1. Esquemas de recursión

```
map :: (a -> b) -> [a] -> [b]
map _ [] = []
map f (x:xs) = (f x) : (map f xs)

filter :: (a -> Bool) -> [a] -> [a]
filter _ [] = []
filter p (x:xs) | (p x)      = x : (filter p xs)
                | otherwise = filter p xs

foldr :: (a -> b -> b) -> b -> [a] -> b
foldr _ z [] = z
foldr f z (x:xs) = f x (foldr f z xs)

foldl :: (b -> a -> b) -> b -> [a] -> b
foldl f z [] = z
foldl f z (x : xs) = foldl f (f z x) xs

recr :: b -> (a -> [a] -> b -> b) -> [a] -> b
recr z _ []= z
recr z f (x:xs) = f x xs (recr z f xs)

type DivideConquer a b  = (a -> Bool) -> (a -> b) -> (a -> [a]) ->
                          ([b] -> b) -> a -> b

divideConquerListas :: DivideConquer [a] b
-- Esto significa que DivideConquerLista es de tipo
-- ([a] -> Bool) -> ([a] -> b) -> ([a] -> [[a]]) -> ([b] -> b)
-- -> [a] -> b

divideConquerListas esTrivial resolver repartir combinar l =
        if (esTrivial l) then resolver l
        else combinar (map dc (repartir l))
where dc = divideConquerListas esTrivial resolver repartir combinar
```

2. Cálculo lambda λ^{bn}

Función Free Variables

$$\begin{aligned}
FV(x) &\stackrel{def}{=} x \\
FV(true) = FV(false) &\stackrel{def}{=} \emptyset \\
FV(if\ M\ then\ P\ else\ Q) &\stackrel{def}{=} FV(M) \cup FV(P) \cup FV(Q) \\
FV(M\ N) &\stackrel{def}{=} FV(M) \cup FV(N) \\
FV(\lambda x : \sigma.M) &\stackrel{def}{=} FV(M) \backslash \{x\}
\end{aligned}$$

Sustitución

$$\begin{aligned}
x\{x \leftarrow N\} &\stackrel{def}{=} N \\
a\{x \leftarrow N\} &\stackrel{def}{=} a \text{ si } a \in \{true, false\} \cup \mathcal{X} \backslash \{x\} \\
(if\ M\ then\ P\ else\ Q)\{x \leftarrow N\} &\stackrel{def}{=} if\ M\{x \leftarrow N\}\ then\ P\{x \leftarrow N\}\ else\ Q\{x \leftarrow N\} \\
(M_1\ M_2)\{x \leftarrow N\} &\stackrel{def}{=} M_1\{x \leftarrow N\}\ M_2\{x \leftarrow N\} \\
(\lambda y : \sigma.M)\{x \leftarrow N\} &\stackrel{def}{=} \lambda y : \sigma.M\{x \leftarrow N\}\ x \neq y,\ y \notin\ FV(N)
\end{aligned}$$

2.0.1. Tipos

$\sigma, \tau \ ::= \ Bool \mid Nat \mid \sigma \to \tau$

Expresiones

$$\begin{aligned}
M, P, Q \ ::= \ & true \mid false \mid if\ M\ then\ P\ else\ Q \\
& \mid M\ N \mid \lambda x : \sigma.M \\
& \mid x \mid 0 \mid succ(M) \mid pred(M) \mid isZero(M)
\end{aligned} \tag{1}$$

2.0.2. Reglas de tipado

$$\frac{}{\Gamma \triangleright true : Bool}\text{(T-True)} \qquad \frac{}{\Gamma \triangleright false : Bool}\text{(T-False)}$$

$$\frac{x : \sigma \in \Gamma}{\Gamma \triangleright x : \sigma}\text{(T-Var)} \qquad \frac{\Gamma \triangleright M : Bool \quad \Gamma \triangleright P : \sigma \quad \Gamma \triangleright Q : \sigma}{\Gamma \triangleright if\ M\ then\ P\ else\ Q : \sigma}\text{(T-If)}$$

$$\frac{\Gamma, x : \sigma \triangleright M : \tau}{\Gamma \triangleright \lambda x : \sigma.M : \sigma \to \tau}\text{(T-Abs)} \qquad \frac{\Gamma \triangleright M : \sigma \to \tau \quad \Gamma \triangleright N : \sigma}{\Gamma \triangleright M\ N : \tau}\text{(T-App)}$$

$$\frac{}{\Gamma \triangleright 0 : Nat}\text{(T-Zero)}$$

$$\frac{\Gamma \triangleright M : Nat}{\Gamma \triangleright succ(M) : Nat}\text{(T-Succ)} \qquad \frac{\Gamma \triangleright M : Nat}{\Gamma \triangleright pred(M) : Nat}\text{(T-Pred)}$$

$$\frac{\Gamma \triangleright M : Nat}{\Gamma \triangleright isZero(M) : Bool}\text{(T-IsZero)}$$

Valores

$V \ ::= \ true \mid false \mid \lambda x : \sigma.M \mid \underline{n}$ donde $\underline{n}$ abrevia $succ^n(0)$

Reglas de semánticas

$$\frac{}{if\ true\ then\ M_1\ else\ M_2 \to M_1}\text{(E-IfTrue)}$$

$$\frac{}{if\ false\ then\ M_1\ else\ M_2 \to M_2}\text{(E-IfFalse)}$$

$$\frac{M_1 \to M_1'}{if\ M_1\ then\ M_2\ else\ M_3 \to if\ M_1'\ then\ M_2\ else\ M_3}\text{(E-If)}$$

$$\frac{M_1 \to M_1'}{M_1\ M_2 \to M_1'\ M_2}\text{(E-App1 / }\mu\text{)} \qquad \frac{M_2 \to M_2'}{V_1\ M_2 \to V_1\ M_2'}\text{(E-App2 / }v\text{)}$$

$$\frac{}{(\lambda x : \sigma.M)\ V \to M\{x \leftarrow V\}}\text{(E-App3 / }\beta\text{)}$$

$$\frac{M_1 \to M_1'}{succ(M_1) \to succ(M_1')}\text{(E-Succ)}$$

$$\frac{}{pred(0) \to 0}\text{(E-PredZero)} \qquad \frac{}{pred(succ(\underline{n})) \to \underline{n}}\text{(E-PredSucc)}$$

$$\frac{M_1 \to M_1'}{pred(M_1) \to pred(M_1')}\text{(E-Pred)}$$

$$\frac{}{isZero(0) \to true}\text{(E-IsZeroZero)} \qquad \frac{}{isZero(succ(\underline{n})) \to false}\text{(E-isZeroSucc)}$$

$$\frac{M_1 \to M_1'}{isZero(M_1) \to isZero(M_1')}\text{(E-isZero)}$$

3. Extensión con memoria λ^{bnu}

Tipos

$$\sigma, \tau \ ::= \ Bool \ |Nat \ | \ Unit \ | \ Ref \ \sigma \ | \ \sigma \to \tau$$

Términos

$$M \ ::= \ \ldots \ | \ unit \ | \ ref \ M \ | \ !M \ | \ M \ := \ N!| \ l$$

Axiomas y reglas de tipado

$$\frac{}{\Gamma|\Sigma \triangleright unit : Unit}\text{(T-Unit)} \qquad \frac{\Gamma|\Sigma \triangleright M_1 : \sigma}{\Gamma|\Sigma \triangleright ref \ M_1 : Ref \ \sigma}\text{(T-Ref)}$$

$$\frac{\Gamma|\Sigma \triangleright M_1 : Ref \ \sigma}{\Gamma\triangleright !M_1 : \sigma}\text{(T-DeRef)}$$

$$\frac{\Gamma|\Sigma \triangleright M_1 : Ref \ \sigma \qquad \Gamma|\Sigma \triangleright M_2 : \sigma}{\Gamma \triangleright M_1 \ := \ M_2 : Unit}\text{(T-Assing)}$$

$$\frac{\Sigma(l) = \sigma}{\Gamma|Signa \triangleright l : Ref \ \sigma}\text{(T-Loc)}$$

Valores

$$V \ ::= \ \ldots \ | \ unit \ | \ l$$

Axiomas y reglas semánticas

$$\frac{M_1|\mu \to M_1'|\mu'}{M_1 \ M_2|\mu \to M_1' \ M_2|\mu'}\text{(E-App1)} \qquad \frac{M_2|\mu \to M_2'|\mu'}{V_1 \ M_2|\mu \to V_1 \ M_2'|\mu'}\text{(E-App2)}$$

$$\frac{}{(\lambda x : \sigma.M) \ V|\mu \to M\{x \leftarrow V\}|\mu'}\text{(E-AppAbs)}$$

$$\frac{M_1|\mu \to M_1'|\mu'}{!M_1|\mu \to !M_1'|\mu'}\text{(E-DeRef)} \qquad \frac{\mu(l) = V}{!l|\mu \to V|\mu}\text{(E-DerefLoc)}$$

$$\frac{M_1|\mu \to M_1'|\mu'}{M_1 \ := \ M_2 \ |\mu \to M_1' \ := \ M_2|\mu'}\text{(E-Assign1)}$$

$$\frac{M_2|\mu \to M_2'|\mu'}{V \ := \ M_2|\mu \to V \ := \ M_2'|\mu'}\text{(E-Assign2)}$$

$$\frac{}{l \ := \ V|\mu \to unit|\mu[l \to V]}\text{(E-Assign)}$$

$$\frac{M_1|\mu \to M_1'|\mu'}{ref \ M_1|\mu \to ref \ M_1'|\mu'}\text{(E-Ref)} \qquad \frac{l \notin Dom(\mu)}{ref \ V|\mu \to l|\mu \oplus (l \to V)}\text{(E-RefV)}$$

4. Extensión con recursión $\lambda^{..r}$

Términos

$$M \;:=\; \ldots \mid \mathit{fix}\ M$$

Regla de tipado

$$\frac{\Gamma \triangleright M : \sigma \to \sigma}{\Gamma \triangleright \mathit{fix}\ M : \sigma}\,(\text{T-Fix})$$

Reglas de evaluación

$$\frac{M_1 \to M_1'}{\mathit{fix}\ M_1 \to \mathit{fix}\ M_1'}\,(\text{E-Fix})$$

$$\frac{}{\mathit{fix}\ (\lambda x : \sigma.M) \to M\{x \leftarrow \mathit{fix}\ \lambda x : \sigma.M\}}\,(\text{E-FixBeta})$$

5. Extensión con Declaraciones Locales ($\lambda^{...let}$)

Con esta extensión, agregamos al lenguaje el término $let\ x : \sigma = M\ in\ N$, que evalúa M a un valor, liga x a V y, luego, evalúa N. Este término solo mejora la legibilidad de los programas que ya podemos definir con el lenguaje hasta ahora definido.

Términos

$$M \;::=\; \ldots \mid let\ x : \sigma = M\ in\ N$$

Axiomas y reglas de tipado

$$\frac{\Gamma \triangleright M : \sigma_1 \qquad \Gamma, x : \sigma_1 \triangleright N : \sigma_2}{\Gamma \triangleright let\ x : \sigma_1 = M\ in\ N : \sigma_2}\,(\text{T-Let})$$

Axiomas y reglas de evaluación

$$\frac{M_1 \to M_1'}{let\ x : \sigma = M_1\ in\ M_2 \to let\ x : \sigma = M_1'\ in\ M_2}\,(\text{E-Let})$$

$$\frac{}{let\ x : \sigma = V_1\ in\ M_2 \to M_2\{x \leftarrow V_1\}}\,(\text{E-LetV})$$

5.0.1. Construcción *let* recursivo (Letrec)

Una construcción alternativa para definir funciones recursivas es

$$letrec\ f : \sigma \to \sigma = \lambda x : \sigma.M\ in\ N$$

Y $letRec$ se puede definir en base a let y fix (definido en **??**) de la siguiente forma:

$$let\ f : \sigma \to \sigma = (\mathit{fix}\ \lambda f : \sigma \to \sigma.\lambda x : \sigma.M)\ in\ N$$

6. Extensión con Registros $\lambda^{..r}$

Tipos

$$\sigma, \tau \ ::= \ ... \mid \{l_i : \sigma_i \ ^{i \in 1..n}\}$$

El tipo $\{l_i : \sigma_i^{i \in 1..n}\}$ representan las estructuras con n atributos tipados, por ejemplo: $\{nombre : String, edad : Nat\}$

Términos

$$M \ ::= \ \ldots \mid \{l_i = M_i \ ^{i \in 1..n}\} \mid M.l$$

Los términos significan:

- El registro $\{l_i = M_i \ ^{i \in 1..n}\}$ evalua $\{l_i = V_i \ ^{i \in 1..n}\}$ donde V_i es el valores al que evalúa M_i para $i \in 1..n$.
- $M.l$: Proyecta el valor de la etiqueta l del registro M

Axiomas y reglas de tipado

$$\frac{\Gamma \triangleright M_i : \sigma_i \text{ para cada } i \in 1..n}{\Gamma \triangleright \{l_i = M_i \ ^{i \in 1..n}\} : \{l_i : \sigma_i \ ^{i \in 1..n}\}} \text{(T-RCD)}$$

$$\frac{\Gamma \triangleright \{l_i = M_i \ ^{i \in 1..n}\} : \{l_i : \sigma_i \ ^{i \in 1..n}\} \qquad j \in 1..n}{\Gamma \triangleright M.l_j : \sigma_j} \text{(T-Proj)}$$

Valores

$$V \ ::= \ \ldots \mid \{l_i = V_i \ ^{i \in 1..n}\}$$

Axiomas y reglas de evaluación

$$\frac{j \in 1..n}{\{l_i = V_i \ ^{i \in 1..n}\}.l_j \to V_j} \text{(E-ProjRcd)}$$

$$\frac{M \to M'}{M.l \to M'.l} \text{(E-Proj)}$$

$$\frac{M_j \to M_j'}{\{l_i = V_i \ ^{i \in 1..j-1}, l_j = M_j, l_i = M_i \ ^{i \in j+1..n}\} \to \{l_i = V_i \ ^{i \in 1..j-1}, l_j = M_j', l_i = M_i \ ^{i \in j+1..n}\}} \text{(E-RCD)}$$

7. Extensión con tuplas

Tipos

$$\sigma, \tau \ ::= \ldots \mid \sigma \times \tau$$

Términos

$$M,\ N \ ::= \ \ldots \mid \ < M, N > \ \mid \pi_1(M) \mid \pi_2(M)$$

Axiomas y reglas de tipado

$$\frac{\Gamma \triangleright M : \sigma \qquad \Gamma \triangleright N : \tau}{\Gamma \triangleright < M, N >: \sigma \times \tau}\text{(T-Tupla)}$$

$$\frac{\Gamma \triangleright M : \sigma \times \tau}{\Gamma \triangleright \pi_1(M) : \sigma}\text{(T-}\pi_1\text{)} \qquad \frac{\Gamma \triangleright M : \sigma \times \tau}{\Gamma \triangleright \pi_2(M) : \tau}\text{(T-}\pi_2\text{)}$$

Valores

$$V \ ::= \ \ldots \mid \ < V, V >$$

Axiomas y reglas de evaluación

$$\frac{M \to M'}{< M, N > \to < M', N >}\text{(E-Tuplas)} \qquad \frac{N \to N'}{< V, N > \to < V, N' >}\text{(E-Tuplas1)}$$

$$\frac{M \to M'}{\pi_1(M) \to \pi_1(M')}\text{(E-}\pi_1\text{)} \qquad \frac{}{\pi_1(< V_1, V_2 >) \to V_1}\text{(E-}\pi_1'\text{)}$$

$$\frac{M \to M'}{\pi_2(M) \to \pi_2(M')}\text{(E-}\pi_2\text{)} \qquad \frac{}{\pi_2(< V_1, V_2 >) \to V_2}\text{(E-}\pi_2'\text{)}$$

8. Extensión con árboles binarios

Tipos

$$\sigma, \tau \ ::= \ldots \mid AB_\sigma$$

Términos

$$M,\ N \ ::= \ \ldots \mid \text{Nil}_\sigma \mid \text{Bin}(M,N,O) \mid \text{raiz}(M) \mid \text{der}(M) \mid \text{izq}(M) \mid \text{esNil}(M)$$

Axiomas y reglas de tipado

$$\frac{}{\Gamma \triangleright \text{Nil}_\sigma : AB_\sigma}\text{(T-Nil)} \qquad \frac{\Gamma \triangleright M : AB_\sigma \quad \Gamma \triangleright N : \sigma \quad \Gamma \triangleright O : AB_\sigma}{\Gamma \triangleright \text{Bin}(M,N,O) : AB_\sigma}\text{(T-Bin)}$$

$$\frac{\Gamma \triangleright M : AB_\sigma}{\Gamma \triangleright \text{raiz}(M) : \sigma}\text{(T-raiz)} \qquad \frac{\Gamma \triangleright M : AB_\sigma}{\Gamma \triangleright \text{der}(M) : AB_\sigma}\text{(T-der)}$$

$$\frac{\Gamma \triangleright M : AB_\sigma}{\Gamma \triangleright \text{izq}(M) : AB_\sigma}\text{(T-izq)} \qquad \frac{\Gamma \triangleright M : AB_\sigma}{\Gamma \triangleright \text{isNil}(M) : \mathit{Bool}}\text{(T-isNil)}$$

Valores

$$V \ ::= \ \ldots \mid \text{Nil} \mid \text{Bin}(V,V,V)$$

Axiomas y reglas de evaluación

$$\frac{M \to M'}{\text{Bin}(M,N,O) \to \text{Bin}(M',N,O)}\text{(E-Bin1)} \qquad \frac{N \to N'}{\text{Bin}(V,N,O) \to \text{Bin}(V,N',O)}\text{(E-Bin2)}$$

$$\frac{O \to O'}{\text{Bin}(V_1,V_2,O) \to \text{Bin}(V_1,V_2,O')}\text{(E-Bin3)}$$

$$\frac{M \to M'}{\text{raiz}(M) \to \text{raiz}(M')}\text{(E-Raiz1)} \qquad \frac{}{\text{raiz}(\text{Bin}(V_1,V_2,V_3)) \to V_2}\text{(E-Bin3)}$$

$$\frac{M \to M'}{\text{der}(M) \to \text{der}(M')}\text{(E-Der1)} \qquad \frac{}{\text{der}(\text{Bin}(V_1,V_2,V_3)) \to V_3}\text{(E-Der2)}$$

$$\frac{M \to M'}{\text{izq}(M) \to \text{izq}(M')}\text{(E-Izq1)} \qquad \frac{}{\text{izq}(\text{Bin}(V_1,V_2,V_3)) \to V_1}\text{(E-Izq2)}$$

$$\frac{}{\text{isNil}(M) \to \text{izq}(M')}\text{(E-isNil1)} \qquad \frac{}{\text{isNil}(\text{Bin}(V_1,V_2,V_3)) \to \mathit{false}}\text{(E-isNilBin)}$$

$$\frac{}{\text{isNil}(\text{Bin}(V_1,V_2,V_3)) \to \mathit{true}}\text{(E-isNilNil)}$$

9. Algoritmo de Martelli-Montanari

1. **Descomposición**

   $\{\sigma_1 \to \sigma_2 \doteq \tau_1 \to \tau_2\} \cup G \mapsto \{\sigma_1 \doteq \tau_1,\ \sigma_2 \doteq \tau_2\} \cup G$

2. **Eliminación de par trivial**

   $\{Nat \doteq Nat\} \cup G \mapsto G$

   $\{\mathrm{Bool} \doteq \mathrm{Bool}\} \cup G \mapsto G$

   $\{\mathrm{s} \doteq \mathrm{s}\} \cup G \mapsto G$

3. **Swap** Si σ no es una variable,

   $\{\sigma \doteq \mathrm{s}\} \cup G \mapsto \{\mathrm{s} \doteq \sigma\} \cup G$

4. **Eliminación de variable** Si $s \notin FV(\sigma)$

   $\{\mathrm{s} \doteq \sigma\} \cup G \mapsto_{\sigma/s} G[\sigma/s]$

5. **Falla**

   $\{\sigma \doteq \tau\} \cup G \mapsto \texttt{falla}$, con $(\sigma, \tau) \in T \cup T^{-1}$ y $T = \{(\mathrm{Bool}, Nat), (Nat, \sigma_1 \to \sigma_2), (\mathrm{Bool}, \sigma_1 \to \sigma_2\}$. Acá, la notación T^{-1} se refiere al conjunto con cada tupla de T invertida.

6. **Occur Check** Si $s \neq \sigma$ y $s \in FV(\sigma)$

   $\{\mathrm{s} \doteq \sigma\} \cup G \mapsto \texttt{falla}$

10. Función $\mathbb{W}$

Constantes y variables

$$\mathbb{W}(true) \stackrel{def}{=} \emptyset \rhd true : Bool$$
$$\mathbb{W}(false) \stackrel{def}{=} \emptyset \rhd false : Bool$$
$$\mathbb{W}(x) \stackrel{def}{=} \{x : s\} \rhd x : s, \ \ s \text{ variable fresca}$$
$$\mathbb{W}(0) \stackrel{def}{=} \emptyset \rhd 0 : Nat$$

Caso *succ*

$$\mathbb{W}(succ(U)) \stackrel{def}{=} S\Gamma \rhd S\ succ(M) : Nat$$

- $\mathbb{W}(U) = \Gamma \rhd M : \tau$
- $S = MGU\{\tau \doteq Nat\}$

Caso *pred*

$$\mathbb{W}(pred(U)) \stackrel{def}{=} S\Gamma \rhd S\ pred(M) : Nat$$

- $\mathbb{W}(U) = \Gamma \rhd M : \tau$
- $S = MGU\{\tau \doteq Nat\}$

Caso *isZero*

$$\mathbb{W}(isZero(U)) \stackrel{def}{=} S\Gamma \rhd S\ isZero(M) : Bool$$

- $\mathbb{W}(U) = \Gamma \rhd M : \tau$
- $S = MGU\{\tau \doteq Nat\}$

Caso *ifThenElse*

$\mathbb{W}(if\ U\ then\ V\ else\ W) \stackrel{def}{=} S\Gamma_1 \cup S\Gamma_2 \cup S\Gamma_3 \triangleright S\ (if\ M\ then\ P\ else\ Q) : S\sigma$

- $\mathbb{W}(U) = \Gamma_1 \triangleright M : \rho$
- $\mathbb{W}(V) = \Gamma_2 \triangleright P : \sigma$
- $\mathbb{W}(W) = \Gamma_3 \triangleright Q : \tau$
- $S = MGU\{\sigma_1 \doteq \sigma_2 \mid x : \sigma_1 \in \Gamma_i\ \wedge\ x : \sigma_2 \in \Gamma_j,\ i \neq j\} \cup \{\sigma \doteq \tau\ \ \rho \doteq Bool\}$

Caso aplicación

$\mathbb{W}(U\ V) \stackrel{def}{=} S\Gamma_1 \cup S\Gamma_2 \triangleright S\ (M\ N) : St$

- $\mathbb{W}(U) = \Gamma_1 \triangleright M : \tau$
- $\mathbb{W}(V) = \Gamma_2 \triangleright N : \rho$
- $S = MGU\{\sigma_1 \doteq \sigma_2 \mid x : \sigma_1 \in \Gamma_i\ \wedge\ x : \sigma_2 \in \Gamma_j,\ i \neq j\} \cup \{\tau \doteq \rho \to t\}$ con t variable fresca

Caso abstracción

Sea $\mathbb{W}(U) = \Gamma \triangleright M : \rho$, si Γ tiene información de tipos para x, es decir $x : \tau \in \Gamma$ para algún τ, entonces:

$$\mathbb{W}(\lambda x.U) \stackrel{def}{=} \Gamma \backslash \{x : \tau\} \triangleright \lambda x : \tau.M : \tau \to \rho$$

Si Γ no tiene información de tipos para x ($x \notin \mathrm{Dom}(\Gamma)$), entonces elegimos una variable fresca s y

$$\mathbb{W}(\lambda x.U) \stackrel{def}{=} \Gamma \triangleright \lambda x : s.M : s \to \rho$$

Caso *fix*

$\mathbb{W}(fix\ (U)) \stackrel{def}{=} S\Gamma \triangleright S\ fix\ (M) : St$

- $\mathbb{W}(U) = \Gamma_1 \triangleright M : \tau$
- $S = MGU\{\tau \doteq t \to t\}$ con t variable fresca