1(3);
     print(!x));
   print(!x);
```

```
let x = ref 0 in
 let pippo xr =
     function n -> xr := !xr + n in
   pippo(x)(3);
   print(!x);
   (let x = ref 0 in
     pippo(x)(3);
     print(!x));
   print(!x);
```

2. Give definitions for the domain Env: Values and Operations
3. Give definitions for the domain Store: Values and Operations