Jean Goubault-Larrecq

λ-calcul

5. Modèles

Modèles

- Un **modèle** du λ-calcul est une fonction qui à tout λ-terme u associe une **valeur** $[\![u]\!]$, dans un certain **domaine** E

  … et telle que si $u =_\beta v$ alors $[\![u]\!]=[\![v]\!]$

- Quelques modèles évidents
- La difficulté avec les modèles
- Le modèle $\mathbf{P}\omega$ de Gordon Plotkin et Dana Scott
- Le modèle D_∞ de Dana Scott (pas ici: voir poly)

Quelques modèles évidents

Le modèle trivial

Un **modèle** du λ-calcul est une fonction qui à tout λ-terme u associe une **valeur** $[\![u]\!]$, dans un certain **domaine** D

… et telle que si $u =_\beta v$ alors $[\![u]\!]=[\![v]\!]$

- On pose $E \stackrel{\text{def}}{=} \{\bullet\}$ (un singleton)

  $[\![u]\!] \stackrel{\text{def}}{=} \bullet$ pour tout terme u

- Utilité: quasi-nulle.

Les modèles syntaxiques

Un **modèle** du λ-calcul est une fonction qui à tout λ-terme u associe une **valeur** $[\![u]\!]$, dans un certain **domaine** D

… et telle que si $u =_\beta v$ alors $[\![u]\!]=[\![v]\!]$

- On pose $E \stackrel{\text{def}}{=}$ {classes d'équivalence de λ-termes mod $=_\beta$}

  $[\![u]\!] \stackrel{\text{def}}{=}$ classe d'équivalence de u

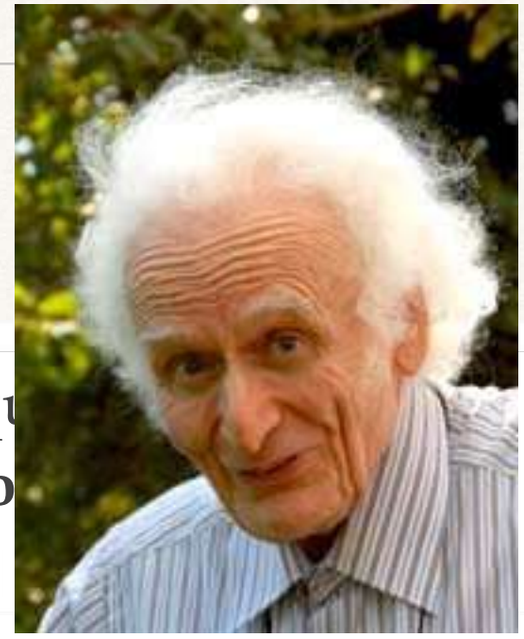
- Modèle non-trivial: par exemple $[\![\ulcorner \text{vrai} \urcorner]\!] \neq [\![\ulcorner \text{faux} \urcorner]\!]$

  … par confluence

- Raisonner dessus n'apporte rien de plus que de raisonner sur la syntaxe, comme on l'a fait jusqu'ici!

- **Note:** au lieu de $=_\beta$, on obtient un autre modèle non trivial en considérant $=_{\beta\eta}$

Les arbres de Böhm

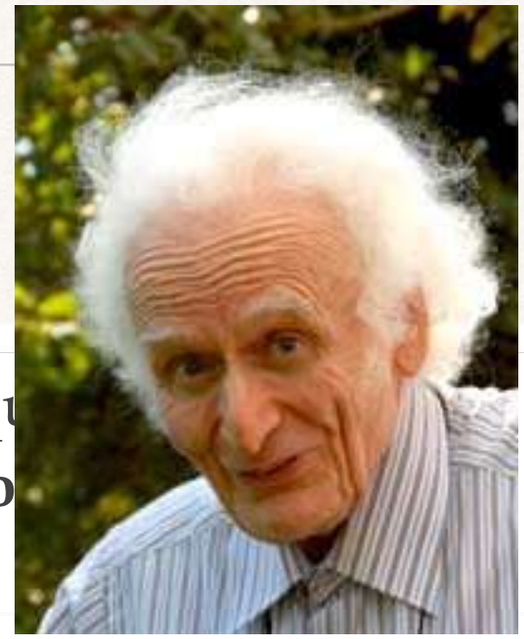


Un **modèle** du λ-calcul est une fonction qu
associe une **valeur** $[\![u]\!]$, dans un certain **do**
… et telle que si $u =_\beta v$ alors $[\![u]\!]=[\![v]\!]$

https://www.computerhope.com/people/pictures/corrado_bohm.jpg

- Un **arbre Böhm-like** est un arbre possiblement infini dont les sommets sont:
  — soit la **feuille** Ω [non, pas le terme $\Omega=\delta\delta$]
  — soit des sommets étiquetés par des
  « formes de tête partielles » $\lambda x_1. \ldots . \lambda x_n.x$

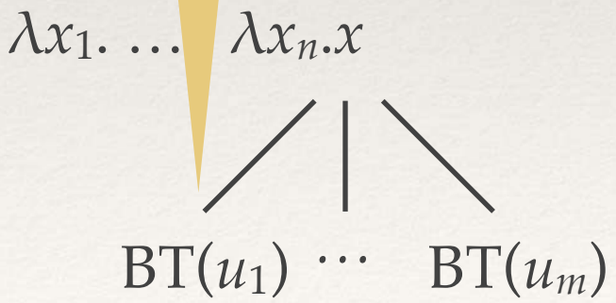
Les arbres de Böhm



Un **modèle** du λ-calcul est une fonction qu
associe une **valeur** $[\![u]\!]$, dans un certain **do**
… et telle que si $u =_\beta v$ alors $[\![u]\!]=[\![v]\!]$

https://www.computerhope.com/people/pictures/corrado_bohm.jpg

- L'**arbre de Böhm** BT(u) d'
  — Ω si u n'a pas de forme
    [=la réduction de tête part
  — sinon, $u \to_t^* \lambda x_1. \ldots . \lambda x_n.x\ u_1\ .\ \ u_m$ normale de tête
    [x variable, parmi les x_i, ou non]
  et alors BT(u) est l'arbre:

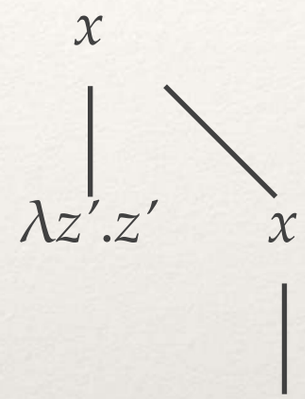


le processus ne s'arrête pas forcément
(formellement, je devrais parler de coinduction)

modulo α-renommage

Exemples d'arbres de Böhm (1/3)

- BT($(\lambda y\,.\,(\lambda x\,.\,yxx)(\lambda z\,.\,z(xy)))(\lambda z'\,.\,z')$) $=$
  car $x(\lambda z'\,.\,z')(x(\lambda z'\,.\,z'))$ est la forme normale



- En général, si u a une forme normale v,
  alors BT(u) est l'écriture sous forme de forme héréditairement normale de tête de v

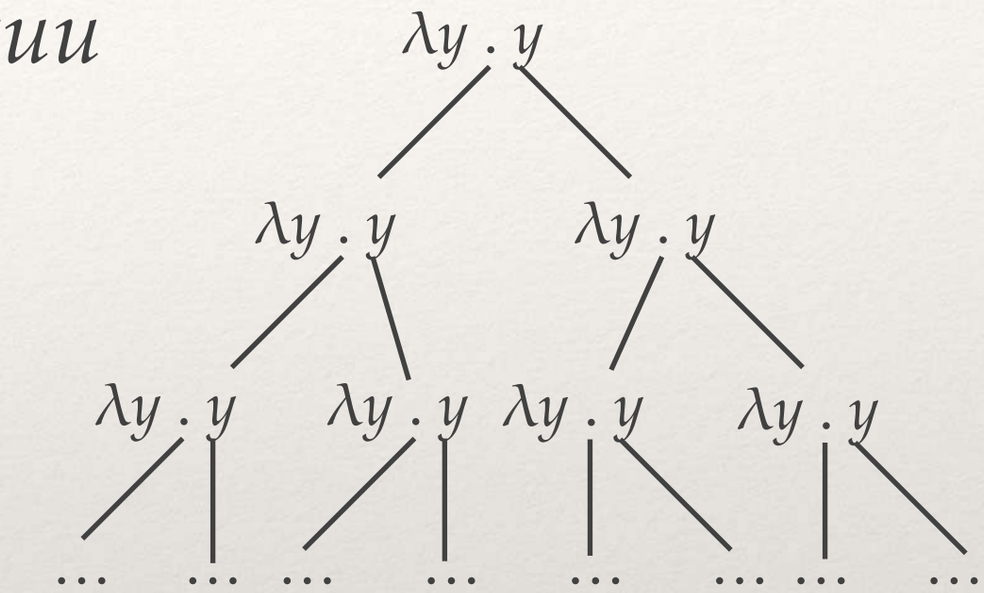
- … c'est le théorème de standardisation!
  comme $u \to^* v$, on a $u \Rightarrow_s v$, et BT(u) se construit par récurrence sur $|v|$

Exemples d'arbres de Böhm (2/3)

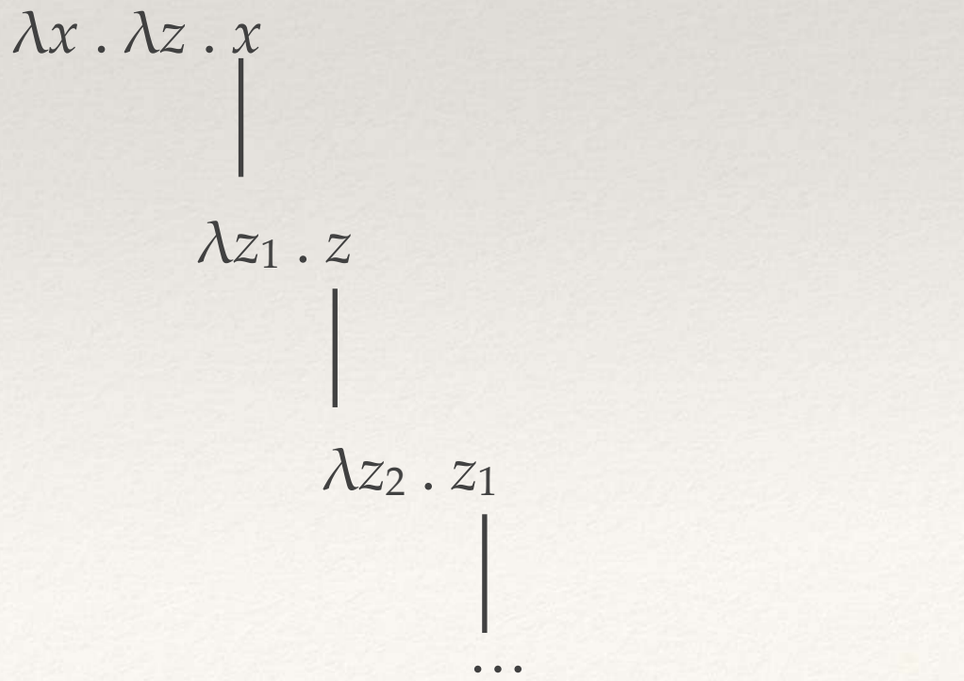
- $BT(\delta\delta) = \Omega$
  car $\delta\delta$ est **irrésoluble** (=pas de forme normale de tête)

Exemples d'arbres de Böhm (3/3)

- Soit $u \overset{\text{def}}{=} \Theta(\lambda x\,.\,\lambda y\,.\,yxx)$: $u \to_t^* \lambda y\,.\,yuu$



- Soit $J \overset{\text{def}}{=} \Theta(\lambda j\,.\,\lambda x\,.\,\lambda z\,.\,x(jz))$:
  $J \to_t^* \lambda x\,.\,\lambda z\,.\,x(Jz)$
  $Jz \to_t^* \lambda z_1\,.\,z(Jz_1)$
  $Jz_1 \to_t^* \lambda z_2\,.\,z_1(Jz_2)$, etc.



Le modèle des arbres de Böhm

- **Prop.** Les arbres de Böhm forment un modèle:
  si $u =_\beta v$ alors $\mathrm{BT}(u)=\mathrm{BT}(v)$.

- *Preuve* (squelette, 1/4).
  Soit $\mathrm{BT}_k(u)$ l'arbre de Böhm de u **tronqué** à la profondeur k.
  [on élimine tous les sommets à profondeur k ou plus]

- On montre par récurrence sur k que
  si $u =_\beta v$ alors $\mathrm{BT}_k(u)=\mathrm{BT}_k(v)$.

- Pour k=0, trivial (« arbre vide » de chaque côté).
  Supposons donc $k\geq 1$.

Le modèle des arbres de Böhm

- **Prop.** Les arbres de Böhm forment un modèle:
  si $u =_\beta v$ alors BT(u)=BT(v).

- *Preuve* (squelette, 2/4).
  Si $u =_\beta v$ et u **irrésoluble** mais pas v
  alors $v \to_t^* \lambda x_1. \ldots . \lambda x_n.x\ v_1 \ldots v_m$ [normale de tête]
  donc $u =_\beta \lambda x_1. \ldots . \lambda x_n.x\ v_1 \ldots v_m$

- Par **confluence**,
  u et $\lambda x_1. \ldots . \lambda x_n.x\ v_1 \ldots v_m$ ont un réduit commun,
  nécessairement de la forme $\lambda x_1. \ldots . \lambda x_n.x\ w_1 \ldots w_m$

- Par **standardisation**, $u \Rightarrow_s \lambda x_1. \ldots . \lambda x_n.x\ w_1 \ldots w_m$

Le modèle des arbres de Böhm

- **Prop.** Les arbres de Böhm forment un modèle:
  si $u =_\beta v$ alors $\mathrm{BT}(u)=\mathrm{BT}(v)$.

- *Preuve* (squelette, 3/4).

  Si $u =_\beta v$ et u **irrésoluble** mais pas v, [rappel]

  $$u \Rightarrow_s \lambda x_1. \ldots . \lambda x_n.x\ w_1 \ldots w_m$$

- Par **définition** de $\Rightarrow_s$, $u \to_t^* \lambda\, x_1 . \ldots . \lambda\, x_n . x\ u_1 \ldots u_m$
  donc u est **résoluble**,
  contradiction
  $$\lambda\, x_1 . \ldots . \lambda\, x_n . x\ w_1 \ldots w_m$$

- Donc soit u et v sont irrésolubles et $\mathrm{BT}_k(u)=\mathrm{BT}_k(v)=\Omega$, soit ils ont tous les deux une forme normale de tête, et…

Le modèle des arbres de Böhm

- **Prop.** Les arbres de Böhm forment un modèle:
  si $u =_\beta v$ alors $\mathrm{BT}(u)=\mathrm{BT}(v)$.

- *Preuve* (squelette, 4/4).
  Si $u \to_t^* \lambda x_1 . \ldots . \lambda x_n . x u_1 \ldots u_m$ et
  $v \to_t^* \lambda x_1 . \ldots . \lambda x_{n'} . x' v_1 \ldots v_{m'}$,

- par **confluence**, ces f. normales de tête ont un réduit commun

- mais les réductions ont lieu à l'intérieur des u_i (resp, v_j), donc
  $n=n'$, $m=m'$, et $u_1 =_\beta v_1$, …, $u_m =_\beta v_m$.

- Par h.r., $\mathrm{BT}_{k-1}(u_i)=\mathrm{BT}_{k-1}(v_i)$ pour tout i, donc $\mathrm{BT}_k(u)=\mathrm{BT}_k(v)$. □

Comparer les modèles

- Tout modèle induit une **théo**
  les équations $u \approx v$ tel

Je m'arrête là sur ce sujet, il y aurait beaucoup à dire. (Notamment, la théorie équationnelle de BT(.) est **la même** que celle du modèle **P**ω à venir!)

- La théorie équationnelle du modèle syntaxique est celle de $=_\beta$
- Quelle est celle de BT(.)?
- Pas la même! Par exemple, BT(δδ) = BT (Y) (=Ω)
  mais δδ $\neq_\beta$ Y (utiliser la confluence)
- BT(.) est un **modèle sensé** (« sensible » en anglais), i.e., qui identifie tous les irrésolubles, contrairement au modèle syntaxique

Vers des modèles sémantiques

Une tentative de modèle sémantique

- On aimerait définir une sémantique **à la Tarski**:

  $[\![x]\!]\varrho \stackrel{\text{def}}{=} \varrho(x)$

  $[\![uv]\!]\varrho \stackrel{\text{def}}{=} [\![u]\!]\varrho\ ([\![v]\!]\varrho)$

  $[\![\lambda x\ .\ u]\!]\varrho \stackrel{\text{def}}{=} (V \in D \mapsto [\![u]\!](\varrho[x{:=}V]))$

- où ϱ est un **environnement** (fonction des variables vers l'ensemble D des valeurs) et $\varrho[x{:=}V]$ envoie x vers V et toute $y{\neq}x$ vers $\varrho(y)$

- (Se relie à la notion précédente de modèle via: $[\![u]\!]$ est la fonction qui à ϱ associe $[\![u]\!]\varrho$ — donc $E \stackrel{\text{def}}{=} (\text{Var} \to D) \to D$)

Une tentative de modèle sémantique

- On aimerait définir une sémantique **à la Tarski**:

  $[\![x]\!]\varrho \stackrel{\text{def}}{=} \varrho(x)$

  $[\![uv]\!]\varrho \stackrel{\text{def}}{=} [\![u]\!]\varrho\,([\![v]\!]\varrho)$

  $[\![\lambda x\,.\,u]\!]\varrho \stackrel{\text{def}}{=} (V \in D \mapsto [\![u]\!](\varrho[x{:=}V]))$

- … à valeurs dans D

Mais alors, ça n'est pas bien typé: il faudrait **convertir** $[\![u]\!]\varrho \in D$ en une fonction de D vers D

Une tentative de modèle sémantique

fonction de conversion

- On aimerait définir une sémantique **à la Tarski**:

  $[\![x]\!]\varrho \stackrel{\text{def}}{=} \varrho(x)$

  $[\![uv]\!]\varrho \stackrel{\text{def}}{=} r\ ([\![u]\!]\varrho)\ ([\![v]\!]\varrho)$

  $[\![\lambda x\ .\ u]\!]\varrho \stackrel{\text{def}}{=} (V \in D \mapsto [\![u]\!](\varrho[x{:=}V]))$

$r : D \to (D \to D)$

- … à valeurs dans D

ça non plus n'est pas bien typé: c'est une fonction de D dans D, et nous devons la **convertir** en un élément de D

Une tentative de modèle sémantique

- On aimerait définir une sémantique **à la Tarski**:

$$[\![x]\!]\varrho \stackrel{\text{def}}{=} \varrho(x)$$

$$[\![uv]\!]\varrho \stackrel{\text{def}}{=} r\ ([\![u]\!]\varrho)\ ([\![v]\!]\varrho)$$

$$[\![\lambda x\ .\ u]\!]\varrho \stackrel{\text{def}}{=} i\ (V \in D \mapsto [\![u]\!](\varrho[x{:=}V]))$$

fonction de conversion

$$r : D \to (D \to D)$$

$$i : (D \to D) \to D$$

- … à valeurs dans D

- Est-ce un modèle?

Un **modèle** du λ-calcul est une fonction qui à tout λ-terme u associe une **valeur** $[\![u]\!]$, dans un certain **domaine** D

… et telle que si $u =_\beta v$ alors $[\![u]\!]=[\![v]\!]$

Une tentative de modèle sémantique

- On aimerait définir une sémantique **à la Tarski**:

$$[\![x]\!]\varrho \stackrel{\text{def}}{=} \varrho(x)$$

$$[\![uv]\!]\varrho \stackrel{\text{def}}{=} r\ ([\![u]\!]\varrho)\ ([\![v]\!]\varrho)$$

$$[\![\lambda x\ .\ u]\!]\varrho \stackrel{\text{def}}{=} i\ (V \in D \mapsto [\![u]\!](\varrho[x{:=}V]))$$

$$r : D \to (D \to D)$$

$$i : (D \to D) \to D$$

- On a: $[\![(\lambda x\ .\ u)v]\!]\varrho = r\ ([\![\lambda x\ .\ u]\!]\varrho)\ ([\![v]\!]\varrho)$

$$= r\ (i\ (V \in D \mapsto [\![u]\!](\varrho[x{:=}V]))\ ([\![v]\!]\varrho)$$

$$= ?$$

Une tentative de modèle sémantique

- On aimerait définir une sémantique **à la Tarski**:

  $[\![x]\!]\varrho \stackrel{\text{def}}{=} \varrho(x)$

  $[\![uv]\!]\varrho \stackrel{\text{def}}{=} r\,([\![u]\!]\varrho)\,([\![v]\!]\varrho)$

  $[\![\lambda x\,.\,u]\!]\varrho \stackrel{\text{def}}{=} i\,(V \in D \mapsto [\![u]\!](\varrho[x{:=}V]))$

$r : D \to (D \to D)$

$i : (D \to D) \to D$

$r \circ i = \mathrm{id}_{D \to D}$

- On a:
$$\begin{aligned}[\![(\lambda x\,.\,u)v]\!]\varrho &= r\,([\![\lambda x\,.\,u]\!]\varrho)\,([\![v]\!]\varrho)\\ &= r\,(i\,(V \in D \mapsto [\![u]\!](\varrho[x{:=}V]))\,([\![v]\!]\varrho)\\ &= (V \in D \mapsto [\![u]\!](\varrho[x{:=}V]))\,([\![v]\!]\varrho)\\ &= [\![u]\!](\varrho[x{:=}[\![v]\!]\varrho])\end{aligned}$$

- et ceci est égal à $[\![u[x{:=}v]]\!]\varrho$, par une récurrence facile sur u

Une tentative de modèle sémantique

- On aimerait définir une sémantique **à la Tarski**:

  $[\![x]\!]\varrho \stackrel{\text{def}}{=} \varrho(x)$

  $[\![uv]\!]\varrho \stackrel{\text{def}}{=} r\ ([\![u]\!]\varrho)\ ([\![v]\!]\varrho)$

  $[\![\lambda x\ .\ u]\!]\varrho \stackrel{\text{def}}{=} i\ (V \in D \mapsto [\![u]\!](\varrho[x{:=}V]))$

$r : D \to (D \to D)$

$i : (D \to D) \to D$

$r \circ i = \mathrm{id}_{D \to D}$

- On a: $[\![(\lambda x\ .\ u)v]\!]\varrho = [\![u[x{:=}v]]\!]\varrho$ (rappel)

- On en déduit facilement que ceci définit bien un **modèle**

Un **modèle** du λ-calcul est une fonction qui à tout λ-terme u associe une **valeur** $[\![u]\!]$, dans un certain **domaine** D

… et telle que si $u =_\beta v$ alors $[\![u]\!]=[\![v]\!]$

Une tentative de modèle sémantique

- On aimerait définir une sémantique **à la Tarski**:

  $[\![x]\!]\varrho \stackrel{\text{def}}{=} \varrho(x)$

  $[\![uv]\!]\varrho \stackrel{\text{def}}{=} r\,([\![u]\!]\varrho)\,([\![v]\!]\varrho)$

  $[\![\lambda x\,.\,u]\!]\varrho \stackrel{\text{def}}{=} i\,(V \in D \mapsto [\![u]\!](\varrho[x{:=}V]))$

$r : D \to (D \to D)$

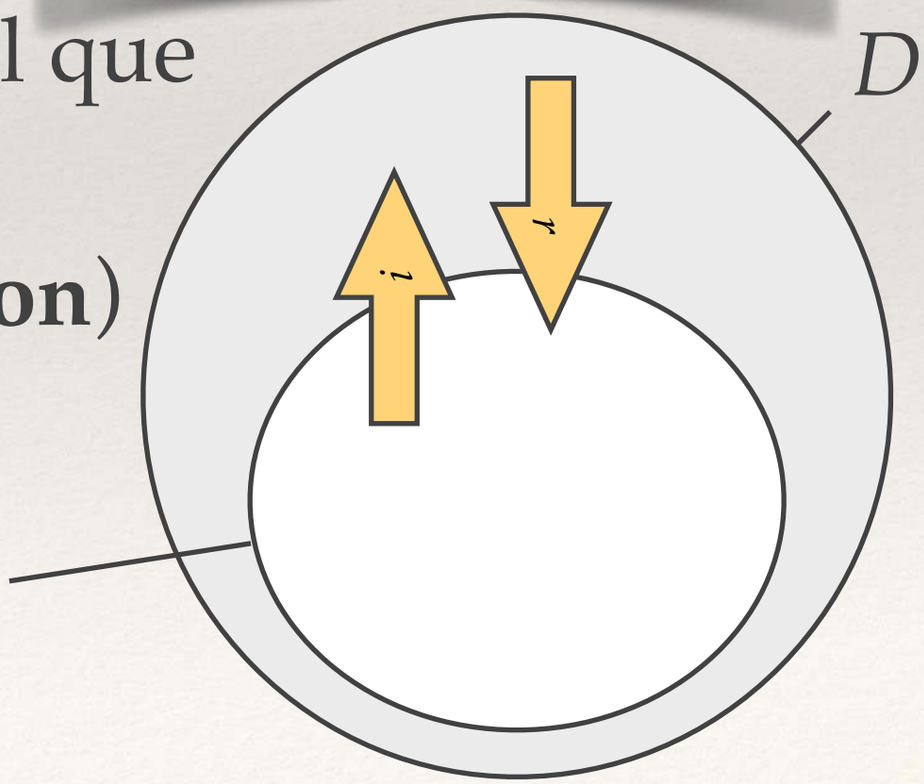
$i : (D \to D) \to D$

$r \circ i = \mathrm{id}_{D \to D}$

- On cherche donc un ensemble D tel que $D \to D$ soit un **rétract** de D

  (r est la **rétraction**, i est la **section**)

- … donc que D soit « plus gros » que $D \to D$ (?)



Parenthèse: modèles extensionnels

- $[\![x]\!]\varrho \stackrel{\text{def}}{=} \varrho(x)$

  $[\![uv]\!]\varrho \stackrel{\text{def}}{=} r\ ([\![u]\!]\varrho)\ ([\![v]\!]\varrho)$

  $[\![\lambda x\ .\ u]\!]\varrho \stackrel{\text{def}}{=} i\ (V \in D \mapsto [\![u]\!](\varrho[x{:=}V]))$

$r : D \to (D \to D)$

$i : (D \to D) \to D$

$r \circ i = \mathrm{id}_{D \to D}$

- Si en plus $i \circ r = \mathrm{id}_D$, r et i définissent un **isomorphisme** entre D et $D \to D$: $(D \to D \cong D)$

- Dans ce cas, si $u =_{\beta\eta} v$ alors $[\![u]\!]=[\![v]\!]$ (modèle **extensionnel**)

- En effet: $[\![\lambda x\ .\ ux]\!]\varrho = i\ (V \in D \mapsto r\ ([\![u]\!](\varrho[x{:=}V]))\ (V))$

  $= i\ (V \in D \mapsto r\ ([\![u]\!]\varrho)\ (V))$ si x pas libre dans u

  $= i\ (r\ ([\![u]\!]\varrho)) = [\![u]\!]\varrho$

Une tentative de modèle sémantique

- Mauvaise nouvelle:

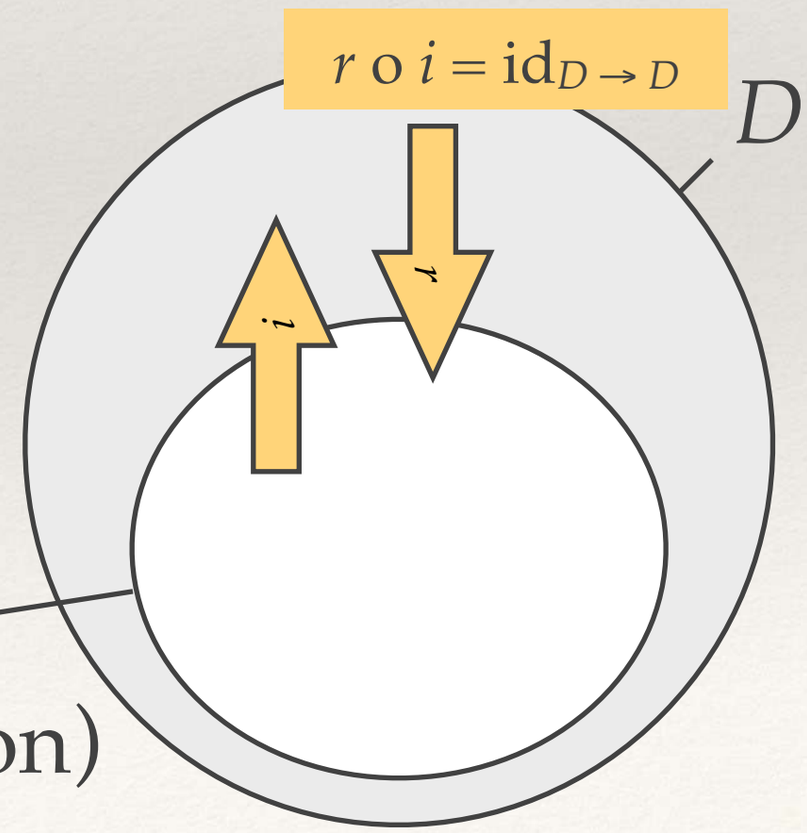
  Si $D \to D$ est un **rétract** de D alors D est un **singleton**.

- *Preuve* (1/2). Si $D=\varnothing$, alors soit $f : D \to D$ la fonction vide; donc $i(f)$ est dans D, contradiction.

  Donc D est non vide.

- Si D contient deux éléments $a \neq b$, alors on peut coder les parties de D comme leurs fonctions caractéristiques (à valeurs dans $\{a,b\}$)

- D'où une injection de $\mathbf{P}(D)$ dans D (non)



Une tentative de modèle sémantique

- Mauvaise nouvelle:

  Si $D \to D$ est un **rétract** de D alors D est un **singleton**.

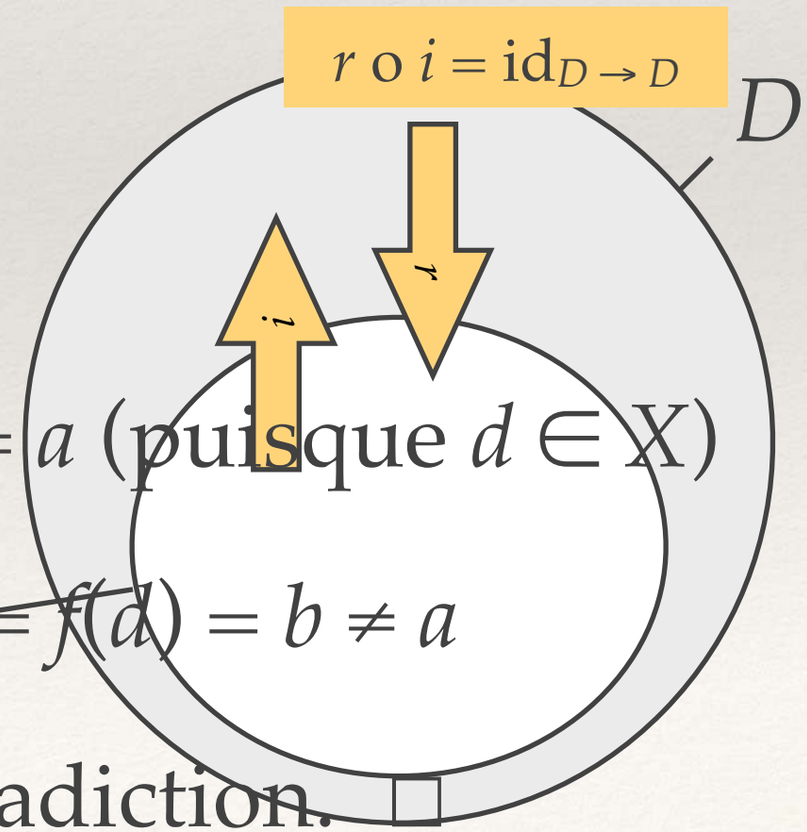
- *Preuve* (2/2). Si D contient $a \neq b$, **explicitement**, posons:

  — $X \stackrel{\text{def}}{=} \{x \in D \mid r(x)(x) \neq a\}$,

  — $f : D \to D : \quad x \in X \mapsto a, \quad x \in D\text{–}X \mapsto b$

  — $d \stackrel{\text{def}}{=} i(f)$

argument diagonal de Cantor



- Si $d \in X$, par déf. de X, $r(d)(d) \neq a$,

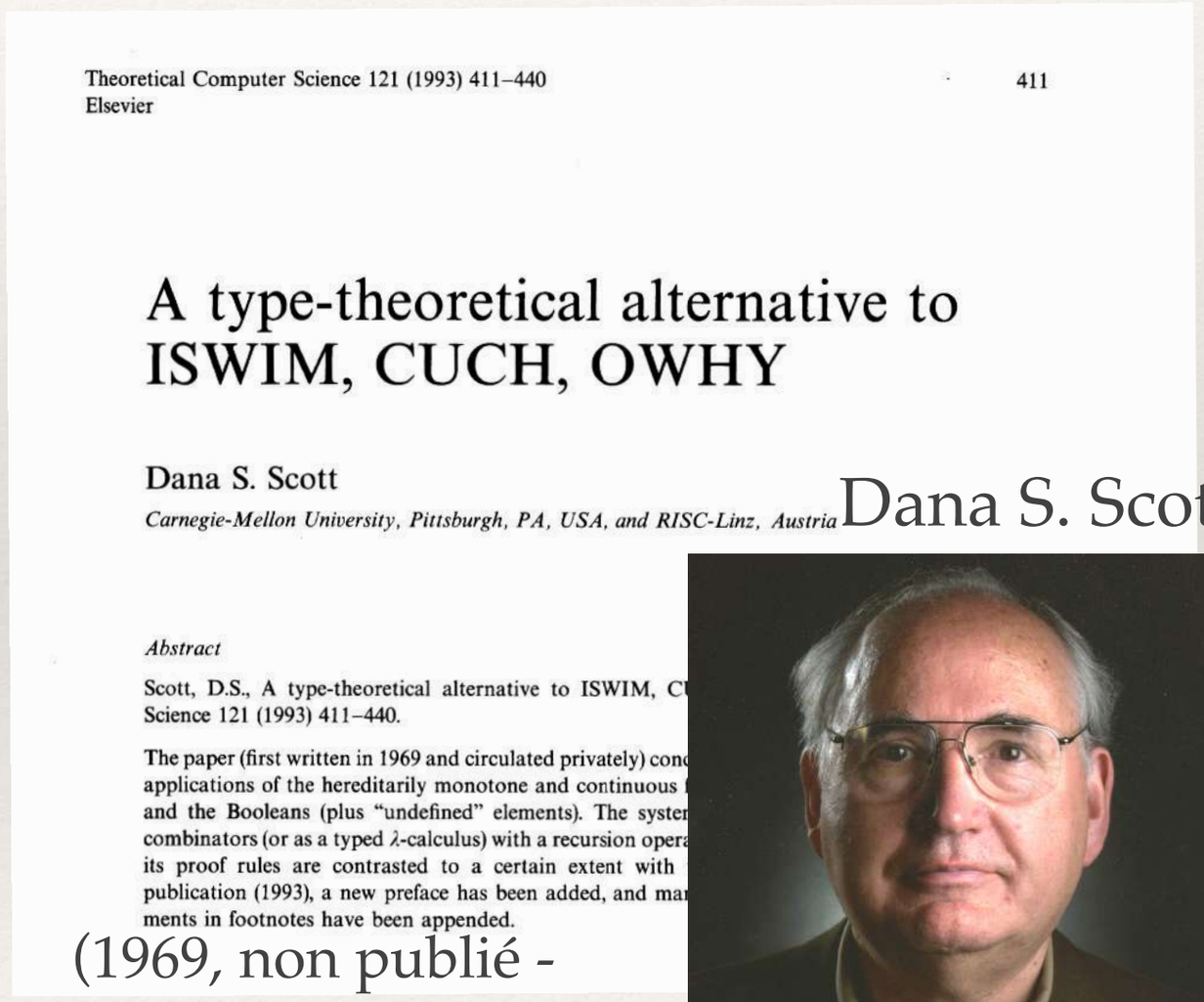
  mais $r(d)(d) = r(i(f))(d) = f(d) = a$ (puisque $d \in X$)

- Donc $d \in D\text{–}X$: d'où $r(d)(d) \neq r(i(f))(d) = f(d) = b \neq a$

- Par déf. de X, d est donc dans X: contradiction. □

Modèles sémantiques?

- Pendant longtemps, le λ-calcul était un calcul **sans sémantique**
- En 1969, Dana Scott, en voulant dénoncer l'utilisation du λ-calcul comme fondement sémantique des langages de programmation…
- en découvrit le premier modèle sémantique!



Theoretical Computer Science 121 (1993) 411–440
Elsevier

A type-theoretical alternative to
ISWIM, CUCH, OWHY

Dana S. Scott
Carnegie-Mellon University, Pittsburgh, PA, USA, and RISC-Linz, Austria

Abstract

Scott, D.S., A type-theoretical alternative to ISWIM, C
Science 121 (1993) 411–440.

The paper (first written in 1969 and circulated privately) conc
applications of the hereditarily monotone and continuous
and the Booleans (plus "undefined" elements). The syste
combinators (or as a typed λ-calculus) with a recursion opera
its proof rules are contrasted to a certain extent with
publication (1993), a new preface has been added, and ma
ments in footnotes have been appended.

(1969, non publié - 1ère publication 1993)

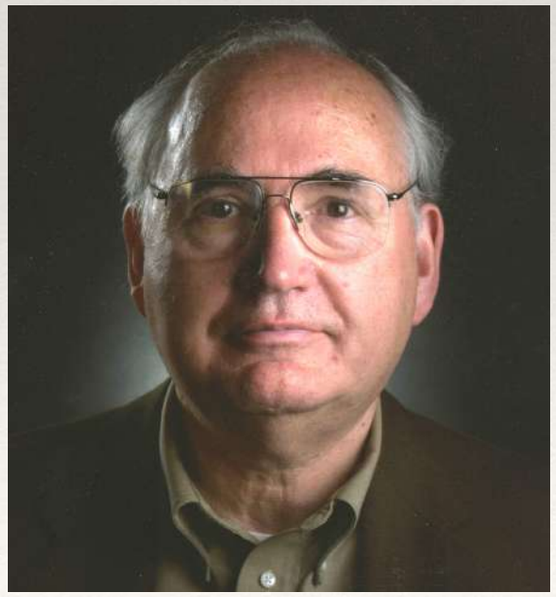
Dana S. Scott

https://www.scs.cmu.edu/sites/default/files/styles/news_item_image/public/DanaScott_b.jpg?itok=T9voOevm

Comment sortir du dilemme

- Un théorème de calculabilité dû à Rice et Shapiro montre que toutes les fonctions calculables sont **continues**, dans un sens précis dû à Scott
- Le domaine $[D \to D]$ des **fonctions (Scott-)continues** de D vers D est beaucoup plus petit que $D \to D$
- Notamment, il est dénombrable si D l'est, contrairement à $D \to D$

Dana S. Scott



https://www.scs.cmu.edu/sites/default/files/styles/news_item_image/public/DanaScott_b.jpg?itok=T9voOevm

Comment sortir du dilemme

- Et Scott a montré qu'on pouvait trouver un domaine D **non trivial** tel que $[D \to D] \cong D$

Theoretical Computer Science 121 (1993) 411–440
Elsevier

A type-theoretical alternative to ISWIM, CUCH, OWHY

Dana S. Scott

Carnegie-Mellon University, Pittsburgh, PA, USA, and RISC-Linz, Austria

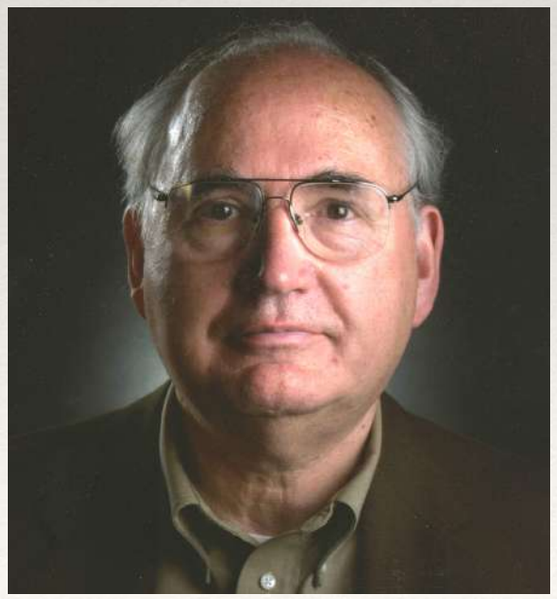
Abstract

Scott, D.S., A type-theoretical alternative to ISWIM, CUCH, OWHY, Theoretical Computer Science 121 (1993) 411–440.

The paper (first written in 1969 and circulated privately) concerns the definition, axiomatization, and applications of the hereditarily monotone and continuous functionals generated from the integers and the Booleans (plus "undefined" elements). The system is formulated as a typed system of combinators (or as a typed λ-calculus) with a recursion operator (the least fixed-point operator), and its proof rules are contrasted to a certain extent with those of the untyped λ-calculus. For publication (1993), a new preface has been added, and many bibliographical references and comments in footnotes have been appended.

(1969, non publié - 1ère publication 1993)

Dana S. Scott



https://www.scs.cmu.edu/sites/default/files/styles/news_item_image/public/DanaScott_b.jpg?itok=T9voOevm

Dcpos, fonctions Scott-continues

(un rappel pour les paris-saclaisiens)

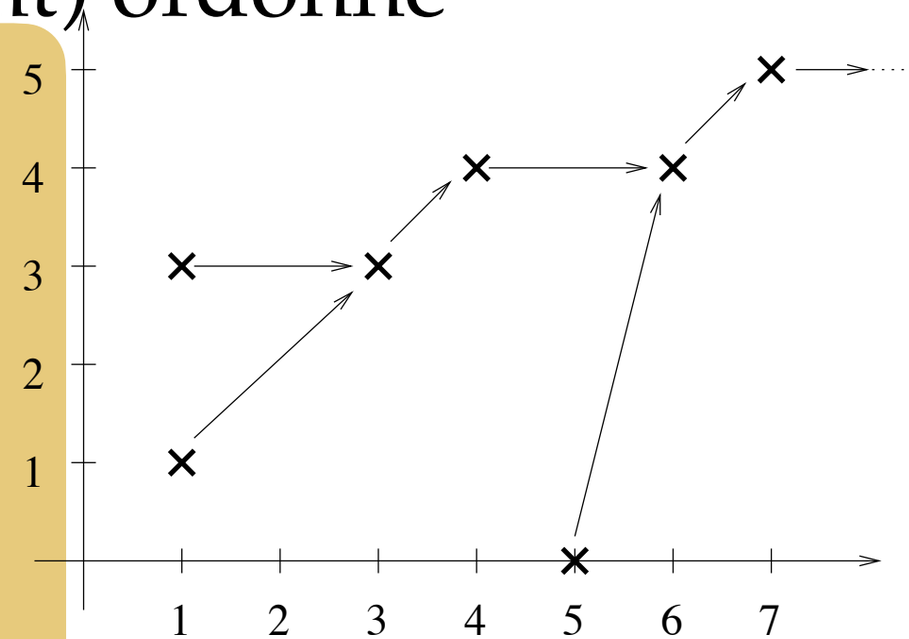
Dcpos

- Un **dcpo** (directed complete partial order) est un ensemble de « valeurs partielles »…
- partiellement ordonné:
  $x \leq y$ ssi « y est plus précis que x »
- avec une notion de « limite » de calculs de plus en plus précis
  (les **bornes supérieures** de familles dirigées)

Familles d

de façon équivalente, ssi
toute sous-famille **finie**
(possiblement vide)
dans D
a un majorant **dans D**

- Soit X ensemble (part ellement) ordonné
- Une famille D est **dirigée** ssi:
  — non vide
  — pour tous $x, y \in D$,
  il existe $z \in D$ / $x, y \leq z$.

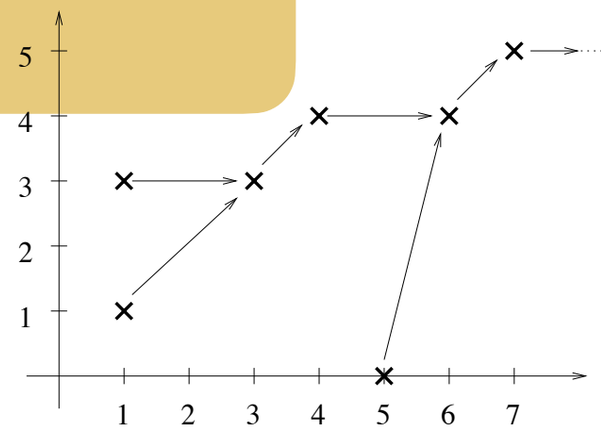


- Exemples: chaînes, $\mathbf{P}_{\text{fin}}(E)$ dans $X \stackrel{\text{def}}{=} (\mathbf{P}(A), \subseteq)$,
  pour toute $E \subseteq A$.

Dcpos

… pour rappeler qu'il s'agit d'un sup **dirigé**

- X est un **dcpo** ssi toute famille dirigée D a une **borne supérieure** $\sup^{\uparrow} D$.



- Exemples: $\{0 < 1\}$, $[0,1]$,
  P(A) pour l'inclusion,
  $\mathbf{N} \cup \{\omega\}$ pour $\leq$,
  tout treillis complet,
  N pour $=$, etc.

- Non-exemples: **N** avec $\leq$, **R**, $\mathbf{P}_{\text{fin}}(A)$ avec $\subseteq$.

Le dcpo $\mathbf{P}(A)$

- $X \stackrel{\text{def}}{=} \mathbf{P}(A)$, avec inclusion
- sups = unions (même si non dirigées)
- Pour toute $E \in \mathbf{P}(A)$, $E = \sup^{\uparrow} \mathbf{P}_{\text{fin}}(E)$
  [oui, ce sup est **dirigé**]
- **Ex:** {entiers pairs} = $\sup^{\uparrow}$ {{0}, {2}, {4}, …, {0,2}, {0,4}, …, {0,2,4}, …} dans $\mathbf{P}(\mathbf{N})$
- En fait, $\mathbf{P}(A)$ est un **dcpo algébrique**

Dcpos algébriques

- Un élément x d'un dcpo X est **fini** (ou **compact**) ssi pour toute famille dirigée D telle que $x \leq \sup^{\uparrow} D$, il existe $y \in D \ / \ x \leq y$
- X est **algébrique** ssi tout $z \in X$
  est $\sup^{\uparrow}$ des éléments finis $\leq z$
  [et, notamment, ils forment une famille dirigée]
- **Prop.** Les éléments finis de $\mathbf{P}(A)$ sont les parties finies (au sens usuel) de A.
- *Preuve*: …

P(*A*) est algébrique

- Un élément x d'un dcpo X est **fini** (ou **compact**) ssi pour toute famille dirigée D telle que $x \leq \sup D$, il existe $y \in D$ / $x \leq y$
- X est **algébrique** ssi tout $z \in X$ est sup dirigé des éléments finis $\leq z$

- **Prop.** Les éléments finis de **P**(A) sont les parties finies (au sens usuel) de A.
- ($\Leftarrow$) Si E est une partie finie $\{x_1, \ldots, x_n\}$, pour toute famille dirigée D / $E \subseteq \bigcup\uparrow D$, il existe $F_1 \in D / x_1 \in F_1$ et … et $F_n \in D / x_n \in F_n$.
- Comme D dirigée, $F_1, \ldots, F_n$ ont un majorant F dans D
- $x_1, \ldots, x_n$ sont dans F, donc $E \subseteq F$

$\mathbf{P}(A)$ est algébrique

- Un élément x d'un dcpo X est **fini** (ou **compact**) ssi pour toute famille dirigée D telle que $x \leq \sup D$, il existe $y \in D$ / $x \leq y$
- X est **algébrique** ssi tout $z \in X$ est sup dirigé des éléments finis $\leq z$

- **Prop.** Les éléments finis de $\mathbf{P}(A)$ sont les parties finies (au sens usuel) de A.
- ($\Rightarrow$) Si E est un élément fini de $\mathbf{P}(A)$, comme $E = \sup^{\uparrow} \mathbf{P}_{\text{fin}}(E)$ il existe $F \in \mathbf{P}_{\text{fin}}(E)$ / $E \subseteq F$
- Donc E est elle-même une partie finie. □
- Et donc $\mathbf{P}(A)$ est **algébrique**: pour toute $E \in \mathbf{P}(A)$, la famille $\mathbf{P}_{\text{fin}}(E)$ des éléments finis $\leq E$ est dirigée et de sup E.

Scott-continuité

- $f : X \to Y$ (dcpos) **Scott-continue** ssi:
  — f est monotone
  — et $\sup^{\uparrow}_{i \in I} f(x_i) = f(\sup^{\uparrow}_{i \in I} x_i)$
  pour toute famille dirigée $(x_i)_{i \in I}$.

- Par exemple, si $f : \mathbf{P}(A) \to \mathbf{P}(B)$ est Scott-continue, alors pour tout $E \subseteq A$,

$$f(E) = \sup^{\uparrow}_{F \text{ finie} \subseteq E} f(F)$$

… car $E = \sup^{\uparrow} \mathbf{P}_{\text{fin}}(E)$ (dirigé)

f est **entièrement déterminée** par ses valeurs sur $\mathbf{P}_{\text{fin}}(A)$

$[X \to Y]$

- Soit $[X \to Y]$ l'ensemble des fonctions **Scott-continues** du dcpo X vers le dcpo Y
- muni de l'ordre **point à point**:

$f \leq g$ ssi pour tout $x \in X$, $f(x) \leq g(x)$

- **Prop.** $[X \to Y]$ est un **dcpo**.
- et les sups dirigés sont **point à point**:

$$(\sup^{\uparrow}_{i \in I} f_i)\,(x) = \sup^{\uparrow}_{i \in I} (f_i\,(x))$$

Extensions

- Si X dcpo **algébrique**, et Y dcpo qq,
  et $K(X)$={éléments finis de X}, alors
  il y a **bijection** entre:
  — les fonctions Scott-continues : $X \to Y$
  — les fonctions monotones : $K(X) \to Y$

- Dans un sens, restriction à $K(X)$.
  Dans l'autre, si g monotone : $K(X) \to Y$
  son unique **extension** Scott-continue à tout X
  est: $\hat{g} : x \mapsto \sup^{\uparrow}_{c \text{ fini} \leq x} g(c)$.

- *Preuve…*

Extension: unicité

Si X dcpo **algébrique**, et Y dcpo qq,
et $K(X)$={éléments finis de X}, alors
il y a **bijection** entre:
— les fonctions Scott-continues : $X \to Y$
— les fonctions monotones : $K(X) \to Y$

- Soit g mono. : $K(X) \to Y$
  Si g a une extension f
  Scott-continue à tout X, alors

$$\forall x \in X, f(x) = \sup^{\uparrow}_{c \textbf{ fini} \leq x} g(c),$$

car $x = \sup^{\uparrow}_{c \textbf{ fini} \leq x} c$

Extension: existence

- **Lemme.** Soit X un dcpo algébrique.
  Pour toute fonction $g : K(X) \to Y$,
  $\hat{g}(x) \stackrel{\text{def}}{=} \sup^{\uparrow}_{c \textbf{ fini} \leq x} g(c)$ est Scott-continue.

- *Preuve* (1/2). $\hat{g}$ monotone: exercice (« si x augmente, il y a davantage de c possibles »)

- Que $\hat{g}$ soit Scott-continue est plus subtil, et nécessite la quantification sur des c **finis**

- **Lemme.** Soit X un dcpo algébrique.
  Pour toute fonction $g : K(X) \to Y$,
  $\hat{g}(x) \stackrel{\text{def}}{=} \sup^{\uparrow}_{c \text{ fini} \leq x} g(c)$ est Scott-continue.

- *Preuve* (2/2). Soit D famille di
  Pour tout d dans D, $\hat{g}(d) \leq \hat{g}(s$
  donc $\sup^{\uparrow}_{d \in D} \hat{g}(d) \leq \hat{g}(\sup D)$

on tente de calculer $\hat{g}(\sup^{\uparrow} D)$,
et on utilise la définition de $\hat{g}$

- Réc., pour tout c fini $\leq \sup^{\uparrow} D$,
- il existe $d \in D$ / $c \leq d$ [car c **fini**!]
- donc $g(c) \leq \hat{g}(d) \leq \sup^{\uparrow}_{d \in D} \hat{g}(d)$
- Donc $\hat{g}(\sup^{\uparrow} D) = \sup^{\uparrow}_{c \text{ fini} \leq \sup D} g(c)$
  $\leq \sup^{\uparrow}_{d \in D} \hat{g}(d)$. □

Extension: fin

- **Fait.** Soit X un dcpo algébrique.
  Pour toute fonction **monotone** $g : K(X) \to Y$,
  $\hat{g}$ étend g, i.e., $\hat{g}_{\restriction K(X)} = g$.

- Pour tout x **fini**,
  $\hat{g}(x) \stackrel{\text{def}}{=} \sup^{\uparrow}_{c \text{ fini} \leq x} g(c) = g(x)$
  car le sup est atteint en $c=x$. □

- Ceci conclut la preuve de:

- Si X dcpo **algébrique**, et Y dcpo qq,
  et $K(X)$={éléments finis de X}, alors
  il y a **bijection** entre:
  — les fonctions Scott-continues : $X \to Y$
  — les fonctions monotones : $K(X) \to Y$
- Dans un sens, restriction à $K(X)$.
  Dans l'autre, si g monotone : $K(X) \to Y$
  son unique extension Scott-continue à tout X
  est: $\hat{g} : x \mapsto \sup^{\uparrow}_{c \text{ fini} \leq x} g(c)$.

Le modèle $\mathbf{P}\omega$ de Gordon Plotkin et Dana Scott

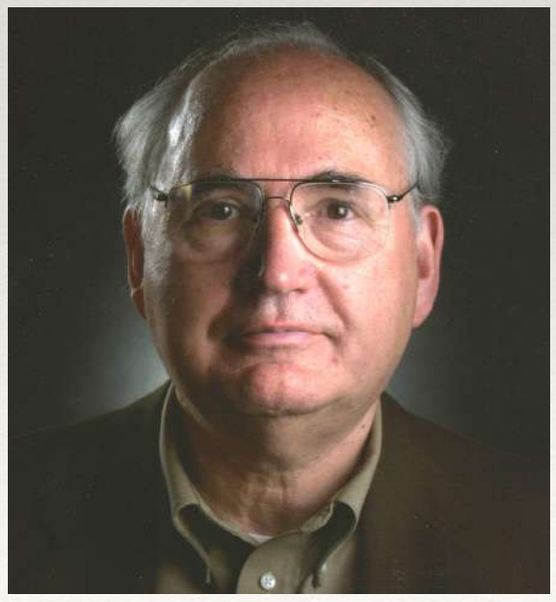
Pω

- **P**ω est juste (**P**(**N**), ⊆)…
  (ω est un nom chic pour
  l'ensemble **N** des entiers naturels)
- On va avoir besoin de coder:
  — des couples d'entiers
  — des ensembles finis d'entiers
  comme des entiers
- Tout le reste vient du fait que **P**(**N**)
  est un dcpo algébrique

Gordon D. Plotkin

https://www.research.ed.ac.uk/files-asset/17539241/gdp.jpg?w=320&f=webp

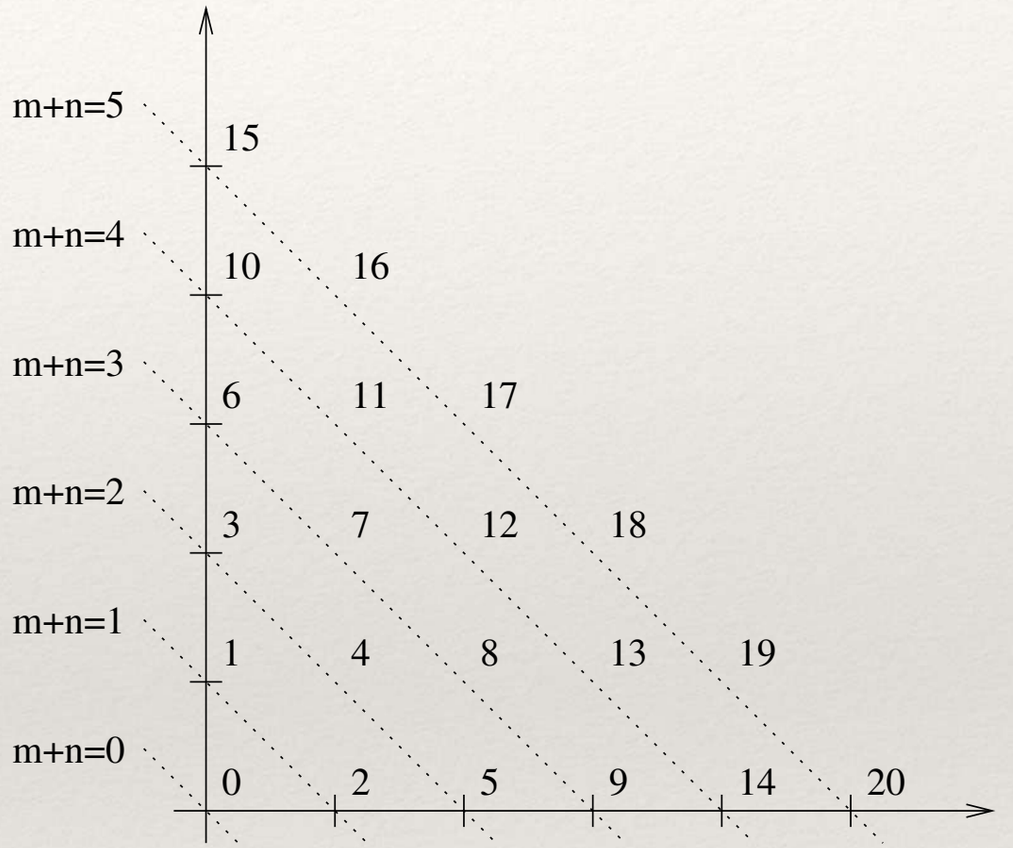
Dana S. Scott



https://www.scs.cmu.edu/sites/default/files/styles/news_item_image/public/DanaScott_b.jpg?itok=T9voOevm

Codage des couples d'entiers

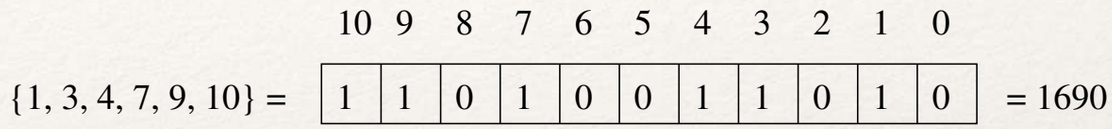
- $\langle m,n\rangle \stackrel{\text{def}}{=} (m+n)(m+n+1)/2+m$
  définit une bijection : $\mathbf{N} \times \mathbf{N} \rightarrow \mathbf{N}$
- Il y en a d'autres, par exemple
  — écrire m et n en binaire
  — entrelacer bit à bit
  — lire le nombre binaire résultant
  (i.e. $\langle 0,0\rangle' \stackrel{\text{def}}{=} 0$,
  $\langle 2m+a,2n+b\rangle' \stackrel{\text{def}}{=} 4\langle m,n\rangle'+2a+b$
  si $2m+a,2n+b \neq 0$, $a,b \in \{0,1\}$)



- Quelle bijection on prend n'aura pas beaucoup d'importance

Codage des ensembles finis

- Pour tout ensemble **finie** F,
  $[F] \stackrel{\text{def}}{=} \sum_{n \in F} 2^n$
  i.e., on écrit le nombre dont
  l'écriture en binaire a le bit n à 1 ssi $n \in F$



La conversion $i : [\mathbf{P}\omega \to \mathbf{P}\omega] \to \mathbf{P}\omega$

- Idée: on encode $f \in [\mathbf{P}\omega \to \mathbf{P}\omega]$ par ses valeurs $f(F)$ sur les ensembles **finis** F, pour chaque $F \in \mathbf{P}_{\text{fin}}(\mathbf{N})$

- **Définition.** $i(f) \stackrel{\text{def}}{=} \{\langle [F], n\rangle \mid F \in \mathbf{P}_{\text{fin}}(\mathbf{N}), n \in f(F)\}$

- **Fait.** i est **Scott-continue** de $[\mathbf{P}\omega \to \mathbf{P}\omega]$ vers $\mathbf{P}\omega$.

- *Preuve*: exercice.
  N'oubliez pas de démontrer la monotonie d'abord!

La conversion $r : \mathbf{P}\omega \to [\mathbf{P}\omega \to \mathbf{P}\omega]$

$$i(f) \stackrel{\text{def}}{=} \{\langle [F], n\rangle \mid F \in \mathbf{P}_{\text{fin}}(\mathbf{N}),\ n \in f(F)\}$$

- Comment retrouve-t-on f à partir d'un ensemble E (censément $i(f)$)?
- Pour chaque ensemble **fini** F, on peut retrouver
  $f(F) = \{n \in \mathbf{N} \mid \langle [F], n\rangle \in E\}$
- I ment
  (et sup
- **Définition** $r(E) \stackrel{\text{def}}{=} (E' \in \mathbf{P}\omega \mapsto \bigcup_{F \text{ fini} \subseteq E'} \{n \in \mathbf{N} \mid \langle [F], n\rangle \in E\})$
- $r(E) = \hat{g}_E$, où $g_E : F \in \mathbf{P}_{\text{fin}}(\mathbf{N}) \mapsto \{n \in \mathbf{N} \mid \langle [F], n\rangle \in E\}$
- Donc $r(E)$ Scott-continue, i.e. $r(E) \in [\mathbf{P}\omega \to \mathbf{P}\omega]$

Note: ceci est pour chaque E fixé;
r elle-même est Scott-continue de $\mathbf{P}\omega$ vers $[\mathbf{P}\omega \to \mathbf{P}\omega]$ (exercice)

- **Lemme.** Soit X un dcpo algébrique.
  Pour toute fonction $g : K(X) \to Y$,
  $\hat{g}(x) \stackrel{\text{def}}{=} \sup{}^{\uparrow}_{c \text{ fini} \leq x}\ g(c)$ est Scott-continue.

$r \circ i = \mathrm{id}$

$$i(f) \stackrel{\text{def}}{=} \{\langle [F], n\rangle \mid F \in \mathbf{P}_{\text{fin}}(\mathbf{N}),\ n \in f(F)\}$$

$$r(E) \stackrel{\text{def}}{=} (E' \in \mathbf{P}\omega \mapsto \bigcup_{F \text{ fini} \subseteq E'} \{n \in \mathbf{N} \mid \langle [F], n\rangle \in E\})$$

- **Fait.** $r \circ i = \mathrm{id}_{[\mathbf{P}\omega \to \mathbf{P}\omega]}$:

Preuve: pour toute $f \in [\mathbf{P}\omega \to \mathbf{P}\omega]$, et tout $E' \in \mathbf{P}\omega$,

- $r(i(f))(E') = \bigcup_{F \text{ fini} \subseteq E'} \{n \in \mathbf{N} \mid \langle [F], n\rangle \in i(f)\}$
- $= \bigcup_{F \text{ fini} \subseteq E'} \{n \in \mathbf{N} \mid n \in f(F)\}$
- $= \bigcup_{F \text{ fini} \subseteq E'} f(F)$
- $= f(E')$ car f **Scott-continue**
- Comme E' arbitraire, $r(i(f)) = f$.

Interprétation du λ-calcul dans Pω

- $[\![x]\!]\varrho \stackrel{\text{def}}{=} \varrho(x)$
  $[\![uv]\!]\varrho \stackrel{\text{def}}{=} r\,([\![u]\!]\varrho)\,([\![v]\!]\varrho)$
  $[\![\lambda x\,.\,u]\!]\varrho \stackrel{\text{def}}{=} i\,(V \in \mathbf{P}\omega \mapsto [\![u]\!](\varrho[x{:=}V]))$

$r : \mathbf{P}\omega \to [\mathbf{P}\omega \to \mathbf{P}\omega]$
$i : [\mathbf{P}\omega \to \mathbf{P}\omega] \to \mathbf{P}\omega$
$r \circ i = \mathrm{id}_{[\mathbf{P}\omega \to \mathbf{P}\omega]}$

- Il faut encore vérifier que ça a un sens…
  par exemple il faut que $V \in \mathbf{P}\omega \mapsto [\![u]\!](\varrho[x{:=}V])$ soit bien dans $[\mathbf{P}\omega \to \mathbf{P}\omega]$, c'est-à-dire **Scott-continue**

- Pour ceci, il suffit de montrer par récurrence sur u que
  $[\![u]\!] : \varrho \mapsto [\![u]\!]\,\varrho$ est bien définie **et Scott-continue**
  de $[\mathrm{Var} \to \mathbf{P}\omega]$ vers $\mathbf{P}\omega$ (exercice… Var ordonné par $=$; vous aurez besoin de la Scott-continuité de r et de i)

Interprétation du λ-calcul dans Pω

- $[\![x]\!]\varrho \stackrel{\text{def}}{=} \varrho(x)$
  $[\![uv]\!]\varrho \stackrel{\text{def}}{=} r\,([\![u]\!]\varrho)\,([\![v]\!]\varrho)$
  $[\![\lambda x\,.\,u]\!]\varrho \stackrel{\text{def}}{=} i\,(V \in \mathbf{P}\omega \mapsto [\![u]\!](\varrho[x{:=}V]))$

$r : \mathbf{P}\omega \to [\mathbf{P}\omega \to \mathbf{P}\omega]$
$i : [\mathbf{P}\omega \to \mathbf{P}\omega] \to \mathbf{P}\omega$
$r \circ i = \mathrm{id}_{[\mathbf{P}\omega \to \mathbf{P}\omega]}$

- **Théorème.** **P**ω est un modèle du λ-calcul:
  si $u =_\beta v$ alors $[\![u]\!]=[\![v]\!]$.

- *Preuve* (extrait — et rappel):

$$[\![(\lambda x\,.\,u)v]\!]\varrho = r\,([\![\lambda x\,.\,u]\!]\varrho)\,([\![v]\!]\varrho)$$
$$= r\,(i\,(V \in D \mapsto [\![u]\!](\varrho[x{:=}V]))\,([\![v]\!]\varrho)$$
$$= (V \in D \mapsto [\![u]\!](\varrho[x{:=}V]))\,([\![v]\!]\varrho)$$
$$= [\![u]\!](\varrho[x{:=}[\![v]\!]\varrho]) \;= [\![u[x{:=}v]]\!]\varrho. \quad \square$$

Interprétation du λ-calcul dans Pω

- $[\![x]\!]\varrho \stackrel{\text{def}}{=} \varrho(x)$
  $[\![uv]\!]\varrho \stackrel{\text{def}}{=} r\,([\![u]\!]\varrho)\,([\![v]\!]\varrho)$
  $[\![\lambda x\,.\,u]\!]\varrho \stackrel{\text{def}}{=} i\,(V \in \mathbf{P}\omega \mapsto [\![u]\!](\varrho[x:=V]))$

$r : \mathbf{P}\omega \to [\mathbf{P}\omega \to \mathbf{P}\omega]$
$i : [\mathbf{P}\omega \to \mathbf{P}\omega] \to \mathbf{P}\omega$
$r \circ i = \mathrm{id}_{[\mathbf{P}\omega \to \mathbf{P}\omega]}$

- **Théorème.** Pω est un modèle du λ
  si $u =_\beta v$ alors $[\![u]\!]=[\![v]\!]$

(sauf si E vide)
par exemple, si $E = \{\langle 0,0\rangle\}$
alors $i(r(E)) = \{\langle m,0\rangle \mid m \in \mathbf{N}\}$

- **Note**: Pω n'est **pas** un modèle extensionnel: $[\![\lambda x\,.\,yx]\!] \neq [\![y]\!]$.

$i(f) \stackrel{\text{def}}{=} \{\langle [F],n\rangle \mid \; \in \mathbf{P}_{\text{fin}}(\mathbf{N}),\ n \in f(F)\}$
$r(E) \stackrel{\text{def}}{=} (E' \in \mathbf{P}\omega \mid \bigcup_{F' \text{ fini} \subseteq E'} \{n \in \mathbf{N} \mid \langle [F'],n\rangle \in$

- Si $E=\varrho(y)$, $[\![\lambda x\,.\,yx]\!]\varrho = i(r(E))$
  $= \{\langle [F],n\rangle \mid F \in \mathbf{P}_{\text{fin}}(\mathbf{N}),\ n \in r(E)(F)\}$
  $= \{\langle [F],n\rangle \mid F \in \mathbf{P}_{\text{fin}}(\mathbf{N}),\ \exists F' \subseteq F,\ \langle [F'],n\rangle \in E\} \neq E = [\![y]\!]\varrho$

Dans le poly…

- Beaucoup d'autres choses, que je ne présenterai pas:
  - le modèle D_∞ de Dana Scott
    (lui est extensionnel — construit comme une limite projective)
  - Une introduction aux fonctions **stables**
  - **Continuations** (on aura le temps de les revoir!)
  - **Effets de bord**

La prochaine fois

La prochaine fois

- λ-calcul simplement typé
- Logique intuitionniste minimale propositionnelle
- Correspondance de Curry-Howard