Modéliser les modèles

La sémantique dénotationnelle des languages de programmation

Damiano Mazza
Laboratoire d'Informatique de Paris Nord
CNRS–Université Paris 13

Lancement du LabEx *Sciences Mathématiques de Paris*
Institut Océanographique, Paris
27 septembre 2011

Un exemple de problème

Est-ce qu'un texte donné est palindrome ?

- radar
- Karine alla en Irak
- Trace l'inégal palindrome. Neige. Bagatelle, dira Hercule. Le brut repentir, cet écrit né Perec. L'arc lu pèse trop, lis à vice-versa. Perte. Cerise d'une vérité banale, le Malstrom, Alep, mort édulcoré, crêpe porté de ce désir brisé d'un iota. **[. . .encore 1155 mots. . .]** à toi, nu désir brisé, décédé, trope percé, roc lu. Détrompe la. Morts : l'Ame, l'Élan abêti, revenu. Désire ce trépas rêvé : Ci va ! S'il porte, sépulcral, ce repentir, cet écrit ne perturbe le lucre : Haridelle, ta gabegie ne mord ni la plage ni l'écart.

Le *Entscheidungsproblem*

Est-ce qu'un enoncé mathématique donné est vrai ?

- $\exists x.(A(x) \Rightarrow \forall y.A(y))$
- $\forall n, a, b, c.(n \geq 3 \Rightarrow (a^n + b^n \neq c^n))$
- $\forall n.\exists p.(p \geq n \wedge P(p) \wedge P(p+2))$

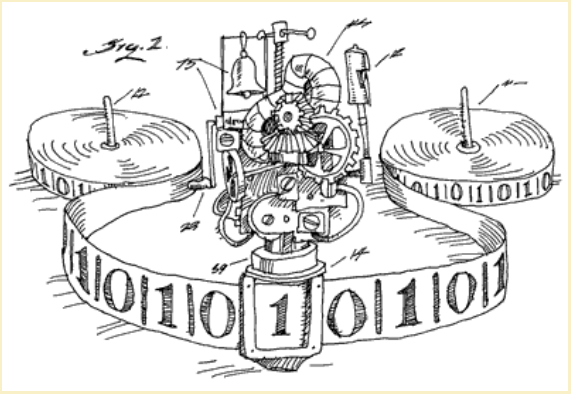
Wir müssen wissen,

wir wierden wissen !

La notion d'algorithme

- Formaliser le concept de procedure effective qui résout un problème : un *algorithme*.
- Cela requiert un **modèle de calcul**.

fonctions recursives



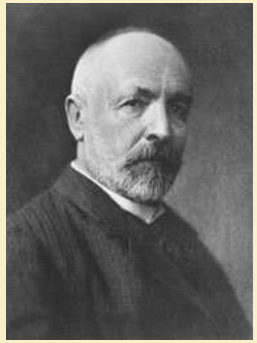
machines de Turing

$$X = \lambda xf.f(xxf)$$
$$\Theta = XX$$
$$\Theta F \rightarrow^* F(\Theta F)$$

λ-calcul

Le λ-calcul : un modèle de langage de programmation

- Basé sur la notion de *fonction*.
- Toutes les fonctions ne sont pas calculables, pour des raisons de *cardinalité*.
- **Thèse de Church :** celles qui le sont, sont calculées par un λ-*terme*.



$\#(\mathbb{N}) < \#(\mathbb{N}^{\mathbb{N}})$

- Les variables $x, y, \ldots$ sont des termes ;
- si t est est un terme, $\lambda x.t$ en est un ;
- si t, u sont des termes, tu en est un.

La règle de calcul :
$(\lambda x.t)u \to t[u/x]$

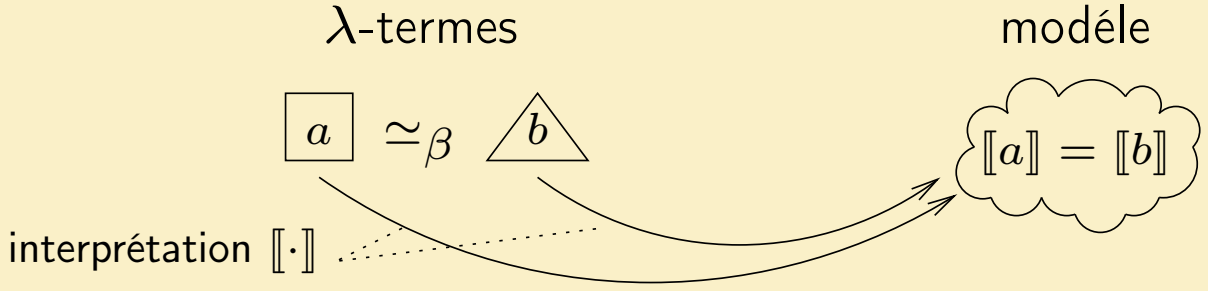
Des modèles du λ-calcul ?

Définition :

$$t \simeq_\beta u \qquad \text{ssi} \qquad \exists v.\ t \to^* v \ {}^*\!\leftarrow u.$$

– On peut démontrer qu'il s'agit d'une relation d'équivalence.

– Est-ce que l'on peut transformer l'équivalence en *égalité* ?



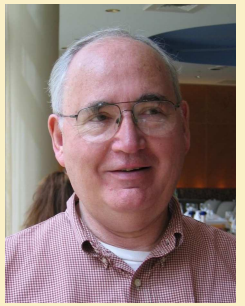
– Il s'agit de trouver un **invariant** du calcul.

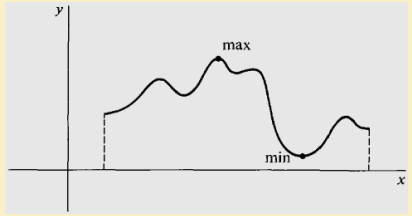
Les modèles de Scott

- L'idée : trouver D tel que D^D " $\subseteq$ " D.
- C'est impossible ! ! ! (Toujours pour la cardinalité).
- Solution : ne considérer que les fonctions **continues** !

$$\#(C^0(\mathbb{R})) = \#(\mathbb{R})$$

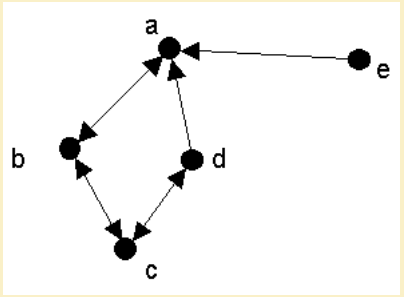
- Exemple : $D = \wp(\mathbb{N})$ avec une topologie telle que $f \in D^D$ continue ssi $f(x) = \bigcup_{a \subseteq^{\text{fin}} x} f(a)$.
- Maintenant, $f \in D^D$ continue peut être representée par $\{\langle \lceil a \rceil, n \rangle \mid n \in f(a),\ a \subseteq^{\text{fin}} \mathbb{N}\} \in D$, où $\langle \cdot, \cdot \rangle$ et $\lceil \cdot \rceil$ sont des bijections de $\mathbb{N}^2$ et $\wp_{\text{fin}}(\mathbb{N})$ dans $\mathbb{N}$.

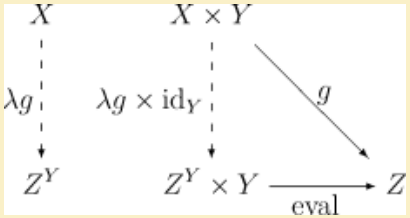




Types, catégories, jeux

- Le λ-calcul peut être *typé* :
  - chaque variable a un type (entiers, listes, etc.) ;
  - si $x : A$ et $t : B$, $\lambda x.t : A \to B$;
  - si $t : A \to B$ et $u : A$, $tu : B$.
- Formulation en **théorie des catégories** :
  - modèle typé = *catégorie Cartésienne close* (CCC).
  - modèle non-typé = *objet réflexif* dans une CCC.
- Tout langage de programmation fonctionnel est une forme de λ-calcul typé.
- Récemment, les modèles basés sur des catégories de **jeux** ont eu un succès particulier.

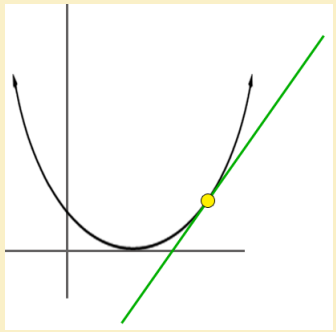




Linéarité, ressources, analyse

- Les modèles dénotationnels ont été fondamentaux pour la découverte de la **linéarité** en informatique.
- Un λ-terme est *linéaire* s'il n'utilise ses arguments qu'**une seule fois**.
- On peut introduire un λ-*calcul avec ressources*, dont le termes correspondent grossomodo à des monômes.
- Les λ-termes habituels peuvent alors être développés en série de Taylor :

$$\lambda x.xx = \sum_{n=0}^{\infty} \frac{1}{n!} \lambda x.x[x^n]$$



Merci ! ☺