

PLP - Recuperatorio del Primer Parcial - 1^{er} cuatrimestre de 2024

#Orden	Nro. Libreta	Apellido(s)	Nombre(s)	
12	1028122	Andrés	SERASÍAN TENACU	
Corregido por	Nota E1	Nota E2	Nota E3	Nota Final
Lucas	B-	B-	B-	A

Este examen se aprueba obteniendo al menos dos ejercicios bien (B+) y uno regular (B). Las notas para cada ejercicio son: -, I, R, B-, B. Entregar cada ejercicio en hojas separadas. Poner nombre, apellido y número de orden en todas las hojas, y numerarlas. Se puede utilizar todo lo definido en las prácticas y todo lo que se dio en clase, colocando referencias claras. El orden de los ejercicios es arbitrario. Recomendamos leer el parcial completo antes de empezar a resolverlo.

Ejercicio 1 - Programación funcional

Aclaración: en este ejercicio no está permitido utilizar recursión explícita, a menos que se indique lo contrario.

En este ejercicio vamos a modelar lógica proposicional en Haskell, de modo de poder construir fórmulas proposicionales y evaluarlas bajo distintas evaluaciones.

```
data Prop = Var String | No Prop | Y Prop Prop | O Prop Prop | Imp Prop Prop

type Valuación = String -> Bool
```

Por ejemplo, la expresión: $Y(\text{Var } 'P') (\text{No}(\text{Imp}(\text{Var } 'Q') (\text{Var } 'R')))$ representa la proposición $P \wedge \neg(Q \Rightarrow R)$.

Las evaluaciones se representan como funciones que a cada variable proposicional le asignan un valor booleano. Por ejemplo la evaluación $\lambda x \rightarrow x == 'P'$ le asigna el valor verdadero a la variable P y falso a todas las otras variables proposicionales.

- Dar el tipo y definir las funciones `foldProp` y `recProp`, que implementan respectivamente los esquemas de recursión estructural y primitiva para el tipo `Prop`. Solo en este inciso se permite usar recursión explícita.
- Definir la función `variables :: Prop -> [String]`, que dada una fórmula devuelve la lista con todas sus variables proposicionales en algún orden, sin elementos repetidos.
Por ejemplo: `variables (O (Var 'P') (No (Y (Var 'Q') (Var 'P'))))` debería devolver la lista `['P', 'Q']` o la lista `['Q', 'P']`.
- Definir la función `evaluar :: Valuación -> Prop -> Bool`, que indica si una fórmula es verdadera o falsa para una evaluación dada.
- Definir la función `estáEnFNN :: Prop -> Bool`, que indica si una fórmula está en Forma Normal Negada. Es decir, si no tiene implicaciones y la negación se aplica únicamente a variables y no a proposiciones más complejas.
Por ejemplo: $Y(\text{Var } 'P') (\text{No}(\text{Imp}(\text{Var } 'Q') (\text{Var } 'R')))$ no está en FNN, y en cambio $Y(\text{Var } 'P') (Y(\text{Var } 'Q') (\text{No}(\text{Var } 'R')))$ sí lo está.

Ejercicio 2 - Demostración e inferencia

Considerar las siguientes definiciones sobre árboles con información en las hojas¹:

```
data AIH a = Hoja a | Bin (AIH a) (AIH a)
    der :: AIH a -> AIH a
    esHoja :: AIH a -> Bool
{E0} esHoja (Hoja x) = True
{E1} esHoja (Bin i d) = False
    mismaEstructura :: AIH a -> AIH a -> Bool
{M0} mismaEstructura (Hoja x) = esHoja
{M1} mismaEstructura (Bin i d) = \t -> not (esHoja t) &&
    mismaEstructura i (izq t) && mismaEstructura d (der t)
    izq :: AIH a -> AIH a
{I} izq (Bin i d) = i
```

a) Demostrar la siguiente propiedad:

$$\forall t :: AIH a . \forall u :: AIH a . \text{mismaEstructura } t u = \text{mismaEstructura } u t$$

Se recomienda hacer inducción en el primer árbol, utilizando extensionalidad en el segundo. Se permite definir macros (poner nombres a expresiones largas para no tener que repetirlas).

No es obligatorio reescribir los \forall correspondientes en cada paso, pero es importante recordar que están presentes y escribir los que correspondan al planteo la propiedad como predicado unario. Recordar también que los $=$ de las definiciones pueden leerse en ambos sentidos.

Se consideran demostradas todas las propiedades conocidas sobre enteros y booleanos.

b) Usar el algoritmo W para inferir juicios de tipado válidos para las siguientes expresiones, o indicar por qué no es posible (recordar que en inferencia **no está permitido** renombrar variables):

- I) $(\lambda x.x(\lambda x.\text{Succ}(x)))(\lambda x.x)$
- II) $\lambda x.\text{if isZero}(x) \text{ then } x \text{ else } x \text{ zero}$

Ejercicio 3 - Cálculo Lambda Tipado

Se desea extender el cálculo lambda simplemente tipado para modelar Árboles con información en las hojas. Para eso se extienden los tipos y expresiones de la siguiente manera:

$$\begin{aligned}\tau ::= \dots &| AIH(\tau) \\ M ::= \dots &| \text{Hoja}(M) | \text{Bin}(M, M) | \text{case } M \text{ of Hoja } x \rightsquigarrow M; \text{Bin}(i, d) \rightsquigarrow M\end{aligned}$$

- $AIH(\tau)$ es el tipo de los árboles con información en las hojas de tipo τ .
- $\text{Hoja}(M)$ es un árbol compuesto por una única hoja con información M .
- $\text{Bin}(M_1, M_2)$ es un árbol compuesto por dos subárboles M_1 y M_2 .
- El observador $\text{case } M \text{ of Hoja } x \rightsquigarrow M_1; \text{Bin}(i, d) \rightsquigarrow M_2$ permite acceder al valor de un árbol que es hoja (el cual se ligará a la variable x que puede aparecer libre en M_2), y a los dos subárboles de un árbol que no es hoja (los cuales se ligarán a las variables i y d que pueden aparecer libres en M_2).

- a. Introducir las reglas de tipado para la extensión propuesta.
- b. Definir el conjunto de valores y las nuevas reglas de semántica operacional a pequeños pasos, tanto de congruencia como de cómputo.
- c. Mostrar paso por paso cómo reduce la expresión:
 $\text{case } (\lambda n : \text{Nat}. \text{Hoja}(n)) \text{ Succ(zero)} \text{ of Hoja } x \rightsquigarrow \text{Succ}(\text{Pred}(x)); \text{Bin}(i, d) \rightsquigarrow \text{zero}$
- d. Definir como macro la función esHoja_τ , que toma un $AIH(\tau)$ y devuelve un booleano que indica si es una hoja.

¹Escritas con recursión explícita para facilitar las demostraciones.

10/28/22

Sebastián Andrade

Hoja 1
orden 12

(1) $\text{data Prop} = \text{Var String} \mid \text{No Prop} \mid \text{Y Prop Prop}$
 $\mid \text{O Prop Prop} \mid \text{Imp Prop Prop}$

$\text{type Valuation} = \text{String} \rightarrow \text{Bool}$

(a)

$\text{recProp} :: (\text{String} \rightarrow b) \rightarrow (\text{Prop} \rightarrow b \rightarrow b) \rightarrow$
 $\quad (\text{Prop} \rightarrow \text{Prop} \rightarrow b \rightarrow b \rightarrow b) \rightarrow$
 $\quad (\text{Prop} \rightarrow \text{Prop} \rightarrow b \rightarrow b \rightarrow b) \rightarrow$
 $\quad (\text{Prop} \rightarrow \text{Prop} \rightarrow b \rightarrow b \rightarrow b) \rightarrow$
 $\quad \text{Prop} \rightarrow b$

$\text{recProp cString cNo cY cO cImp P} = \text{CASE P OF}$
 $\quad \text{Var S} \longrightarrow \text{cString S}$
 $\quad \text{No } f \longrightarrow \text{cNo } f (\text{rec } f)$
 $\quad Y P_1 P_2 \longrightarrow \text{cY } P_1 P_2 (\text{rec } P_1) (\text{rec } P_2)$
 $\quad O P_1 P_2 \longrightarrow \text{cO } P_1 P_2 (\text{rec } P_1) (\text{rec } P_2)$
 $\quad \text{Imp } P_1 P_2 \longrightarrow \text{cImp } P_1 P_2 (\text{rec } P_1) (\text{rec } P_2)$
 $\quad \text{where rec} = \text{recProp cString cY cO cImp}$

(b)

$\text{foldProp} :: (\text{String} \rightarrow b) \rightarrow (b \rightarrow b) \rightarrow (b \rightarrow b \rightarrow b) \rightarrow$
 $\quad (b \rightarrow b \rightarrow b) \rightarrow (b \rightarrow b \rightarrow b) \rightarrow \text{Prop} \rightarrow b$

$\text{foldProp cString cNo cY cO cImp} =$
 $\quad \text{recProp cString } (\lambda \rightarrow \text{cNo}) (\lambda \rightarrow \text{cY})$
 $\quad (\lambda \rightarrow \text{cO}) (\lambda \rightarrow \text{cImp})$

B) Variables :: Prop \rightarrow [String]

* Variables = foldProp ($(\lambda S \rightarrow [S])$) id

Podrías haber pasado (+) para usar aplicación parcial

$$(\lambda n r_2 \rightarrow n ++ r_2)$$

$$(\lambda n r_2 \rightarrow n ++ r_2)$$

$$(\lambda n r_2 \rightarrow n ++ r_2)$$

✓ Esto no filtra repetidos

C) evaluar :: Valuación \rightarrow Prop \rightarrow Bool

* evaluar v = foldProp v \vee not ($\&$) ($\|$)
 $(a c \rightarrow \text{not}(a) \| c)$

sin implicaciones y solo satisface una

D) estáEnFNN :: Prop \rightarrow Bool

* estáEnFNN = recProp ($(\lambda \rightarrow \text{true})$)

Está bien, $\leftarrow (\lambda p r \rightarrow \text{if simple}(p) \text{ then } \text{true} \text{ else false})$ \rightarrow NO

pero podrías

$$(\lambda -- r_1 r_2 \rightarrow r_1 \& r_2)$$

$$(\lambda -- r_1 r_2 \rightarrow r_1 \| r_2)$$

$$(\lambda -- r_1 r_2 \rightarrow \text{false})$$

→ cAY

→ cOO

→ eIMP.

* simple :: Prop \rightarrow Bool

simple p = case p of

Var S \rightarrow true

NO R \rightarrow false

Y R t \rightarrow false

OR t \rightarrow false

IMP RT \rightarrow false

} Usé recProp para conocer la instrucción que estoy respondiendo en el caso MP

} do the s es Vars.
(solo esos pueden estar negados)

10/20/22

Sergio Andes

Hoja 2
orden 12(2) dato $AITH_a = Hoja_a \mid Bin(AITH_a) (AITH_a)$ (a) sol:
 $P(t) = Hu :: AITH_a$. misma estructura t u = mismo u t para $t :: AITH_a$

* Es lo hacen con inducción estructural al árbol t.

Vamos a ver que valga $P(t)$ para cada constructor de $t :: AITH_a$.

① Caso base: t de la forma $t = Hoja_x$, $x :: a$.

 $P(t) = Hu :: AITH_a$ $\underbrace{Hu :: Hoja_x}$ misma estruc $(Hoja_x)$ u = mismo u ($Hoja_x$)
 " no los resumbo por espacio"

 \dots misma estruc $(Hoja_x)$ u = misma estr u ($Hoja_x$)
 $\downarrow 3 Mo$ ✓
 es Hoja u = misma estr u ($Hoja_x$)

En particular $u :: AITH_a$, \rightarrow por extensiónalidad de este estructura vale que:

 $u \xrightarrow{\text{Hoja}_a} \text{Bin}(AITH_a) (AITH_a)$

✓

Vemos cada caso:

 $\rightarrow 1.a) u = Hoja_y$, $y :: a$
 $\rightarrow 1.b) u = \text{Bin } i d$, $i :: AITH_a$, $b :: AITH_a$

1.a) $\mu = \text{Hypo } y : \checkmark$

• $\text{esHypo}(\text{Hypo } y) = \text{mismaEntw}(\text{Hypo } y)(\text{Hypo } x)$
 $\downarrow \{M_0\}$ \checkmark

$\{E_0\} \downarrow \text{esHypo}(\text{Hypo } y) = \text{esHypo}(\text{Hypo } x)$ $\downarrow \{E_0\}$ \checkmark
true = true
vde x Bool

1.b)

$\mu = \text{Bin} \mid d : , \text{iss. Alt a}, \text{B} \dashv, \text{Alt a}$

• $\text{esHypo}(\text{Bin} \mid d) = \text{mismaEstw}(\text{Bin} \mid d)(\text{Hypo } x)$

$\downarrow \{E_1\}$ $\downarrow \{M_1\}$ \checkmark

false = ($\text{it} \rightarrow \text{not}(\text{esHypo}) \& \text{mismaE} \mid (\text{izf t})$
 $\& \text{mismaEst} \mid (\text{dr t})$)
 $(\text{Hypo } x)$

Adicco β_{False} ~~que~~ ~~otro~~ que apriaste β



$\not\models \equiv \text{not}(\text{esHypo}(\text{Hypo } x)) \& \text{mismaE} \mid (\text{izf } (\text{Hypo } x))$
 $\& \text{mismaEst} \mid (\text{dr } (\text{Hypo } x)) \not\models$

$\{E_0\}$

$\equiv \text{not}(\text{true}) \& \text{mismaE} \mid (\text{izf } (\text{Hypo } x))$

~~dos not~~
~~Bool~~

$\& \text{mismaEst} \mid (\text{dr } (\text{Hypo } x))$

$\equiv \text{false} \& \dots \& \dots \not\models \text{Bool}$

$= \text{false} \quad , \quad \text{true} \neq \text{false} \quad (= \not\models)$

1620/22

Sebastián Andri

HA 3

x::=

* Entonces vemos que para $t = \text{Hg}_a \times \text{funcion}$
 ahora vemos caso $t = \text{Bin} i _ d$, $i :: \text{Alt} a$, $d :: \text{Alt} a$

$$\rightarrow P(\text{Bin} i _ d) = \dots$$

$$\equiv \text{MismoEstr} (\text{Bin} i _ d) \quad \mu = \text{MismoEstr} \mu (\text{Bin} i _ d)$$

+ para este caso vamos a usar ~~la hipótesis~~ la siguiente
 hipótesis inductiva:

$$HI = P(i) \wedge P(d).$$

Entonces, suponemos que vale:

$$P(i) = \text{MismoE} i _ w = \text{MismoE} w _ i, \quad \forall w :: \text{Alt} a$$

$$P(d) = \text{MismoE} d _ z = \text{MismoE} z _ d, \quad \forall z :: \text{Alt} a.$$

Luego, probaremos:

$$\text{MismoEstr} (\text{Bin} i _ d) \quad \mu = \text{MismoEstr} \mu (\text{Bin} i _ d)$$

$$\downarrow \{ M_i, \{ \} \}$$

$$\left((it \rightarrow \text{not}(\text{Elgo } t)) \wedge \begin{array}{l} \text{MismoE } i _ (\text{izq}(t)) \\ \& \text{MismoE } d _ (\text{der}(t)) \end{array} \right) \mu = \text{MismoEstr} \mu (\text{Bin} i _ d)$$

↓ Aplicar p

$$\left((\text{not}(\text{Elgo } \mu)) \wedge \begin{array}{l} \text{MismoE } i _ (\text{izq } \mu) \\ \& \text{MismoE } d _ (\text{der } \mu) \end{array} \right) = \text{MismoEstr} \mu (\text{Bin} i _ d)$$

Al igual que en el caso base voy a tener que
 mostrar extensiónalidad de Alt para μ :

$$\rightarrow 2.a) \quad \mu = \text{Hg}_a \times \quad , \quad x = a$$

$$\rightarrow 2.b) \quad \mu = \text{Bin } m _ n _ , \quad m :: \text{Alt} a, \quad n :: \text{Alt} a$$

$$2.9) \underline{M = \text{Hgo } x} \quad x::a$$

$$\left(\begin{array}{l} \text{not} (\text{cHgo} (\text{Hgo } x)) \text{ sf} \\ \text{MSME i } (\text{izp} (\text{Hgo } x)) \text{ sf} \\ \text{MSME d } (\text{der} (\text{Hgo } x)) \end{array} \right) = \text{MSME est } (\text{Hgo } x) \text{ (Bin 1d)}$$

$\downarrow \{E_0\}$

$\downarrow \{M_0\}$

$$\left(\begin{array}{l} \text{not} (\text{true}) \text{ sf} \\ \text{MSME i } (\text{izp} (\text{Hgo } x)) \text{ sf} \\ \text{MSME d } (\text{der} (\text{Hgo } x)) \end{array} \right) = \text{es Hgo (Bin 1d)}$$

$\downarrow \text{not.}$

$\downarrow \{E_1\}$

$$\left(\text{false sf} \dots \text{sf} \dots \right) = \text{false}$$

$\downarrow \{x \text{Bool}\}$

$$\text{false} = \text{false}$$

$\downarrow \text{true}$

$\downarrow \text{Bool}$

Luego este caso es válido.

1020/22

Sebastián Andrés

HGA 4

Order 12

2.b) $M = \text{Bin}(M \cdot N)$, $M :: A1H \alpha$, $N :: A1H \alpha$

$$\left(\begin{array}{l} \text{not}(\text{esHga}(\text{Bin } M \cdot N)) \& \& \\ \text{MISME i } (\text{izq}(\text{Bin } M \cdot N)) \& \& \\ \text{MISME d } (\text{der}(\text{Bin } M \cdot N)) \end{array} \right) \checkmark$$

$\downarrow 3M_1 \beta + \text{aplicar } \beta$

$$\left(\begin{array}{l} \text{not}(\text{false}) \& \& \\ \text{MISME i } (M) \& \& \\ \text{MISME d } (N) \end{array} \right) = \left(\begin{array}{l} \text{not}(\text{esHga}(\text{Bin } 1)) \& \& \\ \text{MISME i } (\text{izq}(\text{Bin } 1)) \& \& \\ \text{MISME d } (\text{der}(\text{Bin } 1)) \end{array} \right) \checkmark$$

$$\left(\begin{array}{l} \text{true} \& \& \\ \& \& \\ \& \& \end{array} \right) = \left(\begin{array}{l} \text{not}(\text{false}) \& \& \\ \text{MISME M i } \& \& \\ \text{MISME N d } \end{array} \right) \checkmark$$

*por q
"saltando"
el true*

not + (d)

$$(\text{MISME i } M \& \& \text{MISME d } N) = (\text{MISME M i } \& \& \\ \text{MISME N d})$$

En particular esto vale por $\text{avanza } H$

$$P(i) = \text{IH} :: A1H \cdot \text{MISME i } M = \text{MISME M i}$$

$$P(d) = \text{VN} :: A1H \cdot \text{MISME d } N = \text{MISME N d}$$

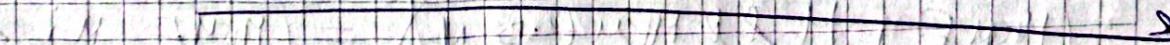
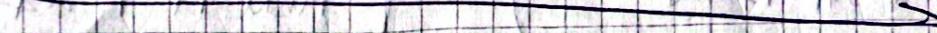
Luego vale que si alguna afirmación H en i es falsa \rightarrow del otro lado pasa \ominus .

Entonces vale este caso.

Como vemos que vale $P(t)$ para todos los constructores del árbol t

$$\rightarrow \forall t \in A \setminus H_2 . P(t)$$

PER



quedo! $\vdash (\lambda x : N \rightarrow N \rightarrow X_4 . x (\lambda x : N . s(x))) (\lambda x : N \rightarrow X) : X_4 \rightarrow X_4$

10/01/22

Saborío Andrés

Order 12

Mgu: 5

B) Inferir tipos tpedo:

$$\boxed{3x:(N \rightarrow N \rightarrow X_4)} \quad \phi \vdash (\lambda x : N \rightarrow N \rightarrow X_4 . x (\lambda x : N . s(x))) (\lambda x : N \rightarrow X) : X_4 \rightarrow X_4$$

$$\boxed{3x:(N \rightarrow N \rightarrow X_4)} \quad \phi \vdash (\lambda x : N \rightarrow N \rightarrow X_4 . x (\lambda x : N . s(x))) (\lambda x : N \rightarrow X) : X_4 \rightarrow X_4$$

$$\boxed{3x:(N \rightarrow N \rightarrow X_4)} \quad \phi \vdash (\lambda x : N \rightarrow N \rightarrow X_4 . x (\lambda x : N . s(x))) (\lambda x : N \rightarrow X) : X_4 \rightarrow X_4$$

$$\boxed{3x:(N \rightarrow N \rightarrow X_4)} \quad \phi \vdash (\lambda x : N \rightarrow N \rightarrow X_4 . x (\lambda x : N . s(x))) (\lambda x : N \rightarrow X) : X_4 \rightarrow X_4$$

$$x : X_1 \beta \vdash x = X_1 \quad \phi \vdash \lambda x : N \rightarrow N . s(x) : N \rightarrow N \quad \checkmark$$

$$3x : N \exists \vdash s(x) : N \quad \star \text{ Te faltó realizar una unificación más } \checkmark$$

Bastante despropósito

$$3x : X_2 \exists \vdash x = X_2 \quad (N \rightarrow N) \rightarrow X_4 \rightarrow X_4 \quad (N \rightarrow N) \rightarrow N \rightarrow N$$

Aux 1 | $|W((x . (\lambda x : N . s(x)))) \rightsquigarrow S(3x : X_1) \cup S(\phi) \vdash \dots : s(X_4)$

$S = Mgu \{ X_1 = (N \rightarrow N) \rightarrow X_4 \}$

Aux 2 | $|W(x (\lambda x : N . s(x))) \rightsquigarrow \{ X_1 : (N \rightarrow N) \rightarrow X_4 \} \vdash \dots : X_4$

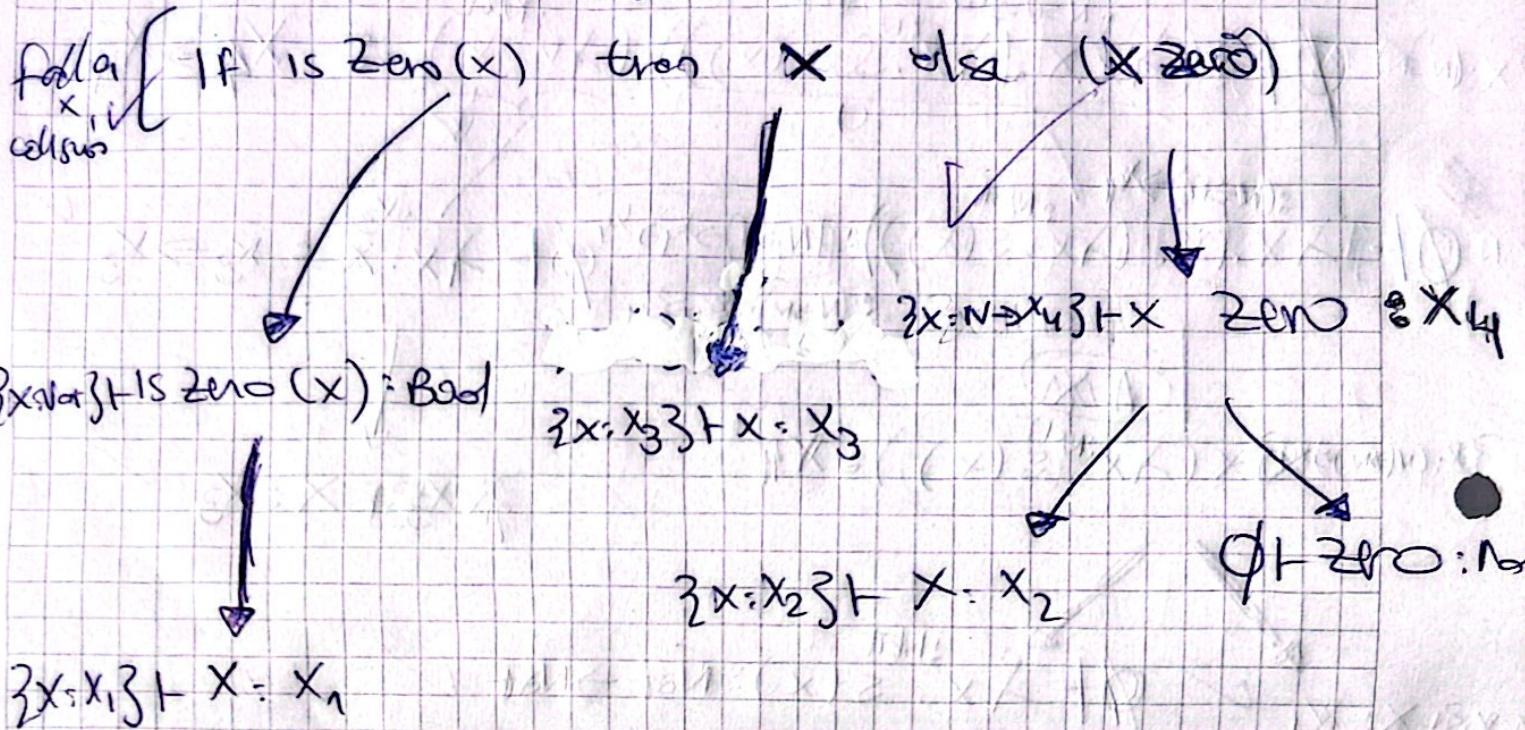
Aux 2 | $|W(UV) \rightsquigarrow S(\phi) \cup S(\phi) + UN : S(X_5)$

$S = Mgu \{ N \rightarrow N \rightarrow X_4 \rightarrow X_4 \equiv X_3 \rightarrow X_3 \rightarrow X_5 \} \rightarrow \{ X_5 = X_4 \}$

$|W(UV) \Rightarrow \phi \vdash UV : X_4 \rightarrow X_4$

⑥ \Rightarrow No se puede unificar
+ RS

$\lambda x. \text{ if } \text{isZero}(x) \text{ then } x \text{ else } (\bar{x} \text{ zero})$
 $\downarrow (\text{AUX})$



* $\text{IN}(\text{if } \text{isZero}(x) \text{ then } x \text{ else } (\bar{x} \text{ zero})) \rightsquigarrow$

$S(\exists x. \{x = \text{Not}\}) \cup S(\exists x. \{x = x_3\}) \cup S(\exists x. \{x = \text{Not} \rightarrow x_4\}) \vdash \text{if } \dots : x_3$

$S = M \wp \cup \{ x_3 = \text{Not} \rightarrow x_4, \text{Bool} = \text{Bool} \}$

$\cup \{ x_3 = \text{Not}, \text{Not} \rightarrow x_4 = x_3, \text{Not} = \text{Not} \rightarrow x_4 \}$

extra no unifica
(colisión)

Luego, ~~una solución~~ vale falso:

u Si E no tiene solución \rightarrow el algoritmo W falla y falso

u Si E tiene solución \rightarrow el alg. llega a \emptyset

Entonces, E no tiene un uniformador posible.

③ Cálculo → tipos:

$$\chi ::= \dots | AITH(\chi)$$

$$M ::= \dots | Hga(M) | \text{BIN}(M, M)$$

| CASE M of $Hga \times \rightarrow M_j$

$$\text{Bin}(1, d) \rightarrow M.$$

(a) Reglas de tipo

$$\frac{\Gamma \vdash M : \chi}{\Gamma \vdash Hga(M) : AITH_\chi} + \text{-Hga.}$$

$$\frac{\Gamma \vdash M : AITH_\chi \quad \Gamma \vdash N : AITH_\chi}{\Gamma \vdash \text{BIN}(M, N) : AITH_\chi} \text{-BIN}$$

$$\frac{\Gamma \vdash M : AITH_\chi \quad \Gamma, x : \chi \vdash N : \sigma}{\Gamma, i : AITH_\chi, d : AITH_\chi \vdash O : \sigma} \text{-CASE}$$

$$\Gamma \vdash \text{CASE } M \text{ of } Hga \times \rightarrow N_j \text{ Bin}(1, d) \rightsquigarrow O : \sigma$$

(b) Cojunto de valores:

- Vemos los términos irreducibles.

$$v ::= \dots | Hga(v) | \text{BIN}(v, v) . \checkmark$$

Reglas semánticas

Reglas de Semântica operacional small step:

① (case hga):

$$\left(\text{CASE } (\text{Hga} \times) \text{ of } (\text{Hga } x) \rightsquigarrow N_j ; (\text{Bin}(i,d)) \rightsquigarrow O \right) \xrightarrow{\quad} N_j \{ x=y \}$$

Deben ser Valores

② (case Bin):

$$\left(\text{CASE } (\text{Bin } M \times N) \text{ of } (\text{Hga } x) \rightsquigarrow N_j ; (\text{Bin}(i,d)) \rightsquigarrow O \right) \xrightarrow{\quad} O \{ i=M, d=N \}$$

y solo hay una regla de congruencia:

$$M \longrightarrow M'$$

$$\left(\text{CASE } M \text{ of } (\text{Hga } x) \rightsquigarrow N_j ; (\text{Bin}(i,d)) \rightsquigarrow O \right) \xrightarrow{\quad} \left(\text{CASE } M' \text{ of } (\text{Hga } x) \rightsquigarrow N_j ; (\text{Bin}(i,d)) \rightsquigarrow O \right)$$

③ Reducir:

$$\text{case } (\lambda n : \text{Nat}. \text{Iota}(n)) \text{ suc(zero)} \text{ of } \text{Hga } x \rightsquigarrow \text{suc}(\text{Pred}(x)) \\ \text{Bin}(1,d) \rightsquigarrow \text{zero}$$

6

1028/22

Sebastián Ardis

May A 7
order 12

CASE 1) $n : \text{Nat} . \text{Hfc}(n) . S(0)$ of $\text{Hfc} X \rightarrow S(\text{P}(X))$;
- $\text{Br}(1,1) \rightarrow 0$

CASE commercial + B

Te confundiste en la
expresión a reducir, tenés
 x , no zero

CASE $Hg_a(s(\underline{o}))$ or $Hg_x \rightarrow s(p(\underline{x}))$
 $Bun(1,2) \rightarrow \underline{o}$

CASE Hints

1

Page 2

1

\hookrightarrow Mol applications

$\rho(0)$)

$$2x = S(p) \}$$

3

F10

$$\left(\text{Pred}(o) \xrightarrow{o} \text{(Aetrica)} \right)$$

Resetas aplicadas.

1

Macro es tipo Σ

10

* $\text{estHga}_x = \lambda A: \text{Alt}(x). \text{ CASE } A \text{ OF } (\text{bf} x) \rightarrow \text{true};$
 $(\text{Bin}(1,2)) \rightarrow \text{false}$