Modèles pour le calcul et la logique

Frédéric Ruyer
Chercheur associé au LAMA, Université de Chambéry

PPS, Université Paris VII
6 Mars 2008

Cadre et Objet

- Modèles : représenter une réalité (syntaxe *vs* sémantique).
- Logique : valeur des énoncés.
- Calcul (λ) : valeur des programmes.
- Cet exposé :
  - Rappels sur les modèles.
  - Treillis premiers : types de données.
  - Treillis applicatifs : calcul.

Logique

- Formules :

$$F = p \mid (F \to F) \mid (F \wedge F) \mid (F \vee F) \mid \neg F$$

  On les note P,Q,...

- Contexte = liste de formules

$$\Gamma = P, (P \to Q), \ldots$$

- Séquent = d'un contexte je déduis ($\vdash$) la conclusion :

$$P, P \to Q \vdash Q$$

Calcul

Le λ-calcul :

- Un langage simple :

$$T = x \mid (TT) \mid \lambda x.T$$

- Les réductions :

$$(\lambda x.t \quad u) \succ_\beta t[x := u]$$

$$\lambda x.(t \quad x) \succ_\eta t \qquad x \notin VLIB(t)$$

Plan

1. Rappels sur les modèles
   - Algèbres de Boole
   - Modèles de Kripke et algèbres de Heyting
   - Réalisabilité
   - Modèles du lambda-calcul
2. Treillis premiers et types de données
   - Treillis premiers
   - Types de données dans les TCP/TCA
   - Isomorphimes
3. Treillis applicatifs
   - Treillis applicatifs et produits
   - Interprétation des termes dans un TA

Les deux vues

- Algèbre de Boole = anneau $< A, +, \times, 0, 1 >$ dans lequel $x^2 = x$.
- Treillis booléen = ensemble ordonné $< A, \leq >$:
  - Sup et Inf de deux éléments : $x \cap y$, $x \cup y$ (treillis).
  - 0 plus petit élément, 1 plus grand élément.
  - $\cap$ et $\cup$ distributives.
  - Complément : $x \cap x' = 0$ et $x \cup x' = 1$

Interprétation des formules

- Valeur de vérité V sur les variables induit valeur de vérité sur les formules :

$$V(P \vee Q) = V(P) \cup V(Q) \qquad V(\neg P) = V(P)^c \ldots$$

- Ordre = Implication, $0 = V(\bot)$, $1 = V(\top)$.
- Complétude : $\vdash_C P$ ssi $V(P) = 1$ dans toute algèbre de Boole.

Modèle de Kripke

- Repère = Ensemble ordonné formé d'étapes $R = \{e_1; e_2; \ldots\}$.
- Une étape valide une formule : $e \Vdash F$.
- Si une formule est validée à une étape, elle l'est "après" : si $e \Vdash F$, et $e \leq e'$, alors $e' \Vdash F$ (monotonie).
- Aucune étape ne valide le faux.

Satisfaction d'une formule par un élément

- $e \Vdash (P \wedge Q)$: $e \Vdash P$ et $e \Vdash Q$.
- $e \Vdash (P \vee Q)$: $e \Vdash P$ ou $e \Vdash Q$.
- $e \Vdash (P \to Q)$: pour tout e', si $e \leq e'$ et $e' \Vdash P$, alors $e' \Vdash Q$.
- $\Vdash P$: dans tout modèle de Kripke, toute étape valide P.
- Complétude : $\vdash_i P$ si et seulement si $\Vdash P$.

Interprétation d'une formule

On interprète les formules par l'ensemble des éléments qui la forcent :

- $I(P) \in \mathcal{P}$, avec $\mathcal{P} \subset P(R)$
- Condition de monotonie : $\mathcal{P}$ est formé des parties saturées par $\leq$.

Algèbres de Heyting

Une algèbre de Heyting $< A, \leq, 0, 1 >$:

- Treillis.
- Plus petit et plus grand élément (0, 1).
- Pour tout a, b : $\{x ; x \cap a \leq b\}$ a un plus grand élément, noté $a \Rightarrow b$.
- Complétude : $\vdash_i P$ si et seulement si $\Vdash P$.

Satisfaction d'une formule par un terme

- A chaque formule atomique p on associe un ensemble de termes $\tilde{p}$.
- $t \Vdash p : t \in \tilde{p}$
- $t \Vdash (P \to Q)$: pour tout t', si $t' \Vdash P$, alors $(t \quad t') \Vdash Q$.
- Pour le modus ponens : $\tilde{p}$ est saturée par β-expansion.
- Adéquation : si $\vdash_i t : P$ alors $t \Vdash P$.

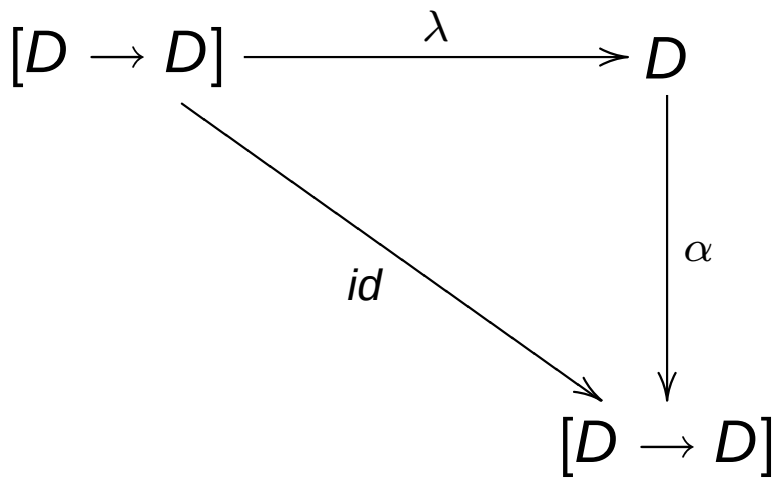
De façon analogue aux modèles de Kripke, on interprète les formules par l'ensemble des éléments qui la réalisent.

Interpréter le calcul

Interprétation des λ-termes :

- $I(\lambda x.t)$ " $=$ " une fonction " $=$ " un élément.
- espace des fonctions considérées doit être équipotent à l'ensemble.
- Condition d'existence de fonction :
  - si j'en sais + sur l'argument, j'en sait + sur le résultat
  - tout élément est approximable par ses minorants "finis".

Domaines



$$\alpha o \lambda = id$$

Interprétation I_v des λ-termes, qui dépend d'une valuation $v : V \to D$ des variables :

- $I_v(x) = v(x)$ si $x \in V$
- $I_v((t \quad u)) = \alpha(I_v(t))(I_v(u))$
- $I_v(\lambda x.t) = \lambda(d \mapsto I_{v[x:=d]}(t))$

Modulo la vérification que $d \mapsto I_{v[x:=d]}(t) \in [D \to D]$, cela définit une interprétation, et on a, par $\alpha o \lambda = id$, et par des lemmes techniques sur les substitutions :

$$\begin{aligned} I_v(\lambda x.t(u)) &= (\alpha o \lambda(d \mapsto I_{v[x:=d]}(t)))(I_v(u)) \\ &= (d \mapsto I_{v[x:=d]}(t))(I_v(u)) \\ &= I_{v[x:=I_v(u)]}(t) \\ &= I_v(t[x := u]) \end{aligned}$$

Domaines de Scott

- dcpo : Tout ensemble dirigé a un sup.
- fonction continue : $f(\cup A) = \cup f(A)$
- élément compact :

$$\text{si } k \leq \cup A, \text{ alors il existe } a \in A \text{ t.q } k \leq a.$$

- ensemble algébrique :

$$\text{pour tout } e,\ e = \cup\{k \text{ compact} ; k \leq e\}.$$

Domaine de Scott = dcpo algébrique tel que tout sous-ensemble majoré a un sup. Correspondance entre l'ensemble des Traces et l'ensemble fournit un domaine.

Exemples (1)

Engeler-Plotkin, 1970

- Types :

$$\begin{cases} T & = & o \mid (\{E\} \to T) \\ E & = & T \mid E; t (t \notin E, t \in T) \end{cases}$$

- $D = \mathcal{P}(T)$.
- $\lambda(f) = \cup\{e \to t; t \in f(e)\}$
- $\alpha(u)(v) = \{w \in T; \exists v' \subset_{fini} v : (v' \to w) \in u\}$

Exemple :

$$|\lambda x.x| = \{(e \to t); t \in e\}$$

Exemples (2)

P_ω Plotkin, 1970

- Codages :
  - des paires d'entiers $< m, n >$.
  - des parties finies de $\mathbb{N}$: e_m, i_e.
- $D = \mathcal{P}(\mathbb{N})$.
- $\lambda(f) = \{< m, n >; n \in f(e_m)\}$
- $\alpha(u)(v) = \{w \in \mathbb{N}; \exists v' \subset_{fini} v :< i_{v'}, w >\in u\}$

Exemple :

$$|\lambda x.x| = \{< m, n >; n \in e_m\}$$

- Points communs (analogies ? ?) : ensembles ordonnés, existence de sups divers, “atomes”, “compacts”, . . .
- Les treillis et la réalisabilité fournissent des modèles qui unifient les points de vue.

1. Rappels sur les modèles
   - Algèbres de Boole
   - Modèles de Kripke et algèbres de Heyting
   - Réalisabilité
   - Modèles du lambda-calcul
2. Treillis premiers et types de données
   - Treillis premiers
   - Types de données dans les TCP/TCA
   - Isomorphimes
3. Treillis applicatifs
   - Treillis applicatifs et produits
   - Interprétation des termes dans un TA

Intuition

- Identifier les programmes qui s'évaluent de la même façon :
  - $p \succ q$ signifie "p s'évalue en q".
  - $\tilde{1} = \{p; p \succ 1\}$
  - $\mathbb{N} = \{p; p \text{ s'évalue en un entier}\}$
- Les objets étudiés sont les ensembles de parties saturées par l'expansion ($\prec$).
- Modéliser les produits par la quantification sur les éléments premiers.
- Notions inspirées de celles de Tarski.

Treillis complets

Treillis complet :

- Ensemble ordonné $(E, \leq, \bot, \top)$
- Toute partie $A \subset E$ admet :
  - une borne supérieure notée $\vee A$
  - une borne inférieure notée $\wedge A$.
- $\bot = \wedge E$ et $\top = \vee E$

Eléments premiers et éléments atteints

- p est premier : pour tout $A \subset E$, si $p \leq \vee A$, alors il existe $a \in A$ tel que $p \leq a$.
- k est atteint : k est premier et pour tout p premier, si $k \leq p$, alors $k = p$.
- Notations :
  - Minorants premiers de $e \in E$ noté P_e
  - $\forall x : a.Q(x) = \forall x \in P_a.Q(x)$

Treillis premiers, treillis atteints

- Treillis complet premier (TCP) :

  pour tout $e \in E$, $e = \vee P_e$.

- Treillis complet atteint (TCA) :

  TCP + tout premier est atteint.

Propriété fondamentale

Fait

Si E est un treillis premier alors pour tout $a, b \in E$:

$$a \leq b \qquad \Longleftrightarrow \forall p \in P_a : p \leq b$$

Schéma de compréhension

Schéma de compréhension de P :

$$SC(P) = \vee\{x \in P_{\top}; P(x)\}$$

(bien défini dans un treillis complet)

Fait

1. *Si E TCP alors :*

$$\forall x : a.P(x) \Rightarrow a \leq SC(P)$$

2. *Si E TCA alors :*

$$\forall x : a.P(x) \Leftrightarrow a \leq SC(P)$$

Projection

ϵ un opérateur de choix :

$$P(\epsilon(P)) \text{ s'il existe un x tel que } P(x).$$

Fait

Si E est un treillis complet, alors pour tout e premier, pour tout $\phi : F \to E$: si on note $e.\phi = \epsilon(f \mapsto e \leq \phi(f))$, on a

$$e \leq \vee\{\phi(f); f \in F\} \Rightarrow e \leq \phi(e.\phi)$$

Induction (1)

Elément inductif associé à une fonction dans un TC :

$$\mu F = \wedge\{x; \forall a \in E(a \leq x \Rightarrow F(a) \leq x)\}$$

Définition plus restrictive que celle usuelle de Tarski :

$$\mu_T F = \wedge\{x; F(x) \leq x\}$$

Fait

1. $\forall x(x \leq \mu F \Rightarrow F(x) \leq \mu F)$
2. $F(\mu F) \leq \mu F$
3. *Si F croit,* $\mu F = \mu_T F$

Induction (2)

Fait (Principe d'induction)

Si E est un TCA :

$$\forall a(\forall x : a.P(x) \Rightarrow \forall x : F(a).P(x)) \Rightarrow \forall x : \mu F.P(x)$$

Démonstration.

On suppose $\forall a(\forall x : a.P(x) \Rightarrow \forall x : F(a).P(x))$. Il suffit d'utiliser le schéma de compréhension $SC(P)$, ce qui entraîne dans un TCA :
$\forall a(a \leq SC(P) \Rightarrow F(a) \leq SC(P))$! □

Co-induction

Elément co-inductif associé à une fonction.

$$\nu F = \vee\{x; \forall a \in E(x \leq a \Rightarrow x \leq F(a))\}$$

Fait (Principe de co-induction)

Si E est un TCA, si F est croissante :

$$\forall a(a \leq F(a) \Rightarrow \forall x : a.P(x)) \Rightarrow \forall x : \nu F.P(x)$$

Parties saturées

- Soit X un ensemble, et R une relation sur X. On dit que $P \subset X$ est *saturée* par R si :

$$\forall x \in P, \forall y \in X : x\,R\,y \Rightarrow y \in P$$

- $\mathcal{P}_R(X)$: ensemble des parties de X saturées par R.
- $\{x\}_R$: la plus petite partie saturée qui contient $\{x\}$.
- ensemble des *singletons* $\mathcal{S}_R(X)$: les $\{x\}_R$.

Isomorphismes (1)

Les parties saturées fournissent des *TCP* :

Fait

Soit X un ensemble et R une relation sur X.

1. $\mathcal{T}_R(X) = (\mathcal{P}_R(X), \subseteq, \emptyset, X)$ *est un TCP, et* $P_{\mathcal{T}_R(X)} = \mathcal{S}_R(X)$.
2. $\mathcal{T}_R(X)$ *est un TCA si et seulement si la clôture transitive de R est symétrique (est une "PER").*

Isomorphismes (2)

Et réciproquement :

Théorème

1. *Tout TCP est isomorphe à un ensemble de parties saturées.*
2. *Tout TCA est isomorphe à un ensemble de parties saturées par une relation d'équivalence.*

- TCA = TCP + complément.
- TCA = cas particulier d'algèbre de Boole (complète atomique).
- La négation ne va pas s'interpréter comme le complément

Treillis Applicatifs

Un treillis applicatif $(E, \leq, \bot, \top, @)$ est la donnée :

- d'un treillis complet $(E, \leq, \bot, \top)$.

- d'une loi $@ : (E \times E) \rightarrow E$ qui commute aux sups (bi-continue au sens de Scott) :

$$\vee\{x@b; x \in A\} = \vee A@b$$

$$\vee\{b@x; x \in A\} = b@ \vee A$$

Produit simple dans un TA

Un TA permet de définir la Flèche :

$$a \to b = \vee\{y ; y @ a \leq b\}$$

Fait

Pour tous $a, b, c \in E$ *:*

$$a \leq (b \to c) \quad \Longleftrightarrow \quad (a @ b) \leq c$$

Produits dépendants dans un TAP :

$$\Pi x : A.B(x) = \vee\{y; \forall x : A(y@x) \leq B(x)\}$$

Fait

- $F \leq \Pi x : A.B(x)$ *si et seulement si* $\forall f : F.\forall x : A.(f@x) \leq B(x)$
- *Si x n'est pas libre dans B,* $\Pi x : A.B(x) = A \rightarrow B$.

Abstraction dans un TA

Un TA permet de définir l'Abstraction :

$$\Lambda f = \wedge\{x \to f(x); x \in E\}$$

Fait

Pour toute fonction f de E dans E :

1. $\Lambda f @ a \leq f(a)$.
2. $a \leq \Lambda(x \mapsto a @ x)$.

Interprétation des termes

On se donne une valuation $\mathcal{V}$ des variables du λ-calcul dans un TA.

- $\mathcal{I}_{\mathcal{V}}(x) = \mathcal{V}(x)$ pour toute variable x.
- $\mathcal{I}_{\mathcal{V}}(uv) = \mathcal{I}_{\mathcal{V}}(u) @ \mathcal{I}_{\mathcal{V}}(v)$.
- $\mathcal{I}_{\mathcal{V}}(\lambda x.t) = \Lambda(e \mapsto \mathcal{I}_{\mathcal{V}[x:=e]}(t))$.
- Cela suffit à fournir un modèle de $\Lambda_{\prec_\beta \cup \succ_\eta}$ dans le sens suivant :

Théorème

- *Si* $u \succ_\beta v$, *alors* $\mathcal{I}(u) \leq \mathcal{I}(v)$
- *Si* $u \succ_\eta v$, *alors* $\mathcal{I}(u) \geq \mathcal{I}(v)$

Réalisabilité abstraite

Si on se donne une interprétation des types simples, on peut définir une forme de réalisabilité “abstraite” dans tout TA :

$$t \vDash_a A \qquad = \qquad \mathcal{I}(t) \leq \mathcal{I}(A)$$

Fait

Dans un TAP, si l'interprétation de tout λ-terme est un élément premier, alors : Si $\vdash t : A$, alors $t \vDash_a A$.

Remarque

- *La partie délicate est l'introduction de la flèche.*
- *Le résultat se généralise au système ST , dans le cas atteint.*

Conclusion

- Un TA est λ-atteint si l'interprétation de tout terme est atteinte : il fournit en ce cas un modèle de $\Lambda_{=_\beta \cup =_\eta}$.
- Notions proches de l'algèbre de Heyting, mais la conjonction est remplacée par l'application.
- On en a un exemple simple : $\Lambda_{=_\beta \cup =_\eta}$, qui est de plus un TCA (est ce le seul ?).
- Les TCA λ-atteints fournissent un cadre "simple" d'interprétation du système ST.

Pistes de recherche

- Compréhension des modèles.
- Complétude vis-à-vis de la réalisabilité formelle.
- Extension de ST : logique classique (accepté), logique mixte (MST), . . .
- Typage plus réaliste, Extension des traits, Implémentation . . .

Courte bibliographie

- Chantal Berline *Cours de DEA*. P7, 2002.
- René Cori, Daniel Lascar *Logique mathématique*. Masson 1993.
- Jean-Louis Krivine *Lambda calcul, types et modèles*. Masson 1990.
- René Lalement *Logique, Réduction, Résolution*. Masson 1990.
- Christine Paulin-Mohring, *Extraction de Programmes dans le Calcul des Constructions*, Thèse U.P7 1989.
- Christophe Raffalli, *System ST, toward a Type System for Extraction and Proof of Programs*, APAL 2003.
- Christophe Raffalli, *System ST, beta-reduction and completeness*, LICS 2003.
- Raffalli,Christophe et Ruyer,Frédéric, *Realizability of the axiom of choice in HOL : an analysis of Krivine's work* Fundamenta Informatica 2007.
- Cardelli,Luca et Leroy,Xavier, *Abstract types and the dot notation*,Digital Equipment Corporation SRC, 1990.
- Frédéric Ruyer ,*Preuves, Types et Sous-types*, Thèse de l'U. Savoie, 2006.