Universidad Nacional Autónoma de México
Facultad de Ciencias
Lenguajes de Programación

•••

Karla Ramírez Pulido

Propiedades de un lenguaje: consistencia y seguridad

Los programas en general

- Terminan con un valor esperado tal que ese valor, llamemos v es de tipo T

- Algunos programas regresan:

  -1 ó 0

  (estos valores NO los definió el programador)

- Algunos otros terminan:

  Excepción

Propiedad de CONSISTENCIA

Propiedad de consistencia o solidez

(En inglés: soundness)

Para todo programa p, si el tipo de p es T

entonces éste terminará con un valor v tal que v:T o

bien se levantará una excepción, de un conjunto bien definido de excepciones.

El siguiente programa:

```
{if0 {+ 1 2}
     {{fun {x : number} : number {+ 1 x}} 7}
     {{fun {x : number} : number {+ 1 {+ 2 x}}} 1}}
```

¿Qué hace?

Siempre se evalúa la rama del else del if0, por lo que siempre se ejecuta ésta. Sin embargo el verificador de tipos SIEMPRE debe verificar ambas ramas.

Propiedad de SEGURIDAD

Ninguna operación primitiva se aplicará a los tipos incorrectos.

Ejemplo: (+ number number)

1. (+ 2 3)

2. (+ 2 a)

Algunos ejemplos de lenguajes

Lenguajes seguros

Lenguajes no seguros

	statically checked	not statically checked
type safe	ML, Java	Scheme
type unsafe	C, C++	assembly

Tipos principales de Hindley-Milner

Para un término t, considera el tipo T.

T es el tipo principal de t sí para cualquier otro tipo T' el cual tipifica a t, existe una sustitución (posiblemente vacía) que cuando se aplica T, T' se mantiene.

Refraseando la definición de Hindley-Milner:

- El sistema de tipos infiere el tipo "más general" para un término.
- El tipo generado por el sistema de tipos de Hindley-Milner impone algunas restricciones al comportamiento del programa. En particular impone algunas restricciones necesarias para garantizar la consistencia de tipos.

Ejemplo

(+ 1 t2)

(+ 1 : number t2 : number)

(+ 1 3.5)

(+ 1 : integer 3.5 : float)

(+ number number)

Seguridad = Progreso + Preservación

Los términos bien tipificados no serán incorrectos (erróneos): esto significa que no alcanzarán un estado que se quede atascado, es decir, que no se termine con valor esperado o donde las reglas de evaluación no nos digan qué paso seguir.

Queremos saber si los términos están bien tipificados y no se queda el proceso atascado en la evaluación. Para ello definimos los teoremas de Progreso y Preservación.

Teorema del Progreso

Supongamos un término t bien-tipificado (esto es, que t : T para algún T). Entonces t es un valor o bien existe algún t' con t → t'.

Progreso: un término bien tipificado no se debe estancar (ya que es un valor o se puede llegar en algún paso acorde a las reglas de evaluación).

Ejemplo: sintaxis y sistema

t ::= true | false | if t then t else t | 0 | succ t | pred t | iszero t

Cuando evaluamos un término obtenemos un valor v
v ::= true | false | n

Valores numéricos:
n ::= 0 | succ n

Tipos:
T ::= Bool | Nat

Reglas de tipificado

$t : T$

$0 : Nat$ (T-ZERO)

$true : Bool$ (T-TRUE)

$false : Bool$ (T-FALSE)

$$\frac{t_1 : Bool \quad t_2 : T \quad t_3 : T}{\text{if } t_1 \text{ then } t_2 \text{ else } t_3} \text{ (T-IF)}$$

$$\frac{t_1 : Nat}{succ\ t_1 : Nat} \text{ (T-SUCC)}$$

$$\frac{t_1 : Nat}{pred\ t_1 : Nat} \text{ (T-PRED)}$$

$$\frac{t_1 : Nat}{iszero\ t_1 : Bool} \text{ (T-ISZERO)}$$

Lema de Inversión de la Relación de Tipificado

1. Si true:R entonces R=Bool
2. Si false:R entonces R=Bool
3. Si if t_1 then t_2 else t_3 : R entonces t_1=Bool, t_2:R, y t_3:R
4. Si 0:R entonces R=Nat
5. Si succ t_1:R entonces R=Nat y t_1:Nat
6. Si pred t_1:R entonces R=Nat y t_1:Nat
7. Si iszero t_1:R entonces R=Bool y t_1:Nat

Demostración es directa usando las definiciones de la relación de tipificado.

Lema para Formas Canónicas

1. Si v es un valor de tipo Bool, entonces v es true o false.
2. Si v es un valor de tipo Nat, entonces v es un valor numérico de acuerdo con la gramática siguiente:

t ::= .... | 0 | succ t | pred t | iszero t

v ::= ... | n

n ::= 0 | succ n

Lema para Formas Canónicas

Reglas de evaluación $t \longrightarrow t'$

$$\frac{t\_1 \rightarrow t\_1'}{\text{succ } t\_1 \longrightarrow \text{succ } t\_1'} \quad \text{(E-SUCC)}$$

$$\text{pred } 0 \rightarrow 0 \quad \text{(E-PREDZERO)}$$

$$\frac{t\_1 \rightarrow t\_1'}{\text{pred } t\_1 \longrightarrow \text{pred } t\_1'} \quad \text{(E-PRED)}$$

$$\text{iszero } 0 \rightarrow \text{true} \quad \text{(E-ISZERO-ZERO)}$$

$$\text{iszero } (\text{succ } n\_1) \rightarrow \text{false} \quad \text{(E-ISZEROSUCC)}$$

$$\frac{t\_1 \rightarrow t\_1'}{\text{iszero } t\_1 \longrightarrow \text{iszero } t\_1'} \quad \text{(E-ISZERO)}$$

$$\text{if true then t2 else t3} \rightarrow \text{t2} \quad \text{(E-IFTRUE)}$$

$$\text{if false then t2 else t3} \rightarrow \text{t3} \quad \text{(E-IFFALSE)}$$

$$\frac{t1 \rightarrow t1'}{\text{if t1 then t2 else t3} \rightarrow \text{if t1' then t2 else t3}} \quad \text{(E-IF)}$$

Teorema de Progreso

Supongamos un término t bien-tipificado (esto es, que t : T para algún T). Entonces t es un valor o bien existe algún t' con t → t'.

Dem: Por Inducción sobre la derivación de t:T.
Caso T-TRUE y T-FALSE y T-ZERO se cumplen de inmediato, ya que t en estos casos es un valor.
Para los siguientes casos tenemos:

Caso T-IF
t= if t_1 then t_2 else t_3
t_1: Bool t_2:T t_3:T
Por H. I. t_1 es un valor o es algún término t_1' tal que t_1 → t_1'.

Si t_1 es un valor entonces por el Lema de las formas canónicas podemos garantizar que puede ser true o false, en cuyo caso E-IFTRUE o E-IFFALSE se pueden aplicar a t.

Por otro lado, si t_1 → t_1' entonces por E-IF
t → if t_1' then t_2 else t_3

Teorema de Progreso

Caso T-SUCC
t = succ t_1 y t_1: Nat
Por H. I. t_1 es un valor o es algún término t_1' tal que t_1 → t_1'.

Si t_1 es un valor entonces por el Lema de las formas canónicas, éste debe ser un valor numérico, en cuyo caso es t.

Por otro lado, si t_1 → t_1' entonces por E-SUCC, succ t_1 → succ t_1'

Teorema de Progreso

Caso T-PRED
t = pred t_1 y t_1: Nat
Por H. I. t_1 es un valor o es algún término t_1' tal que t_1 → t_1'.

Si t_1 es un valor entonces por el Lema de las formas canónicas, éste debe ser un valor numérico, i.e. es 0 ó succ n para algún n, y una de las siguientes reglas E-PREDZERO, o E-PREDSUCC se pueden aplicar a t_1.

Por otro lado, si t_1 → t_1' entonces por E-PRED, pred t_1 → pred t_1'

Teorema de preservación

$$\text{Si } t : T \text{ y } t \longrightarrow t' \text{ entonces } t' : T.$$

Preservación: si un término bien tipificado toma un paso en la evaluación, entonces el término resultante también estará bien tipificado.

Teorema de Preservación

Supongamos un término t bien-tipificado (esto es, que t : T para algún T). Entonces t es un valor o bien existe algún t' con t → t':T.

Demostración:
Por Inducción sobre la derivación de t:T. En cada paso de la inducción, asumimos que la propiedad deseada se mantiene para todas las subderivaciones (i.e. que si s:S y s → s', entonces s':S, cuando s:S sea probado por una subderivación de la actual) y procedemos por análisis de casos en la regla final de la derivación.

Caso T-TRUE
t = true
T = Bool

En la última regla de derivación es T-TRUE, entonces sabemos por la forma de esta regla que t debe ser la constante true y T debe ser Bool. Pero si t es un valor, entonces no podemos tener el caso de t → t' para cualquier t', y los requerimientos del teorema se satisfacen por vacuidad.

Teorema de Preservación

Caso T-IF
t= if t_1 then t_2 else t_3
t_1:Bool t_2:T t_3:T

Si la última regla de derivación de T-IF, entonces sabemos por la forma de la regla que t debe tener la forma if t_1 then t_2 else t_3 para algún t_1, t_2 y t_3.

Ahora, buscando las reglas de evaluación con if en el lado izquierdo, encontramos que podemos aplicar 3 tipos de reglas para que t →t' pueda ser derivado

Reglas de evaluación:

1.
if true then t2 else t3 → t2
(E-IFTRUE)

2.
if false then t2 else t3 → t3 (E-IFFALSE)

3.
$$\frac{t1 \rightarrow t1'}{\text{if } t1 \text{ then } t2 \text{ else } t3 \rightarrow \text{if } t1' \text{ then } t2 \text{ else } t3} \quad \text{(E-IF)}$$

Veamos por casos, caso 1.
Sub caso E-IFTRUE: t1 = true t'= t2

Teorema de Preservación

Veamos por casos, caso 1.
Sub caso E-IFTRUE: t1 = true t'= t2

Si t → t' es derivado usando E-IFTRUE, entonces por la forma de esta regla, t1 debe ser true y el resultado del término t' es la segunda expresión t2.
Esto significa que hemos terminado, dado que sabemos (por el antecedente de T-IF) que t2:T, que es lo que necesitamos.

Los otros casos 2 y 3 son similares.

CASO T-ZERO
t=0 T=Nat

No puede suceder (por las mismas razones que T-TRUE)

CASO T-SUCC
t= succ t1 T=Nat t1:Nat

Viendo la regla de (E-SUCC) solo hay una única regla que puede ser usada para derivar t → t'.
La forma de esta regla nos dice que t1 → t1´. Dado que sabemos que t1:Nat, podemos aplicar la H.I. para obtener t1':Nat, de lo cual obtenemos succ t1':Nat, i.e. t':T aplicando la regla T-SUCC.

¿Dudas?

Gracias