Lenguajes de Programación 2016-1
Tarea 1

Ricardo Garcia Garcia Juan Carlos López López
Luis Rodrigo Rojo Morales

21 de septiembre de 2015
Facultad de Ciencias UNAM

1. Problema I

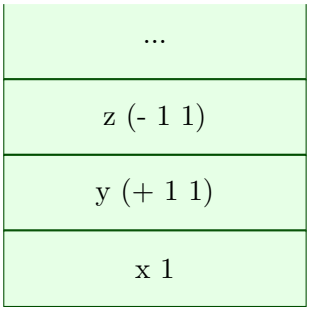
Hemos visto en clase que la definición de sustitución resulta en una operación ineficiente: en el peor caso es de orden cuadrático en relación al tamaño del programa (considerando el tamaño del programa como el numero de nodos en el árbol de sintaxis abstracta). También se vio la alternativa de diferir la sustitución por medio ambientes. Sin embargo, implementar un ambiente usando un stack no parece ser mucho mas eficiente.

Responde las siguientes preguntas.

- Provee un esquema para un programa que ilustre la no-linealidad de la implementación de ambientes basada en un stack. Explica brevemente porque su ejecución en tiempo no es lineal con respecto al tamaño de su entrada.
  Respuesta: Por ejemplo:

```
 > {subst {with {x 1}
    {with {y {+ x 1}}
      {with {z {- x 1}}
{+ x {+ y z}}}}}}
 > {subst {with {y {+ x 1}}
     {with {z {- x 1}}
      {+ x {+ y z}}}}}
 > {subst {with {z {- x 1}}
    {+ x {+ y z}}}}
 ------------------------------------
 > {+ x {+ y z}}
```

```
> {+ 1 {+ y z}}
> {+ 1 {+ 2 z}}
> {+ 1 {+ 2 0}}
> {+ 1 2}
> (num 3)
```



Para la pila primero entra x, luego entra y y para saber su valor tiene que recorrer toda la pila hasta x, luego entra z y para saber su valor tiene que recorrer toda la pila hasta x, luego para evaluar para sacar el valor de x hay que recorrer toda la pila hasta x, para y hasta y y para z hasta z, usando evaluación glotona, pero para cada elemento se tuvo que recorrer toda la pila al menos una vez por lo que no es lineal, es cuadrático.

- Describe una estructura de datos para un ambiente que un interprete de `FWAE` pueda usar para mejorar su complejidad

  Respuesta: Utilizando una lista para poder simular el comportamiento de un stack, esto nos ayuda a poder reducir la complejidad en tiempo hasta O(n) debido a que a lo más recorreremos la lista 1 sola vez realizando las comparaciones correspondientes.

- Muestra como usaría el interpreté esta nueva estructura de datos.

  Respuesta: Dada la secciòn de codigo por analizar, basta con usarla del modo (expresion env) donde env es la implementacion de lista (que recordemos simula un stack) y guardara en cada subambiente, las variables y el ambiente siguiente, de la forma:
  env 0 = ()
  env 1 = ((var1 val1))
  env 2 = ((var2 val2),(var1, val1))

...
env n = ((varn valn), ... ,(var1 val1))
con esto, la implementacion de nuestro interprete se modifica. Recibiremos una expresion y su ambiente (por default siempre sera env0). Analizamos los casos como venimos haciendo normalmente (Obs: para esto hacemos varias funciones que nos quiten las etiquetas del Lenguaje). Los casos num, add y sub son similares a los anteriores, simplemente llamamos recursivamente la funcion interp a los lados del operador. Para el caso id v: simplemente ejecutamos la funcion lookup para poder buscar el valor de v en el ambiente proporcionado. Para el caso de las funciones basta con usar nuestro tipo ClosureV, quedando de esta forma el caso: [fun (bound-id bound-body) (closureV bound-id bound-body env)] Finalmente para el caso de la aplicacion de funcion:

```
[app (fun-expr arg-expr)
;Creamos variable local
  (local ([define fun-val (interp fun-expr env)])
    ;Si esa variable es un closure, llamamos la funcion interp
    ;sobre el cuerpo del closure y la variable de la funcion
    (if (closureV? fun-val)
        (interp (closureV-body fun-val)
        ;Ademas crearemos un sub-cache (sub ambiente)
                (aSub (closureV-param fun-val)
                      (interp arg-expr env)
                      (closureV-env fun-val)))
                      ;Sino, simplemente devolvemos un error
         (error 'interp (string-append
                         (~a fun-expr) " expression is not a function"))))]))
```

- Indica cual es la nueva complejidad del interprete (análisis del peor caso) y de forma informal pero rigurosa pruébalo.
  La nueva complejidad es O(n).

2. Problema II

Dada la siguiente expresión de `FWAE`:

```
> {with {x 4}
    {with {f {fun {y} {+ x y}}}
     {with {x 5}
```

```
        {f 10}}}}
```

debe evaluar a (*num* 14) usando alcance estático, mientras que usando alance dinámico se obtendría (*num* 15),

Respuesta:

Evaluando usando alcance estatico.

```
> {subst {with {x 4}
    {with {f {fun {y} {+ x y}}}
     {with {x 5}
       {f 10}}}}}
> {subst {with {f {fun {y} {+ 4 y}}}
    {with {x 5}
      {f 10}}}}}
> {subst {with {x 5}
      {f 10}}}}}
-------------------------------------
>{f {fun {y} {+ x y}}
>{f {fun {y} {+ 4 y}}
>{f {fun {10} {+ 4 10}}
>{f {fun {10} {14}}
>(num 14)
```

Ambiente.

...
y 10
x 5
f (fun (y) (+ 4 y))
x 4

Evaluando usando alcance dinamico.

```
> {subst {with {x 4}
```

```
    {with {f {fun {y} {+ x y}}}
     {with {x 5}
       {f 10}}}}}
> {subst {with {f {fun {y} {+ x y}}}
    {with {x 5}
      {f 10}}}}}
> {subst {with {x 5}
      {f 10}}}}}
---------------------------------------
>{f {fun {y} {+ x y}}
>{f {fun {y} {+ 5 y}}
>{f {fun {10} {+ 5 10}}
>{f {fun {10} {15}}
>(num 15)
```

Ambiente.

...
y 10
x 5
f (fun (y) (+ 5 y))
x 4

Ahora Ben un agudo pero excéntrico estudiante dice que podemos seguir usando alcance dinámico mientras tomemos el valor mas viejo de `x` en el ambiente en vez del nuevo y para este ejemplo el tiene razón.

- ¿ Lo que dice Ben esta bien en general? si es el caso justifícalo.
  Respuesta: No, lo que dice Ben no esta bien para el caso general, lo que dice solo funciona en este tipo de casos.

- Si Ben esta equivocado entonces da un programa de contraejemplo y explica por que la estrategia de evaluación de Ben podría producir una respuesta incorrecta.
  Respuesta: Por ejemplo tenemos la siguiente expresión de FWAE:

```
>{with {x 4}
  {with {x 6}
   {with {x 7}
    {with {f {fun {y} {+ x y}}}
     {with {x 5}
      {with {x 3}
       {f 10}}}}}}}
```

El ambiente seria:

...
x 3
x 5
f (fun (y) (+ x y))
x 7
x 6
x 4

El valor mas viejo de x es x 4, pero en alcance dinamico el valor de x seria x 5 y en alcance estatico el valor de x seria x 7, por lo tanto lo que dice Ben no se cumple.

3. Problema III

Dada la siguiente expresión de `FWAE` con `with` multi-parametrico:

```
{with {{x 5} {adder {fun {x} {fun {y} {+ x y}}}} {z 3}}
    {with {{y 10} {add5 {adder x}}}
          {add5 {with {{x {+ 10 z}} {y {add5 0}}}
                  {+ {+ y x} z}}}}}
```

- Da la forma Bruijn de la expresión anterior.
  Respuesta:

```
{with {{5} {adder {fun {x} {fun {y} {+ x y}}}} {3}}
    {with {{10} {add5 {adder <0 0>}}}
```

```
{add5 {with {{+ 10 <1 2>} {y {add5 0}}}
        {+ {+ <0 1> <0 0>} <2 2>}}}}
```

- Realiza la corrida de esta expresión, es decir escribe explícitamente cada una de las llamadas tanto para `subst` y `interp`, escribiendo además los resultados parciales en sintaxis concreta.