Lenguajes y Compiladores - Trabajo práctico 8 - Año 2006

(1) Calcular la semántica operacional de las siguientes expresiones en el lenguaje aplicativo eager con referencias y asignación:

(a) $\textbf{newvar}\ x := 1\ \textbf{in}\ (1 + \textbf{val}\ x);\ x := 2 + \textbf{val}\ x$

(b) $\textbf{let}\ x \equiv \textbf{mkref}\ (-1)\ \textbf{in}$
$\textbf{while}\ (\textbf{val}\ x \neq 0)\ \textbf{do}\ (x := 1 + \textbf{val}\ x;\ x)$

(2) Considere la expresión

$$\textbf{let}\ r \equiv \textbf{mkref}\ 0\ \textbf{in}\ (\lambda x.(r := 1; x))(\textbf{val}\ r).$$

Dar el resultado de evaluar la expresión, y también el resultado de evaluar la expresión contrayendo previamente la redex β.

(3) Para el lenguaje aplicativo eager con referencias y asignación, calcular la semántica de continuaciones de $x := x + 1$ en un ambiente que asigna el valor $\iota_{\text{ref}}\ l$ a x y un estado que asigna $\iota_0\ n$ a l. Idem, en un ambiente que asigna $\iota_{\text{int}}\ n$ a x y un estado cualquiera.

(4) Dar la semántica directa del lenguaje aplicativo eager con referencias y asignación. Dar las ecuaciones semánticas para

$$-e,\ e_0 + e_1,\ e_0\ e_1,\ \textbf{let},\ \textbf{mkref},\ \textbf{val},\ e_0 =_{\textbf{ref}} e_1,\ e_0 := e_1.$$

(5) Agregar el comando $\textbf{for}\ i := e_0\ \textbf{to}\ e_1\ \textbf{do}\ c$ y dar su semántica directa de modo que no requiera que i esté previamente declarada, evalúe sólo una vez cada una de las expresiones e_0 y e_1, y dé error en caso de que se intente asignar a i en el cuerpo c del **for**. Idem con la semántica de continuaciones.