型付きλ計算

Jacques Garrigue, 2007 年 5 月 25 日

λ計算の関数は数学的関数と違い、定義域と値域を定めていない。例えば、前回で見た c_+ は整数以外の物に適用しても意味がないが、そのような項は正しくないと判断できない。

1 型 (type) と項

型付きλ計算では、そのような領域を**型**で表す。単純な計算系では全ての値は一つだけの型に属する。型には二種類があり、基底型と関数・構造型に分けられる。

$$\begin{array}{rcl} b & ::= & \mathsf{int} \mid \mathsf{bool} \mid \ldots \\ t & ::= & b \mid t \to t \mid t \times t \end{array}$$

項の中にも、型情報を入れ、定数も追加する。

$$M ::= x \mid c_t \mid \lambda x{:}t.M \mid (M\ M) \mid (M, M)$$

簡約規則には、β に加えて定数に関するδ規則も導入される。

$$\begin{array}{lcl} (\lambda x{:}\tau.M)\ N & \to & [N/x]M \\ (\mathsf{fst}_{\tau\times\theta\to\tau}\ (M,N)) & \to & M \\ (\mathsf{snd}_{\tau\times\theta\to\theta}\ (M,N)) & \to & N \\ (\mathsf{s}_{int\to int}\ n_{\mathsf{int}}) & \to & (n+1)_{\mathsf{int}} \\ (\mathsf{if0}_{\mathsf{int}\to\tau\to\tau\to\tau}\ 0_{\mathsf{int}}\ M\ N) & \to & M \\ (\mathsf{if0}_{\mathsf{int}\to\tau\to\tau\to\tau}\ n_{\mathsf{int}}\ M\ N) & \to & N \\ \ldots & & \end{array}$$

上記のδ規則では s は固定の型を持っているが、fst, snd と if0 は様々な型について定義しなければならない。τ や θ はそういう任意に決められる型を表している。

2 型推論

ある項の正しさ (well-formedness) を型判定式で表す。

$$\Gamma \vdash M : \tau$$

M は項であり、τ は型である。そして Γ は型宣言列と呼ばれる変数名と型の集合 $(x_1 : \tau_1, \ldots, x_n : \tau_n)$。次の型推論規則で型判定式が証明された項を正しいとする。

変数 $\quad \Gamma \vdash x : \tau \quad (x : \tau \text{は} \Gamma \text{に含まれる})$

定数 $\quad \Gamma \vdash c_\tau : \tau$

抽象 $\dfrac{\Gamma, x:\theta \vdash M:\tau}{\Gamma \vdash \lambda x{:}\theta.M:\tau}$

適用 $\dfrac{\Gamma \vdash M:\theta\to\tau \quad \Gamma\vdash N:\theta}{\Gamma\vdash (M\ N):\tau}$

直積 $\dfrac{\Gamma\vdash M:\tau \quad \Gamma\vdash N:\theta}{\Gamma\vdash (M,N):\tau\times\theta}$

証明の例

$$\dfrac{\dfrac{\dfrac{x:\mathsf{int}\vdash \mathsf{s}_{\mathsf{int}\to\mathsf{int}}:\mathsf{int}\to\mathsf{int} \quad x:\mathsf{int}\vdash x:\mathsf{int}}{x:\mathsf{int}\vdash(\mathsf{s}_{\mathsf{int}\to\mathsf{int}}\ x):\mathsf{int}}}{\vdash\lambda x{:}\mathsf{int}.(\mathsf{s}_{\mathsf{int}\to\mathsf{int}}\ x):\mathsf{int}\to\mathsf{int}} \qquad \vdash 1_{\mathsf{int}}:\mathsf{int}}{\vdash((\lambda x{:}\mathsf{int}.(\mathsf{s}_{\mathsf{int}\to\mathsf{int}}\ x))\ 1_{\mathsf{int}}):\mathsf{int}}$$

2.1 性質

次の定理は fst, snd 以外の δ 規則を含まない計算系に関するものである。

定理 1 (主部簡約) $\Gamma\vdash M:\tau$ と $M\to N$ が成り立てば、$\Gamma\vdash N:\tau$ が成り立つ。

定理 2 (停止性) $\Gamma\vdash M:\tau$ ならば、無限な簡約列 $(M\to M_1\to M_2\to\ldots)$ は存在しない。

停止性が成り立てば、定義できないものがある。例えば、型のないλ計算で停止しなかった $(\lambda x.x\ x)(\lambda x.x\ x)$ は定義できない。それを見せるために、$\lambda x:\tau.x\ x$ の証明を書いてみよう。

$$\dfrac{\dfrac{x:\tau\vdash x:\tau\to\theta \quad x:\tau\vdash x:\tau}{x:\tau\vdash x\ x:\theta}}{\vdash\lambda x:\tau.x\ x:\theta}$$

項の構造から、以上の形の証明しかできないが、条件として $\tau=(\tau\to\theta)$ を満す必要がある。当然ながら、上記の型の定義では、こんな方程式を満すような τ は存在しない。

同様に Y を導入すると、止らない計算ができるので、型付λ計算では Y は定義できない。

3 論理との関係

各推論規則の中で型のところだけを残せば、λ計算を作る動機となった直観主義論理の推論規則が得られる。定数は公理になる。

例えば、$\lambda x:\tau\times\theta.(\mathsf{snd}\ x,\mathsf{fst}\ x)$ の証明は

$$\dfrac{\dfrac{\Gamma\vdash\mathsf{snd}:\tau\times\theta\to\theta \quad \Gamma\vdash x:\tau\times\theta}{\Gamma\vdash(\mathsf{snd}\ x):\theta} \quad \dfrac{\vdash\mathsf{fst}:\tau\times\theta\to\tau \quad \Gamma\vdash x:\tau\times\theta}{\Gamma\vdash(\mathsf{fst}\ x):\tau}}{\dfrac{\Gamma=x:\tau\times\theta\vdash(\mathsf{snd}\ x,\mathsf{fst}\ x):\theta\times\tau}{\vdash\lambda x:\tau\times\theta.(\mathsf{snd}\ x,\mathsf{fst}\ x):\tau\times\theta\to\theta\times\tau}}$$

を次の直観主義論理の証明に変えられる

$$\dfrac{\dfrac{\dfrac{A\wedge B\to B \quad A\wedge B^{(1)}}{B} \quad \dfrac{A\wedge B\to A \quad A\wedge B^{(1)}}{A}}{B\wedge A}}{A\wedge B\to B\wedge A}\,{}^{(1)}$$

しかも最初の証明は項だけから再構築できるので、ある項はその型の証明である、ともいう。

4 万能性

以上のδ規則と停止性で分かるように、単体では型付λ計算は万能ではない。ただし、二種類の異る問題がある。

整数などに関するδ規則について、チャーチ数が型付けできないのは原因である。細かく言えば、型付けはできるが、汎用性が足りない。

$$\vdash \lambda f : \tau \to \tau . \lambda x : \tau . f^n \ x : (\tau \to \tau) \to \tau \to \tau$$

問題は、この関数はある特定の型τにおいてしか使えない。チャーチ数の計算は様々なものをxに代入することを利用していたので、これでは意味のあるエンコーディングはできない。この問題は、計算をδ規則に移すことで解決される。

同様に、再帰的な計算をしたければ、Yのδ規則を入れる必要がある。

$$\mathsf{Y}_{(\tau\to\tau)\to\tau} \ M \to M \ (\mathsf{Y}_{(\tau\to\tau)\to\tau} \ M)$$

整数とYだけを加えれば、型付λ計算は万能になる。実行にはδ規則を使うか、定数を型のないλ計算の項に変えて型のないλ計算で実行するかのどちらもができる。

型体系を強くすれば、汎用性のある定義ができるようになり、チャーチ数の定義がそのまま使える。二次λ計算では型変数が用意されている。

$$\begin{array}{rcl} t & ::= & \ldots \mid \tau \mid \forall \tau . t \\ M & ::= & \ldots \mid \Lambda \tau . M \mid M[t] \end{array}$$

例えば二次λ計算と言われる体系では、次のように表現する。

$$\begin{array}{l} \vdash \mathsf{c}_n = \Lambda \tau . \lambda f : \tau \to \tau . \lambda x : \tau . f^n x : \forall \tau . (\tau \to \tau) \to \tau \to \tau \\ \vdash \mathsf{c}_+ = \lambda m : Int . \lambda n : Int . \Lambda \tau . \lambda f : \tau \to \tau . \lambda x : \tau . (m[\tau] \ x \ (n[\tau] \ f \ x)) : Int \to Int \to Int \\ \vdash \mathsf{c}_\times = \lambda m : Int . \lambda n : Int . \Lambda \tau . \lambda f : \tau \to \tau . (m[\tau] \ (n[\tau] \ f)) : Int \to Int \to Int \\ \vdash \mathsf{c}_{\mathsf{exp}} = \lambda m : Int . \lambda n : Int . \Lambda \tau . n[\tau \to \tau] \ (m[\tau]) : Int \to Int \to Int \end{array}$$

ここでは$Int = \forall \tau . (\tau \to \tau) \to \tau \to \tau$。

ただし、二次λ計算は型付λ計算と同様に停止性を持っているので、相変わらずYは定義できない。それは欠点とは限らない。全ての計算が止るということは、全ての項に計算できる意味があるということでもある。

さらに進んで、再帰的な型を持ったλ計算を使えば、Yも定義できるようになるが、停止性が失われる。