Programación para la Inteligencia Artificial
Práctica 6 (Parte II)
El número de Meertens

El ejercicio consiste en encontrar un número que posee una propiedad peculiar. El problema está relacionada con la descomposición de un número en factores primos. Por ejemplo, $400 = 2^4 3^0 5^2$ y $432 = 2^4 3^3 5^0$. Supongamos que definimos $g(n) = 2^{d_1} 3^{d_2} ... p_k^{d_k}$, donde $d_1, d_2, ..., d_k$ es la representación decimal de n y p_k es el k-ésimo número primo. Es decir, $g(402) = 400$ y $g(430) = 432$. La mayor parte de las veces, los números n y $g(n)$ no son tan cercanos como los de los ejemplos. La pregunta que nos hacemos es: **¿coinciden alguna vez?**

La función g recibe el nombre en honor de Kurt Gödel, que usó esta idea para codificar una secuencia de caracteres mediante un sólo número en su famoso artículo sobre la incompletitud de la aritmética. Aquí usaremos la misma idea para codificar la representación decimal de los números.

Cuando $n = g(n)$ decimos que n es un *número de Meertens*. Actualmente sólo se conoce un único número de Meertens. Resolviendo este ejercicio trataremos de descubrirlo.

Para buscarlo se pide realizar un programa en Haskell según las siguientes indicaciones[1]:

A) Crear un módulo[2] `Rosadelfa` en el que se defina el tipo de árbol *rosadelfa* (`RAdelfa`) tal y como se hace en la última diapositiva del tema 6. Además:

1. Declarar el tipo como una instancia de la clase Show, de modo que se visualicen los nodos más sangrados a medida que vaya aumentando la profundidad en el árbol (ver el árbol del anexo como ejemplo).
2. Definir la función

   ```
   podar :: RAdelfa a -> RAdelfa a
   ```

   de forma que `podar (Nodo x ts)` elimine el hijo más a la izquierda de ese Nodo.
3. Definir la función

   ```
   aplanar :: RAdelfa a -> [a]
   ```

   que devuelva la lista que resulta de aplanar un árbol mediante un recorrido en profundidad.

[1]No es necesario usar en la solución todas las funciones propuestas en cada apartado. Por otro lado pueden definirse todas las funciones que se consideren necesarias.

[2]Usar, si se quiere, el fichero "Rosadelfa.hs" como punto de partida.

B) Crear un fichero que importe el módulo anterior y:

1. Definir por renombramiento un tipo denominado `Valor` como una pareja de `Integer`. Definir por renombramiento un tipo denominado `Adelfa` como una `RAdelfa` cuyos nodos se etiqueten con datos de tipo `Valor`.
2. Definir la función `godel` que tome como entrada un número entero y devuelva $g(n)$.
3. Definir la función `digitos` que tome como entrada un número entero y devuelva una lista con sus dígitos en orden, es decir, `digitos 345 = [3,4,5]`.
4. Definir la función `pots` que tome como entrada un número entero y devuelva una lista infinita de potencias de ese número, es decir, `take 10 (pots 3) = [1,3,9,27,81,243,729,2187,6561,19683]`.
5. Ahora suponemos que queremos construir un árbol con nodos de tipo `Valor`, que representen parejas de `(n,g(n))`. La raíz del árbol sería `(0,1)`, es decir $(0, 2^0)$. Cada nodo tendría 10 hijos, uno por cada dígito, aunque la raíz tendría 9, pues no existen números que comiencen por cero. Se etiqueta cada nodo del árbol con dos enteros (`n` y `g`), donde `g=godel n` tal y como hemos dicho antes. La representación decimal de `n` define el camino en el árbol desde la raíz hasta el nodo dado. Por tanto, una posible solución al problema pasa por definir las siguientes funciones:
   (a) Una función `gArbol` que tome como entrada un número entero (`k`) y genere el árbol para números con un límite de `k` dígitos, es decir, un árbol de profundidad `k`. Por ejemplo, `gArbol 2` sería el árbol que aparece en el anexo.
   Para que la generación sea más eficiente tener en cuenta, al menos, estas dos cuestiones:
      - No tiene sentido generar un subárbol cuya raíz sea `(n,g)` y que cumpla que $g \geq 10^k$ porque, por construcción, todos los valores `(n,g)` en el árbol cumplen $n < 10^k$.
      - Será necesario podar la primera rama puesto que ningún número comienza por 0 y así se evita la generación del subárbol correspondiente.

   (b) Una función `meertens` que tome como entrada la profundidad del árbol (límite de dígitos para los números) y devuelva la lista con los números de Meertens localizados. Esta función podrá implementarse aplanando el árbol generado en la función anterior.

Número de Meertens encontrado con 8 dígitos:

ANEXO: Árbol de profundidad 2 generado por `gArbol 2`

```
(0,1)
  (1,2)
    (10,2)
    (11,6)
    (12,18)
    (13,54)

  (2,4)
    (20,4)
    (21,12)
    (22,36)

  (3,8)
    (30,8)
    (31,24)
    (32,72)

  (4,16)
    (40,16)
    (41,48)

  (5,32)
    (50,32)
    (51,96)

  (6,64)
    (60,64)
```