

# Abordaje Funcional a EDSLs

Alberto Pardo (pardo@fing.edu.uy)

Marcos Viera (mviera@fing.edu.uy)

ECI 2024

Se desea implementar un pequeño EDSL para representar proposiciones basadas en comparaciones entre números naturales, el cual tiene las siguientes construcciones:

- **val**: valor natural
- **eq**: igualdad entre naturales
- **lt**: relación de menor entre naturales
- **not**: negación
- **and**: conjunción
- **or**: disyunción

Asociado al EDSL se tiene un sistema de tipos que está formado por las reglas que se presentan a continuación. Como es habitual, el juicio  $\vdash p : \tau$  significa que la expresión  $p$  tiene tipo  $\tau$ . El lenguaje maneja dos tipos (*integer* y *boolean*).

$$\begin{array}{c} \frac{}{\vdash \text{val } n : \text{integer}} \quad \frac{\vdash p_1 : \text{integer} \quad \vdash p_2 : \text{integer}}{\vdash \text{eq } p_1 p_2 : \text{boolean}} \quad \frac{\vdash p_1 : \text{integer} \quad \vdash p_2 : \text{integer}}{\vdash \text{lt } p_1 p_2 : \text{boolean}} \\[10pt] \frac{\vdash p : \text{boolean}}{\vdash \text{not } p : \text{boolean}} \quad \frac{\vdash p_1 : \text{boolean} \quad \vdash p_2 : \text{boolean}}{\vdash \text{and } p_1 p_2 : \text{boolean}} \quad \frac{\vdash p_1 : \text{boolean} \quad \vdash p_2 : \text{boolean}}{\vdash \text{or } p_1 p_2 : \text{boolean}} \end{array}$$

Por ejemplo, las siguientes son expresiones bien tipadas del EDSL:

```
 $\vdash \text{var } 4 : \text{integer}$   
 $\vdash \text{lt } (\text{var } 4) (\text{var } 5) : \text{boolean}$   
 $\vdash \text{or } (\text{not } (\text{lt } (\text{var } 4) (\text{var } 3))) (\text{eq } (\text{var } 4) (\text{var } 3)) : \text{boolean}$ 
```

En cambio, las siguientes no lo son:

```
 $\text{or } (\text{var } 4) (\text{var } 5)$   
 $\text{not } (\text{var } 4)$   
 $\text{eq } (\text{lt } (\text{var } 4) (\text{var } 3))(\text{var } 5)$ 
```

Considerando que la interpretación estándar de las proposiciones es su valor de verdad o el propio natural  $n$  en el caso de expresiones de la forma `val  $n$` , se pide:

1. Implementar el EDSL como un **shallow embedding** bien tipado en Haskell siguiendo el enfoque *tagless-final*. Interprete los tipos *integer* y *boolean* como tipos de Haskell.
2. Implementar el EDSL como un **deep embedding** bien tipado en Haskell utilizando GADTs. Definir la función *eval* que evalúa una expresión bien tipada de tipo  $t$  y retorna un valor de ese tipo.
3. Dada la siguiente gramática que describe una sintaxis concreta para el lenguaje:

$$\begin{aligned} \text{prop} &::= \text{term } "\backslash/" \text{ prop} \mid \text{term} \\ \text{term} &::= \text{factor } "/" \text{ term} \mid \text{factor} \\ \text{factor} &::= '\sim' \text{ prop} \mid '(' \text{ prop } ')' \mid '(' \text{ prop } '=' \text{ prop } ')' \mid '(' \text{ prop } '<' \text{ prop } ')' \mid \mathbb{N} \end{aligned}$$

- (a) Definir una nueva interpretación para 1. y 2. que retorne un *String* que represente el programa en esa sintaxis.
- (b) Definir un tipo *UProp*, de kind  $*$ , que represente el árbol de sintaxis abstracta no tipado del lenguaje.
- (c) Escribir un parser del lenguaje utilizando los combinadores vistos en el curso (puede optar por usar los combinadores aplicativos o monádicos) y que retorne el valor de tipo *UProp* correspondiente.
4. (OPCIONAL) Considere la siguiente extensión al lenguaje, en la que se agrega:

- **var**: variable proposicional

con la siguiente regla de tipado:

$$\frac{}{\vdash \text{var } x : \text{boolean}}$$

Ahora la interpretación depende de un *ambiente de variables* en el que se le asocian valores de verdad a las variables.

- (a) Extienda el EDSL definido en 1.
- (b) Extienda el EDSL definido en 2.
- (c) Extienda el tipo definido en 3.a y el parser definido en 3.b.
- (d) Defina una función  $\text{typeProp} :: \text{Ty } t \rightarrow \text{UProp} \rightarrow \text{Prop } t$  donde  $\text{Prop } t$  es el GADT que definió para el deep embedding y  $\text{Ty } t$  es el siguiente tipo que representa tipos:

```
data Ty :: * -> * where
  TInt  :: Ty Int
  TBool :: Ty Bool
```

**Sobre la entrega:**

- Enviar un archivo `NombreApellido.zip` conteniendo los archivos `.hs` con la implementación de sus soluciones a `pardo@fing.edu.uy` y `mviera@fing.edu.uy`. Incluya sus datos (nombre, apellido, universidad, email) en todos los archivos que entregue.
- La prueba consiste de las partes 1 a 3. La parte 4. es simplemente opcional para quienes deseen extender el EDSL con variables proposicionales. No cuenta en la nota final.