Práctica 5

Galeana Araujo Emiliano (No. de Cuenta: 314340520)

and Miranda Sánchez Kevin Ricardo (No. de Cuenta: 314011163)

Facultad de Ciencias, UNAM

28 Noviembre 2018

En esta práctica se desarrollaron dos temas vistos en clase cuales son **Excepciones** y **Continuaciones**. Y se modifico el ultimo semanal de modo que pudiéramos crear 3 módulos. Estos son:

1. FEAB. Aquí la forma más simple de una excepción es un error que no tiene información asociada. Este error se puede interceptar y convertir en un cómputo correcto transfiriendo el control a otra expresión. Haciendo uso de las **Catch Expr Expr** y **Error** como extensión del lenguaje EAB que siguen la semántica dinámica:

$$\frac{}{k \triangleright error \to_{\mathcal{K}} k \blacktriangleleft error}$$

$$\frac{}{k \triangleright catch(e_1, e_2) \to_{\mathcal{K}} k; catch(-, e_2) \triangleright e_1}$$

$$\frac{}{k; catch(-, e_2) \triangleleft v \to_{\mathcal{K}} k \triangleright v}$$

$$\frac{}{k; catch(-, e_2) \blacktriangleleft error \to_{\mathcal{K}} k \triangleright e_2}$$

$$\frac{f \neq catch(-, e_2)}{k; f \blacktriangleleft error \to_{\mathcal{K}} k \blacktriangleleft error}$$

2. XEAB. Aquí el argumento de la expresión **Raise** se evaluará para determinar el valor que se pasará al manejador. La expresión **Handle** liga la variable x a la expresión manejadora. El valor asociado de la excepción estará ligado en la expresión manejadora en caso de que se genere una excepción cuando se evalúa la primera expresión.

$$\overline{k \rhd raise(e) \rightarrow_{\mathcal{K}} k; raise(-) \rhd e}$$

$$\overline{k; raise(-) \lhd v \rightarrow_{\mathcal{K}} k \blacktriangleleft raise(v)}$$

$$\overline{k \rhd handle(e_1, x.e_2) \rightarrow_{\mathcal{K}} k; handle(-, x.e_2) \rhd e_1}$$

$$\overline{k; handle(-, x.e_2) \lhd v \rightarrow_{\mathcal{K}} k \lhd v}$$

$$\overline{k; handle(-, x.e_2) \blacktriangleleft raise(v) \rightarrow_{\mathcal{K}} k \rhd e_2[x := v]}$$

$$\frac{f \neq handle(-, x.e_2)}{\overline{k; f \blacktriangleleft raise(v) \rightarrow_{\mathcal{K}} k \blacktriangleleft raise(v)}}$$

$$\overline{k \rhd write(s, e) \rightarrow_{\mathcal{K}} error_p(s, e)}$$

3. KEAB.Aquí la idea para las continuaciones es considerar a la pila de control de un programa como un valor el cual puede devolverse o pasarse como argumento a otra función.

$$\overline{k \rhd letCC(x.e) \rightarrow_{\mathcal{K}} k \rhd e[x := cont(k)]}$$

$$\overline{k \rhd continue(e_1, e_2) \rightarrow_{\mathcal{K}} k; continue(-, e_2) \rhd e_1}$$

$$\overline{k; continue(-, e_2) \lhd v_1 \rightarrow_{\mathcal{K}} k; continue(v_1, -) \rhd e_2}$$

$$\overline{k; continue(cont(k'), -) \lhd v_2 \rightarrow_{\mathcal{K}} k' \lhd v_2}$$

1. Entrada y ejecución

Para ejecutar el programa se debe estar ubicado el directorio src, abrir una terminal y escribir **ghci <archvio>.hs** donde archivo debe ser elegido entre FEAB.hs, XEAB.hs y KEAB.hs, inmediatamente después debe aparecer:

```
GHCi, version 7.10.3: http://www.haskell.org/ghc/  :? for help
[1 of 1] Compiling KEAB             ( KEAB.hs, interpreted )
Ok, modules loaded: KEAB.
*KEAB>
```

Como ejemplo del programa del archivo KEAB.hs:

```
*KEAB> eval (LetCC "k"(Succ (Continue (V "k")(Continue (V "k") (I 1)))))
N[1]
```

Como ejemplo del programa del archivo XEAB.hs:

```
*XEAB> eval (Handle (Add ( Let "x" (I 0) (If(Eq(I 0)(I 0))
(Raise(I 10))(Add(I 9)(V "x"))))(I(-1)))"h"(Mul(I 2)(V "h" )))
N[20]
```

Como ejemplo del programa del archivo FEAB.hs:

```
*FEAB> v t [] (Add (I 1) (Suc Error)) Integer
True
```

2. Conclusiones

Fue complicado programarlos, es notable la diferencia entre las definiciones teóricas y la implementación ya que para algunas funciones se debía encontrar algunos casos especiales. Por otro lado, el separar los temas y crear nuevos módulos nos ayudo a un mejor entendimiento de los temas.

Referencias

[1] lp191n12.pdf, lp191n13.pdf, Archivero, curso de Lenguajes de Programacion 2019-1.