Inducción estructural

Mario Román

<2015-03-14 Sat 15:02>

Normalmente aplicamos inducción sobre los números naturales, y cuando necesitamos aplicar inducción en otro contexto lo hacemos corresponder con los números naturales. Por ejemplo, si queremos demostrar una propiedad sobre los árboles binarios, la demostraríamos por inducción sobre la altura del árbol. Pero el proceso de llevar todo a los naturales puede ser incómodo, tedioso y puede complicar la demostración innecesariamente. En este post vamos a desarrollar una forma de ampliar la inducción a la estructura de los tipos de datos para simplificar todas esas demostraciones.

Conjuntos bien fundados

Vamos a definir las relaciones bien fundadas, que nos permitirán definir una inducción generalizada. [1]

- **Relación bien fundada:** una relación en un conjunto de elementos es bien fundada si todo subconjunto no vacío tiene un elemento minimal. Dado un orden parcial, es bien fundado si todo subconjunto no vacío tiene un elemento tal que ninguno es menor que él.

Y podemos realizar inducción sobre cualquier conjunto con una relación bien fundada.

- **Inducción noetheriana:** sea X un conjunto bien fundado con $A \subset X$. Si se cumple:

$$(y < x \Rightarrow y \in A) \Rightarrow x \in A$$

  Entonces $A = X$.

Inducción sobre tipos

Ahora vamos a aplicar esto a teoría de tipos. Sea un tipo con sus constructores. Para todas las instancias constructibles del tipo (es decir, aquellas que pueden generarse en un número finito de pasos desde sus constructores), definimos un orden parcial:

- **Orden constructivo:** para dos instancias del tipo: $a, b :: A$, b se construye con a si el constructor de b toma a a como argumento. La clausura transitiva de esta relación forma un orden parcial:

$$a \leq b \Rightarrow a \text{ se usa en la construcción de } b$$

Y ahora tenemos una inducción sobre los constructores de los tipos, que describimos ahora.

- **Inducción sobre tipos**: sea un tipo A con constructores y sea $P :: A \to Bool$ una propiedad. Siendo $a_1, a_2 \ldots a_i :: A$ argumentos del constructor, si se cumple la condición de inducción para cada constructor C_i:

$$P(a_1) \wedge P(a_2) \wedge \ldots P(a_i) \Rightarrow P(C_i(a_1, a_2, \ldots, b_0, b_1 \ldots))$$

Entonces $a :: A \Rightarrow P(a)$.

Ejemplo 1: Naturales

Nuestro primer ejemplo va a ser obtener la inducción sobre los naturales como caso particular. Damos una definición de los naturales en lenguaje Haskell, con los axiomas de Peano, un natural es 0 o el siguiente de un natural:

```
data Nat = O
         | S Nat
```

Que equivale a la definición en Coq:

```
Inductive nat : Type :=
  | O : nat
  | S : nat -> nat
```

Es decir, si lo demostramos para `O` y para `S n` sabiéndolo para `n`, lo hemos demostrado para todos los naturales.

Ejemplo 2: Árboles binarios

Ahora vamos a intentar el ejemplo que motivó esta búsqueda. Definimos un árbol binario como un árbol vacío o como un nodo del que surgen dos árboles binarios, en Haskell:

```
data Tree a = Empty
            | Node a (Tree a) (Tree a)
```

Que equivale a la definición en Coq:

```
Inductive tree (X:Type) : Type :=
  | nilt : tree X
  | node : X -> tree X -> tree X -> tree X.
```

Es decir, si demostramos una propiedad para el árbol vacío y para un árbol sabiendo que la cumplen sus subárboles derecho e izquierdo, la hemos demostrado para todos los árboles binarios.

En el repositorio mroman42/recorridosArboles hay varias demostraciones por inducción sobre árboles binarios, explicados en lenguaje natural y demostrados luego sobre el asistente de demostraciones Coq.

[1] Post sobre generalizaciones de la inducción en Stack Overflow.