Ejercicio Semanal 3

Emiliano Galeana Araujo

Facultad de Ciencias, UNAM

Fecha de entrega: 08 de Noviembre de 2018

1. Descripción del programa

En este semanal implementamos un lenguaje llamado miniC, el cual es un lenguaje imperativo que tiene como núcleo el lenguaje de expresiones aritmetico booleanas.

1.1. MiniC

Recordemos el lenguaje de EAB y agregamos el comportamiento de nuevos contrusctores para miniC.

```
data Exp = V identifier | I Int | B Bool
  | Add Exp Exp | Mul Exp Exp | Succ Exp | Pred Exp
  | And Exp Exp | Or Exp Exp  | Not Exp
  | Lt Exp Exp  | Gt Exp Exp  | Eq Exp Exp
  | If Exp Exp Exp
  | Let Identifier Exp Exp
  | L Int
  | Alloc Exp
  | Deref Exp
  | Assig Exp Exp
  | Void
  | Seq
  | While Exp Exp
```

Un ejemplo trivial de un programa en miniC se puede ver como sigue:

```
While (B True) Void
```

Agregamos nuevos tipos para poder representar la memoria y direcciones.

```
type Addr = Exp

type Value = Exp

type Cell = (Addr, Value)

type Mem = [Cell]
```

Las siguientes son funciones que implementamos para poder representar el lenguaje miniC.

```
-- | domain. Dada una memoria, obtiene todas sus direcciones.
domain :: Mem → [Int]

-- | newL. Dada una memoria genera una nueva direccion.
newL :: Mem → Addr

-- | accessM. Dada una direccion de memoria, devuelve el valor contenido en
-- |          la celda con tal direccion.
accessM :: Addr → Mem → Maybe Value

-- | updateM. Dada una celda de memoria, actualiza el valor.
updateM :: Cell → Mem → Mem

-- | frVars. Extiende esta funcion del lenguaje EAB con las nuevas funciones.
frVars :: Exp → [Identifier]

-- | subst. Extiende esta funcion del lenguaje EAB con las nuevas funciones.
subst :: Exp → Substitution → Exp

-- | eval1. Extiende esta funcion del lenguaje EAB con las nuevas funciones.
eval1 :: (Mem, Exp) → (Mem, Exp)

-- | evals. Extiende esta funcion del lenguaje EAB con las nuevas funciones.
evals :: (Mem, Exp) → (Mem, Exp)

-- | eval. Extiende esta funcion del lenguaje EAB con las nuevas funciones.
eval :: Exp → Exp
```

2. Entrada y ejecución

El programa es interpretado por `GCHI` de la siguiente forma

```
~:ghci EjerSem03.hs
```

2.1. Funciones

Ya en el programa, los siguientes son ejemplos de ejecuciones de las funciones antes mencionadas.

```
*EjerSem03> domain []
[]
*EjerSem03> domain [(L 0, B False), (L 2, I 9)]
[0,2]
*EjerSem03> newL []
[]
*EjerSem03> newL [(L 0, B False),(L 2, I 9)]
L 1
*EjerSem03> accessM (L 3) []
Nothing
*EjerSem03> accessM (L 2) [(L 0, I 21), (L 1, Void), (L 2, I 12)]
Just (I 12)
*EjerSem03>
*EjerSem03> updateM (L 3, B False) []
***Exception: Memory addres does not exit.
*EjerSem03> updateM (L 0, I 22) [(L 0, I 21),(L 1, Void),(L 2, I 12)]
[(L 0, I 22),(L 1, Void),(L 2, I 12)]
*EjerSem03> frVars (Assig (L 2) (Add (I 0) (V ``z'''')))
[``z'''']
*EjerSem03> subst (Assig (L 2) (Add (I 0) (V ``z'''')))  (``z'''', B False)
(Assig (L 2) (Add (I 0) (B False)))
*EjerSem03> eval1 ([(L 0, B False)], (Add (I 1) (I 2)))
([(L 0, B False)], I 3)
*EjerSem03> eval1 ([], While (B True) (Add (I 1) (I 1)))
([], If (B True) (Seq (Add (I 1) (I 1)) (While (B True) (Add (I 1) (I 1)))) Void)
*EjerSem03> evals ([], (Let ``x'''' (Add (I 1) (I 1)) (Eq (V ``x'''') (I 0))))
([], B False)
*EjerSem03> evals ([], Assig (Alloc (B False)) (Add (I 1) (I 9)))
([(L 0, I 10)], Void)
```

```
*EjerSem03> eval (While (B True) Void)

*EjerSem03> eval (Or (Eq (Add (I 0) (I 0)) (I 0)) (Eq (I 1) (I 1)))
B True
```

3. Conclusiones

Fue algo sencillo, pues ya teníamos eval1, evals, eval del lenguaje EAB, sin embargo, como había que hacer cosas con la memoria no podíamos pasarlo literal, así que se tuvieron que hacer unas modificaciones, y todo salió bien, como eval1 tiene que regresar un error si le pasamos un terminal, se implementó evals porque en algún punto no dejaba continuar con la ejecución.

Un problema fue que al atrapar errores del while, no caen ahí, sino que caen en el if, eso es malo, pues un while true, no debería regresar un error, sin embargo, regresa uno por parte del if.

Referencias

[1] Archivero, curso de Estructuras Discretas 2017-1