

---

## Polymorphisme

---

### Les différentes formes du polymorphisme

---

- Polymorphisme ad-hoc ou surcharge
- Polymorphisme d'inclusion ou sous-typage
- Polymorphisme paramétrique ou générique

### Motivations

---

- Décrire le comportement d'une opération de façon générale, i.e. de façon indépendante de la nature (ou type) de ses paramètres.
- Employer une même opération avec plusieurs types différents. Ceci rend le programme plus claire et plus succinct.
- Faciliter la réutilisation.
- Faire porter le même nom à des opérations qui se "ressemblent" d'un point de vue sémantique.

1

### Le polymorphisme ad-hoc

---

Dans les langages orientés objet  $\Rightarrow$  définir des méthodes de même nom dans deux classes indépendantes.

Dans les langages procéduraux  $\Rightarrow$  définir deux fonctions de même nom mais ayant des paramètres différents (type ou nombre).

**Exemple :** Opérateur  $+$  pour réaliser l'addition entre deux nombres complexes, entiers ou réels.

**Difficulté :** Pour chaque utilisation de l'opérateur le compilateur doit décider quel code exécuter.

2

3

## Le polymorphisme d'inclusion

---

**Définition :** Un type  $A'$  est un **sous-type** (ou **est inclus** dans) un type  $A$  si n'importe quelle opération définie sur  $A$  est aussi définie sur  $A'$ .

**Intuition :** Un objet de type  $A'$  peut remplacer sans danger un objet de type  $A$ .

### Sous-typage entre objets :

Soient  $A = \langle m_1 : A_1; \dots; m_n : A_n \rangle$  et  $A' = \langle m_1 : A'_1; \dots; m_n : A'_n; m_{n+1} : C_{n+1}; \dots; m_k : C_k \rangle$ . On dit que  $A'$  est un sous-type de  $A$ , noté  $A' \leq A$ , ssi  $A'_i < A_i$  pour  $i \in \{1, \dots, n\}$ .

**Exemple :** Le type **colorpoint** est un sous-type du type **point**.

4

## Le polymorphisme paramétrique

---

### Motivation :

Plusieurs fonctions identité :

$Id_{int} : int \rightarrow int$ ,  $Id_{bool} : bool \rightarrow bool$ ,

$Id_{int \rightarrow int} : (int \rightarrow int) \rightarrow (int \rightarrow int)$ , ...

Plusieurs fonctions pour rajouter un élément à une liste :

$Aj_{int} : int \times list(int) \rightarrow list(int)$ ,

$Aj_{bool} : bool \times list(bool) \rightarrow list(bool)$ , ...

6

### Sous-typage dans les appels de fonctions :

Si  $\Gamma \vdash f : A \rightarrow B$ ,  $\Gamma \vdash M : A'$  et  $A' \leq A$  alors  $fM : B$ .

**Exemple :** Le polymorphisme d'inclusion permet de construire des listes hétérogènes où chaque élément est sous-type du type des éléments de la liste. Ainsi, on peut construire une liste de points (de couleur ou pas) et appliquer une fonction sur cette liste.

5

### Idée :

$Id : \forall \alpha. \alpha \rightarrow \alpha$

et donc

$Id_{int} = Id[int]$

$Id_{bool} = Id[bool]$

$Id_{int \rightarrow int} = Id[int \rightarrow int]$

**Remarque :** Ce qui caractérise le polymorphisme paramétrique est la relation d'instantiation qui existe entre le type général de la fonction polymorphe  $\alpha$ , et le type particulier de chaque spécialisation  $int, bool, int \rightarrow int$ .

7

## Le polymorphisme à la Girard-Reynolds

---

**Types :**  $A ::= \mathcal{T} \mid \alpha \mid A \times A \mid A \rightarrow A \mid \forall \alpha. A$

**Notation :**  $\forall \alpha \forall \beta. A = \forall (\alpha, \beta). A$ .

**Expressions :**

$$M ::= x \mid cte \mid \langle M, M \rangle \mid M M \mid$$
$$\lambda x : A. M \mid \text{let } x : A = M \text{ in } M \mid$$
$$M[A] \mid \Lambda \alpha. M$$

8

## Exemple

---

$Id = \Lambda \alpha. \lambda x : \alpha. x$

$(Id[int]) \Rightarrow \lambda x : int. x$

$(Id[int]) 3 \Rightarrow (\lambda x : int. x) 3 \Rightarrow 3$

10

## Règles de réduction

---

$(\lambda x : A. M) N \Rightarrow M\{x/N\}$

$\text{let } x : A = N \text{ in } M \Rightarrow M\{x/N\}$

$fix M \Rightarrow M (fix M)$

$fst \langle M, N \rangle \Rightarrow M$

$snd \langle M, N \rangle \Rightarrow N$

$\text{if true then } M \text{ else } N \Rightarrow M$

$\text{if false then } M \text{ else } N \Rightarrow N$

$(\Lambda \alpha. M)[A] \Rightarrow M\{\alpha/A\}$

9

**Définition :** Les variables libres d'un type sont définies par :

$VarLib(\mathcal{T}) = \emptyset$

$VarLib(\alpha) = \{\alpha\}$

$VarLib(\forall \alpha. A) = VarLib(A) \setminus \{\alpha\}$

$VarLib(A \times B) = VarLib(A) \cup VarLib(B)$

$VarLib(A \rightarrow B) = VarLib(A) \cup VarLib(B)$

Un type  $A$  est **clos** ssi  $VarLib(A) = \emptyset$ .

**Exemple :**  $VarLib(\beta \rightarrow (\forall \alpha. \alpha \times \gamma)) = \{\beta, \gamma\}$  et  $\forall \alpha \forall \beta. (\alpha \rightarrow \beta)$  est clos.

11

## Règles du typage polymorphe à la Church

---

L'ensemble de **variables libres d'un environnement** est donné par  $VarLib(\Gamma) = \bigcup_{x \in \Gamma} VarLib(\Gamma(x))$ .

Pour obtenir les règles du typage polymorphe à la Church on rajoute aux règles monomorphes :

$$\frac{\Gamma \vdash M : A \quad \alpha \notin VarLib(\Gamma)}{\Gamma \vdash \Lambda\alpha.M : \forall\alpha.A} \quad \frac{\Gamma \vdash M : \forall\alpha.A}{\Gamma \vdash M[B] : A\{\alpha/B\}}$$

12

## Algorithme de typage

---

Étant donné un terme  $M$ , le **problème de l'inférence de types** consiste à savoir si  $\exists \Gamma$  et  $\exists A$  t.q.  $\Gamma \vdash M : A$ .

**Théorème :** (Schubert) Le problème de l'inférence de types dans le système de Girard est **indécidable**.

**Idée :** Réduire la grammaire de types  $\rightarrow$  polymorphisme à la ML.

14

**Exemple :**

$$\frac{\frac{\frac{x : \alpha \vdash x : \alpha}{\vdash \lambda x : \alpha.x : \alpha \rightarrow \alpha}}{\vdash \Lambda\alpha.\lambda x : \alpha.x : \forall\alpha.(\alpha \rightarrow \alpha)}}{\vdash (\Lambda\alpha.\lambda x : \alpha.x)[int] : int \rightarrow int} \quad 3 : int}{\vdash ((\Lambda\alpha.\lambda x : \alpha.x)[int]) 3 : int}$$

13

## Le polymorphisme à la ML

---

**Type matrice :**  $M ::= \mathcal{T} \mid \alpha \mid M \times M \mid M \rightarrow M$

**Type :**  $S ::= \forall\alpha_1 \dots \forall\alpha_n.M$

Un type matrice est un cas particulier de type.

**Expressions :** Comme les expressions à la Curry

15

## Typage de constantes

---

$$\begin{aligned} TC(+): & \quad int \times int \rightarrow int \\ TC(fst): & \quad \forall(\alpha, \beta).(\alpha \times \beta) \rightarrow \alpha \\ TC(snd): & \quad \forall(\alpha, \beta).(\alpha \times \beta) \rightarrow \beta \\ TC(ifthenelse): & \quad \forall\alpha.(bool \times \alpha \times \alpha) \rightarrow \alpha \\ TC(fix): & \quad \forall\alpha.(\alpha \rightarrow \alpha) \rightarrow \alpha \end{aligned}$$

16

## Généralisations d'un type

---

**Définition :** L'opérateur  $Gen$  est donné par

$$Gen(A, \Gamma) = \forall(\alpha_1, \dots, \alpha_n).A, \text{ où } \{\alpha_1, \dots, \alpha_n\} = VarLib(A) \setminus VarLib(\Gamma).$$

**Exemple :** Soit  $A = \alpha \rightarrow \beta$  et  $\Gamma = x : \beta, y : \forall\alpha.\alpha$ . Alors

$$Gen(A, \Gamma) = \forall\alpha.\alpha \rightarrow \beta.$$

18

## Instances d'un type

---

**Définition :** Le type  $A$  est une instance du type  $\forall(\alpha_1, \dots, \alpha_n).B$ , noté  $A \leq \forall(\alpha_1, \dots, \alpha_n).B$  ssi ils existent  $C_1, \dots, C_n$  t.q.  $A = B\{\alpha_1, \dots, \alpha_n/C_1, \dots, C_n\}$ .

En particulier, pour  $n = 0$ , on a  $A \leq B$  ssi  $A \equiv B$ .

**Exemple :**

$$\begin{aligned} int \rightarrow int & \leq \forall\alpha.\alpha \rightarrow \alpha \\ bool \rightarrow int & \leq \forall(\alpha, \beta).\alpha \rightarrow \beta \\ bool \rightarrow int & \not\leq \forall\alpha.\alpha \rightarrow \alpha \end{aligned}$$

17

## Règles du typage polymorphe à la Curry

---

$$\frac{A \leq \Gamma(x)}{\Gamma \vdash x : A} \quad \frac{A \leq TC(cte)}{\Gamma \vdash cte : A}$$

$$\frac{\Gamma \vdash M : A \rightarrow B \quad \Gamma \vdash N : A}{\Gamma \vdash M N : B} \quad \frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : A \rightarrow B}$$

$$\frac{\Gamma \vdash M : A \quad \Gamma \vdash N : B}{\Gamma \vdash \langle M, N \rangle : A \times B} \quad \frac{\Gamma \vdash M : A \quad \Gamma, x : Gen(A, \Gamma) \vdash N : B}{\Gamma \vdash \text{let } x = M \text{ in } N : B}$$

19

## Exemple de dérivation

---

Soit  $A = \forall\alpha.\alpha \rightarrow \alpha$

$$\frac{\frac{y : \alpha \vdash y : \alpha}{\vdash \lambda y.y : \alpha \rightarrow \alpha} \quad \frac{\frac{int \rightarrow int \leq A}{f : A \vdash f : int \rightarrow int} \quad f : A \vdash 1 : int}{f : \forall\alpha.\alpha \rightarrow \alpha \vdash f 1 : int}}{\vdash \text{let } f = \lambda y.y \text{ in } f 1 : int}$$

20

## Inférence de types polymorphes

---

Soit  $M \equiv \text{let } x = \lambda y.y \text{ in } x$ .

On considère l'ensemble d'équations :

$$\{\alpha_M \doteq \alpha_x, \alpha_x \doteq \text{Gen}(\alpha_B, \emptyset), \alpha_B \doteq \alpha_y \rightarrow \alpha_y\}$$

où  $\alpha_B$  est le type de  $B \equiv \lambda y.y$ .

Si l'on traite la deuxième équation on obtient  $\alpha_x \equiv \forall\alpha_B.\alpha_B$ ,

**incorrect** car  $\alpha_x$  doit être un type fonctionnel.

22

## Propriétés

---

**(Stabilité)** Si  $\Gamma, x : \forall(\alpha_1, \dots, \alpha_n).B \vdash M : A$  et  $\Gamma \vdash N : B$ , alors  $\Gamma \vdash M\{x/N\} : A$ .

**(Préservation)** Si  $\Gamma \vdash M : A$  et  $M \Rightarrow M'$ , alors  $\Gamma \vdash M' : A$ .

21

## Vers un algorithme de typage : notations

---

Soit  $\Gamma = x_1 : A_1, \dots, x_n : A_n$  un environnement et soit  $\sigma$  une substitution de types. Alors  $\sigma(\Gamma) = x_1 : \sigma(A_1), \dots, x_n : \sigma(A_n)$ .

On note  $\text{inst}(\forall(\alpha_1, \dots, \alpha_n).A)$  le type  $A\{\alpha_1, \dots, \alpha_n/\beta_1, \dots, \beta_n\}$ , où  $\beta_1, \dots, \beta_n$  sont de variables **fraîches**.

23

## Sécification de l'algorithme

---

**Entrée** : Un environnement  $\Gamma$  et un terme  $M$  t.q.  $FV(M) \subseteq \Gamma$ .

**Sortie** : Un type  $A$  et une substitution  $\sigma$  t.q.  $\sigma(\Gamma) \vdash M : A$ .

24

## Premier exemple

---

$$W(\emptyset, \lambda x.x + x) = (int \rightarrow int, \{\alpha/int, \beta/int\})$$

$$W(x : \alpha, x + x) = (int, \{\alpha/int, \beta/int\})$$

$$W(x : \alpha, +) = (int \times int \rightarrow int, id)$$

$$W(id(x : \alpha), \langle x, x \rangle) = (\alpha \times \alpha, id)$$

$$W(x : \alpha, x) = (\alpha, id)$$

$$MGU(int \times int \rightarrow int \doteq \alpha \times \alpha \rightarrow \beta) = \{\alpha/int, \beta/int\}$$

26

## Algorithme de Damas-Milner-Tofte

---

$$W((\Delta, x : A), x) = (inst(A), id)$$

$$W(\Delta, cte) = (inst(A), id) \text{ où } A \text{ est le type de la constante } cte$$

$$W(\Delta, \lambda x.N) = (\rho(\alpha) \rightarrow B, \rho), \text{ où } W((\Delta, x : \alpha), N) = (B, \rho) \text{ et } \alpha \text{ est une variable fraiche}$$

$$W(\Delta, N L) = (\mu(\alpha), \mu \circ \rho_C \circ \rho_B), \text{ où } W(\Delta, N) = (B, \rho_B),$$

$$W(\rho_B(\Delta), L) = (C, \rho_C), \alpha \text{ est une variable fraiche et } \mu = MGU(\rho_C(B) \doteq C \rightarrow \alpha)$$

$$W(\Delta, \langle N, L \rangle) = (\rho_C(B) \times C, \rho_C \circ \rho_B), W(\Delta, N) = (B, \rho_B),$$

$$W(\rho_B(\Delta), L) = (C, \rho_C)$$

$$W(\Delta, \mathbf{let } x = N \mathbf{ in } L) = (C, \rho_C \circ \rho_B), \text{ où } W(\Delta, N) = (B, \rho_B),$$

$$W((\rho_B(\Delta), x : Gen(B, \rho_B(\Delta))), L) = (C, \rho_C).$$

25

## Deuxième exemple

---

$$W(\emptyset, \mathbf{let } f = \lambda x.x \mathbf{ in } f 2) = (int, \{\beta/int, \gamma/int\})$$

$$W(\emptyset, \lambda x.x) = (\alpha \rightarrow \alpha, id)$$

$$W(x : \alpha, x) = (\alpha, id)$$

$$W(f : \forall \alpha. \alpha \rightarrow \alpha, f 2) = (int, \{\beta/int, \gamma/int\})$$

$$W(f : \forall \alpha. \alpha \rightarrow \alpha, f) = (\beta \rightarrow \beta, id)$$

$$W(f : \forall \alpha. \alpha \rightarrow \alpha, 2) = (int, id)$$

$$MGU(\beta \rightarrow \beta \doteq int \rightarrow \gamma) = \{\beta/int, \gamma/int\}$$

27

## Propriétés de l'algorithme

---

**Théorème : (Correction)** Si  $W(\Delta, M) = (A, \sigma)$ , alors  $\sigma(\Delta) \vdash M : A$ .

**Théorème : (Complétude)** Si  $\tau(\Delta) \vdash M : B$ , alors  $W(\Delta, M) = (A, \sigma)$ , où  $B$  est une instance de  $A$  et  $\tau$  est une instance de  $\sigma$ .

**Corollaire :**  $\emptyset \vdash M : A$  ssi  $W(\emptyset, M) = (B, \sigma)$  et  $A$  est une instance de  $B$ .