Ejercisios Explicit-Refs

Francisco Javier Atondo Nubes

21 Octubre 2022

1 4.8

Muestra exactamente en que parte de nuestra implementacion del almacenamiento las operaciones toman tiempo lineal en lugar de tiempo constante.

Se puede argumentar que el tiempo constante es tomado durante la ejecucion de value-of:
(value-of exp1 p o) = (val1 , o1)

```
  Por lo que si usamos:
 let x = newref(0)
in letrec even(dummy)
 = if zero?(deref(x ))
  then 1
 else begin setref(x , -(deref(x ),1)); (odd 666) end
 odd(dummy)
 = if zero?(deref(x ))
  then 0
  else begin setref(x , -(deref(x ),1)); (even 666) end
in begin setref(x ,13); (odd 666) end
```

como ejemplo de la implementacion del almacenamiento, se dice que las operaciones toman tiempo lineal en la definicion de la operacion, osea "let x = newref(0)".

2 4.9

Implementa el almacenamiento en tiempo constante representandolo comoun vector de Racket. ¿Qu´e perdemos al usar esta representacion?

3 4.10

Implementa las expresiones begin como se especifican en el ejercicio 4.4.

4 4.11

Implementa list del ejercicio 4.5.

let x = 0 let list = newref(0)

5 4.12

Nuestro entendimiento del almacenamiento, en este int´erprete, depende del significado de efectos en Racket. En particular, depende en que nosotros sepamos cuando estos efectos ocurren en un programa de Racket. Podemos evitar esta dependencia al escribir un interprete que siga mas de cerca la especificacion. En este interprete, value-of regresaria tanto el valor como el estado de almacenamiento. Implementa esta version del interprete.