Implantation et sémantique opérationnelle :

mémoire, partage, stratégies

Les modèles de calcul

- *Les modèles impératifs*
  - machines de Turing,
  - machines de Von Neumann.
- *Les modèles fonctionnels* (λ-calcul)

  Ils ont des inconvénients :
  - Les calculs peuvent être dupliqués.
  - La substitution n'a aucun «coût».

Quels modèles de calcul?

Peut-on remédier à cela, pour faire du modèle fonctionnel une *véritable alternative* au modèle impératif:

- sur le plan *théorique* (analyse de complexité),
- sur le plan *pragmatique*?

Oui avec un calcul avec adresses.

ALORS ALLONS Y !

Les objectifs d'un calcul avec adresses

- Donner un *statut* au modèle fonctionnel,
- Combler le *fossé* entre la théorie et la mise en œuvre.
- Internaliser la *substitution*,
- Incorporer les concepts de *mémoire* dans le calcul,
- Prendre en compte le *partage* de sous-termes,
- Faire un certain nombre de *preuves de propriétés* d'implantation:
  - absence de fuite de mémoire dans la machine STG (Peyton-Jones),
  - correction de la machine de Krivine à partage,
  - pas de «*method not found*» dans les langages à objets.

Plan de l’exposé

- Les langages fonctionnels
  - substitutions explicites
  - substitutions explicites avec adresses
  - réécriture simultanée et admissibilité
  - stratégies
  - récursion
- Les langages à objets,
- la syntaxe,
- les modules.

Les langages fonctionnels

Le calcul faible : λx

Termes

$$M, N ::= \lambda x.M \mid M\,N \mid x \qquad \text{(Pure)}$$

$$W ::= M[s] \mid W\,W \qquad \text{(Weak)}$$

$$s ::= W/x; s \mid \mathsf{id} \qquad \text{(Substitution)}$$

Beta-reduction

$$(\lambda x.M)[s]\,W \to M[W/x; s] \qquad \text{(B)}$$

Élimination des substitutions

$$(M\,N)[s] \to M[s]\,N[s] \qquad \text{(App)}$$

$$x[\ldots W/x \ldots] \to W \qquad \text{(Var)}$$

Le calcul faible, indices de de Bruijn : $\lambda\sigma_w$

Termes

$$W ::= \underline{n} \;\mid\; W W \;\mid\; M[s] \qquad \text{(Weak)}$$

$$M, N ::= \lambda M \;\mid\; M N \;\mid\; \underline{n} \qquad \text{(Pure)}$$

$$s ::= W \cdot s \;\mid\; \text{id} \qquad \text{(Substitution)}$$

Beta-reduction

$$(\lambda M)[s]\, W \to M[W \cdot s] \qquad \text{(B)}$$

Élimination des substitutions

$$(M N)[s] \to M[s]\, N[s] \qquad \text{(App)}$$

$$\underline{0}[W \cdot s] \to W \qquad \text{(FVar)}$$

$$\underline{n+1}[W \cdot s] \to \underline{n}[s] \qquad \text{(RVar)}$$

$$\underline{n}[\text{id}] \to \underline{n} \qquad \text{(VarId)}$$

Le calcul faible avec adresses : $\lambda\sigma_w^a$

La syntaxe

$$
\begin{array}{rll}
T, U, V ::= E^a \mid \perp & & \text{(Addressed)} \\
E, F ::= M[s] \mid UV \mid \underline{n} & & \text{(Evaluation Context)} \\
M, N ::= \lambda M \mid MN \mid \underline{n} & & \text{(Pure)} \\
s, t ::= \mathsf{id} \mid U \cdot s & & \text{(Substitution)}
\end{array}
$$

Les règles

Élimination des β-redex :

$\xrightarrow[B]{}$ est définie par la règle

$$((\lambda M)[s]^b\, U)^a \to M[U \cdot s]^a \qquad \text{(B)}$$

Élimination des substitutions : $\xrightarrow[\mathtt{x}]{}$ est définie par la règle

$$(MN)[s]^a \to (M[s]^b\, N[s]^c)^a \qquad b, c \text{ nouvelle} \qquad \text{(App)}$$

$$\underline{0}[E^b \cdot s]^a \to E^b \qquad \text{(FVarG)}$$

$$\underline{0}[E^b \cdot s]^a \to E^a \qquad \text{(FVarE)}$$

$$\underline{n+1}[U \cdot s]^a \to \underline{n}[s]^a \qquad \text{(RVar)}$$

$$\underline{n}[\mathsf{id}]^a \to \underline{n}^a \qquad \text{(VarId)}$$

Nettoyage:

$\xrightarrow[C]{}$ est définie par le règle

$$M[s]^a \to M[s|_{\mathsf{fv}(M)}]^a \qquad s \neq s|_{\mathsf{fv}(M)} \qquad \text{(Collect)}$$

où le *nettoyage de l'environnement* (*environment trimming*) est défini par $s|_I = s|_I^0$ où

- $\mathsf{id}|_I^i = \mathsf{id}$
- et $(U \cdot s)|_I^i = \begin{cases} U \cdot s|_I^{i+1} \text{ quand } i \in I \\ \perp \cdot s|_I^{i+1} \text{ quand } i \notin I. \end{cases}$

Admissibilité et simultanéité

- On ne considère que les termes *admissibles*, c-à-d ceux dont tous les termes à la même adresse sont identiques.
- La *réécriture simultanée* réécrit en même temps tous les termes à la même adresse.
- Donc la réécriture simultanée *préserve l'admissibilité*.

Réécriture de Graphes $\longleftrightarrow$ Machines à environnement

Les règles *(FVarE)* et *(FVarG)* marquent la bifurcation entre les machines à réécriture de graphes et les machines à environnement.

Qu'est-ce qu'une stratégie ?

On sépare complètement les stratégies de réduction des règles elles-mêmes. Le cadre est *générique*.

- Une stratégie est une relation $\vdash_{\mathcal{S}}$ qui fait correspondre une adresse à un terme.
- Pour une stratégie $\mathcal{S}$, la $\mathcal{S}$*-réduction* $\xrightarrow[X/\mathcal{S}]{}$ est définie par :

$$\boxed{\begin{array}{c} U_1 \xrightarrow[X/\mathcal{S}]{} U_2 \\ \text{si et seulement si} \\ U_1 \vdash_{\mathcal{S}} A \text{ and } U_1 \xrightarrow[A]{X} U_2. \end{array}}$$

Les règles de stratégie

$$\frac{}{M[s]^a \vdash \{a\}} \ \text{(Scl)} \qquad \frac{U \vdash A_1 \quad U \vdash A_2}{U \vdash A} \ A = A_1 \cup A_2 \ \text{(Par)}$$

$$\frac{s \vdash A}{M[s]^a \vdash A} \ \text{(Sub)} \qquad \frac{U \vdash A}{U \cdot s \vdash A} \ \text{(Hd)} \qquad \frac{s \vdash A}{U \cdot s \vdash A} \ \text{(Tl)}$$

$$\frac{}{(UV)^a \vdash \{a\}} \ \text{(Sap)} \qquad \frac{U \vdash A}{(UV)^a \vdash A} \ \text{(Lap)} \qquad \frac{V \vdash A}{(UV)^a \vdash A} \ \text{(Rap)}$$

L’appel par valeur

L’appel par valeur, c’est

(SCL) + (SAP) (si U et V sont des valeurs) + (LAP) (si U n’est pas une valeur)

+ (RAP) (si U est une valeur et V n’est pas une valeur)

L’appel par nécessité

L’appel par nécessité, c’est

(SCL) + (SAP) (si U est une valeur) + (LAP) (si U n’est pas une valeur)

La récursion

La règle traditionnelle :

$$\mu M[s]^a \to M[\mu M[s]^a \cdot s]^b \qquad b \text{ nouvelle} \qquad \text{(Unfold)}$$

Une règle avec *pointeurs en arrière* :

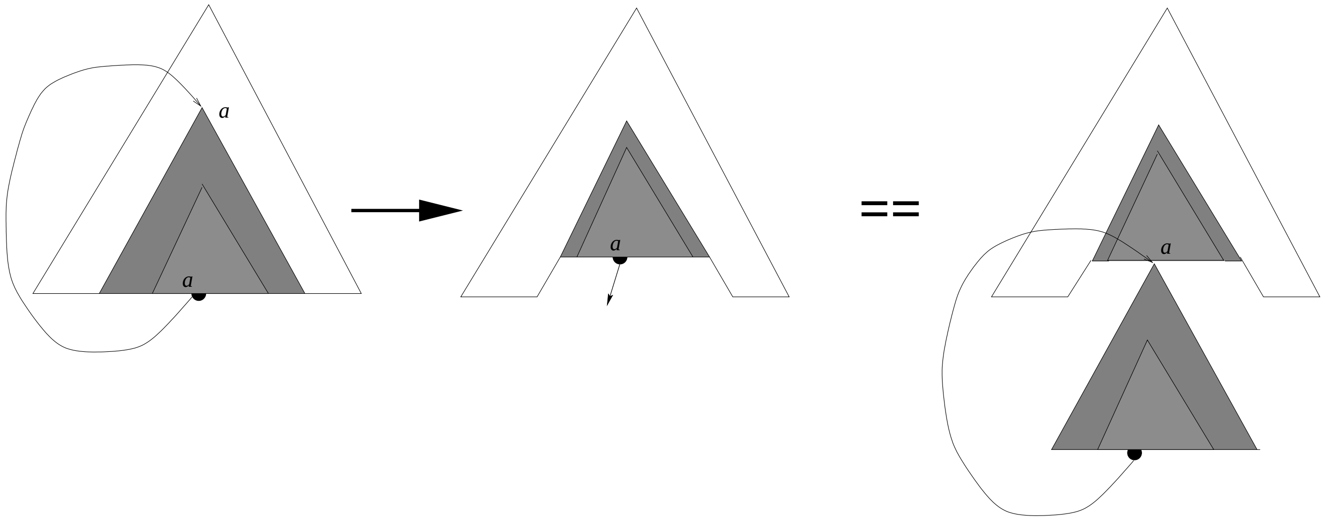
$$\mu M[s]^a \to M[\bullet^a \cdot s]^a \qquad \text{(Cycle)}$$

Un terme avec $\bullet$ est *admissible* si $\bullet^a$ n'apparaît que dans des termes E^a.

La récursion

Q: Qu'arrive-t-il à $\bullet^a$ quand on fait disparaître l'adresse a du terme E^a qui le contient ?

R: On peut créer un pointeur «pendant».



La récursion

Il faut «compenser» le «manque» en remplaçant les $\bullet^a$ par des E^a. Il y a trois règles concernées.

$$((\lambda M)[s]^b\, U)^a \to M[U \cdot (s\{\bullet^b := \lambda M[s]^b\})]^a \qquad (\mathrm{B}_w^\bullet)$$

$$\underline{0}[E^b \cdot s]^a \to (E\{\bullet^a := \underline{0}[E^b \cdot s]^a\})^b \qquad (\mathrm{FVarG}^\bullet)$$

$$\underline{0}[E^b \cdot s]^a \to (E\{\bullet^b := E^b\})^a \qquad (\mathrm{FVarE}^\bullet)$$

Les langages à objets

La sémantique opérationnelle des objets

Quelques objectifs

- Être générique
- Traiter dans un même cadre
  - les objets mutables
  - et les objets mutables,
- Classifier
  - langages à “embedding” (à “plongement”)
  - et langages à délégation
- Pouvoir faire des preuves rigoureuses de propriétés,
- Séparer règles et stratégies,

La Syntaxe

Code

$$
\begin{array}{rlr}
M, N ::= & x \mid \lambda x.M \mid M N \mid c & \text{(Lambda Terms)} \\
\mid & M \Leftarrow m & \text{(Message Sending)} \\
\mid & \langle\rangle & \text{(Object Initialization)} \\
\mid & \langle M \leftarrow m = N \rangle \mid \langle M \leftarrow\!:\ m = N \rangle & \\
 & & \text{(Object Modifications)} \\
\mid & \mathsf{shallow}(x) \mid \mathsf{refresh}(x) & \\
 & & \text{(Duplication Primitives)}
\end{array}
$$

Evaluation Contexts

$$
\begin{array}{rll}
U, V ::= & M[s]^a & \text{(Closure)} \\
\mid & (UV)^a & \text{(Application)} \\
\mid & (U \Leftarrow m)^a & \text{(Message Sending)} \\
\mid & \langle U \leftarrow m = V \rangle^a \mid \langle U \leftarrow: m = V \rangle^a & \\
 & & \text{(Object Modifications)} \\
\mid & \lceil O \rceil^a \mid \bullet^a & \text{(Object)} \\
\mid & (O \hookleftarrow m)^a & \text{(Lookup)}
\end{array}
$$

Objects-Structures

$$
\begin{array}{rll}
O ::= & \langle\rangle^a \mid \langle O \leftarrow m = V \rangle^a & \text{(Internal Objects)} \\
\mid & \mathsf{copy}(O)^a \mid \bullet^a & \text{(Duplicator)}
\end{array}
$$

Environments

$$
s ::= U/x; s \mid \mathsf{id} \qquad \text{(Bindings)}
$$

Les modules

$\lambda\mathcal{O}bj^{+a}$ a 4 modules :

L : est $\lambda\mathrm{x}_w^a$ le calcul faible avec *nom*, *substituions explicites*, et *adresses*.

C : contient les opérations communes sur les objets :

- *instanciation* d'un objet,
- *envoi* d'un message,
- *recherche* d'une méthode.

F : décrit les *altérations non mutables* d'un objet.

I : décrit

- les *modifications en place* d'un objet,
- les opérations de *clonage*.

Le module L

$$\big((\lambda x.M)[s]^b\, U\big)^a \to M[U/x; s]^a \qquad \text{(B)}$$

$$(MN)[s]^a \to (M[s]^b\, N[s]^c)^a \qquad b, c \text{ nouvelle} \qquad \text{(App)}$$

$$x[\ldots U/x \ldots]^a \to U \qquad \text{(VarG)}$$

$$x[\ldots E^b/x \ldots]^a \to E^a \quad \text{where } E \equiv (\lambda y.M)[s] \text{ or } c[s] \qquad \text{(VarE)}$$

Le module commun C

Instantiation

$$\langle\rangle[s]^a \to \lceil\langle\rangle^b\rceil^a \qquad b\text{nouvelle} \qquad \text{(NO)}$$

Message Sending

$$(M \Leftarrow m)[s]^a \to (M[s]^b \Leftarrow m)^a \qquad b \text{ nouvelle} \qquad \text{(SP)}$$

$$\left(\lceil O\rceil^b \Leftarrow m\right)^a \to \left((O \hookleftarrow m)^c \lceil O\rceil^b\right)^a \qquad c \text{ nouvelle} \qquad \text{(SA)}$$

$$(\langle O \leftarrow m = V\rangle^b \hookleftarrow m)^a \to V \qquad \text{(SG)}$$

$$(\langle O \leftarrow m = E^c\rangle^b \hookleftarrow m)^a \to E^a \qquad \text{(SE)}$$

$$(\langle O \leftarrow n = V\rangle^b \hookleftarrow m)^a \to (O \hookleftarrow m)^a \qquad \text{(NL)}$$

Le module «non mtutable» F

Functional Update

$$\langle M \leftarrow m = N \rangle [s]^a \rightarrow \langle M[s]^b \leftarrow m = N[s]^c \rangle^a \quad b, c\text{nouvelle} \tag{FP}$$

$$\langle \lceil O \rceil^b \leftarrow m = V \rangle^a \rightarrow \lceil \langle O \leftarrow m = V \rangle^c \rceil^a \quad c\text{nouvelle} \tag{FC}$$

Le module «mutable» I

Imperative Update

$$\langle M \leftarrow\!\!: \; m = N\rangle[s]^a \to \langle M[s]^b \leftarrow\!\!: \; m = N[s]^c\rangle^a \qquad b, c \text{ nouvelle} \qquad \text{(IP)}$$

$$\langle\lceil O\rceil^b \leftarrow\!\!: \; m = V\rangle^a \to \lceil\langle O \leftarrow m = V\rangle^c\rceil^b \qquad c \text{ nouvelle} \qquad \text{(IC)}$$

Le module «mutable» I *(suite)*

Cloning Primitives

$$\mathsf{shallow}(x)[\ldots\lceil O\rceil^{b}/x\ldots]^{a} \to \lceil O\rceil^{a} \tag{SC}$$

$$\mathsf{refresh}(x)[\ldots\lceil O\rceil^{b}/x\ldots]^{a} \to \lceil \mathsf{copy}(O)^{c}\rceil^{b} \qquad c \text{ nouvelle} \tag{RE}$$

$$\mathsf{copy}(\langle\rangle^{b})^{a} \to \langle\rangle^{a} \tag{CE}$$

$$\mathsf{copy}(\langle O \leftarrow m = V\rangle^{b})^{a} \to \langle \mathsf{copy}(O)^{c} \leftarrow m = V\rangle^{a} \qquad c \text{ nouvelle} \tag{CO}$$

Travaux futurs

- *Développer* $\lambda\mathcal{O}bj^{+a}$,
- Peaufiner le *système de types*, notamment le type des constructions mutables pour lesquelles il faut démontrer la réduction du sujet
- Formaliser la *réécriture simultanée*.