計算と論理

Jacques Garrigue, 2017 年 1 月 20 日

10　命題論理

前回見た命題論理は含意と論理積に限定したものであった．ここでは，さらに否定と論理和を含めることで，完成させる．

10.1　構文と意味論

$$
\begin{array}{rcll}
P, Q & ::= & A & \text{アトム} \\
 & | & P \supset Q & \text{含意} \\
 & | & P \wedge Q & \text{論理積} \\
 & | & P \vee Q & \text{論理和} \\
 & | & \neg P & \text{否定} \\
 & | & \top & \text{真} \\
 & | & \bot & \text{偽}
\end{array}
$$

$v : Atom \to \{\top, \bot\}$ という割り当て関数を用意すると，その割り当てに対する論理式の真偽 $[\![P]\!]_v$ が計算できる．以下の真偽表を使う．

P	Q	$P \wedge Q$	$P \vee Q$	$\neg P$	$P \supset Q$
$\top$	$\top$	$\top$	$\top$	$\bot$	$\top$
$\top$	$\bot$	$\bot$	$\top$	$\bot$	$\bot$
$\bot$	$\top$	$\bot$	$\top$	$\top$	$\top$
$\bot$	$\bot$	$\bot$	$\bot$	$\top$	$\top$

例えば $v(A) = \top$ で $v(B) = \bot$ ならば，

$$[\![A \wedge (\neg B)]\!]_v = \top \wedge \neg \bot = \top \wedge \top = \top$$

任意の割り当てについて $[\![P]\!]_v = \top$ ならば，P を**恒真**と言う．また，任意の割り当てについて $[\![P]\!]_v = [\![Q]\!]_v$ ならば，P と Q は**論理的同値**と言う．
以下の論理式は論理的同値である．

$$
\begin{array}{rcl}
\neg A & \Leftrightarrow & A \supset \bot \\
A \supset B & \Leftrightarrow & \neg A \vee B \\
\neg(A \wedge B) & \Leftrightarrow & \neg A \vee \neg B \\
\neg(A \vee B) & \Leftrightarrow & \neg A \wedge \neg B
\end{array}
$$

なお，任意の論理式 P, Q について，P と Q が論理的同値であることと $(P \supset Q) \wedge (Q \supset P)$ が恒真であることは同値である．

10.2 証明論

前説の意味論による定義は，式の全ての割あえてにおける計算を必要としている．アトムが多くなると，2^n 個の場合を考えなければならない．

証明論は式の証明をあらかじめ定めた公理と推論規則から導出することで，もっと簡潔に正しさを保障できる．

$$\supset \text{導入}\ \frac{\begin{array}{c}[P]^{(n)}\\ \vdots \\ Q\end{array}}{P \supset Q}{}^{(n)} \qquad \supset \text{除去}\ \frac{P \supset Q \quad P}{Q}$$

$$\wedge\text{導入}\ \frac{P \quad Q}{P \wedge Q} \qquad \wedge \text{除去}\ P \wedge Q \supset P \qquad P \wedge Q \supset Q$$

$$\vee\text{導入}\ \frac{P}{P \vee Q} \qquad \frac{Q}{P \vee Q} \qquad \vee \text{除去}\ \frac{P \supset R \quad Q \supset R}{(P \vee Q) \supset R}$$

$$\text{背理法}\ \frac{\neg P \supset \bot}{P} \qquad \text{恒真} \quad \top$$

ここでは $\neg P$ を $P \supset \bot$ の略記法とみなす．

例えば以下のように排中律が証明できる．

$$\dfrac{\dfrac{\neg(P \vee \neg P)^{(2)} \qquad \dfrac{\dfrac{\dfrac{\neg P^{(1)}}{P \vee \neg P} \quad \neg(P \vee \neg P)^{(2)}}{\bot}}{\neg P \supset \bot}{}^{(1)}}{P}}{P \vee \neg P} \qquad \dfrac{\dfrac{\bot}{\neg(P \vee \neg P) \supset \bot}{}^{(2)}}{P \vee \neg P}$$

証明論と意味論の間以下の二つの定理が成り立つ．

定理 1 (健全性) 論理式 P の証明が導出できるなら，P は意味論において恒真である．

定理 2 (完全性) 論理式 P が意味論において恒真なら，P の証明が存在する．

10.3 型付き λ 計算と直観主義命題論理

λ 計算を拡張することで，Curry-Howard 同型を上記の命題論理に拡張できる．

$$M ::= \ldots \mid \mathsf{left}\ M \mid \mathsf{right}\ M$$

以下の δ 規則も追加する．

$$\begin{array}{lcl} \mathsf{case}_{(\tau_1 \to \theta) \to (\tau_2 \to \theta) \to \tau_1 + \tau_2 \to \theta}\ M_1\ M_2\ (\mathsf{left}\ M) & \to & M_1\ M \\ \mathsf{case}_{(\tau_1 \to \theta) \to (\tau_2 \to \theta) \to \tau_1 + \tau_2 \to \theta}\ M_1\ M_2\ (\mathsf{right}\ M) & \to & M_2\ M \end{array}$$

併せて，以下の推論規則を追加する．

$$\text{直和左}\ \frac{\Gamma \vdash M : \tau}{\Gamma \vdash \mathsf{left}\ M : \tau + \theta} \qquad \text{直和右}\ \frac{\Gamma \vdash M : \theta}{\Gamma \vdash \mathsf{right}\ M : \tau + \theta}$$

背理法と恒真のために以下の定数を用意する．

$$\mathsf{classic}_{((\tau\to\bot)\to\bot)\to\tau} \qquad \mathsf{unit}_{\top}$$

この拡張で今までどおり型付き λ 計算と命題論理の証明の間に同型ができる．しかし，背理法を実装した classic に対する δ 規則がないので，背理法を使った証明に対して計算ができない．

これを克服するために，**直観主義論理**という背理法を含まない論理が作られた．厳密には直観主義論理が先にに Brouwer や Gentzen に証明の上の計算が可能な論理として提案され，型付き λ 計算はそれに対する計算系となる．

背理法を除くだけでは，否定に関する扱いが弱いので，代わりに推論規則を追加する．

$$\text{矛盾}\ \frac{\bot}{P}$$

要するに，$\bot$ が証明できれば何でも証明できる．

$\bot$ の証明は絶対に作れないので，λ 計算では以下の δ 規則が使える．

$$\begin{array}{rcl} (\mathsf{abort}_{\bot\to\tau\to\theta}\ M)\ N & \to & \mathsf{abort}_{\bot\to\theta}\ M \\ M_{\tau\to\theta}\ (\mathsf{abort}_{\bot\to\tau}\ N) & \to & \mathsf{abort}_{\bot\to\theta}\ N \end{array}$$

要するに，矛盾が証明できた時点で計算を終了する．

直観主義論理は前述の意味論に対して健全だが，完全ではない（証明できないものもある）．ただし，論理式を二重否定で始めるものに限らせると完全になる．

定理 3 (健全性) 論理式 P の証明が直観主義論理で導出できるなら，P は意味論において恒真である．

定理 4 (完全性) 論理式 P が意味論において恒真なら，直観主義論理において $\neg\neg P$ の証明が存在する．

11 述語論理と依存型

上記の二次 λ 計算は計算能力という面で単純型付 λ 計算を拡張するが，論理の面では相変わらず命題論理にしか対応していない．述語論理に対応するために，異なる種類の拡張が必要になる．

述語論理の命題は多相型に似ている．

$$\begin{array}{rcll} t & ::= & x \mid a \mid f(t,\ldots) & \text{項} \\ A & ::= & \bot \mid A \to A \mid A \wedge A \mid A \vee A & \\ & \mid & p(t,\ldots) & \text{述語} \\ & \mid & \forall x.A & \text{全称} \\ & \mid & \exists x.A & \text{存在} \end{array}$$

しかし，よく見ると，ここでは t や x で表されているのは型ではなく，項である．そこで，二次 λ 計算と少し違う拡張が考えられる．二次 λ 計算では項の中に型が入ってもよかったが，今度は型の中に項を入れる．

$$\begin{array}{rcll} t & ::= & b \mid \bot \mid p_t\ M \mid \Pi x{:}t.t \mid t \times t & \text{型} \\ M & ::= & x \mid c_t \mid \lambda x{:}t.M \mid (MM) \mid (M, M) & \text{項} \end{array}$$

二つ目の型に x が現れない場合，$\Pi x{:}t_1.t_2$ (依存関数型) は $t_1 \to t_2$ と書ける．

推論規則も増やす．特に，型の構成を確認する必要がある．

型
$$\frac{\Gamma \vdash M : t}{\Gamma \vdash p_t\ M\ \mathrm{ok}} \qquad \frac{\Gamma, x : t \vdash t'\ \mathrm{ok}}{\Gamma \vdash \Pi x{:}t.t'\ \mathrm{ok}} \qquad \frac{\Gamma, x : t \vdash t'\ \mathrm{ok}}{\Gamma \vdash \Sigma x{:}t.t'\ \mathrm{ok}}$$

抽象
$$\frac{\Gamma, x : t \vdash M : t'}{\Gamma \vdash \lambda x{:}t.M : \Pi x{:}t.t'}$$

適用
$$\frac{\Gamma \vdash M : \Pi x{:}t.t' \quad \Gamma \vdash N : t}{\Gamma \vdash (M\ N) : [N/x]t'}$$

変換
$$\frac{\Gamma \vdash M : [N/x]t \quad N =_{\beta\delta} N'}{\Gamma \vdash M : [N'/x]t}$$

否定
$$\frac{\Gamma \vdash M : \bot \quad \Gamma \vdash t\ \mathrm{ok}}{\Gamma \vdash M : t}$$

以上の定義を使えば，述語論理を表現することができる．実は，述語論理で表現できないものも入ってしまうので，準同型にしかならないが，さらなる制限を加えると同型になる．

例えば,「人間は死ぬ，ソクラテスは人間である，すなわちソクラテスは死ぬ」は以下の型として表現できる．

$$(\Pi x{:}Name.Human\ x \to Mortal\ x) \to Human\ Socrates \to Mortal\ Socrates$$

以下の項によって証明される．

$$\lambda mortal{:}(\Pi x{:}Name.Human\ x \to Mortal\ x).\lambda human{:}(Human\ Socrates).mortal\ Socrates\ human$$

式「$\forall x.x + x = 2 \times x$」も以下のように書ける．

$$\Pi x{:}\mathsf{Nat}.eqnat(add\ x\ x, mult\ 2\ x)$$

δ 規則は以下のとおり．

$$\begin{array}{l} add\ 0\ n \to n \\ add\ (sm)\ n \to s\ (add\ m\ n) \\ mult\ 0\ n \to 0 \\ mult\ (sm)\ n \to add\ n\ (mult\ m\ n) \end{array}$$

以下の定数 (公理) を使って証明できる．

$$\begin{array}{l} add\_sym : \Pi m{:}\mathsf{Nat}.\Pi n{:}\mathsf{Nat}.eqnat(add\ m\ n, add\ n\ m) \\ eq\_sub : \Pi f{:}(\mathsf{Nat} \to \mathsf{Nat}).\Pi m{:}\mathsf{Nat}.\Pi n{:}\mathsf{Nat}.eqnat(m, n) \to eqnat(f\ m, f\ n) \end{array}$$

以下の項によって証明される．

$$\lambda x{:}\mathsf{Nat}.eq\_sub\ (\lambda y{:}\mathsf{Nat}.add\ x\ y)\ (add\_sym\ 0\ x)$$