1　一階述語論理　— “理想化された推論者” たち

古典論理によれば、どんな学校のどんなクラスにも、ある生徒 x が存在し、x がテストで 100 点を取るならば、クラスの全員が 100 点を取るという。あなたは信じられるだろうか？

直観主義論理によれば、背理法のようなごく当たり前の論法すら制限されるべきだという。あなたは耐えられるだろうか？

数学には何ができて何ができないのか。その可能性と限界が知りたい。そこで我々がとる方針は、“理想化された数学者” たちを定義し、彼らの能力や性質を調べていくことだ。

本章ではその第一段階として、より一般的で汎用性のある “理想化された推論者” たちを定義する。後々で、推論者にさまざまな理論（自然数論、解析学など）の公理を追加していく。言い換えれば、推論者にさまざまな理論を “教え込んでいく。” それにより一人の数学者が誕生するのである。この一連のプロセスを**形式化** (formalization) という。

実際には、二人の推論者を定義する。一方は**古典的** (classical) であり、他方は**直観主義的** (intuitionistic) である。どちらも文を主語—述語に分解して考えるため**述語論理** (predicate logic) と呼ばれる。また、「全ての ⋯ について」「ある ⋯ が存在して」などの**量化** (quantification) は、個体についてしか行わず、性質については行わない。それゆえより正確には**一階** (first order) 述語論理と呼ばれる。結局のところ、これから定義するのは、古典一階述語論理と直観主義一階述語論理の二つである。

1.1　論理式

言語　まずは、“理想化された推論者” が推論をする際に用いる “理想化された言葉” の定義から始めよう。最も重要なのは、「かつ」「または」「ならば」など、文の論理構造を表す言葉である。これらを表すのに、以下の記号を用いる。

かつ	または	ならば	真	矛盾	すべて	存在
$\wedge$	$\vee$	$\to$	$\top$	$\bot$	$\forall$	$\exists$

しかしこれだけでは文を構成することができないので、対象や関数、述語を表す記号も必要である。具体的にどんな記号が必要になるかは各理論に依存するので、ここでは一般的な定義を与えておく。

定義 1.1 (言語) **言語** *(language)* L は以下二種類の記号の集合により定めることができる。

- **関数記号** *(function symbol)* $\{f, g, h, \dots\}$
- **述語記号** *(predicate symbol)* $\{p, q, r, \dots\}$

各記号 f や p には**項数** *(arity)* $n \in \mathbb{N}$ が定められているものとする。f や p が項数 n を持つとき、$f^{(n)}$, $p^{(n)}$ のように書く。とくに項数として $n = 0$ も許す。0 項関数記号のことを**定数記号** *(constant symbol)* といい、0 項述語記号のことを**命題記号** *(propositional symbol)* という。

以下に言語の例を二つ挙げる。

例 1.2 (算術の言語 L_0)

- 関数記号 $\{0^{(0)}, S^{(1)}, +^{(2)}, \cdot^{(2)}\}$
- 述語記号 $\{=^{(2)}\}$

$+$ や $\cdot$ は足し算、掛け算を表す記号であり、二つの引数をとるので二項関数記号である。同様に $=$ は二項述語記号である。0 は引数をとらない表現なので 0 項関数記号（つまり定数記号）である。S は x が与えられたとき $x+1$ を返す関数を表すので、引数を一つとる。ゆえに一項関数記号である。

例 1.3 (イソノ家の言語 L_1)

- 関数記号 $\{$ サザエ$^{(0)}$, カツオ$^{(0)}$, ワカメ$^{(0)}$, タラ$^{(0)}$, ナミヘイ$^{(0)}$, フネ$^{(0)}$, マスオ$^{(0)}\}$
- 述語記号 $\{$ 男$^{(1)}$, 親$^{(2)}$, 夫婦$^{(2)}$, 年上$^{(2)}$, $=^{(2)}\}$

サザエ、カツオ等は当然定数記号として考える。その他に「男である」ことを表す *1* 項述語記号、「$\cdots$ は $\cdots$ の親である」「$\cdots$ と $\cdots$ は夫婦である」「$\cdots$ は $\cdots$ より年上である」を表す *2* 項述語記号および等号記号を入れておく。

項と論理式　関数記号や述語記号とは別に**変数** (variable) の集合 $\{x, y, z, \dots\}$ を用意しておく。これらの記号を用いて “理想化された文” を構成する。

定義 1.4 (L 項, L 論理式) 言語 L が与えられたとき、L **項** *(L-term)* 全体の集合 $T(L)$ を以下のように定義する。

1. $x, y, z, \cdots \in T(L)$.
2. f が L の n 項関数記号で $t_1, \dots, t_n \in T(L)$ ならば、$f(t_1, \dots, t_n) \in T(L)$.　とくに f が定数記号ならば $f \in T(L)$.
3. $T(L)$ は *1,2* を満たす最小の集合である。

同様にして、L **論理式** *(L-formula)* 全体の集合 $F(L)$ を以下のように定義する。

1. p が L の n 項述語記号で $t_1, \dots, t_n \in T(L)$ ならば、$p(t_1, \dots, t_n) \in F(L)$.　とくに p が命題記号ならば $p \in F(L)$.
2. $\top, \bot \in F(L)$.
3. $\varphi, \psi \in F(L)$ ならば $(\varphi \wedge \psi), (\varphi \vee \psi), (\varphi \to \psi) \in F(L)$.
4. $\varphi \in F(L)$ ならば $\forall x.\varphi, \exists x.\varphi \in F(L)$.
5. $F(L)$ は *1, 2, 3, 4* を満たす最小の集合である。

以上が公式的な定義である。これによれば、L_0 項とは $+(S(x), 0)$ のような表現であり、L_0 論理式とは $\forall x.(= (+(S(x), 0), S(y)) \to \bot)$ のような表現である。しかしこれではあまりにも読みにくいので、次のような非公式的な書き方も認めることにする。

- $+(t, u)$ や $\cdot(t, u)$ と書く代わりに $(t + u)$, $(t \cdot u)$ と書く。
- $= (t, u)$ と書く代わりに $t = u$ と書く。
- $\varphi \to \bot$ と書く代わりに $\neg\varphi$ と書く。さらに φ が等式 $t = u$ のときには、$t \neq u$ とも書く。
- $((\varphi \to \psi) \wedge (\psi \to \varphi))$ と書く代わりに $(\varphi \leftrightarrow \psi)$ と書く。
- 不要なカッコは適宜省略する。

すると、上の L_0 論理式は $\forall x.S(x) + 0 \neq S(y)$ と書ける。ただし、これは我々人間にとって読みやすい簡便な書き方であるにすぎず、"理想化された推論者" はあくまでも上の公式的な定義に従って推論を進めていくことになる。

なお、L 項の定義の 3 段目で $T(L)$ は 1,2 を満たす最小の集合であると断っているのは、$T(L)$ の中には余計なものが含まれていないことを保証するためである。たとえば、$S(+)$ は 1,2 を用いたのでは構成できないので、3 により L_0 項ではないと言える。

例 1.5 算術の言語 L_0 は自然数についての性質や文を表すための言語である。まず、各自然数 $0, 1, 2, 3, \ldots$ は次のように L_0 項を用いて表すことができる。

$$0, \quad S(0), \quad S(S(0)), \quad S(S(S(0))), \quad \ldots$$

このような項を**数項** *(numeral)* とよび、自然数 $n \in \mathbb{N}$ に対応する数項を n と書く。

論理式 $\exists z.z + x = y$ は "x は y 以下である" ことを表す。この論理式を $x \leq y$ と書く。このように派生論理式を定義するときには、

$$x \leq y \quad \equiv \quad \exists z.z + x = y$$

のように書くことにする。同様にして

$$\begin{aligned}
\forall x \leq t.\varphi(x) &\equiv \forall x.(x \leq t \to \varphi(x)) \\
\exists x \leq t.\varphi(x) &\equiv \exists x.(x \leq t \wedge \varphi(x)) \\
\exists! x.\varphi(x) &\equiv \exists x(\varphi(x) \wedge \forall y(\varphi(y) \to x = y)) \\
x|y &\equiv \exists z.x \cdot z = y \\
\mathsf{even}(x) &\equiv \mathsf{2}|x \\
\mathsf{prime}(x) &\equiv \mathsf{2} \leq x \wedge \forall z \leq x(z|x \to (z = \mathsf{1} \vee z = x))
\end{aligned}$$

このようにして、複雑な概念を表す L_0 論理式を次々に定義していくことができる。これらの派生論理式を用いれば、「6 は素数ではない」ことは $\neg\mathsf{prime}(\mathsf{6})$ と書け、「素数は無限に多く存在する」ことは $\forall x.\exists y(x \leq y \wedge \mathsf{prime}(y))$ と書ける。

例 1.6 イソノ家の言語 L_1 はイソノ家の内部事情について語るための言語である。L_1 の記号を用いれば、

$$\begin{aligned} \text{女}(x) &\equiv \neg\text{男}\,(x) \\ \text{兄弟}\,(x,y) &\equiv \exists z.\text{親}\,(z,x) \wedge \text{親}\,(z,y) \\ \text{妹}\,(x,y) &\equiv \text{兄弟}\,(x,y) \wedge \text{女}\,(x) \wedge \text{年上}\,(y,\,x) \end{aligned}$$

などの概念を新たに定義することができる。また、「兄弟の兄弟は兄弟である」のように一般的な事柄も $\forall x\forall y\forall z(\text{兄弟}\,(x,y) \wedge \text{兄弟}\,(y,z) \to \text{兄弟}\,(x,z))$ と書ける。(注意：いま問題にしているのは、ある事柄が論理式を用いて書けるかどうかであり、それが真か偽かは別問題である。)

BNF 記法　定義 1.4 は、基本的な対象から始めて、より複雑な対象を作っていく仕方を記述しているものと見ることができる。このような定義を**帰納的定義** (inductive definition) という。これから先、帰納的定義を用いる機会は多々あるだろう。帰納的定義を簡潔に述べるには、**BNF 記法**を用いるのが便利である。定義 1.4 は、(ややインフォーマルな) BNF 記法を用いると以下のように簡潔に書ける。

$$\begin{array}{lrcl} L \text{ 項} & t & ::= & x \mid f(t_1,\ldots,t_n) \\ L \text{ 論理式} & \varphi,\psi & ::= & p(t_1,\ldots,t_n) \mid \top \mid \bot \mid (\varphi\wedge\psi) \mid (\varphi\vee\psi) \mid (\varphi\to\psi) \mid \forall x.\varphi \mid \exists x.\varphi. \end{array}$$

ただし x は変数、f, p はそれぞれ L の関数記号、述語記号を表し、その項数は引数の数 n と一致しているものとする。

これまでの定義でわかる通り、項、論理式の定義は言語 L に依存する。しかし具体的にどんな L を考えているのかは重要ではないときには、L 項、L 論理式というかわりに、単に項、論理式ということにする。

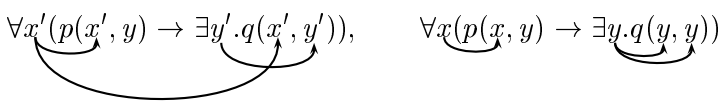
束縛変数・自由変数　変数には二通りの使い方がある。例えば論理式 $\forall x.\exists y.x = S(y) + z$ を考えると、この中には 3 つの変数 x, y, z が現れるが、そのうち x と y は、量化子 $\forall x$, $\exists y$ により使われ方が規定されている。つまり、x, y はそれぞれ「すべての x について $\cdots$」「ある y が存在して $\cdots$」という文脈において使われている。このような変数を**束縛変数** (bound variable) という。一方、z はいかなる量化子によっても使われ方が規定されていない。このよう変数を**自由変数** (free variable) という。束縛変数・自由変数については、もっと厳密な定義を与えることはいくらでもできるが、そうすると煩雑になるので、むしろ以下の諸例を通して直感的に理解してもらったほうがよい。大切なのは、論理式を見たら常に "束縛関係" を意識することである。

$$\forall x(p(x,y) \to \exists y.q(x,y))$$

束縛変数について大事な取り決めをしておく。

- 束縛変数の名前は、束縛関係を変えない限り自由に付け変えてよい。

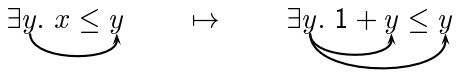
たとえば、左下の論理式は上と同じ論理式を表すが、右下は異なる論理式を表す。



束縛変数について大切なのは束縛関係であり、その関係さえ保たれていれば、x だろうが w だろうが、どんな名前をつけても構わないのである。このあたりの事情は積分をするときに $\int g(x)dx \equiv \int g(y)dy$ としてよいのと同様である。一方、自由変数についてはそのような名前のつけかえは許さないことにする。

項の代入　論理式 φ において、その中に現れる自由変数 x に着目するとき、$\varphi(x)$ のように書く。そして x に項 t を代入した結果得られる論理式を $\varphi(t)$ のように書く。例えば $\varphi(x) \equiv p(x,x,y) \wedge \forall x.q(x)$ で、$t \equiv f(x)$ のとき、$\varphi(t) \equiv p(f(x),f(x),y) \wedge \forall x.q(x)$ である。

ただし、代入を行う際には、**変数の衝突** (variable clash) に注意しなければならない。例えば $\varphi(x) \equiv \exists y.x \leq y$ は、x にどんな自然数を代入しても成り立つ。それゆえ 0 や $z+5$ を代入しても、結果として得られる論理式はやはり成り立つはずである。しかし変数 y を含む項、たとえば $1+y$ を不用意に代入すると、結果は $\exists y.1+y \leq y$ となるが、これは成り立たない。この問題の原因は、$1+y$ を代入することにより新たな束縛関係が発生してしまったことにある。



このような事態を避けるため、次の約束をしておく。

- 項を論理式に代入するときには、新たな束縛関係が生じないよう、事前に束縛変数の名前を付け替えておく。

例えば $\varphi(x)$ に $1+y$ を代入するときには、事前に束縛変数 y を新しい変数 z に名前換えし、$\varphi(x) \equiv \exists z.x \leq z$ としておいてから $1+y$ を代入するのである。結果としてえられる論理式は $\varphi(y+1) \equiv \exists z.1+y \leq z$ である。

最後に重要な定義をして本節を終えることにする。

定義 1.7 (文) 自由変数を含まない論理式を**文**という。

例えば例 1.5 における $\mathsf{even}(x)$ や $\mathsf{prime}(x)$ は文ではないが、$\forall x.\exists y(x \leq y \wedge \mathsf{prime}(y))$ は文である。

練習問題 1.8 以下の事柄を L_0 論理式を用いて表せ。

1. $\mathsf{twinprime}(x,y)$：(x,y) は双子素数である。
2. 双子素数予想：双子素数は無限に多く存在する。
3. ゴールドバッハの予想：4 以上の全ての偶数は、二つの素数の和で表すことができる。

練習問題 1.9 以下の事柄を L_1 論理式を用いて表せ。

1. ワカメは結婚していない。
2. カツオはタラの伯父である。
3. タラの父方の祖父はナミヘイではない。
4. すべての親は自分の子よりも年上である。
5. どんな兄弟も夫婦ではない。
6. いとこ同士の夫婦が存在する。

練習問題 1.10 次のことを証明せよ。言語 L において、関数記号の項数の最大値 $N = \max\{n : f^{(n)} \in L\}$ が存在するとする。t を L 項とするとき、その中に含まれる変数の個数は高々 N^k 個である。ただし k は t に含まれる関数記号の個数である。

練習問題 1.11 $\varphi(x) \equiv p(x) \wedge \forall y.q(y,x) \wedge \forall x.r(x)$、$t \equiv f(y)$ とするとき、$\varphi(t)$ を求めよ。

1.2 証明体系

次の課題は、“理想化された推論者” たちがどのようにして推論を進めていくのかを形式化することである。そのために、古典論理・直観主義論理の両者について**証明体系** (proof system) を導入する。これにはいろいろな流儀があるのだが、本講義ではゲンツェンによる**自然演繹** (natural deduction) に則って説明する。

当面の間、言語 L を固定し、L 項、L 論理式という代わりに単に項、論理式ということにする。

定義 1.12 (シークエント) 論理式の有限集合を表すのに $\Gamma, \Delta, \cdots$ などの記号を用いる。**シークエント** *(sequent)* とは $\Gamma \vdash \varphi$ の形の表現である。

シークエント $\Gamma \vdash \varphi$ は、直感的には「仮定 Γ のもとで φ が成り立つ」ことを表す。つまり論理式 φ に加えてそれが依って立つ仮定 Γ をも明示的に表したのがシークエントである。もちろんシークエントの中には正しい仮定–結論関係を表すものもあれば、そうでないものもある。正しい仮定–結論関係のみを、有限個の規則を用いて導出したい。そのために以下の**推論規則** (inference rule) を考える。今後、有限集合 $\Gamma = \{\varphi_1, \ldots, \varphi_n\}$ のことを単に $\varphi_1, \ldots, \varphi_n$ と書き、集合 Γ に要素 ψ を加えるときには、$\Gamma \cup \{\psi\}$ と書く代わりに単に Γ, ψ と書く。

$$\frac{}{\Gamma \vdash \varphi}\ (init) \quad \text{ただし} \varphi \in \Gamma \qquad \frac{\Gamma, \neg\varphi \vdash \bot}{\Gamma \vdash \varphi}\ (abs)$$

$$\frac{}{\Gamma \vdash \top}\ (\top I) \qquad \frac{\Gamma \vdash \bot}{\Gamma \vdash \varphi}\ (\bot E)$$

$$\frac{\Gamma \vdash \varphi \quad \Gamma \vdash \psi}{\Gamma \vdash \varphi \wedge \psi}\ (\wedge I) \qquad \frac{\Gamma \vdash \varphi \wedge \psi}{\Gamma \vdash \varphi}\ (\wedge E) \qquad \frac{\Gamma \vdash \varphi \wedge \psi}{\Gamma \vdash \psi}\ (\wedge E)$$

$$\frac{\Gamma \vdash \varphi}{\Gamma \vdash \varphi \vee \psi}\ (\vee I) \qquad \frac{\Gamma \vdash \psi}{\Gamma \vdash \varphi \vee \psi}\ (\vee I) \qquad \frac{\Gamma \vdash \varphi \vee \psi \quad \Gamma, \varphi \vdash \xi \quad \Gamma, \psi \vdash \xi}{\Gamma \vdash \xi}\ (\vee E)$$

$$\frac{\Gamma, \varphi \vdash \psi}{\Gamma \vdash \varphi \to \psi}\ (\to I) \qquad \frac{\Gamma \vdash \varphi \to \psi \quad \Gamma \vdash \varphi}{\Gamma \vdash \psi}\ (\to E)$$

$$\frac{\Gamma \vdash \varphi(x)}{\Gamma \vdash \forall x.\varphi(x)}\ (\forall I)^* \qquad \frac{\Gamma \vdash \forall x.\varphi(x)}{\Gamma \vdash \varphi(t)}\ (\forall E)$$

$$\frac{\Gamma \vdash \varphi(t)}{\Gamma \vdash \exists x.\varphi(x)}\ (\exists I) \quad \frac{\Gamma \vdash \exists x.\varphi(x) \quad \Gamma, \varphi(x) \vdash \xi}{\Gamma \vdash \xi}\ (\exists E)^*$$

ただし、$(\forall I)$ および $(\exists E)$ 規則は次の**固有変数条件** (eigenvariable condition) を満たすものとする:

(*) 仮定 Γ（および $(\exists E)$ の場合は結論 ξ）は x を自由変数として含まない。

各規則の基本的な読み方は、「横線の上に書いてある事柄が全て成り立てば、下に書いてある事柄も成り立つ」である。一見すると多くの規則があって大変そうだが、以下の点に注意すれば、多少は整理して理解できるだろう。

- 論理記号 $\wedge, \vee, \to, \forall, \exists$ それぞれについて導入規則（I 規則）と除去規則（E 規則）が一つずつある。ただし $(\wedge E)$ と $(\vee I)$ については、二つの規則で一つと考える。
- $(\wedge E)$ と $(\vee I)$、$(\forall E)$ と $(\exists I)$ は上下さかさまの関係にある。

各規則について説明を加えていこう。

- $(init)$: Γ の中に φ が含まれるときには、もちろん仮定 Γ のもとで φ が成り立つ。
- $(\wedge I)$: Γ を仮定しよう。もしも φ と ψ が両方とも成り立つならば、$\varphi \wedge \psi$ が成り立つ。（以下では、Γ を仮定することをいちいち断らない。）
- $(\wedge E)$: $\varphi \wedge \psi$ が成り立つならば、φ も ψ も成り立つ。$(\vee I)$ についても同様。
- $(\vee E)$: これはいわゆる場合分け論法に相当する。いま、$\varphi \vee \psi$ が成り立つとする。このとき、φ を仮定した場合に ξ が成り立ち、また ψ を仮定した場合にも ξ が成り立つならば、そのような仮定なしで ξ が成り立つと言える。
- $(\to I)$: $\varphi \to \psi$ が成り立つことを示すには、φ を仮定すれば ψ が成り立つことを示せばよい。
- $(\to E)$: これは Modus Ponens と呼ばれる最も基本的な論理推論である。$\varphi \to \psi$ と φ が成り立つならば ψ が成り立つ。$\neg\varphi \equiv \varphi \to \bot$ と定義したことを思い出せば、$(\to I)$, $(\to E)$ の特別な場合として以下の規則が導かれる：

$$\frac{\Gamma, \varphi \vdash \bot}{\Gamma \vdash \neg\varphi}\ (\to I) \qquad \frac{\Gamma \vdash \neg\varphi \quad \Gamma \vdash \varphi}{\Gamma \vdash \bot}\ (\to E)$$

その他の推論規則を説明する前に、いくつか例を挙げよう。論理推論は、推論規則を組み合わせることにより行うことができる。

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\varphi \wedge \psi \vdash \varphi \wedge \psi}\ (init)}{\varphi \wedge \psi \vdash \psi}\ (\wedge E) \quad \dfrac{\dfrac{}{\varphi \wedge \psi \vdash \varphi \wedge \psi}\ (init)}{\varphi \wedge \psi \vdash \varphi}\ (\wedge E)}{\varphi \wedge \psi \vdash \psi \wedge \varphi}\ (\wedge I)}{\vdash \varphi \wedge \psi \to \psi \wedge \varphi}\ (\to I)$$

このように推論規則を組み合わせてできる木構造を**証明図** (proof figure) あるいは単に**証明** (proof) と呼ぶ。ただし葉の部分（上端）に来てよいのは $(init)$ および $(\top I)$ 規則のみである。根の部分（下端）には証明されるべき論理式がくる。ゆえにより詳しくいえば、上の証明図は $\vdash \varphi \wedge \psi \to \psi \wedge \varphi$ **の証明図**である。ところで、この最後のシークエントは仮定を含まない（$\vdash$ の左側が空っぽである）特別な形をしている。こういうときには $\vdash$ は無視して、上の証明図は $\varphi \wedge \psi \to \psi \wedge \varphi$ の証明図であるともいう。

もう一つの例として、次のものを挙げておく。ただし $\Gamma = \{\neg\varphi \wedge \neg\psi, \varphi \vee \psi\}$ である。

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash \varphi \vee \psi}\,(init) \quad \dfrac{\dfrac{\dfrac{}{\Gamma, \varphi \vdash \neg\varphi \wedge \neg\psi}\,(init)}{\Gamma, \varphi \vdash \neg\varphi}\,(\wedge E) \quad \dfrac{}{\Gamma, \varphi \vdash \varphi}\,(init)}{\Gamma, \varphi \vdash \bot}\,(\to E) \quad \dfrac{\dfrac{\dfrac{}{\Gamma, \psi \vdash \neg\varphi \wedge \neg\psi}\,(init)}{\Gamma, \psi \vdash \neg\psi}\,(\wedge E) \quad \dfrac{}{\Gamma, \psi \vdash \psi}\,(init)}{\Gamma, \psi \vdash \bot}\,(\to E)}{\Gamma \vdash \bot}\,(\vee E)}{\neg\varphi \wedge \neg\psi \vdash \neg(\varphi \vee \psi)}\,(\to I)}{\vdash \neg\varphi \wedge \neg\psi \to \neg(\varphi \vee \psi)}\,(\to I)$$

次に量化子に関わる推論規則について説明する。

- $(\forall I)$: $\forall x.\varphi(x)$ が成り立つことを示すには、任意の x について $\varphi(x)$ が成り立つことを示せばよい。このとき x は任意の対象でなければならず、余計な条件がついていてはならない。たとえば「全ての三角形は内角の和が 180 度である」ということを示すには、任意の三角形 x をとってきて、その内角の和が 180 度であることを示せばよいのだが、このときとってくる x はあくまでも任意の三角形でなければならない（正三角形や直角三角形などの特別な三角形を念頭に置いてはいけない）。
- $(\forall E)$, $(\exists I)$ の意味は明らかだろう。
- $(\exists E)$: 論証の過程で $\exists x.\varphi(x)$、つまり性質 φ を満たす対象の存在が明らかになったとする。このとき、そのような対象に仮に x と名前をつけて論証を続けることができる。ただし x について仮定してよいのは、それが $\varphi(x)$ を満たすということのみであり、それ以外に余計なことを仮定してはいけない。また、x というのは一時的な仮の名前にすぎないから、論証の結論に出てきてはいけない。これが固有変数条件の意味である。

これらの規則を用いると、たとえば次のような証明図が書ける。ただし $\Gamma = \{\neg\exists x.\varphi(x), \varphi(x)\}$ とする。

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash \neg\exists x.\varphi(x)}\,(init) \quad \dfrac{\dfrac{}{\Gamma \vdash \varphi(x)}\,(init)}{\Gamma \vdash \exists x.\varphi(x)}\,(\exists I)}{\Gamma \vdash \bot}\,(\to E)}{\neg\exists x.\varphi(x) \vdash \neg\varphi(x)}\,(\to I)}{\neg\exists x.\varphi(x) \vdash \forall x.\neg\varphi(x)}\,(\forall I)}{\vdash \neg\exists x.\varphi(x) \to \forall x.\neg\varphi(x)}\,(\to I)$$

ここで $(\forall I)$ 規則を使う際に、確かに固有変数条件が満たされていることを確認してほしい。もしも固有変数条件がなかったら、次のような推論が出来てしまう。

$$\dfrac{\dfrac{\dfrac{}{\exists x.\varphi(x) \vdash \exists x.\varphi(x)}\,(init) \quad \dfrac{\dfrac{}{\exists x.\varphi(x), \varphi(x) \vdash \varphi(x)}\,(init)}{\exists x.\varphi(x), \varphi(x) \vdash \forall x.\varphi(x)}\,(\forall I)}{\exists x.\varphi(x) \vdash \forall x.\varphi(x)}\,(\exists E)}{\vdash \exists x.\varphi(x) \to \forall x.\varphi(x)}\,(\to I)$$

しかし $\varphi(x)$ であるような x が存在するならば全ての x について $\varphi(x)$ であるというのは、明らかに不合理である。原因は、上で $(\forall I)$ 規則を使用する際に固有変数条件が守られていなかったことにある。

- $(\bot E)$: これは仮定 Γ のもとで矛盾が生じたなら、同じ仮定のもとで何でも成り立つことを表す。実際の論証においては、場合分け論法などとの組み合わせにより効果を発揮する。たとえば $\Gamma = \{\varphi \vee \psi, \neg\varphi\}$ として

$$\dfrac{\dfrac{}{\Gamma \vdash \varphi \vee \psi}\ (init) \qquad \dfrac{\dfrac{\dfrac{}{\Gamma, \varphi \vdash \neg\varphi}\ (init) \quad \dfrac{}{\Gamma, \varphi \vdash \varphi}\ (init)}{\Gamma, \varphi \vdash \bot}\ (\to E)}{\Gamma, \varphi \vdash \psi}\ (\bot E) \qquad \dfrac{}{\Gamma, \psi \vdash \psi}\ (init)}{\Gamma \vdash \psi}\ (\vee E)$$

  ゆえに $\varphi \vee \psi$、$\neg\varphi$ という仮定のもとで ψ が成り立つ。

- (abs): これはいわゆる背理法に相当する。φ が成り立つことを示すには、$\neg\varphi$ を仮定して矛盾を導けばよい。この規則を $\neg\varphi$ に関する $(\to I)$ の特別な場合と混同ないように注意。この背理法を認めるか否かで、古典論理と直観主義論理の二つに分かれる。

さて、これでようやく二人の “理想化された推論者” を設定することができる。上に挙げた推論規則全てを用いることのできる推論者を古典論理（正確には古典一階述語論理）とよび **CL** により表す。また (abs) を用いるのを拒否する推論者を直観主義論理とよび **IL** により表す。より正確には、両者の証明能力を表す論理的帰結関係 $\vdash_{\mathbf{CL}}$ および $\vdash_{\mathbf{IL}}$ を次のように定義する（$\vdash$ と紛らわしいが、慣用に従うことにする）。

定義 1.13 (証明可能性) Φ を論理式の集合とする（有限集合でなくてもよい）。ある有限部分集合 $\Gamma \subseteq \Phi$ が存在し、$\Gamma \vdash \varphi$ の証明図が存在するとき、$\Phi \vdash \varphi$ は**古典論理で証明可能** *(provable in classical logic)* であるといい、$\Phi \vdash_{\mathbf{CL}} \varphi$ と書く。特に $\Phi = \emptyset$ のときには単に φ は古典論理で証明可能であるといい、$\vdash_{\mathbf{CL}} \varphi$ と書く。

直観主義論理で証明可能 *(provable in intuitionistic logic)* であることを表す表現 $\Phi \vdash_{\mathbf{IL}} \varphi$ も同様に定義する。ただしその際には、証明図において (abs) 規則を用いてはならない。

明らかに $\Phi \vdash_{\mathbf{IL}} \varphi$ ならば $\Phi \vdash_{\mathbf{CL}} \varphi$ であるが、逆は一般には成り立たない。古典論理では証明可能であるが直観主義論理では証明不可能な論理式の例をいくつか挙げておく。

1. 二重否定の除去則：$\neg\neg\varphi \leftrightarrow \varphi$. 一方向 $\varphi \to \neg\neg\varphi$ は直観主義論理で証明可能であるが、逆方向を示すには、一般には背理法 (abs) が必要である。

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\neg\neg\varphi, \neg\varphi \vdash \neg\neg\varphi}\ (init) \quad \dfrac{}{\neg\neg\varphi, \neg\varphi \vdash \neg\varphi}\ (init)}{\neg\neg\varphi, \neg\varphi \vdash \bot}\ (\to E)}{\neg\neg\varphi \vdash \varphi}\ (abs)}{\vdash \neg\neg\varphi \to \varphi}\ (\to I)$$

2. 排中律：$\varphi \vee \neg\varphi$.

3. ド・モルガンの法則：$\neg(\varphi \wedge \psi) \leftrightarrow \neg\varphi \vee \neg\psi$, $\qquad \neg\exists x.\varphi(x) \leftrightarrow \forall x.\neg\varphi(x)$.

我々が普段当たり前に用いているこれらの論理法則が使えないというのは、さぞかし不便だろうと思われるに違いない。実際その通りなのだが、だからといって直観主義論理が古典論理よりも制限された論理であるといったり、表現能力の劣る論理であるというのは、必ずしも実情を反映していない。なぜなら **CL** は翻訳を通して **IL** に埋め込むことができるからである。このあたりの事情については後ほど説明する。

一方、古典論理にも問題がある。直観主義論理の証明能力が弱すぎるように見えるのに対して、古典論理の証明能力は強すぎるように見えてしまうのである。例えば、一見不合理にみえる次のような論理式が古典論理では証明可能である。

4. ダメットの法則：$(\varphi \to \psi) \vee (\psi \to \varphi)$

5. 酒場の法則：$\exists x.(\varphi(x) \to \forall y.\varphi(y))$

いま、$\varphi \equiv$「双子素数予想は正しい」、$\psi \equiv$「AKB が解散する」としよう。ダメットの法則によれば、「双子素数予想が正しいならば AKB は解散する、または AKB が解散するならば双子素数予想は正しい」ことが論理的に成り立つというのである。しかし AKB と双子素数予想の間に一体どんな関係があるというのだろうか。

また酒場の法則 (drinker's formula) によれば、どんな酒場にも必ずキーパーソン x が存在し、x が酒を飲んでいるならば、その場の全員が酒を飲んでいるというのである。また、冒頭で述べたように、どんな学校のどんなクラスにも必ずキーパーソン x が存在し、x が 100 点をとるならば、クラスの全員が 100 点をとるというのである。これが論理的に成り立つというのは、あたかもカンニングが常態化しているかのようで、不気味である。

このように古典論理で証明可能な法則と、論理推論についての日常的な感覚の間には多少のギャップがある。このことから、より日常感覚に近い論理体系を求める非古典論理研究の派閥もあるが、本講義では立ち入らない。日常的に自然な論理を求めると、数学的に不自然な論理が出来上がることが多いからである。

練習問題 1.14 次の論理式が直観主義論理で証明可能なことを示せ。

1. $((\varphi \wedge \psi) \to \xi) \leftrightarrow (\varphi \to (\psi \to \xi))$
2. $\varphi \wedge (\psi \vee \xi) \leftrightarrow (\varphi \wedge \psi) \vee (\varphi \wedge \xi)$
3. $\neg\varphi \vee \neg\psi \to \neg(\varphi \wedge \psi)$
4. $\exists x.\neg\varphi(x) \to \neg\forall x.\varphi(x)$
5. $\forall x(\varphi(x) \to \varphi(f(x))) \to \forall x(\varphi(x) \to \varphi(f(f(x))))$

練習問題 1.15 次の論理式が古典論理で証明可能なことを示せ。

1. $\neg(\varphi \wedge \psi) \to \neg\varphi \vee \neg\psi$
2. $\neg\forall x.\varphi(x) \to \exists x.\neg\varphi(x)$
3. 排中立
4. ダメットの法則
5. 酒場の法則

等号について　一般に一階述語論理では、等号 $=$ について特別な取り扱いをすることが多い。言語 L に二項述語記号 $=^{(2)}$ が含まれるときには、次の推論規則を考えることにする。

$$\frac{}{\Gamma \vdash t = t}\ (eq1) \qquad \frac{\Gamma \vdash t_1 = u_1 \quad \cdots \quad \Gamma \vdash t_n = u_n}{\Gamma \vdash f(t_1, \ldots, t_n) = f(u_1, \ldots, u_n)}\ (eq2)$$

$$\frac{\Gamma \vdash t_1 = u_1 \quad \cdots \quad \Gamma \vdash t_n = u_n \quad \Gamma \vdash p(t_1, \ldots, t_n)}{\Gamma \vdash p(u_1, \ldots, u_n)}\ (eq3)$$

特に (eq3) で p として $=$ をとれば、

$$\frac{\Gamma \vdash t_1 = u_1 \quad \Gamma \vdash t_2 = u_2 \quad \Gamma \vdash t_1 = t_2}{\Gamma \vdash u_1 = u_2}\ (eq3)$$

となる。ここで $t_1 \equiv t_2 \equiv u_2 \equiv t$, $u_1 \equiv u$ とおけば、次の証明図が得られる。

$$\dfrac{\dfrac{\dfrac{}{t = u \vdash t = u}\ (init) \quad \dfrac{}{t = u \vdash t = t}\ (eq1) \quad \dfrac{}{t = u \vdash t = t}\ (eq1)}{t = u \vdash u = t}\ (eq3)}{\vdash t = u \to u = t}\ (\to I)$$

つまり、$\vdash_{\mathbf{IL}} t = u \to u = t$ である。

練習問題 1.16　次の論理式が **IL** で証明可能であることを示せ。

1. $t = u \wedge u = v \to t = v$
2. $t = u \wedge \varphi(t) \to \varphi(u)$. ただし $\varphi(x)$ は任意の論理式とする。（やや難：証明は φ の構成についての帰納法による。）

理論　“理想化された数学者” の構築は、“理想化された推論者” に個別理論を “教え込む” ことにより完結する。散文的な言い方をすれば、これは **CL** や **IL** にいくつかの公理を付け加えることに相当する。

この最後のステップについては、算術の理論を例にとって後ほど詳しく解説するが、先の見通しをよくするために、ここで理論一般の実現の仕方について少しだけ触れておく。話を簡単にするため、当面は古典論理 **CL** をベースとして話を進めていく。直観主義論理を考える場合にはいちいち断ることにする。

定義 1.17 (理論) L を言語とする。L 文の集合 Φ のことを L **理論** *(L-theory)* あるいは単に理論と呼ぶ。**公理系** *(axiomatic system)* ということもある。

$\Phi \vdash_{\mathbf{CL}} \varphi$ が成り立つとき、φ は理論 Φ の**定理** *(theorem)* であるといい、$\vdash_\Phi \varphi$ と書く。

理論 Φ が**矛盾する** *(inconsistent)* のは $\vdash_\Phi \bot$ が成り立つときである。さもなければ Φ は**無矛盾である** *(consistent)*。

例 1.18　算術の最も基本的な理論 **Q** は以下の論理式の全称閉包からなる。

$$\begin{array}{lll} S(x) \neq 0 & S(x) = S(y) \to x = y & x \neq 0 \to \exists y(x = S(y)) \\ x + 0 = x & x + S(y) = S(x + y) & \\ x \cdot 0 = 0 & x \cdot S(y) = x \cdot y + x & \end{array}$$

ここで全称閉包とは自由変数をすべて$\forall$で束縛したもののことである。たとえば、$x+S(y)=S(x+y)$の全称閉包は$\forall x\forall y(x+S(y)=S(x+y))$である。

$Q\vdash_{\mathbf{CL}}1+2=2+1$となることは容易に確かめることができる。ゆえに$\vdash_{\mathbf{Q}}1+2=2+1$であり、$1+2=2+1$は$\mathbf{Q}$の定理である（“数学者$\mathbf{Q}$は$1+2=2+1$を証明できる”）。

一方で、$\vdash_{\mathbf{Q}}\forall x\forall y(x+y=y+x)$は成り立たない（“数学者$\mathbf{Q}$は加法の可換性を証明できない”）。

このように$\mathbf{Q}$は足し算に関するもっとも基本的な事実すら証明できない。そこで公理を付け加えて$\mathbf{Q}$を拡張する必要性が出てくる。最も一般的なのは、数学的帰納法の公理を付け加えることである。これについては後ほど詳述する。

例 1.19 群の言語L_2は関数記号$\{e^{(0)},inv^{(1)},\circ^{(2)}\}$および述語記号$\{=^{(2)}\}$からなる。以下、$inv(t)$と書くかわりに$t^{-1}$と書き、$t\circ u$と書くかわりに$tu$と書く。

群の公理系Gは次の論理式の全称閉包からなる：

$$\begin{array}{rclcrcl} x(yz) & = & (xy)z & & & & \\ xe & = & x & & ex & = & x \\ xx^{-1} & = & e & & x^{-1}x & = & e \end{array}$$

すると、たとえば$tu=v\to t=vu^{-1}$はGの定理となる。

しかしこのような路線で群論を展開すること、すなわち$\vdash_G\varphi$という形で群論の定理を示していくことは不可能である。なぜならばL_2文を用いたのでは、群の要素については語ることはできても、群そのものや、部分群について語ることはできず、正規部分群による商群の構成など、もっとも基本的な群の構成法すら記述できないからである。

最大の原因は、ベース論理が**一階**述語論理であることによる。一階述語論理では、「個体」について語ることはできても「個体の集合」については十分に語ることができないので、群そのものや部分群自体を理論の対象として捉えることができないのである。群論を十分に展開するためには、ZF集合論の内部で群を集合論的に構成するか（これなら一階述語論理ベースで話を進められる）、あるいは個体の集合について語ることのできる**二階**述語論理をベースとして理論展開を行う等の方策をとる必要がある。

1.3 文の真偽

これまでは、与えられた文（より一般には論理式）が証明できるか否かという**推論**の問題を扱ってきたが、ここでやや別の視点に立って、与えられた文が真かどうかという**解釈**の問題を考える。前者が証明論的な発想だとすれば、後者はモデル論的な発想である。発想が根本的に異なるので、一度前節の内容はリセットして考えるとよいだろう。

構造　言語L_0における文$\varphi\equiv\neg\exists x.0=S(x)$を考える。これは「$0=S(x)$であるような$x$は存在しない」ことを意味するのだが、果たしてこの文は真だろうか、偽だろうか？もしもxの変域が自然数全体の集合$\mathbb{N}$であり、かつ$S(x)$が$x+1$を意味するならば、φは真である。しかし、xの変域が-1を含むならばφは偽となる。また、$S(x)$が何か全然別の関数を表すときにもやっぱり偽となりうる。このことからわかるように、

- 文の真偽は変数の変域と各記号の解釈に依存する。

要は文の真偽はそれ単独で決まるのではなく、状況に依存するのである。そのような状況を記述するのが、次で定義する構造である。

定義 1.20 (構造) L を言語とする。L **構造** *(L-structure)*$\mathbf{A}$ は以下の構成要素からなる。

1. 空でない集合 A（変数の変域、**個体領域** *(individual domain)* などとも呼ばれる*)*
2. 各関数記号 $f^{(n)} \in L$ について、A 上の n 項関数 $f_{\mathbf{A}} : A^n \longrightarrow A$. とくに各定数記号 $f^{(0)} \in L$ について、A の要素 $f_{\mathbf{A}} \in A$.
3. 各述語記号 $p^{(n)} \in L$ について、A 上の n 項関数 $p_{\mathbf{A}} : A^n \longrightarrow \{\mathsf{T}, \mathsf{F}\}$. とくに各命題記号 $p^{(0)} \in L$ について、$p_{\mathbf{A}} \in \{\mathsf{T}, \mathsf{F}\}$.

ただし L に等号 $=$ が含まれる場合には、$=_{\mathbf{A}}$ は常に次の関数を表すものとする。

$$\begin{aligned} =_{\mathbf{A}}(a, b) &= \mathsf{T} \quad (a = b \text{ のとき}) \\ &= \mathsf{F} \quad (a \neq b \text{ のとき}) \end{aligned}$$

ここで T, F はそれぞれ**真** (true), **偽** (false) を表す文字である。

例 1.21 L_0 構造 $\mathbf{N}$ を次のように定義する。

1. 個体領域は $\mathbb{N}$.
2. $0_{\mathbf{N}} = 0$ （同じ文字を使っているが、左の *0* は定数記号、右の *0* は自然数であることに注意）
3. $+_{\mathbf{N}}, \cdot_{\mathbf{N}}$ はそれぞれ自然数上の足し算、掛け算.
4. $S_{\mathbf{N}}$ は $S_{\mathbf{N}}(n) = n + 1$ により定義される関数.

これは L_0 の意図通りの解釈であり、**算術の標準モデル** *(standard model of arithmetic)* と呼ばれる。他方で、意図に反する構造も多々存在する。中でも**算術の超準モデル** *(nonstandard model of arithmetic)* と呼ばれる諸構造は重要な役割を果たすが、本講義では取り上げない。

論理式の真理値　論理式の真偽を定めるには、さらに自由変数に値を割り当てる必要がある。

定義 1.22 (付値) $\mathbf{A}$ を L 構造とする。このとき各変数 x に A の要素 $v(x)$ を割り当てる関数 v を $\mathbf{A}$ 上の**付値** *(valuation)* という。

付値 v が与えられると、各項 t の値 $v(t) \in A$ は次の式により帰納的に定めることができる。

$$v(f(t_1, \ldots, t_n)) \quad = \quad f_{\mathbf{A}}(v(t_1), \ldots, v(t_n)).$$

例えば、標準モデル $\mathbf{N}$ 上の付値を $v(x) = 3, v(y) = 5$ により定義する。すると、

$$\begin{aligned} v(S(x) + y) &= +_{\mathbf{N}}(v(S(x)), v(y)) \\ &= +_{\mathbf{N}}(S_{\mathbf{N}}(v(x)), 5) \\ &= +_{\mathbf{N}}(S_{\mathbf{N}}(3), 5) \\ &= +_{\mathbf{N}}(4, 5) \\ &= 9. \end{aligned}$$

定義 1.23 (論理式の真理値) $\mathbf{A}$ を L 構造、v を $\mathbf{A}$ 上の付値とする。このとき L 論理式 φ の**真理値** *(truth value)* $v(\varphi) \in \{\mathsf{T}, \mathsf{F}\}$ を次のように帰納的に定義する。

- $v(p(t_1, \ldots, t_n)) = p_{\mathbf{A}}(v(t_1), \ldots, v(t_n))$.
- $v(\top) = \mathsf{T}$, $v(\bot) = \mathsf{F}$.
- $\varphi \wedge \psi, \varphi \vee \psi, \varphi \to \psi$ の真理値は次の真理値表で定める。

$v(\varphi)$	$v(\psi)$	$v(\varphi \wedge \psi)$	$v(\varphi \vee \psi)$	$v(\varphi \to \psi)$
T	T	T	T	T
T	F	F	T	F
F	T	F	T	T
F	F	F	F	T

- $v(\forall x.\varphi) = \mathsf{T} \iff$ すべての $a \in A$ について $v[x \mapsto a](\varphi) = \mathsf{T}$.
- $v(\exists x.\varphi) = \mathsf{T} \iff$ ある $a \in A$ について $v[x \mapsto a](\varphi) = \mathsf{T}$.

ここで $v[x \mapsto a]$ は $v[x \mapsto a](x) = a$ であること以外は v と変わらない付値関数を表す。

上の定義によれば、$v(\varphi \to \psi) = \mathsf{F}$ となるのは、$v(\varphi) = \mathsf{T}$ かつ $v(\psi) = \mathsf{F}$ となるときに限る（他の場合はすべて $v(\varphi \to \psi) = \mathsf{T}$ である）。これは「“φ ならば ψ” ということを否定しようと思ったら、φ が成り立つのに ψ は成り立たないということを示せばよい」という数学における「ならば」の用法に（ある程度）合致する。また、

$$v(\neg\varphi) = \mathsf{T} \iff v(\varphi) = \mathsf{F}$$

となることもわかる。

明らかに、φ が文のとき（自由変数を含まないとき）は、$v(\varphi)$ の値は付値 v の選び方によらない。

次に、構造に関する標準的な用語をいくつか定義する。

定義 1.24 (妥当、論理的帰結、モデル) φ を L 論理式、Φ を L 論理式の集合とする。

1. $\mathbf{A}$ を L 構造とする。$\mathbf{A}$ 上のすべての付値 v について $v(\varphi) = \mathsf{T}$ となるとき、$\mathbf{A} \models \varphi$ と書く。
2. すべての L 構造 $\mathbf{A}$ について $\mathbf{A} \models \varphi$ となるとき、φ は**妥当** *(valid)* であるといい $\models \varphi$ と書く。

3. すべての L 構造 $\mathbf{A}$、$\mathbf{A}$ 上のすべての付値 v について、

$$\text{すべての } \psi \in \Phi \text{ について } v(\psi) = \mathsf{T} \quad \Longrightarrow \quad v(\varphi) = \mathsf{T}$$

となるとき、φ は Φ の**論理的帰結** *(logical consequence)* であるといい、$\Phi \models \varphi$ と書く。

4. $\mathbf{A}$ を L 構造とする。すべての $\psi \in \Phi$ について $\mathbf{A} \models \psi$ が成り立つとき、$\mathbf{A}$ を Φ の**モデル** *(model)* という。

例 1.25

1. 標準モデル $\mathbf{N}$ は理論 $\mathbf{Q}$（例 *1.18*）のモデルである。

2. $\mathbf{A}$ を公理系 G（例 *1.19*）のモデルとする。すると $\mathbf{A}$ は $(A, \circ_{\mathbf{A}}, inv_{\mathbf{A}}, e_{\mathbf{A}})$ と表すことができ、この構造が公理系 G を満たすわけである。これは $\mathbf{A}$ が群であるということに他ならない。同様に環や体の公理系のモデルとは、環や体そのもののことである。

練習問題 1.26 ダメットの法則、酒場の法則が妥当であることを確かめよ。

1.4 古典一階述語論理の完全性

1.2 では推論規則を通して古典一階述語論理を導入し、1.3 では構造上での解釈を通して論理式の真偽を定めた。本節ではこの二つの立場を関係づける完全性定理について説明する。まず重要なのは、次の補題である。これは古典一階述語論理の**健全性** (soundness) とよばれる。

補題 1.27 $\Gamma \vdash \varphi$ をシークエントとする。$\Gamma \vdash_{\mathbf{CL}} \varphi$ ならば $\Gamma \models \varphi$ である。
特別な場合として、φ が古典論理で証明可能ならば、φ は妥当である。

証明 $\Gamma \vdash_{\mathbf{CL}} \varphi$ というのは、つまり $\Gamma \vdash \varphi$ の証明図が存在するということである。証明図は推論規則を組み合わせてできているので、次のことを示せば十分である（正確な証明は証明図の構成に関する帰納法による）。

- $(init)$, $(\top I)$ 規則に現れるシークエントは論理的帰結関係 $\Gamma \models \varphi$ を満たす。
- その他の規則については、上に現れるシークエントが論理的帰結関係を満たすならば、下に現れるシークエントも論理的帰結関係を満たす。

いくつかの場合についてこのことを確かめておこう。$\mathbf{A}$ を任意の構造とする。

(1) $\dfrac{}{\Gamma \vdash \varphi}\ (init)$ ただし$\varphi \in \Gamma$

v を $\mathbf{A}$ 上の任意の付値とし、全ての $\chi \in \Gamma$ について $v(\chi) = \mathsf{T}$ と仮定する。すると $\varphi \in \Gamma$ であるから、当然 $v(\varphi) = \mathsf{T}$ である。ゆえに $\Gamma \models \varphi$.

(2) $$\frac{\Gamma \vdash \varphi \vee \psi \quad \Gamma, \varphi \vdash \xi \quad \Gamma, \psi \vdash \xi}{\Gamma \vdash \xi}\ (\vee E)$$

$\Gamma \models \varphi \vee \psi$ と $\Gamma, \varphi \models \xi$ と $\Gamma, \psi \models \xi$ を仮定する。我々の目的は $\Gamma \models \xi$ を示すことである。

そこで υ を $\mathbf{A}$ 上の任意の付値とし、全ての $\chi \in \Gamma$ について $\upsilon(\chi) = \mathsf{T}$ とする（以後このことを省略して $\upsilon(\Gamma) = \mathsf{T}$ と書くことにする）。すると一番目の仮定より $\upsilon(\varphi \vee \psi) = \mathsf{T}$ であり、真理値表より $\upsilon(\varphi) = \mathsf{T}$ または $\upsilon(\psi) = \mathsf{T}$ のどちらかである。$\upsilon(\varphi) = \mathsf{T}$ が成り立つときには、二番目の仮定より $\upsilon(\xi) = \mathsf{T}$ である。$\upsilon(\psi) = \mathsf{T}$ が成り立つときには、三番目の仮定より $\upsilon(\xi) = \mathsf{T}$ である。いずれにせよ $\upsilon(\xi) = \mathsf{T}$ であり、$\Gamma \models \xi$ が示せた。

(3) $$\frac{\Gamma, \varphi \vdash \psi}{\Gamma \vdash \varphi \to \psi}\ (\to I)$$

$\Gamma, \varphi \models \psi$ を仮定する。我々の目的は $\Gamma \models \varphi \to \psi$ を示すことである。

そこで υ を $\mathbf{A}$ 上の任意の付値とし、$\upsilon(\Gamma) = \mathsf{T}$ とする。$\upsilon(\varphi \to \psi) = \mathsf{T}$ を示すには、$\upsilon(\varphi) = \mathsf{T}$ を仮定して $\upsilon(\psi) = \mathsf{T}$ を示せばよいが、これは $\Gamma, \varphi \models \psi$ から直接に帰結する。

(4) $$\frac{\Gamma \vdash \varphi(x)}{\Gamma \vdash \forall x.\varphi(x)}\ (\forall I)$$

$\Gamma \models \varphi(x)$ を仮定する。また、υ を $\mathbf{A}$ 上の任意の付値とし、$\upsilon(\Gamma) = \mathsf{T}$ とする。定義より、$\upsilon(\forall x.\varphi(x)) = \mathsf{T}$ を示すには、任意の $a \in A$ について $\upsilon[x \mapsto a](\varphi(x)) = \mathsf{T}$ を示せばよい。固有変数条件により変数 x は Γ の中で自由変数としては用いられていないので、$\upsilon(\Gamma)$ の値は $\upsilon(x)$ の値に関係なく定まる。ゆえに $\upsilon[x \mapsto a](\Gamma) = \upsilon(\Gamma) = \mathsf{T}$ である。よって仮定 $\Gamma \models \varphi(x)$ より $\upsilon[x \mapsto a](\varphi(x)) = \mathsf{T}$ であり、$\Gamma \models \forall x.\varphi(x)$ が示せた。

(5) $$\frac{\Gamma \vdash \forall x.\varphi(x)}{\Gamma \vdash \varphi(t)}\ (\forall E)$$

$\Gamma \models \forall x.\varphi(x)$ を仮定する。また、υ を $\mathbf{A}$ 上の任意の付値とし、$\upsilon(\Gamma) = \mathsf{T}$ とする。仮定より、任意の $a \in A$ について $\upsilon[x \mapsto a](\varphi(x)) = \mathsf{T}$ である。特に $\upsilon(t) \in A$ であるから $\upsilon[x \mapsto \upsilon(t)](\varphi(x)) = \mathsf{T}$ であるが、これは $\upsilon(\varphi(t)) = \mathsf{T}$ を意味する（実際にはこのことを示すのには多少の論証が必要である）。ゆえに $\Gamma \models \varphi(t)$ が示せた。 ■

練習問題 1.28 その他の規則についても同様の確認を行うことにより、補題 *1.27* の証明を補完せよ。

上の補題は、何らかの文が古典論理で**証明できない**ことを示すのに有効である。たとえば文 $\varphi \equiv \forall x(p(x) \to q(x)) \to \forall x(q(x) \to p(x))$ を考える。この文は明らかに論理的に正しくないが、そのことを正確に示すために、次のような構造 $\mathbf{A}$ を考える。

- 個体領域 $A = \{a\}$.
- $p_{\mathbf{A}}(a) = \mathsf{F}$, $q_{\mathbf{A}}(a) = \mathsf{T}$.

すると $\mathbf{A} \not\models \varphi$ となることは容易に確かめることができる。ゆえに φ は妥当ではなく、補題 1.27 の対偶により文 φ は古典論理では証明できないことが言える。このようなモデル $\mathbf{A}$ を φ の**反例モデル** (countermodel) という。

練習問題 1.29 次の文の反例モデルを作れ。

1. $\forall x(p(x) \vee q(x)) \to (\forall x.p(x)) \vee (\forall x.q(x))$
2. $\forall x \exists y.p(x,y) \to \exists y \forall x.p(x,y)$

このように、反例モデルを構築するのは文の証明不可能性を示すための有効な手段である。しかし、このことは常に可能なのだろうか？すなわち、証明不可能な文には必ず反例モデルが存在すると言えるのだろうか？答えは“ある意味で”イエスである。そのことを保証するのが、ゲーデルが 1929 年に学位論文で証明した古典一階述語論理の**完全性定理** (completeness theorem) である。

定理 1.30 (一階述語論理の完全性) L を言語、φ を L 文、Φ を L 文の集合とする。このとき、

$$\Phi \vdash_{\mathbf{CL}} \varphi \quad \Longleftrightarrow \quad \Phi \models \varphi.$$

ここでは証明は与えずに、いくつかコメントするにとどめる。

- この定理の“$\Longrightarrow$”方向は、補題 1.27 から直接的に帰結する。問題は逆方向である。話を簡単にするために $\Phi = \emptyset$ とすると、$\vdash_{\mathbf{CL}} \varphi$ が成り立たないときに φ の反例モデルが存在することを言えばよいのだが、それをどうやって構築するかが問題なのである。一つのアイデアは、シークエント $\vdash \varphi$ に推論規則を下から上に（“ボトムアップに”）次々と適用していくことにより、$\vdash \varphi$ の証明図を構築しようと試みることである（ボトムアップ証明探索)。もちろんそのような試みは失敗するわけだが、なぜ失敗したのかをよくよく分析してみると、反例モデルが作れるのである。失敗を反省することは役に立つ、というのがこの論法からえられる教訓である。

- 上で、「答えは“ある意味で”イエスである」と限定をつけた。これは二つの理由による。第一に、どんな証明不可能な文にも反例モデルが存在するというのは確かなのだが、その反例モデルは無限構造である（個体領域が無限集合である）こともありうる。つまり、反例モデルが存在することは確かなのだが、だからといって我々人間が常に有限の時間内に見つけられるということにはならない。このことは後ほど説明する一階述語論理の決定不能性と密接に関わってくる。

- 第二の理由は、後ほど説明するゲーデルの不完全性定理に関係する。たとえば算術の文 $1 = 2$ を考えると、これは明らかに標準モデル $\mathbf{N}$ において偽である。つまり $\mathbf{N}$ は $1 = 2$ の反例モデルである。もしもこのことが一般化できて、算術の理論で証明不可能な文すべてについて $\mathbf{N}$ が反例モデルとなってくれるならば大変ありがたいのだが、そうはうまくいかない。上の完全性定理により得られる反例モデルは、標準モデルになるとは限らないのである（いわゆる超準モデルになる)。これが不完全性定理の意味である。

- 具体的な文、たとえば $\varphi \equiv \exists x(p(x) \to \forall y.p(y))$（酒場の法則）が与えられたとき、我々はそれが妥当であることを $\models \varphi$ の定義に従って確かめることができる。練習問題 1.26 をやっていただいた方にはわかってもらえると思うが、その際の論証はあく

までの “我々” が行う論証である。他方で、$\vdash_{\mathbf{CL}} \varphi$ も成り立つ（練習問題 1.15）。後者が示しているのは、“理想化された推論者” たる **CL** が、彼（女）の推論規則を用いて φ を論証できるということである。完全性定理が述べているのは、ある意味でこの二つの論証（我々の論証と **CL** の論証）のポテンシャルは等しいということである。つまり、（適切な仮定のもとで）我々の推論能力と **CL** の推論能力は一致するのである。

完全性定理の帰結として次の系が成り立つ。実際には完全性定理と同値なので、下の系のことを完全性定理と呼ぶことも多い。

系 1.31 Φ を理論（すなわち文の集合）とすると、

$$\Phi \text{ は無矛盾（つまり } \Phi \nvdash_{\mathbf{CL}} \bot\text{）} \quad \Longleftrightarrow \quad \Phi \text{ はモデルを持つ。}$$

証明 完全性定理により、$\Phi \nvdash_{\mathbf{CL}} \bot \Longleftrightarrow \Phi \not\models \bot$. 後者が意味するのは、ある構造 $\mathbf{A}$ が存在して（どんな付値 v についても）、$v(\Phi) = \mathsf{T}$（すなわちすべての $\chi \in \Phi$ について $v(\chi) = \mathsf{T}$）かつ $v(\bot) = \mathsf{F}$ となるということである。そのような $\mathbf{A}$ は Φ のモデルに他ならない。 ■

1.5 双対原理と標準形

双対原理 まずは、古典論理について成り立つ諸法則のうち、量化子を含まないものについてまとめておこう。

含意・二重否定除去	$(\varphi \to \psi)$	$\leftrightarrow$	$\neg\varphi \vee \psi$	$\neg\neg\varphi$	$\leftrightarrow$	φ
結合性	$(\varphi \wedge \psi) \wedge \xi$	$\leftrightarrow$	$\varphi \wedge (\psi \wedge \xi)$	$(\varphi \vee \psi) \vee \xi$	$\leftrightarrow$	$\varphi \vee (\psi \vee \xi)$
可換性	$\varphi \wedge \psi$	$\leftrightarrow$	$\psi \wedge \varphi$	$\varphi \vee \psi$	$\leftrightarrow$	$\psi \vee \varphi$
単位元	$\varphi \wedge \top$	$\leftrightarrow$	φ	$\varphi \vee \bot$	$\leftrightarrow$	φ
べき等性	$\varphi \wedge \varphi$	$\leftrightarrow$	φ	$\varphi \vee \varphi$	$\leftrightarrow$	φ
分配則	$\varphi \wedge (\psi \vee \xi)$	$\leftrightarrow$	$(\varphi \wedge \psi) \vee (\varphi \wedge \xi)$	$\varphi \vee (\psi \wedge \xi)$	$\leftrightarrow$	$(\varphi \vee \psi) \wedge (\varphi \vee \xi)$
	$\varphi \wedge \bot$	$\leftrightarrow$	$\bot$	$\varphi \vee \top$	$\leftrightarrow$	$\top$
ドモルガンの法則	$\neg(\varphi \wedge \psi)$	$\leftrightarrow$	$\neg\varphi \vee \neg\psi$	$\neg(\varphi \vee \psi)$	$\leftrightarrow$	$\neg\varphi \wedge \neg\psi$
	$\neg\top$	$\leftrightarrow$	$\bot$	$\neg\bot$	$\leftrightarrow$	$\top$

これらは全て妥当な論理式であり、古典論理で証明可能である。ここで注意してほしいのは、含意除去・二重否定除去を除く全ての法則は二つの形をもち、一方は他方の $\wedge$ と $\vee$ を入れ替え、$\top$ と $\bot$ を入れ替えた形をしているということである。このことは決して偶然ではなく、古典論理の最も重要な性質である**双対原理** (duality principle) の最たる帰結である。

含意記号 $\to$ を含まない論理式 φ （ただし $\neg$ は用いてよい）が与えられたとき、

$$\wedge \quad \rightleftharpoons \quad \vee, \qquad \top \quad \rightleftharpoons \quad \bot, \qquad \forall \quad \rightleftharpoons \quad \exists$$

という入れ替えを施して得られる論理式を φ^d と書く。このとき、次のことが成り立つ。

定理 1.32 (双対原理) φ を含意記号 $\to$ を含まない論理式とする（ただし $\neg$ は用いられていてもよい）。このとき、

$$\vdash_{\mathbf{CL}} \varphi \to \psi \quad \Longleftrightarrow \quad \vdash_{\mathbf{CL}} \psi^d \to \varphi^d.$$

それゆえ

$$\vdash_{\mathbf{CL}} \varphi \leftrightarrow \psi \quad \Longleftrightarrow \quad \vdash_{\mathbf{CL}} \varphi^d \leftrightarrow \psi^d.$$

連言標準形と選言標準形　複素係数の多項式 $P(x)$ を考える。そのような多項式は常に $\alpha_0 + \alpha_1 x + \cdots + \alpha_n x^n$ の形に展開できる。また、$\beta_0(x-\beta_1)\cdots(x-\beta_n)$ の形に因数分解できる。ゆえに複素係数の多項式は二種類の標準形を持っていると言ってよい。

論理においてこれらに相当するのは、選言標準形と連言標準形である。

定義 1.33 (連言標準形、選言標準形)

1. $\forall, \exists$ を含まない論理式を**開論理式** *(open formula)* という。
2. 原子論理式 $p(t_1, \ldots, t_n)$ およびその否定 $\neg p(t_1, \ldots, t_n)$ のことを**リテラル** *(literal)* という。
3. 有限個のリテラル $\alpha_1, \ldots, \alpha_n$ を $\vee$ で結んでできる $\alpha_1 \vee \cdots \vee \alpha_n$ を**節** *(clause)* という。（$\vee$ は結合性を満たすので結合の順番は無視してよい。）
4. 有限個の節 $\varphi_1, \ldots, \varphi_n$ を $\wedge$ で結んでできる $\varphi_1 \wedge \cdots \wedge \varphi_n$ を**連言標準形** *(conjunctive normal form)* という。
5. 連言標準形 φ の双対 φ^d を**選言標準形** *(disjunctive normal form)* という。

つまり、連言標準形、選言標準形はそれぞれ次の形をしている：

$$\bigwedge_{i=1}^{n} \bigvee_{j=1}^{m_i} \alpha_{ij}, \qquad \bigvee_{i=1}^{n} \bigwedge_{j=1}^{m_i} \alpha_{ij}.$$

ただし各 α_{ij} は原子論理式かその否定である。

本節の最初に挙げた諸法則を用いれば、どんな開論理式も連言標準形に直すことができる。

1. 開論理式 φ_0 が与えられたら、まずは含意除去を用いて $\to$ を除去する（ただし $\neg$ は残してよい）：
$$\varphi \to \psi \quad \mapsto \quad \neg\varphi \vee \psi.$$
2. 次にドモルガンの法則を用いて $\neg$ を可能な限り内側に “押し込む” ：
$$\neg(\varphi \wedge \psi) \quad \mapsto \quad \neg\varphi \vee \neg\psi, \qquad \neg(\varphi \vee \psi) \quad \mapsto \quad \neg\varphi \wedge \neg\psi, \qquad \neg\top \quad \mapsto \quad \bot, \qquad \neg\bot \quad \mapsto \quad \top.$$
3. 二重否定除去を用いて $\neg\neg$ を全部取り除く：
$$\neg\neg\varphi \quad \mapsto \quad \varphi.$$

4. $\bot, \top$ を単位元・分配則を用いて取り除く：

$$\varphi \vee \bot \quad \mapsto \quad \varphi, \qquad \varphi \wedge \top \quad \mapsto \quad \varphi, \qquad \varphi \wedge \bot \quad \mapsto \quad \bot, \qquad etc.$$

5. 最後に分配則を用いて $\wedge$ の外側に現れる $\vee$ を可能な限り内側に “押し込む”：

$$\varphi \vee (\psi \wedge \xi) \quad \mapsto \quad (\varphi \vee \psi) \wedge (\varphi \vee \xi), \qquad (\psi \wedge \xi) \vee \varphi \quad \mapsto \quad (\psi \vee \varphi) \wedge (\psi \vee \varphi).$$

結果として得られる論理式が連言標準形となることは明らかである。選言標準形の求め方も同様である（最後に用いる分配則を変えるだけである）。

定理 1.34 (連言標準形、選言標準形) どんな開論理式 φ に対しても連言標準形 $\varphi_\wedge$ と選言標準形 $\varphi_\vee$ が存在し、

$$\vdash_{\mathbf{CL}} \varphi \leftrightarrow \varphi_\wedge, \qquad \vdash_{\mathbf{CL}} \varphi \leftrightarrow \varphi_\vee.$$

余談：標準形と $P \neq NP$ 予想　いま、言語 L は命題記号（0 項述語記号）しか含まないとし、量化子 $\forall, \exists$ を含まない論理式のみを考えるとする（たとえば $p \wedge (q \to \neg p)$）。そのような論理式には項は含まれない。特に変数は一切含まれない。このような論理式を**命題論理式** (propositional formula) という。

命題論理式の真理値は、各命題記号 p へ T か F を割り当てることにより完全に定まる。実際、命題記号の真理値から論理式全体の真理値を求めるには、前節で与えた真理値表を用いればよい。

どんな風に真理値を割り当てても真となる論理式を**トートロジー** (tautology) と呼ぶ。一方、ある仕方で真理値を割り当てると真となる論理式を**充足可能** (satisfiable) であるという。完全性定理により、

$$\vdash_{\mathbf{CL}} \varphi \quad \Longleftrightarrow \quad \varphi \text{ はトートロジー}$$

である。また定義からすぐにわかるように

$$\varphi \text{ はトートロジー} \quad \Longleftrightarrow \quad \neg\varphi \text{ は充足不可能}$$

である。

さて、φ が選言標準形

$$\varphi \quad \equiv \quad \varphi_1 \vee \cdots \vee \varphi_n$$

のときには、φ が充足可能かどうかは簡単にわかる。実際、φ が充足可能なのは、ある φ_i $(1 \leq i \leq n)$ が充足可能なときに限る。そして φ_i は $\alpha_1 \wedge \cdots \wedge \alpha_m$ という形をしているので、これが充足可能なのは、$\alpha_1, \ldots, \alpha_n$ がある命題記号 p とその否定 $\neg p$ を同時に含まないときに限る。これはコンピュータを使えば高速にわかることである。実際、φ のサイズ（φ に含まれる記号の数）を n とすれば、上の判定法は高々 n の定数倍ステップくらいで実行できる。

同様にして、φ が連言標準形のときには、φ がトートロジーかどうかは簡単にわかる（このことを判定する方法を考えよ）。

さて、ここで問題なのは、一般の命題論理式の充足可能性を判定する問題である。

- **充足可能性問題** (satisfiability problem) : 命題論理式 φ が与えられたとき、φ が充足可能かどうかを判定せよ。

一見すると、これは簡単なように思える。なぜならどんな論理式も選言標準形に直すことができるのであるから、φ を一旦選言標準形に直した上で、上の判定法を用いればよい。

しかしここには穴がある。論理式を選言標準形定理に直す過程では分配則を使う必要がある：$\varphi \wedge (\psi \vee \xi) \quad \mapsto \quad (\varphi \wedge \psi) \vee (\varphi \wedge \psi)$ この書き換えを行うと φ が複製されてしまう。ゆえに最悪の状況（φ が非常に大きくて、ψ, ξ が非常に小さい場合）を考えると、論理式のサイズはほぼ 2 倍になってしまう。ゆえに元の論理式のサイズを n とすると、最悪の場合選言標準形のサイズは 2^n ほどにもなってしまう。すなわち計算には**指数関数時間** (exponential time) を要する。n が 3 桁、4 桁の数ともなると、これはどんなに高速なコンピュータを用いても、現在知られているどんなにうまいアルゴリズムを用いても、計算は何兆世紀もかかることになる。

上のような指数関数時間アルゴリズムと対比されるのは**多項式時間** (polynomial time) アルゴリズム、すなわち計算に高々 cn^k (c, k は定数) ステップくらいしかかからないアルゴリズムである。これはコンピュータを用いて高速に実行可能なアルゴリズムのある程度よい近似になっている。

充足可能性に関する根本問題は次のものである。

- 充足可能性問題を多項式時間で解くアルゴリズムは存在するか？

これはいわゆる P vs. NP 問題の一つの表現である。もしもそのようなアルゴリズムが存在するなら $P = NP$ であり、存在しないなら $P \neq NP$ である。大方の予想は $P \neq NP$ であるが、who knows である。いずれにせよ、もしも上の問題がとけたならクレイ数学研究所から 100 万ドルもらえる（http://www.claymath.org/millennium/）。

この問題が面白いのは、離散数学（グラフ論、整数論、組み合わせ論）における多くの困難な問題と**一蓮托生**であることにある。すなわち充足可能性問題に多項式時間アルゴリズムが存在すれば、多くの困難な問題にも多項式時間アルゴリズムが存在することになり、計算の世界の見取り図は一新することになる。逆にいえば、この一蓮托生性が、充足可能性問題の困難さを裏付けているのである。

冠頭標準形　つぎに $\forall, \exists$ を含む論理式一般の標準形について考える。その際に基本となるのは、以下の論理法則である。

結合性	$\varphi_0 \wedge \forall x.\psi(x) \leftrightarrow \forall x.\varphi_0 \wedge \psi(x)$	$\varphi_0 \vee \exists x.\psi(x) \leftrightarrow \exists x.\varphi_0 \vee \psi(x)$
分配則	$\varphi_0 \vee \forall x.\psi(x) \leftrightarrow \forall x.\varphi_0 \vee \psi(x)$	$\varphi_0 \wedge \exists x.\psi(x) \leftrightarrow \exists x.\varphi_0 \wedge \psi(x)$
ドモルガンの法則	$\neg\forall x.\varphi(x) \leftrightarrow \exists x.\neg\varphi(x)$	$\neg\exists x.\varphi(x) \leftrightarrow \forall x.\neg\varphi(x)$

(*) ただし結合性・分配則において φ_0 は x を自由変数として含まない。

これらもやはり妥当な論理式であり、古典論理で証明可能である。これらの法則を使うと、どんな論理式も次の冠頭標準形に直すことができる。

定義 1.35 (冠頭標準形) 次の形の論理式を**冠頭標準形** *(prenex normal form)* という：

$$Q_1 x_1 \cdots Q_n x_n.\varphi$$

ここで $n \geq 0$、各 Q_i は $\forall$ か $\exists$ かのどちらかであり、φ は開論理式である。

与えられた論理式を冠頭標準形に直すには、まず $\to$ を含意除去により取り除き、量化子の外側に現れる $\wedge, \vee, \neg, \top, \bot$ を結合性、分配則、ドモルガンの法則を用いて取り除けばよい。

定理 1.36 (冠頭標準形) どんな論理式 φ に対しても冠頭標準形 $\varphi_{\mathbf{Q}}$ が存在し、

$$\vdash_{\mathbf{CL}} \varphi \leftrightarrow \varphi_{\mathbf{Q}}.$$

冠頭標準形は、論理式の**論理的な複雑さ** (logical complexity) を測る上で重要な役割を果たす。すなわち文 φ を冠頭標準形に直すと、

$$\forall x_1 \exists x_2 \exists x_3 \forall x_4 \forall x_5 . \psi$$

というふうに $\forall$ と $\exists$ が連なって現れるわけだが、ここで $\forall$ から $\exists$ へ、または $\exists$ から $\forall$ へと交代する回数をもって φ の複雑さを定義するのである。

2　テューリング機械 — “理想化された計算者”

> 無限大学には毎年無限人の学生が受験する。合格発表の日には、掲示板に合格者の受験番号がひとつひとつ、順不同に掲示されていく。さて、掲示板をずっとながめていれば、合格者はいつかは自分が合格したことがわかるだろう。しかし不合格者はいつまでたっても自分が不合格なことがわからない。これではあまりにも不都合なので、大学当局はある対策を講じた。その対策とは何か？

前章では、人間が推論する過程や例を作る過程を分析することにより、“理想化された推論者” や “理想化された反論者” を定義した。本章では、同じ路線にのっとって人間が計算を行う過程を分析し、それにより “理想化された計算者” を定義する。この作業を最初に行ったのは論理学者テューリングである。1936 年に彼の定義した計算者 (computer) は、当初は単なる理論的概念にすぎなかったが、その発想に基づいてやがてハードウェアとしてのコンピュータが誕生するに至ったのである。テューリングの定義した最初の計算者は、現在では**テューリング機械** (Turing machine) と呼ばれており、計算可能性や計算量について論じる際の基準としての役割を担っている。

計算過程の分析　テューリングはまず、我々人間がノートと鉛筆を行って計算を行う際に正確には何が行われているかを徹底的に分析し、その本質を捉えようとした。例えば、足し算「528 + 645」を計算するとき、一体何が行われているのだろうか?一つの考え方は次の通りである。まず、一の位にある二つの記号「8」と「5」を読み込む。そして二数の和「13」を求め、一時的にノートに書き記す。次に二の位の記号「2」と「4」を読み込み、繰り上がりを考慮に入れた上で、「7」を「1」の上に上書きする。そして三の位に進む。そうやって全ての位を処理し終えたら、鉛筆をおいて停止する。最終的にノートに書かれた記号列が答えである。

ここで重要なのは、一桁の足し算「8+5」や「2+4+1」は高々有限通りしかないので、有限の表なり記憶なりを参照すれば、規則に従う能力のある主体である限り、誰でも簡単に遂行できるという点である。これが基本ステップであり、この有限通りの単純な作業を繰り返すことで、理論上無限通りの組み合わせが可能な二数の足し算が行えるのである。このことは足し算に限らず、掛け算割り算だろうが、微分積分だろうが、行列演算だろうが変わらない。計算の本質は、「あらかじめ決められた規則に従って基本ステップを繰り返すこと」なのである。

その際に一時的な結果をノート（計算用紙）に書きこんでよい。一時的な結果は後で上書きしてもよいし、消去してもよい。ノートは足りなくなったらいくらでも追加できるので、原理上無限であるとしてよい。ところで、現実のノートは二次元に広がっているが、このことは本質的だろうか？そんなことはない。仮にノートの形状が左右に伸びる長いテープのようなものだったとしても、計算には支障ない（二次元のノートに比べて左右にいったりきたりする手間はかかるだろうが)。また、テープはマス目に区切られており、一つのマス目には一つの記号しか書けないと仮定してよい。記号の種類はもちろん有限である。有限個の記号を組み合わせることにより、無限に多くの事柄を表現することができるのである。

我々はテープに書かれた記号を一度に一字ずつ読むとしてよい。その際テープを読む装

置（人間の場合は眼）のことをヘッドと呼ぶ。複数の文字を読むのには、ヘッドを左右に一マスずつずらしていけばよい。

最後に、我々が規則に従って計算を行う際には、計算過程のどの段階にあるのかを意識している必要がある。人間の意識状態は複雑であるが、およそ規則に従って機械的に作業を行う限りにおいては、それに必要な意識状態の種類は、（どんなに多数であろうとも）有限通りであると仮定してよい。無限の長さのテープ（ノート）が、有限の意識状態を補完してくれるからである。（「博士の愛した数式」を思い出してほしい。記憶に収まりきらず忘れてしまうことはノートに書いておけばよいのである。）

以上の分析により、計算には最低限次のような道具立てが必要なことがわかる。

1. 有限種類の記号
2. 無限の長さのテープ
3. ヘッド
4. 有限種類の状態
5. 計算規則

これらを数学的に表したものがテューリング機械である。ただしここで「無限の長さのテープ」と言っているが、これは字義どおりに無限ということではなく（そんなテープは物理的に存在しえない！）、「必要ならばいくらでも後から継ぎ足せる」という意味で可能的に無限であるということにすぎない。

定義 2.1 (テューリング機械) テューリング機械 *(Turing machine)* は 5つ組 $(\Sigma, Q, \delta, q_I, q_F)$ により定められる。ここで

- Σ は**記号** *(symbol)* の有限集合。Σ の中には常に $0, 1, \sqcup$ が含まれるものとする。最後の記号 $\sqcup$ は**空白** *(blank)* を表す。
- $\mathbf{Q}$ は**状態** *(state)* の有限集合。
- δ は**遷移関数** *(transition function)*

$$\delta : \Sigma \times Q \longrightarrow \Sigma \times Q \times \{\leftarrow, \rightarrow\}.$$

- q_I, q_F は $\mathbf{Q}$ の要素であり、それぞれ**初期状態** *(initial state)*、**停止状態** *(final state)* を表す。

X を集合とするとき、X の要素からなる有限列全体の集合を X^* により表す。たとえば $\{0,1\}^*$ は 0 と 1 からなる有限列全体の集合を表す。

さて、**入力** (input) $w \in \{0,1\}^*$ が与えられたとき、テューリング機械 $M = (\Sigma, Q, \delta, q_I, q_F)$ は次のように動作する。

- 最初、テープには w が（一マスに一字ずつ）書かれている。残りの部分には空白が書かれている。

- M は初期状態 q_I にあり、そのヘッドは w の左端にある。
- 次のことを繰り返す。
  1. M のヘッドが記号 $a \in \Sigma$ を読んでおり状態 $p \in Q$ にあるとする。このとき、遷移関数を参照し、$\delta(a,p) = (b,q,\leftarrow)$ ならば、現在のマス目に記号 b を上書きし、ヘッドを左に一マス動かし、状態 q に移る。($\leftarrow$ の代わりに $\rightarrow$ のときには、ヘッドを右に一マス動かす。)
  2. もしも $q = q_F$ ならば停止する。さもなければ 1 に戻る。
- 停止状態 q_F に到達したときにテープに書かれている記号列が**出力** (output) である。ただし空白記号は無視する。M は停止しないで計算を永遠に続ける可能性があることに注意。そのときには M は何も出力しない。

例 2.2 次のテューリング機械 $M = (\Sigma, Q, \delta, q_I, q_F)$ は二進数が入力として与えられたとき、その数 $+1$ を出力する。

- $\Sigma = \{0, 1, \sqcup\}$
- $Q = \{q_I, q_F, q_0\}$
- $\delta : \Sigma \times Q \longrightarrow \Sigma \times Q \times \{\leftarrow, \rightarrow\}$ は次のように定義される :

$$\begin{array}{rcl} (0, q_I) & \mapsto & (0, q_I, \rightarrow) \\ (1, q_I) & \mapsto & (1, q_I, \rightarrow) \\ (\sqcup, q_I) & \mapsto & (\sqcup, q_0, \leftarrow) \\ (0, q_0) & \mapsto & (1, q_F, \leftarrow) \\ (1, q_0) & \mapsto & (0, q_0, \leftarrow) \\ (\sqcup, q_0) & \mapsto & (1, q_F, \leftarrow) \end{array}$$

例 2.3 次のテューリング機械 $M = (\Sigma, Q, \delta, q_I, q_F)$ は *0-1* 列 w が入力として与えられたとき、w が回文ならば 1 を、回文でなければ 0 を出力する。

- $\Sigma = \{0, 1, \sqcup\}$
- $Q = \{q_I, q_F, q_0, q_0', q_1, q_1', q_y, q_n\}$

- $\delta : \Sigma \times Q \longrightarrow \Sigma \times Q \times \{\leftarrow, \rightarrow\}$ は次のように定義される：

$$
\begin{array}{llllll}
(0, q_I) & \mapsto & (\sqcup, q_0, \rightarrow) & & & \\
(1, q_I) & \mapsto & (\sqcup, q_1, \rightarrow) & & & \\
(\sqcup, q_I) & \mapsto & (1, q_F, \rightarrow) & & & \\
(0, q_0) & \mapsto & (0, q_0, \rightarrow) & (0, q_1) & \mapsto & (0, q_1, \rightarrow) \\
(1, q_0) & \mapsto & (1, q_0, \rightarrow) & (1, q_1) & \mapsto & (1, q_1, \rightarrow) \\
(\sqcup, q_0) & \mapsto & (\sqcup, q'_0, \leftarrow) & (\sqcup, q_1) & \mapsto & (\sqcup, q'_1, \leftarrow) \\
(0, q'_0) & \mapsto & (\sqcup, q_y, \leftarrow) & (0, q'_1) & \mapsto & (\sqcup, q_n, \leftarrow) \\
(1, q'_0) & \mapsto & (\sqcup, q_n, \leftarrow) & (1, q'_1) & \mapsto & (\sqcup, q_y, \leftarrow) \\
(\sqcup, q'_0) & \mapsto & (1, q_F, \leftarrow) & (\sqcup, q'_1) & \mapsto & (1, q_F, \leftarrow) \\
(0, q_y) & \mapsto & (0, q_y, \leftarrow) & (0, q_n) & \mapsto & (\sqcup, q_n, \leftarrow) \\
(1, q_y) & \mapsto & (1, q_y, \leftarrow) & (1, q_n) & \mapsto & (\sqcup, q_n, \leftarrow) \\
(\sqcup, q_y) & \mapsto & (\sqcup, q_I, \rightarrow) & (\sqcup, q_n) & \mapsto & (0, q_F, \leftarrow)
\end{array}
$$

どんな離散的対象（自然数、有理数、数式、日本語の文章など）も 0-1 列を用いて**コード化** (encoding) することができるので、入力も出力も $\{0,1\}^*$ の要素に限って話を進めて差し支えない。

上の回文の例のように、テューリング機械に与えられる課題は、しばしば次の形をとる。

回文問題

入力: 文字列 $w \in \{0,1\}^*$

問題: w は回文か？

テューリング機械は、答えがイエスのときには 1 を出力し、ノーのときには 0 を出力することが求められる。このような課題を**決定問題** (decision problem) という。決定問題の例としては、次のようなものがある。

一筆書き問題

入力: グラフ G

問題: G は一筆書き可能か？

ディオファントス方程式問題

入力: 整係数多変数多項式 $P(x_1, \ldots, x_n)$

問題: $P(x_1, \ldots, x_n) = 0$ は整数解を持つか？

さて、グラフや多項式は 0-1 列を用いてコード化することに注意すれば、上の各決定問題を $\{0,1\}^*$ の部分集合で表すことができる。実際には、次のような集合を考えればよい：

$$
\begin{array}{lll}
L_1 & = & \{w \in \{0,1\}^* : w \text{ は回文 } \} \\
L_2 & = & \{w \in \{0,1\}^* : w \text{ は一筆書き可能なグラフを表す } \} \\
L_3 & = & \{w \in \{0,1\}^* : w \text{ は整数解をもつ整係数多項式を表す } \}
\end{array}
$$

このような $\{0,1\}^*$ の部分集合のことを**言語** (language) という（1 章の語法とオーバーラップするが、慣用に従うことにする）。これにより、決定問題を解くことは、与えられた入力 w が言語 L に属するかどうかを判定する問題に帰着する。たとえば、テューリング機械 M がディオファントス方程式問題を解くというのは、入力 w が与えられたとき、M は $w \in L_3$ かどうかを判定することに他ならない。厳密な定義を与えるために、以下の記法を用いる。

- 入力 $w \in \{0,1\}^*$ が与えられたとき、M が $u \in \{0,1\}^*$ を出力するときには、$M(w) = u$ と書く。
- 入力 $w \in \{0,1\}^*$ が与えられたとき、M が停止せず永遠に計算を続けるか、0-1 列以外を出力して停止する場合には $M(w) = \uparrow$ と書く。

定義 2.4 (受理、決定、計算) M をテューリング機械とし、言語 $L \subseteq \{0,1\}^*$ および関数 $f : \{0,1\}^* \longrightarrow \{0,1\}^*$ を考える。

- どんな $w \in \{0,1\}^*$ に対しても M が

$$w \in L \quad \Longleftrightarrow \quad M(w) = 1$$

を満たすとき、M は L を**受理する** *(accept)* という。

- どんな $w \in \{0,1\}^*$ に対しても M が

$$\begin{aligned} w \in L &\Rightarrow M(w) = 1 \\ w \notin L &\Rightarrow M(w) = 0 \end{aligned}$$

を満たすとき、M は L を**決定する** *(decide)* という。

- どんな $w \in \{0,1\}^*$ に対しても M が

$$M(w) = f(w)$$

を満たすとき、M は f を**計算する** *(compute)* という。

定義により、例 2.3 のテューリング機械は言語 L_1 を決定するし、また受理もする。「受理」と「決定」の違いは、L に含まれない要素 w が与えられたときの M の振る舞いにある。M が L を決定する場合には、M は 0 を出力して停止する。一方、M が L を単に受理するだけの場合には、M は 1 を出力しない限りどのように振る舞ってもよい。特に M は停止せず、計算を永遠に続けてもよい。

例えば、ディオファントス方程式問題に相当する言語 L_3 について考える。L_3 を受理するテューリング機械 M を与えるのは比較的簡単である。素朴に考えて、$P(x) = 0$ が整数解を持つかどうかを調べるには、$P(x)$ に $x = 0, \pm 1, \pm 2, \ldots$ を次々に代入して、値が 0 になるかどうかを調べて行けばよいからである。もしそのような x が見つかったら停止する。見つからない限り永遠に代入を続ける。そのようなテューリング機械を定義してやればよい。

しかしこの方法では、どこまで計算を続けても $P(x) = 0$ が整数解を**持たない**ことを確かめることはできない。たとえ $x = \pm 1000000000$ まで試して解が見つからなかったとしても、次の数 1000000001 が解となる可能性は捨てきれないからである。ゆえに上で考えたテューリング機械は L_3 を決定しはしない。

ディオファントス方程式が与えられたときにそれが整数解をもつかどうかを判定する方法を考案せよというのは、ヒルベルトが 1900 年に提起した**ヒルベルトの 23 問題**のうちの 10 番目にあたる。この問題は 70 年を経て、マチャセヴィッチにより否定的に解決された。

定理 2.5 L_3 を決定するテューリング機械は存在しない。

ゆえに「受理」と「決定」は本質的に異なるのである。

チャーチ・テューリングのテーゼ　マチャセヴェッチの定理はなぜそれほど重要なのか？それはテューリング機械のもつ普遍性による。テューリング機械の定義は単純であるが、そのポテンシャルは計り知れない。実際、四則演算・微分積分どころか、およそ我々が計算できることは何でも代行することができる。さらに、現代のコンピュータと比べてもひけをとらない（ただし計算コストの問題は度外視する）。実際、Windows だろうが Mac だろうがどんなコンピュータもテューリング機械を使って模倣することができる。この普遍性にかんがみて、1930 年代に次のテーゼが立てられた。

- **チャーチ・テューリングのテーゼ**: 関数 f が（アルゴリズム的に）計算可能であるとは、f はテューリング機械を用いて計算可能であることに他ならない。また決定問題 L が（アルゴリズム的）解法を持つとは、L がテューリング機械を用いて決定可能であることに他ならない。

つまりテューリング機械を “計算” の標準として受け入れようというのである。このテーゼを受け入れれば、マチャセヴィッチの定理から次のことが言える。すなわち、与えられたディオファントス方程式が整数解を持つかどうかを判定するアルゴリズム的解法は存在しない。

チャーチ・テューリングのテーゼを受け入れる根拠としては次のようなことが挙げられる。

- アルゴリズム的に計算可能で、かつテューリング機械によっては計算不可能な関数（や問題）は今まで見つかっていない。
- 全く異なる発想により考案された他の計算モデル（ラムダ計算、再帰的関数）と計算能力がぴったり一致する。
- テューリング機械は人間の行う計算を分析し、抽象化することにより得られた “理想化された計算者” である。ゆえに “計算” の理論的標準としてふさわしい。

もちろん、将来テューリング機械では手に負えないような関数の値が求められるようになる可能性は完全には否定しきれない。しかし、たとえそのような方法が現れたとしても、それは「計算」とは呼ばれずに何か別の呼称で呼ばれるのではないだろうか（例えば魔術や超能力など）。それくらいにチャーチ・テューリングのテーゼは確固としている。

このテーゼは証明可能な定理といったような代物ではなく、「およそ計算可能なものはテューリング機械を用いて計算可能なはずだ」という経験的な主張と、「計算可能という語をテューリング機械で計算可能という意味で使いましょう」いう規範的な役割を併せたものにすぎない。にも関わらず、以下では定理を証明するときに、このテーゼを（無害な形で）多用していく。それにより、計算を巡る議論が大幅に簡略化されるからである。

定義 2.6　言語 $L \subseteq \{0,1\}^*$ および関数 $f : \{0,1\}^* \longrightarrow \{0,1\}^*$ を考える。

- L を受理するテューリング機械が存在するとき、L は**再帰的枚挙可能** *(recursively enumerable)* であるという。
- L を決定するテューリング機械が存在するとき、L は**再帰的** *(recursive)* であるという。
- f を計算するテューリング機械が存在するとき、f は**再帰的** *(recursive)* であるという。

「再帰的」「再帰的枚挙可能」という語を用いるのは歴史的な経緯による。定義により、再帰的な言語は再帰的枚挙可能でもある。一方、L_3 は再帰的枚挙可能ではあるが、再帰的ではない。

定理 2.7 言語 $L \subseteq \{0,1\}^*$ が再帰的なのは、L とその補集合 $\overline{L}$ が共に再帰的枚挙可能なとき、かつその時に限る。

証明　言語 $L \subseteq \{0,1\}^*$ が再帰的ならば、L も $\overline{L}$ も再帰的枚挙可能なことはすぐにわかる。逆に L と $\overline{L}$ がともに再帰的枚挙可能だとすると、L, $\overline{L}$ を受理するテューリング機械 M_1, M_2 が存在する。このとき、次のような手順を考えれば、$w \in L$ かどうかを決定することができる。

1. 入力 $w \in \{0,1\}^*$ が与えられたとき、M_1 と M_2 の計算を 1 ステップずつ、交互にシミュレートする。
2. もしも M_1 の計算が先に停止し、1 を出力するならば、$w \in L$ である。
3. もしも M_2 の計算が先に停止し、1 を出力するならば、$w \in \overline{L}$、すなわち $x \notin L$ である。
4. $w \in L \cup \overline{L}$ なので、M_1 か M_2 のどちらか一方は必ず有限ステップで停止するはずである。

上の手順は L を決定するアルゴリズム的解法を与えているから、チャーチ・テューリングのテーゼにより、L を決定するテューリング機械が存在する。すなわち L は再帰的である。

■

L が再帰的枚挙可能であるという状況は、冒頭で挙げた無限大学の合格発表の状況に似ている。合格者掲示板に次々と受験番号が発表されていく（L は再帰的枚挙可能）ので、

自分が合格しているならば（$w \in L$ ならば）いつかは合格していることがわかる（テューリング機械が 1 を出力する）。しかし自分が不合格の場合には、いつまでたっても不合格であることはわからない（テューリング機械は停止しない）。この状況を改善するには、合格者掲示板に加えて、不合格者掲示板も作ればよい（$\overline{L}$ も再帰的枚挙可能であればよい）。そうすれば、どんな受験番号も合格者掲示板と不合格者掲示板のどちらか一方に必ずあらわれるので、両掲示板を交互に見ていけば、合格者も不合格者も必ず有限時間内に自分の合否がわかるはずである（L は再帰的である）。これが上の定理の証明のアイデアに他ならない。

練習問題 2.8 次の言語を決定するテューリング機械を定義せよ。

$$\begin{aligned} L_1 &= \{w \in \{0,1\}^* : w \text{ は偶数個の } \textit{1} \text{ を含む}\} \\ L_2 &= \{w \in \{0,1\}^* : w \text{ は } \textit{0} \text{ と } \textit{1} \text{ を同数個ずつ含む}\} \end{aligned}$$

練習問題 2.9 $L \subseteq \{0,1\}^*$ を言語とする。

1. テープの適当な領域に L の要素を次々と順不同で列挙していくようなテューリング機械 M が存在するとする：

   $$010 \sqcup 100101010 \sqcup 01 \sqcup 1010 \cdots$$

   このとき L は再帰的枚挙可能なことを証明せよ。

2. L が無限集合で、長さの短い順に L の要素を列挙していくようなテューリング機械 M が存在するとする：

   $$01 \sqcup 010 \sqcup 1010 \sqcup 100101010 \cdots$$

   このとき L は再帰的言語であることを証明せよ。

（チャーチ・テューリングのテーゼを用いてよい。）

3　算術階層 — なぜ我々はかくも不完全なのかについて

3.1　算術の論理式

本章ではもっぱら算術の理論について論じる。そこで算術の言語とは何であったかを思い出しておこう。算術の言語 L_0 は以下の記号からなる。

- 関数記号 $\{0^{(0)}, S^{(1)}, +^{(2)}, \cdot^{(2)}\}$
- 述語記号 $\{=^{(2)}\}$

これらを用いて項や論理式を組み立てていくわけである。念のため項の定義を算術の言語に合わせて書きなおしておく。**算術の項** (arithmetical term) は次のようにして作られる。

$$t, u \ ::= \ x \mid 0 \mid S(t) \mid (t + u) \mid (t \cdot u).$$

このようにして作られる項のみが算術の項である。

$$0, \quad S(0), \quad S(S(0)), \quad S(S(S(0))), \quad \ldots$$

のような項を**数項** (numeral) とよび、自然数 $n \in \mathbb{N}$ に対応する数項を n と書く。**算術の論理式** (arithmetical formula) は次のようにして作られる。

$$\varphi, \psi \ ::= \ \top \mid \bot \mid t = u \mid (\varphi \wedge \psi) \mid (\varphi \vee \psi) \mid (\varphi \to \psi) \mid \forall x.\varphi \mid \exists x.\varphi.$$

自由変数を含まない論理式を**算術の文** (arithmetical sentence) といい、自由変数を高々1つだけ含む論理式を（算術の）**一変数論理式** (one-variable formula) という。

次の派生表現を用いる。

$$\begin{aligned} x \leq y &\equiv \exists z.z + x = y \\ \forall x \leq t.\varphi(x) &\equiv \forall x.(x \leq t \to \varphi(x)) \\ \exists x \leq t.\varphi(x) &\equiv \exists x.(x \leq t \wedge \varphi(x)) \\ \exists! x.\varphi(x) &\equiv \exists x(\varphi(x) \wedge \forall y(\varphi(y) \to x = y)) \end{aligned}$$

ゲーデル数　算術の項や論理式は記号列なので自然数を用いてコード化することができる。すなわち、$T(L_0)$, $F(L_0)$ をそれぞれ算術の項全体の集合、算術の論理式全体の集合とするとき、ある再帰的な（つまり計算可能な）関数

$$\ulcorner \ \urcorner : T(L_0) \cup F(L_0) \longrightarrow \mathbb{N}$$

が存在する。この関数はもちろん単射でなければならない：

$$t \not\equiv u \quad \Longrightarrow \quad \ulcorner t \urcorner \neq \ulcorner u \urcorner, \qquad \varphi \not\equiv \psi \quad \Longrightarrow \quad \ulcorner \varphi \urcorner \neq \ulcorner \psi \urcorner.$$

また、$\ulcorner \ \urcorner$ の逆操作

$$\mathsf{decode} : \mathbb{N} \longrightarrow T(L_0) \cup F(L_0) \cup \{\mathsf{error}\}$$

も再帰的でなければならない。自然数 $\ulcorner t \urcorner$, $\ulcorner \varphi \urcorner$ を t, φ の**ゲーデル数** (Gödel number) と呼ぶ。$\mathsf{decode}(n) = \mathsf{error}$ となるのは、n がどんな項や論理式のゲーデル数でもない場合である。

算術の真理集合　算術の文 φ が普通の意味で真であることを $\mathbf{N} \models \varphi$ と書く。形式的にはこれは標準モデル $\mathbf{N}$ 上で文 φ が成り立つことを意味するが、あまり意識する必要はない。例えば

$$\mathbf{N} \models \forall x \exists y (x \leq y \wedge x \neq y), \qquad \mathbf{N} \not\models \exists y \forall x (x \leq y \wedge x \neq y)$$

であるが、このことは厳密な定義を参照しなくとも直感的に明らかだろう。

さて、ゲーデル数を用いれば、次のような自然数の集合を考えることができる。

$$\mathsf{TA} := \{\ulcorner \varphi \urcorner \ : \ \mathbf{N} \models \varphi\}.$$

この集合 $\mathsf{TA} \subseteq \mathbb{N}$ はいわば "真実の書" であり、この書を手にする者は、初等数論の全ての真理を掌中におさめているといってよい。たとえば、双子素数予想を表す論理式を φ とし、そのゲーデル数を $n = \ulcorner \varphi \urcorner$ とする。このとき双子素数予想が成り立つための必要十分条件は $n \in \mathsf{TA}$ である。このようにして全ての真理を記述する書というのは、相当に複雑であろうというのは容易に予想できる。では一体、正確にいってどの程度複雑なのだろうか？この問いに答えるためには、自然数の集合の複雑さを測るための尺度を導入しなければならない。それが論理的複雑さである。

3.2　論理的複雑さ

自然数の集合は非可算無限に存在する。それらのうちの大部分は、いかなる定義をも受け付けない混沌たるものであるが、中には算術の論理式により定義できるような集合も存在する。例えば、偶数の集合は $\exists y(\mathsf{2} \cdot y = \mathsf{n})$ を満たす自然数 n の集合と同一視することができる。そのような集合は算術の論理式により記述可能であるという。ここでは、算術の論理式により記述可能な集合全体の集まりは、**算術階層** (arithmetical hierarchy) と呼ばれる階層性を成すことを示す。

定義 3.1 (限定論理式) 次の *BNF* 記法により定義される論理式を**限定論理式** *(bounded formula)* という。

$$\varphi, \psi \ ::= \ t = u \mid \top \mid \bot \mid (\varphi \wedge \psi) \mid (\varphi \vee \psi) \mid (\varphi \to \psi) \mid \forall x \leq t.\varphi \mid \exists x \leq t.\varphi.$$

すなわち、量化子 $\forall, \exists$ が用いられる際には、常に**限定量化子** (bounded quantifier) $\forall x \leq t, \exists x \leq t$ の形になっているような論理式が限定論理式である。

定義 3.2 (Σ_i 論理式, Π_i 論理式) Σ_0 **論理式**とは限定論理式のことに他ならない。同様に、Π_0 **論理式**とは限定論理式のことに他ならない。各 $i = 1, 2, \ldots$ について Σ_i **論理式**、Π_i **論理式**を次のように帰納的に定義する。

1. φ が Π_{i-1} 論理式ならば、$\exists x_1 \cdots \exists x_n.\varphi$ は Σ_i 論理式である。
2. φ が Σ_{i-1} 論理式ならば、$\forall x_1 \cdots \forall x_n.\varphi$ は Π_i 論理式である。

ここで $n \in \mathbb{N}$ は任意であり、特に $n = 0$ の場合も含める。

定義 3.3 (集合の記述) $\varphi \equiv \varphi(x)$ を一変数論理式とするとき、集合 $X_\varphi \subseteq \mathbb{N}$ を

$$X_\varphi := \{n \mid \mathbf{N} \models \varphi(\mathsf{n})\}$$

と定義する。集合 $X \subseteq \mathbb{N}$ に対して $X = X_\varphi$ となるとき、X は一変数論理式 φ により**記述できる**という。

定義 3.4 (算術階層) 何らかの Σ_i 論理式により記述できるような集合のことを Σ_i **集合**と呼ぶ。同様に、Π_i 論理式により記述できる集合を Π_i **集合**と呼ぶ。Σ_i 集合であり、かつ Π_i 集合でもあるような集合を Δ_i 集合と呼ぶ。

Δ_i 集合全ての集まりのことを単に Δ_i と書く。Σ_i、Π_i についても同様である。

例えば、偶数の集合は Σ_1 論理式 $\exists y(x = 2 \cdot y)$ により記述できるので Σ_1 集合である。実際には、

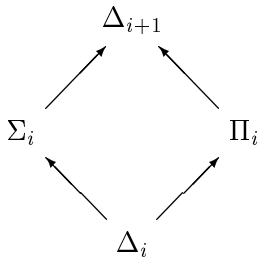
$$\mathsf{even}(x) \equiv \exists y \leq x(x = 2 \cdot y)$$

というように限定論理式（$= \Sigma_0$ 論理式、Π_0 論理式）によっても記述できるので、Δ_0 集合であるとも言える。また、素数の集合も

$$\begin{aligned} y|x &\equiv \exists z \leq x(y \cdot z = x) \\ \mathsf{prime}(x) &\equiv 2 \leq x \wedge \forall y \leq x(y|x \to (y = 1 \vee y = x)) \end{aligned}$$

というように限定論理式により記述できるので、Δ_0 集合である。

補題 3.5 *1.* 各 $i \in \mathbb{N}$ について、Δ_i, Σ_i, Π_i は以下の包含関係にある。（ここで $\longrightarrow$ は包含関係 $\subseteq$ を表す。例えば、$\Delta_i \longrightarrow \Sigma_i$ は、集合 X が Δ_i 集合ならば、それは Σ_i 集合でもあるということを表す。）



2. 算術の論理式により記述可能な集合 X はすべて算術階層に属する。即ち $X \in \bigcup_{i\in\mathbb{N}} \Sigma_i$ である。

3. $X \in \Sigma_i \Longleftrightarrow \overline{X} \in \Pi_i$ （ここで $\overline{X}$ は $\mathbb{N}$ に関する X の補集合を表す。）

証明

1. 定義より、$\Delta_i \subseteq \Sigma_i$, $\Delta_i \subseteq \Pi_i$、$\Sigma_i \subseteq \Pi_{i+1}$、$\Pi_i \subseteq \Sigma_{i+1}$ である。さらに $\Sigma_i \subseteq \Sigma_{i+1}$、$\Pi_i \subseteq \Pi_{i+1}$ であることは、i に関する帰納法により証明することができる。ゆえに

$$\Sigma_i \subseteq \Sigma_{i+1} \cap \Pi_{i+1} = \Delta_{i+1}, \qquad \Pi_i \subseteq \Sigma_{i+1} \cap \Pi_{i+1} = \Delta_{i+1}.$$

2. $X = X_\varphi$ とする。冠頭標準形定理により、φ は $Q_1x_1Q_2x_2\cdots Q_nx_n.\psi$ という形の論理式と同値である（ここで各 Q_i は $\forall$ または $\exists$、ψ は開論理式）。$Q_1, \ldots, Q_n$ における $\forall$ と $\exists$ の交替の回数が k 回であるとすると、それは Σ_{k+1} 論理式である。ゆえに $X = X_\varphi$ は Σ_{k+1} 集合である。

3. ドモルガンの法則より。例えば X が Σ_2 論理式 $\exists y\forall z.\psi$ により記述できるならば、$\overline{X}$ は $\neg\exists y\forall z.\psi$ により記述できるが、後者は Π_2 論理式 $\forall y\exists z.\neg\psi$ と論理的に同値である。 ■

このようにして、算術の論理式により記述できる集合は算術階層の中に分類できることがわかった。念のため確認しておくが、すべての自然数の集合 $X \subseteq \mathbb{N}$ が算術階層に属するわけではない。なぜならばそのような集合は非可算無限個あるが、論理式により記述できる集合は可算無限個しかないからである。そこで次のような疑問がわいてくる。

1. 算術階層に属さない集合の具体例としてはどのようなものがあるか？

2. 定義により $\Sigma_0 = \Pi_0 = \Delta_0$ であるが、$i \geq 1$ のときには、補題 3.5(1) で挙げた包含関係は真なる包含関係になっているのだろうか？

話を先取りすれば、“真実の書”TA がまさに 1. の具体例になっているわけだが、このことは詰まるところ 2. の問題に帰着する。実際、2. の答えはイエスであることを後で示す。算術階層は、いわば絶対強固にそびえたつ高層建築であり、どこか途中の階で崩壊していたりすることはない。この算術階層の絶対強固性が、推論者・計算者としての我々人間に限界を与えているのである。

関数の記述　これまでは集合の記述について考えてきたが、次に関数の記述について考える。

定義 3.6 (Δ_1 関数) 関数 $f : \mathbb{N} \longrightarrow \mathbb{N}$ に対して次のような Σ_1 論理式 $\varphi_f(x, y)$ が存在するとき、f は **Δ_1 関数**であるという：任意の $n \in \mathbb{N}$ について、

$$f(n) = m \quad \Longleftrightarrow \quad \mathbf{N} \models \varphi_f(\mathsf{n}, \mathsf{m}).$$

上の性質を満たす Σ_1 論理式 φ_f が与えられれば、同じ性質を満たす Π_1 論理式を構成することが常に可能である（練習問題）。f が Σ_1 ではなく Δ_1 関数と呼ばれるのはそのためである。

補題 3.7 $i \geq 1$ とする。X を Σ_i 集合、f を Δ_1 関数とすると、X の f による逆像 $f^{-1}(X)$ は Σ_i 集合である。同様に、X を Π_i 集合、f を Δ_1 関数とすると $f^{-1}(X)$ は Π_i 集合である。

証明　集合 X が Σ_i 論理式 $\varphi_X(x)$ により記述でき、関数 f が Σ_1 論理式 $\varphi_f(x, y)$ により記述できるとき、集合 $f^{-1}(X)$ は論理式 $\exists y(\varphi_f(x, y) \wedge \varphi_X(y))$ により記述できる。実際、任意の $n \in \mathbb{N}$ について以下が成り立つ：

$$\begin{aligned} n \in f^{-1}(X) &\iff \text{ある } m \text{ について } f(n) = m \text{ かつ } m \in X \\ &\iff \text{ある } m \text{ について } \mathbf{N} \models \varphi_f(\mathsf{n}, \mathsf{m}) \text{ かつ } \mathbf{N} \models \varphi_X(\mathsf{m}) \\ &\iff \mathbf{N} \models \exists y(\varphi_f(\mathsf{n}, y) \wedge \varphi_X(y)) \end{aligned}$$

この論理式が Σ_i 論理式と同値であることは容易に確かめることができる。
二番目の主張については練習問題とする。 ■

練習問題 3.8

1. Δ_1 関数の定義において、Σ_1 論理式 $\varphi_f(x,y)$ は $\varphi_f(x,y) \equiv \exists z.\psi(x,y,z)$ の形であるとする。ただし $\psi(x,y,z)$ は Δ_0 論理式である。このとき、Π_1 論理式

$$\varphi'_f(x,y) \quad \equiv \quad \forall z \forall w(\psi(x,w,z) \to w = y)$$

をとれば、全ての $n, m \in \mathbb{N}$ について

$$f(n) = m \quad \Longleftrightarrow \quad \mathbf{N} \models \varphi'_f(\mathsf{n}, \mathsf{m})$$

を満たすことを確かめよ。

2. 補題 *3.7* の二番目の主張を確かめよ。

3.3 論理的複雑さと計算の複雑さ

前節で、0-1 列の集合 X が再帰的であるとはどういうことか、また再帰的枚挙可能であるとはどういうことかを定義した。実をいうと、これらは論理の言葉でいえば Δ_1 集合であること、Σ_1 集合であることと正確に対応するのである。（自然数は 2 進数（0-1 列）を用いて表せるので、再帰性・再帰的枚挙可能性は自然数の集合に関する性質だと思って差し支えない。）

定理 3.9 X を自然数の集合とする。

1. X は再帰的枚挙可能である $\Longleftrightarrow$ X は Σ_1 集合である。
2. X は再帰的である $\Longleftrightarrow$ X は Δ_1 集合である。
3. f は再帰的である $\Longleftrightarrow$ f は Δ_1 関数である。

証明 1 の $\Longleftarrow$ を示す。まず、限定文の真偽は有限時間で判定可能であることに注意する。たとえば、$\mathsf{n}_1 + \mathsf{n}_2 = \mathsf{n}_3$ の真偽を判定するには、$n_1 + n_2$ を計算して、その結果を n_3 と比べればよい。また、$\forall x \le t.\varphi(x)$ の真偽を調べるためには、まず t の値を計算する。仮にそれが n だとすると、$\forall x \le t.\varphi(x)$ の真偽は

$$\varphi(\mathsf{0}) \wedge \varphi(\mathsf{1}) \wedge \cdots \wedge \varphi(\mathsf{n})$$

の真偽と一致するから、$\varphi(\mathsf{0}), \ldots, \varphi(\mathsf{n})$ の真偽を確かめてゆけばよい。
さて、X を Σ_1 集合とする。X が再帰的枚挙可能であることを示すには、

$$n \in X \quad \Longleftrightarrow \quad M(n) = 1$$

を満たすテューリング機械 M が存在することを言えばよい。定義により X を記述する Σ_1 論理式 $\varphi(x) \equiv \exists y.\psi(x,y)$ が存在し $n \in X \iff \mathbf{N} \models \varphi(\mathsf{n})$ が成り立つことに注意する。(話を簡単にするために、量化子∃の数は一つと仮定している。$\psi(x,y)$ は限定論理式である。)

次のようなアルゴリズムを考える。まず $\psi(\mathsf{n},0)$ の真偽を判定し、もし真ならば "1" を出力する。さもなければ $\psi(\mathsf{n},1)$ の真偽を判定し、もし真ならば "1" を出力する。さもなければ $\psi(\mathsf{n},2)$ の真偽を判定する。このようにして、

$$\psi(\mathsf{n},0), \quad \psi(\mathsf{n},1), \quad \psi(\mathsf{n},2), \quad \psi(\mathsf{n},3), \ldots$$

と真偽判定のプロセスを続けていく。各 $\psi(\mathsf{n},\mathsf{k})$ は限定文だから、その真偽は有限時間で判定可能である。また、$\mathbf{N} \models \varphi(\mathsf{n})$ が成り立つことはある $k \in \mathbb{N}$ について $\mathbf{N} \models \psi(\mathsf{n},\mathsf{k})$ が成り立つことと同値である。ゆえに、上記のアルゴリズムは $\mathbf{N} \models \varphi(\mathsf{n})$、つまり $n \in X$ のとき、またそのときに限り "1" を有限時間内に出力する。チャーチ・テューリングのテーゼにより上のアルゴリズムを実現するテューリング機械が存在する。

1 の $\Longrightarrow$ については、証明の概略を以下で述べる。

1 が成り立てば、2 は定理 2.7 と補題 3.5 を用いて簡単に証明できる :

$$\begin{aligned} X \text{ は再帰的である} &\iff X \text{ と } \overline{X} \text{ は再帰的枚挙可能である} \\ &\iff X \text{ と } \overline{X} \text{ は } \Sigma_1 \text{ 集合である} \\ &\iff X \text{ は } \Sigma_1 \text{ 集合であり、かつ } \Pi_1 \text{ 集合でもある} \\ &\iff X \text{ は } \Delta_1 \text{ 集合である。} \end{aligned}$$

3 の証明は省略する。 ■

3.4 算術階層の厳密性

本節では、前章で導入した算術階層が厳密であることを示す。即ち、$\Sigma_0 = \Pi_0 = \Delta_0$ であることを除いて、クラス間の包含関係 $\subseteq$ は実際には $\subsetneq$ であることを示す。証明はカントールの**対角線論法**による。

Σ_1 集合とは一変数 Σ_1 論理式により定義される集合のことであった。一変数 Σ_1 論理式は、ゲーデル数が小さいほうから順番に数えることで、0 番目の一変数 Σ_1 論理式 $\varphi_0(x)$、1 番目の一変数 Σ_1 論理式 $\varphi_1(x)$、2 番目の一変数 Σ_1 論理式 $\varphi_2(x)$ というようにして全て枚挙することができる :

$$\varphi_0(x), \quad \varphi_1(x), \quad \varphi_2(x), \ldots$$

よって各 $\varphi_i(x)$ が記述する Σ_1 集合を X_i とおけば、

$$X_0, \quad X_1, \quad X_2, \ldots$$

というふうに Σ_1 集合をすべて枚挙することができる (同じ集合が何度も重複してあらわれることになるが気にしない)。いま、集合 K を $K = \{n | n \in X_n\}$ により定義する。すると次の性質が成り立つ。

補題 3.10 (Σ_1 対角化)

1. $K \in \Sigma_1$

2. $K \notin \Pi_1$

証明 1. 定理 3.9 より、K を受理するテューリング機械の存在を言えばよい。まず、関数 $n \mapsto \varphi_n(x)$ は明らかに計算可能である。そして

$$n \in K \quad \Longleftrightarrow \quad n \in X_n \quad \Longleftrightarrow \quad \mathbf{N} \models \varphi_n(\mathsf{n})$$

である。$\mathbf{N} \models \varphi_n(\mathsf{n})$ が成り立つかどうかの判定方法は定理 3.9 の証明中で示した通りである。これらを組み合わせてチャーチ・テューリングのテーゼを適用すれば、

$$n \in K \quad \Longleftrightarrow \quad M(n) = 1$$

を満たすテューリング機械 M が得られる。

2. 仮に K が Π_1 集合であるとすると、補題 3.5 により $\overline{K}$ は Σ_1 集合となる。ゆえにある k について $X_k = \overline{K}$ となるはずであるが、

$$k \notin \overline{K} \quad \Longleftrightarrow \quad k \in K \quad \Longleftrightarrow \quad k \in X_k \quad \Longleftrightarrow \quad k \in \overline{K}$$

となり矛盾する。 ■

ここで用いられた論法が対角線論法と呼ばれる理由は次のとおりである。

自然数の集合 $X \subseteq \mathbb{N}$ が与えられたとき、

$$\begin{aligned} x_i &= 1 \quad i \in X \text{ のとき} \\ &= 0 \quad i \notin X \text{ のとき} \end{aligned}$$

とおく。すると X に対して、0 と 1 から成る無限列

$$x_0, x_1, x_2, \ldots$$

を対応させることができる。例えば、偶数の集合 $Even = \{0, 2, 4, \ldots\}$ には

$$1, 0, 1, 0, 1, 0, \ldots$$

が対応し、素数の集合 $Prime = \{2, 3, 5, 7, 11 \ldots\}$ には

$$0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, \ldots$$

が対応する。逆にこのような無限列が与えられたら、そこから自然数の集合 X を一意に復元することができる。

さて、本章の最初で定義した Σ_1 集合の列 $X_0, X_1, X_2, \ldots$ の各要素 X_i に対して上のように 0,1 の無限列を対応させると、次のような表を得ることができる。

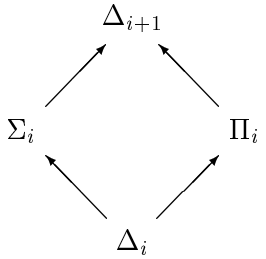
	0	1	2	3	$\cdots$
X_0	0	0	1	1	$\cdots$
X_1	0	1	1	0	$\cdots$
X_2	0	0	1	0	$\cdots$
X_3	1	0	1	0	$\cdots$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$

この表の対角線 $0, 1, 1, 0, \ldots$ をとると、それ自体 0,1 の無限列になっている。集合 K はこの無限列に相当する。そしてその補集合 $\overline{K}$ は、対角線の 0,1 を逆にして得られる無限列 $1, 0, 0, 1, \ldots$ に相当する。補題 3.10 が示しているのは、K はこの表の中に何らかの X_k として現れるが、$\overline{K}$ はこの表の中には現れないということである。(なぜならばいかなる X_k についても、$\overline{K}$ と X_k は、その k 番目の要素について異なっているからである。)

補題 3.10 およびその一般形から、次の定理が帰結する。

定理 3.11 (算術階層の厳密性)

1. $\Sigma_1 \not\subseteq \Pi_1$、$\Pi_1 \not\subseteq \Sigma_1$
2. $\Delta_1 \subsetneq \Sigma_1$、$\Delta_1 \subsetneq \Pi_1$
3. $\Sigma_1 \subsetneq \Delta_2$、$\Pi_1 \subsetneq \Delta_2$
4. 一般に算術階層は厳密である。すなわち、各 $i \geq 1$ について包含関係



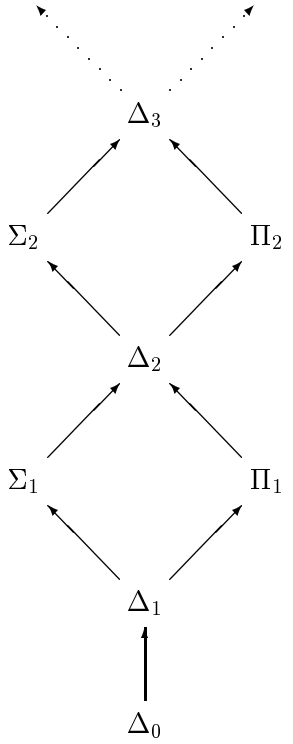
は厳密である。

証明　1. は補題 3.10 より、$K \in \Sigma_1$ かつ $K \notin \Pi_1$ であること、また逆に $\overline{K} \in \Pi_1$ かつ $\overline{K} \notin \Sigma_1$ であることからいえる。

2., 3. は 1. から容易に示すことができる。

4. は以上の事柄の一般化である。 ■

かくして強固にそびえたつ高層建築・算術階層の構造が明らかになった。



3.5 整数や有限数列の表現

算術の言語は一見貧弱である。実際、関数記号は $+, \cdot, S$ しか含んでいないので、引き算 $x - y$ やべき乗 x^y が表現できるかどうかはそのままではわからない。引き算を表すには、自然数のみならず整数全体へと数の領域を拡張する必要がある。べき乗を表すには、有限数列 $(a_1, \ldots, a_n)$ を一つの自然数で表す必要が出てくる。本節では、算術の言語を用いて整数を表したり、べき乗を表したりすることができることを説明する。

順序対の表現　最初に、自然数の順序対 (m, n) を一つの自然数で表す方法を述べる。このことは $\mathbb{N}^2$ から $\mathbb{N}$ への単射を与えることに相当する。まず、$\mathbb{N}^2$ の要素を次のように並べてみる。

$$
\begin{array}{l}
(0,0) \\
(1,0), (0,1) \\
(2,0), (1,1), (0,2) \\
(3,0), (2,1), (1,2), (0,3) \\
(4,0), (3,1), (2,2), (1,3), (0,4) \\
\quad \vdots
\end{array}
$$

この並べ方において、

- 一番上の行を 0 行目、一番左の列を 0 列目と考える。すると (m,n) は $m+n$ 行目の n 列目に現れる（たとえば $(1,2)$ は 3 行目の 2 列目に現れる）。
- i 行目は $i+1$ 個の要素からなる（例えば 4 行目は 5 個の要素からなる）。

以上のことに注意すると、(m,n) は

$$\left(\sum_{i=0}^{i=m+n-1}(i+1)\right)+n=\left(\sum_{i=1}^{i=m+n} i\right)+n=\frac{(m+n)(m+n+1)}{2}+n$$

番目に登場することがわかる。言い換えれば、2 項関数

$$\langle x,y\rangle := \frac{(x+y)(x+y+1)}{2}+y$$

は $\mathbb{N}^2$ から $\mathbb{N}$ への全単射を与えることがわかる。つまり、算術の論理式

$$\mathsf{op}(x,y,z) \quad \equiv \quad 2z=(x+y)(x+y+1)+2y$$

を考えると、次のことが成り立つ。

$$\mathbf{N}\models \forall x\forall y\exists! z.\mathsf{op}(x,y,z), \qquad \mathbf{N}\models \forall x\exists! x\exists! y.\mathsf{op}(x,y,z).$$

以後、$\mathsf{op}(x,y,z)$ と書く代わりに $\langle x,y\rangle = z$ と書く。また、$\exists y \leq z.\mathsf{op}(x,y,z)$, $\exists x \leq z.\mathsf{op}(x,y,z)$ の代わりに $\pi_1(z)=x$, $\pi_2(z)=y$ と書く。これらの表現は他の関数と組み合わせることができる。たとえば $\langle t,u\rangle + v = w$ は、算術の論理式を用いて

$$\exists z(\mathsf{op}(t,u,z)\wedge z+v=w)$$

と書き下すことができる。また、$\langle t,u\rangle \leq (t+u)(t+u+1)$ が成り立つから、同じことは限定論理式を用いて

$$\exists z\leq (t+u)(t+u+1)(\mathsf{op}(t,u,z)\wedge z+v=w)$$

と書くこともできる。

順序対を用いれば、３つ組や４つ組も自然に表すことができる。

$$\begin{aligned}\langle m_1,m_2,m_3\rangle &:= \langle m_1,\langle m_2,m_3\rangle\rangle\\ \langle m_1,m_2,m_3,m_4\rangle &:= \langle m_1,\langle m_2,\langle m_3,m_4\rangle\rangle\rangle\end{aligned}$$

ゆえに、算術の言語があれば、どんな**固定長**の数列も一つの自然数を用いて表すことができる（つまり任意の自然数 $k\geq 1$ について $\mathbb{N}^k$ から $\mathbb{N}$ への全単射を与えることができる）。より難易度が高いのは**可変長**の数列を一つの自然数を用いて表すこと（つまり $\mathbb{N}^*$ から $\mathbb{N}$ への全単射を与えること）である。これについては後で論じる。

整数の表現　次に自然数を用いて整数を表す方法について考える。最も自然なのは「整数とは二つの自然数の差である」という考え方である。たとえば順序対 $(5,3)$ は整数 $5-3=2$ を、$(3,5)$ は整数 $3-5=-2$ を表すものとみなすのである。すると、(m,n) が整数 k を表すとき、$-k$ は単に (n,m) となり、大変都合がよい。ただしこの考え方の弱点は、一つの整数を表す順序対がたくさん存在してしまうことである。たとえば $(3,5),(4,6),(5,7),\ldots$ はみな同じ整数 -2 を表してしまう。しかしこの問題を回避するのは簡単で、整数の表現法に合わせて等号 $=$ を再定義してしまえばよいのである。いま、順序対の間の同値関係 $\doteq$ を

$$(n,m) \doteq (k,l) \quad \Longleftrightarrow \quad n+l=m+k$$

により定義する。ここで $n,m,k,l \in \mathbb{N}$ であり右辺の $=$ は自然数間の等号である。すると $(3,5) \doteq (4,6) \doteq (5,7) \doteq \cdots$ となり、どんな整数も自然数の順序対により（$\doteq$ の意味で同値なものを除いて）一意に表すことができる。さらに次のように演算 $-,+,\cdot$ を定義できる。

$$\begin{aligned} -(n,m) &:= (m,n) \\ (n,m)+(k,l) &:= (n+k,m+l) \\ (n,m)\cdot(k,l) &:= (n\cdot k+m\cdot l, n\cdot l+m\cdot k) \end{aligned}$$

上の定義の右辺は関数記号 $+,\cdot$ しか使っていないから、算術の言語を用いて表現できる。また前項で示したとおり、自然数の順序対は一つの自然数で表すことができる。ゆえに算術の言語さえあれば、整数環 $\mathbb{Z}=(\mathbb{Z},+,\cdot,-,0,1)$ について十分に語ることができる。

似たような発想に従えば、有理数体 $\mathbb{Q}=(\mathbb{Q},+,\cdot,-,0,1)$ も算術の言語を用いて語ることができる。

数列の表現　次に可変長の数列を一つの自然数で表す方法について考える。いま、$m \geq 1$, $a,b \in \mathbb{Z}$ とするとき、m を法とする合同関係を

$$a \equiv b \pmod{m} \quad \Longleftrightarrow \quad m|(a-b)$$

により定義する。また、a と b が 1 以外の公約数を持たないとき、a と b は**互いに素** (relatively prime) であるという。すると次のことが成り立つ。

補題 3.12 (中国剰余定理) $k \geq 1$, $a_1,\ldots,a_k \in \mathbb{Z}$ とする。また $m_1,\ldots,m_k \geq 1$ は互いに素であるとする。このとき、連立合同方程式

$$\begin{aligned} x &\equiv a_1 \pmod{m_1} \\ &\vdots \\ x &\equiv a_k \pmod{m_k} \end{aligned}$$

は常に解 $a \in \mathbb{Z}$ を持つ。しかもそのような a は $m_1 \cdots m_k$ を法として一意に定まる。

上の定理は $a_1,\ldots,a_k$ を自然数に限っても当然成り立つ。ゆえに十分に大きくかつ互いに素であるような k 個の自然数 $m_1,\ldots,m_k$ を生成することができれば、k 個の自然数 $a_1,\ldots,a_k$ を一つの自然数 a で表すことができる。実際、上の方程式を満たす自然数 a は常に存在するし、逆にそのような a が与えられたら、m_i で割って余りを求めれば a_i を復元することができる。

互いに素な自然数は次のようにすれば簡単に生成することができる。

補題 3.13 $0 \leq i < j \leq m$ ならば、$1 + im!$ と $1 + jm!$ は互いに素である。

証明 仮に $d \neq 1$ かつ $d|(1 + im!)$, $d|(1 + jm!)$ となる $d \in \mathbb{N}$ が存在したとする。そのような中で最小の数を選ぶことにより、d は素数であると仮定してよい。すると $d|(1 + jm!) - (1 + im!)$、つまり $d|(j - i)m!$ である。d は素数であり、$j - i \leq m$ だから $d|m!$、ゆえに $d|jm!$ である。このことと仮定 $d|(1 + jm!)$ を合わせると $d|1$ となるがこれは矛盾である。 ■

以上で数列を表現する準備が整った。限定論理式 $\mathsf{rm}(x, y, z)$ および $\mathsf{beta}(x, i, z)$ を次のように定義する。

$$\begin{aligned} \mathsf{rm}(x, y, z) &\equiv \exists w \leq x(x = yw + z \wedge z < y) \\ \mathsf{beta}(x, i, z) &\equiv \exists a \leq x. \exists m_0 \leq x(x = \langle a, m_0 \rangle \wedge \mathsf{rm}(a, 1 + im_0, z)). \end{aligned}$$

すると $\mathbf{N} \models \forall x \forall i \exists! z. \mathsf{beta}(x, y, z)$ が成り立つことは明らかである。実際 x が与えられれば $x = \langle a, m_0 \rangle$ となる a, m_0 は一意に定まるし、さらに i が与えられれば $\mathsf{rm}(a, 1 + im_0, z)$ となる z も一意に定まるからである。以下、$\mathsf{beta}(x, i, z)$ と書く代わりに $\beta(x, i) = z$ と書く。これがゲーデルの β **関数**である（解析学におけるベータ関数とは関係ない）。

β 関数は次の性質を満たす。

補題 3.14 自然数 $a_1, \ldots, a_k$ が与えられたとき、ある n が存在し、$\beta(n, 1) = a_1, \ldots, \beta(n, k) = a_k$ が成り立つ。

すなわちどんな可変長の自然数列 $a_1, \ldots, a_k$ も一つの自然数 n で表すことができ、逆に n が与えられたとき、β 関数を用いれば $a_1, \ldots, a_k$ を復元することができる。

証明 $m := \max(k, a_1, \ldots, a_k)$ とし、$m_0 := m!$ とおく。すると $1+m_0, 1+2m_0, \ldots, 1+km_0$ は互い素である。ゆえに連立合同方程式

$$\begin{aligned} x &\equiv a_1 \pmod{1 + m_0} \\ &\vdots \\ x &\equiv a_k \pmod{1 + km_0} \end{aligned}$$

は解 $a \in \mathbb{N}$ を持つので、$n := \langle a, m_0 \rangle$ とすればよい。 ■

β 関数を定義するのには算術の言語で十分であることに注意しよう。特に階乗関数 $x!$ は β 関数の定義自体には必要ない。また定義全体は限定論理式になっている。これを用いることにより、様々な関数が Σ_1 論理式を用いて定義可能になる。例えば、指数関数 x^y は次のように定義できる。

$$x^y = z \quad \equiv \quad \exists w. \beta(w, 1) = 1 \wedge \forall 1 \leq i \leq y(\beta(w, i + 1) = x \cdot \beta(w, i)) \wedge \beta(w, y + 1) = z.$$

ただしここで $\forall 1 \leq i \leq y. \varphi$ は $\forall i \leq y(1 \leq i \to \varphi)$ の省略表現である。この定義は自然数上の指数関数の定義式

$$x^0 := 1, \qquad x^{i+1} := xx^i$$

にのっとっている。このように i についての帰納法により関数を構成する方法を**原始再帰法** (primitive recursion) という。上のように β 関数をうまく使えば、原始再帰法により構成できる関数はすべて Σ_1 論理式を用いて記述することができる。

3.6 テューリング機械の表現

定理 3.9(1) で再帰的枚挙可能集合と Σ_1 集合が一致することを主張したが、まだ証明のうち一方向が残っていた。すなわち、どんな再帰的枚挙可能集合も Σ_1 集合であることを示さなければならない。このことは結局、テューリング機械の動作は算術の論理式を用いて記述できるということに帰着する。ここで本質的な役割を果たすのが β 関数である。

テューリング機械による計算は、テープ上で記号を読み書きしたり、ヘッドを左右に動かしたりすることで進んでいく。いま、テープの状況、ヘッドの位置、内部状態を合わせて**計算状況** (configuration) ということにすれば、テューリング機械による計算は、遷移関数 δ に従って計算状況を変えていくことだと考えることができる。

$$c_1 \quad \stackrel{\delta}{\longrightarrow} \quad c_2 \quad \stackrel{\delta}{\longrightarrow} \quad c_3 \quad \stackrel{\delta}{\longrightarrow} \quad \cdots$$

さて、各計算状況を一つの自然数で表すことを考える。まずテープの状況についてだが、テープは左右に無限にのびてはいるものの、有限時間内に消費することができるのはそのうちの有限部分にすぎない。ゆえに、Σ に含まれる記号の数を n とすれば、テープの状況は n 進数を用いて表すことができる。たとえば $\Sigma = \{0, 1, \sqcup\}$ のときには 3 進数を用いる。また、ヘッドの位置を表すには、テープをヘッドの位置で二分すればよい。最後に、内部状態 $q \in Q$ はもちろん自然数で表すことができる。

いま、テューリング機械 $M = (\Sigma, Q, \delta, q_I, q_F)$ が与えられているとし、$\Sigma = \{0, 1, \sqcup\}$ とする。次のような計算状況を考える。

$$\begin{array}{ll} \text{テープの状況：} & \cdots \sqcup 011 \sqcup 101\underline{0}111 \sqcup 100 \sqcup \cdots \\ \text{内部状態：} & \qquad\qquad\qquad\quad q \end{array}$$

ただし下線はヘッドの位置をあらわし、$q \in \mathbb{N}$ とする。このような計算状況は自然数

$$c \quad = \quad \langle (20112101)_3, (200121110)_3, q \rangle$$

で表すことができる。ここで 2 は空白記号に対応し、$(20112101)_3$ は 3 進数表記すると 20112101 となる自然数を表す。第一要素はヘッドの左側のテープの状況を表し、第二要素ははヘッド位置を含めた右側のテープの状況を反転したものを表す。第三要素は内部状態である。

一般に計算状況 $\langle l, r, q \rangle$ が与えられたとき、ヘッド位置の記号を読むには r を 3 で割って余り a を求めればよい（テープの右側は反転されているので、r の 1 の位がヘッドの位置に対応する)。いま $\delta(q, a) = (q', a', \rightarrow)$ とすると、次の計算状況を求めるには、

$$r = 3r' + a, \quad a \in \{0, 1, 2\}$$

とした上で、$l' := 3l + a'$ とする。このとき $\langle l', r', q' \rangle$ が新しい計算状況を表す。

$$\langle l, r, q \rangle \quad \stackrel{\delta}{\longrightarrow} \quad \langle l', r', q' \rangle.$$

このように考えれば、次のような事柄を表す Σ_1 論理式を構成することができるはずである。

$$\begin{array}{rcl} \mathsf{init}(w, c) & \Longleftrightarrow & c \text{ は入力 } w \in \mathbb{N} \text{ が与えられたときの初期計算状況である} \\ \mathsf{tran}(c_1, c_2) & \Longleftrightarrow & \text{遷移関数 } \delta \text{ に従えば } c_1 \text{ の次の計算状況は } c_2 \text{ である} \\ \mathsf{accept}(c) & \Longleftrightarrow & c \text{ は状態 } q_F \text{ でありテープには 1 が書かれているような計算状況である} \end{array}$$

(ここでは入力 w は自然数と考えている。)

すると次のことが成り立つ。

$$\begin{aligned} M(w)=1 &\iff \mathsf{init}(w,c_1)\text{かつ } c_1 \xrightarrow{\delta} \cdots \xrightarrow{\delta} c_{k+1}\text{かつ } \mathsf{accept}(c_{k+1}) \\ &\quad \text{となる計算状況列 } c_1,\ldots,c_{k+1} \text{ が存在する} \\ &\iff \mathbf{N} \models \exists x.\exists k.\ \ \mathsf{init}(w,\beta(x,1)) \\ &\qquad\qquad \wedge \forall 1 \le i \le k.\mathsf{tran}(\beta(x,i),\beta(x,i+1)) \\ &\qquad\qquad \wedge \mathsf{accept}(\beta(x,k+1)). \end{aligned}$$

ここで右辺の論理式は Σ_1 論理式である。

以上のことから、どんな再帰的枚挙可能集合も Σ_1 集合であることがわかり、定理 3.9(1) の $\Longrightarrow$ が成り立つことがわかる。

3.7 真理述語の定義不能性

本章では、算術階層の厳密性の第一の帰結として、真理述語の定義不能性（タルスキ）を証明する。

$\varphi \equiv \varphi(x)$ を一変数論理式とする。このとき関数 $f_\varphi : \mathbb{N} \longrightarrow \mathbb{N}$ を次のように定義する。

$$f_\varphi(n) := \ulcorner \varphi(\mathsf{n}) \urcorner.$$

適切なゲーデル符号化のもとでは、この関数は再帰的である（Δ_1 関数である）と仮定してよい。

このとき $f_\varphi^{-1}(\mathsf{TA})$ は、$\varphi(x)$ により記述される集合 X_φ と一致する。実際、

$$n \in X_\varphi \iff \mathbf{N} \models \varphi(\mathsf{n}) \iff \ulcorner \varphi(\mathsf{n}) \urcorner \in \mathsf{TA} \iff f_\varphi(n) \in \mathsf{TA} \iff n \in f_\varphi^{-1}(\mathsf{TA})$$

が成り立つ。

定理 3.15 (真理述語の定義不能性) 算術の論理式について「真である」という性質は、算術の論理式によっては定義することができない。すなわち、真な文（のゲーデル数）の集合 TA は算術の論理式によっては記述することができない。

証明 仮に TA が算術の論理式によって記述可能であるとすると、補題 3.5 により、TA はある k について Σ_k 集合となるはずである。ゆえに補題 3.7 により任意の一変数論理式 $\varphi(x)$ について $X_\varphi = f_\varphi^{-1}(\mathsf{TA})$ も Σ_k 集合となるはずである。よって算術階層に属する全ての集合が Σ_k 集合となり、算術階層は崩壊してしまう。しかしこのことは算術階層の厳密性に反する。 ■

よって、算術の論理式一般に関する真理概念は算術の言語によっては定義することができない。“真実の書” は算術階層の “外” にある。

一方、各 Σ_n についての “真実の書” はちゃんと算術階層の中にある。

定理 3.16 (各層ごとの真理述語の定義可能性) 各 $n \ge 1$ について、真な Σ_n 文（のゲーデル数）全体の集合 $\mathsf{TA}_n \subseteq \mathbb{N}$ は Σ_n 集合である。

3.8 理論 Q：小学生レベルの数学者

例 1.18 で紹介した理論 $\mathbf{Q}$ を思い出そう。$\mathbf{Q}$ は以下の論理式の全称閉包からなる。

$$\begin{array}{lll} S(x) \neq 0 & S(x) = S(y) \to x = y & x \neq 0 \to \exists y(x = S(y)) \\ x + 0 = x & x + S(y) = S(x + y) & \\ x \cdot 0 = 0 & x \cdot S(y) = x \cdot y + x & \end{array}$$

古典論理 $\mathbf{CL}$ においてこれらの公理から論理式 φ が導出できるとき、$\mathbf{Q}$ は φ を証明できるといい $\vdash_{\mathbf{Q}} \varphi$ と書く。この $\mathbf{Q}$ の証明能力について調べるのが本節の課題である。

補題 3.17 n, m を自然数とする。

1. $n + m = k$ ならば $\vdash_{\mathbf{Q}} \mathsf{n} + \mathsf{m} = \mathsf{k}$.
2. $nm = k$ ならば $\vdash_{\mathbf{Q}} \mathsf{n} \cdot \mathsf{m} = \mathsf{k}$.
3. $n \neq m$ ならば $\vdash_{\mathbf{Q}} \mathsf{n} \neq \mathsf{m}$.
4. $\vdash_{\mathbf{Q}} x \leq \mathsf{n} \to x = \mathsf{0} \vee x = \mathsf{1} \vee \cdots \vee x = \mathsf{n}$.

証明 1. m についての数学的帰納法による。$m = 0$ のときには $\mathbf{Q}$ の公理 $\forall x.x + 0 = 0$ に $(\forall E)$ 規則を適用して $\vdash_{\mathbf{Q}} \mathsf{n} + \mathsf{0} = \mathsf{n}$ が得られる。$m = m' + 1$ のときには $\mathbf{Q}$ の公理から $\vdash_{\mathbf{Q}} \mathsf{n} + \mathsf{m} = S(\mathsf{n} + \mathsf{m}')$. $n + m' = k'$ とすると、帰納法の仮定から $\vdash_{\mathbf{Q}} \mathsf{n} + \mathsf{m}' = \mathsf{k}'$. これらに等号の公理を用いれば、$\vdash_{\mathbf{Q}} \mathsf{n} + \mathsf{m} = S(\mathsf{k}')$、すなわち $\vdash_{\mathbf{Q}} \mathsf{n} + \mathsf{m} = \mathsf{k}$ が言える。

2., 4. は省略。

3. については一般性を失わずに $n > m$、すなわちある $k \in \mathbb{N}$ について $n = m + k + 1$ としてよい。公理 $\forall x.\forall y.S(x) = S(y) \to x = y$ を m 回用いることにより $\mathsf{n} = \mathsf{m} \vdash_{\mathbf{Q}} S(\mathsf{k}) = 0$ がいえる。ここで公理 $\forall x.S(x) \neq 0$ を用いれば $\mathsf{n} = \mathsf{m} \vdash_{\mathbf{Q}} \bot$、よって $(\to I)$ 規則により $\vdash_{\mathbf{Q}} \mathsf{n} = \mathsf{m} \to \bot$. ■

補題 3.18 φ を限定文とする。

1. $\mathbf{N} \models \varphi$ ならば $\vdash_{\mathbf{Q}} \varphi$.
2. $\mathbf{N} \not\models \varphi$ ならば $\vdash_{\mathbf{Q}} \neg\varphi$.

証明 まず、t を変数を含まない項とし、その値を n とするとき、$\vdash_{\mathbf{Q}} t = \mathsf{n}$ が成り立つことに注意する。

補題の証明は φ の構造に関する帰納法による。いくつかの場合だけ取り上げる。

- φ が原子文 $t = u$ のとき。t, u の値をそれぞれ n, m とすると、$\vdash_{\mathbf{Q}} t = \mathsf{n}$ および $\vdash_{\mathbf{Q}} u = \mathsf{m}$ が成り立つ。

  いま、$\mathbf{N} \models t = u$ のときには $n = m$ であるからから、等号の公理より $\vdash_{\mathbf{Q}} t = u$ が言える。

  一方、$\mathbf{N} \not\models t = u$ のときには $n \neq m$ であるから上の補題により $\vdash_{\mathbf{Q}} \mathsf{n} \neq \mathsf{m}$. これを用いて $\vdash_{\mathbf{Q}} t \neq u$ を導くことができる。

- φ が $\neg\psi$ の形のとき。$\mathbf{N} \models \varphi$ ならば、$\mathbf{N} \not\models \psi$. よって帰納法の仮定により $\vdash_{\mathbf{Q}} \neg\psi$. $\mathbf{N} \not\models \varphi$ ならば、$\mathbf{N} \models \psi$. 帰納法の仮定により $\vdash_{\mathbf{Q}} \psi$. これより $\vdash_{\mathbf{Q}} \neg\neg\psi$ が帰結する。
- φ が $\forall x \leq t.\psi(x)$ の形のとき。$\mathbf{N} \models \varphi$ ならば、t の値を n として

$$\mathbf{N} \models \psi(\mathsf{0}), \quad \mathbf{N} \models \psi(\mathsf{1}), \quad \cdots, \quad \mathbf{N} \models \psi(\mathsf{n}).$$

帰納法の仮定により

$$\vdash_{\mathbf{Q}} \psi(\mathsf{0}), \quad \vdash_{\mathbf{Q}} \psi(\mathsf{1}), \quad \cdots, \quad \vdash_{\mathbf{Q}} \psi(\mathsf{n}).$$

練習問題 1.16(2) を用いて

$$\vdash_{\mathbf{Q}} x = \mathsf{0} \to \psi(x), \quad \vdash_{\mathbf{Q}} x = \mathsf{1} \to \psi(x), \quad \cdots, \quad \vdash_{\mathbf{Q}} x = \mathsf{n} \to \psi(x).$$

よって $\vdash_{\mathbf{Q}} x = \mathsf{0} \vee \cdots \vee x = \mathsf{n} \to \psi(x)$. 補題 3.17(4) により $\vdash_{\mathbf{Q}} x \leq \mathsf{n} \to \psi(x)$. これより $\vdash_{\mathbf{Q}} \forall x \leq t.\psi(x)$ が従う。$\mathbf{N} \not\models \varphi$ の場合は省略。 ■

定理 3.19 (Q の Σ_1 完全性) φ を Σ_1 文とする。$\mathbf{N} \models \varphi$ ならば $\vdash_{\mathbf{Q}} \varphi$.

証明 $\varphi \equiv \exists x.\psi(x)$ とする（ψ は限定文）。$\mathbf{N} \models \varphi$ とすると、ある $n \in \mathbb{N}$ が存在し $\mathbf{N} \models \psi(\mathsf{n})$. 補題 3.18 より $\vdash_{\mathbf{Q}} \psi(\mathsf{n})$. よって $(\exists I)$ 規則により $\vdash_{\mathbf{Q}} \exists x.\psi(x)$. ■

ゆえに $\mathbf{Q}$ は Σ_1 文については完全な証明能力を持っている。例えば $\mathsf{prime(x)}$ は限定論理式であり、$\mathsf{prime(223)}$ は真な限定文、よって真な Σ_1 文である。それゆえ $\mathbf{Q}$ は $\mathsf{prime(223)}$ を証明できる。さらに $\varphi \equiv \exists y(\mathsf{1000000000} \leq y \wedge \mathsf{prime(y)})$（“1000000000 以上の素数が存在する”）を考えると、これも真な Σ_1 文であるから、$\vdash_{\mathbf{Q}} \varphi$. しかし $\psi \equiv \forall x \exists y(x \leq y \wedge \mathsf{prime(y)})$（“素数は無限に多く存在する”）を考えるとこれは Π_2 文であるから、$\vdash_{\mathbf{Q}} \psi$ とは言えない。（実際 $\nvdash_{\mathbf{Q}} \psi$ である。）

また、$\vdash_{\mathbf{Q}} \mathsf{n+m} = \mathsf{m+n}$ がどんな $n, m \in \mathbb{N}$ についても成り立つが、一方で $\nvdash_{\mathbf{Q}} \forall x \forall y(x + y = y + x)$ である。（前者は限定文、それゆえ Σ_1 文であるのに対し、後者は Π_1 文である。）

このように $\mathbf{Q}$ は具体的な数に関する計算については十分な能力を持つが、加法の可換性が証明できないなど、全称量化された変数を含む文になると途端におぼつかなくなる。変数の扱いが不十分という点で、$\mathbf{Q}$ は “理想化された小学生レベルの数学者” であると考えることができるかもしれない。

3.9 Q と CL の決定不能性

理論 $\mathbf{Q}$ は任意の Σ_1 文 φ について

$$\mathbf{N} \models \varphi \quad \Longrightarrow \quad \vdash_{\mathbf{Q}} \varphi$$

を満たすが、逆にどんな文 φ についても

$$\vdash_{\mathbf{Q}} \varphi \quad \Longrightarrow \quad \mathbf{N} \models \varphi$$

を満たす。このことを $\mathbf{Q}$ は**健全である** (sound) という。以上をまとめると次のことが成り立つ。

定理 3.20 (Σ_1 弱表現可能性) $\varphi \equiv \varphi(x)$ を一変数 Σ_1 論理式とする。このとき

$$n \in X_\varphi \quad \Longleftrightarrow \quad \mathbf{N} \models \varphi(\mathsf{n}) \quad \Longleftrightarrow \quad \vdash_{\mathbf{Q}} \varphi(\mathsf{n}).$$

このことから即座に $\mathbf{Q}$ の**決定不能性** (undecidability) が帰結する。すなわち文 φ が与えられたとき、$\vdash_{\mathbf{Q}} \varphi$ が成り立つかどうかをアルゴリズム的に判定する方法は存在しない。

定理 3.21 ($\mathbf{Q}$ の決定不能性) 集合 $\mathsf{Prov}_{\mathbf{Q}} = \{\ulcorner\varphi\urcorner \mid \vdash_{\mathbf{Q}} \varphi\}$ は再帰的ではない。

証明　定理 3.20 により、任意の一変数 Σ_1 論理式 $\varphi(x)$ について

$$n \in X_\varphi \quad \Longleftrightarrow \quad \vdash_{\mathbf{Q}} \varphi(\mathsf{n}) \quad \Longleftrightarrow \quad \ulcorner\varphi(\mathsf{n})\urcorner \in \mathsf{Prov}_{\mathbf{Q}} \quad \Longleftrightarrow \quad n \in f_\varphi^{-1}(\mathsf{Prov}_{\mathbf{Q}})$$

が成り立つ。仮に $\mathsf{Prov}_{\mathbf{Q}}$ が再帰的であるとすると、定理 3.9 より Δ_1 集合である。ゆえに補題 3.7 により $f_\varphi^{-1}(\mathsf{Prov}_{\mathbf{Q}}) = X_\varphi$ も Δ_1 集合となる。よって全ての Σ_1 集合は Δ_1 集合であることになってしまうが、このことは算術階層の厳密性に反する。 ■

さらに $\mathbf{Q}$ が有限個の公理の集合であることから、古典一階述語論理の決定不能性が帰結する。

系 3.22 (一階述語論理の決定不能性) 集合 $\mathsf{Prov}_{\mathbf{CL}} = \{\ulcorner\varphi\urcorner \mid \vdash_{\mathbf{CL}} \varphi\}$ は再帰的ではない。

証明　$\mathbf{Q}$ の公理を全て $\wedge$ により結んで得られる文を $\bigwedge Q$ と書く。すると、

$$\vdash_{\mathbf{Q}} \varphi \quad \Longleftrightarrow \quad \vdash_{\mathbf{CL}} \bigwedge Q \to \varphi$$

が成り立つ。ゆえに、もしも一階述語論理の文 ψ について $\vdash_{\mathbf{CL}} \psi$ かどうかを判定する方法があれば、そのような方法は $\mathbf{Q}$ についても使うことができることになるが、これは上の定理に反する。 ■

この系が述べているのは、つまり「どんな問いにも正しく答えてくれるような万能コンピュータは存在しない」ということである。SF などに出てくる万能コンピュータの典型的イメージは次のようなものだろう。そのコンピュータには世界に関するありとあらゆる情報（個人情報、地図情報、etc.）そしてありとあらゆる法則（物理法則、国家法律、etc.）が入力されており、質問者が「φ か？」（Panasonic の株価は半月以内に上昇するか？、etc.）というふうに質問を入力すると、コンピュータはデータベースにある情報から論理的推論（およびその他の推論）に基づいてイエスかノーかの解答を与えてくれる。もしもそんなコンピュータができてしまったらどうしようか、というのが大昔の SF の典型的な問題意識だったのだが、何もそんな心配をする必要はない。そのような万能コンピュータはそもそも存在しえないからである。

もちろん、だからといってコンピュータによる知的問題解決は完全に不可能だということではなく、例えば人工知能分野においては、限られた分野（医療、法律等）に関する限られた種類の質問に対して信頼度の高い答えを返してくれるエキスパートシステムの研究が継続されている。その際には一階述語論理全体の決定不能性を踏まえた上で、それを回避するよう問題設定を工夫するのが常である。このように、決定不能性は確かに否定的な結果ではあるが、実用的システム構築の指針として積極的な役割をも果たしうるのである。

3.10　理論 PA:中学生レベル？の数学者

3.8 節で述べたとおり、理論 **Q** は計算能力はあるが証明能力は非常に弱い。その最大の理由は、数論の最大の武器である**数学的帰納法** (mathematical induction) を **Q** が備えていないことにある。そこで **Q** に数学的帰納法を“教え込む。”そのようにして得られるのが**ペアノ算術** (Peano arithmetic) **PA** である。

理論 **PA** は **Q** に次の形の論理式の全称閉包（数学的帰納法の公理）をすべて加えることにより得られる：

$$(\forall x.\varphi(x) \to \varphi(S(x))) \to \varphi(0) \to \forall x.\varphi(x).$$

ここで φ は任意の算術の論理式である。それゆえ **Q** と異なり PA は無限に多くの文を含む。

この拡張により、**PA** は次のような推論を行うことができる。

$$\frac{\Gamma \vdash \varphi(0) \quad \Gamma, \varphi(x) \vdash \varphi(S(x))}{\Gamma \vdash \forall x.\varphi(x)}\ (ind)$$

ただし Γ は自由変数 x を含まないものとする。

この新たな推論規則を用いれば、基本的な命題はすべて証明できる。たとえば **PA** は加法の可換性を証明できることを見てみよう。

命題 3.23 **PA** は以下の文を証明できる。

1. $\forall x.0 + x = x.$
2. $\forall x.S(y) + x = S(y + x).$
3. $\forall x.\forall y.x + y = y + x.$

証明　1. Q の公理、等号の公理、(ind) を用いて次のような証明図を書くことができる。

$$\dfrac{\dfrac{}{\vdash \forall x.x + 0 = x}\ (\forall E)}{\vdash 0 + 0 = 0} \qquad \dfrac{\dfrac{0 + x = x \vdash \forall y.y + S(x) = S(y + x)}{0 + x = x \vdash 0 + S(x) = S(0 + x)} \quad \dfrac{\dfrac{}{0 + x = x \vdash 0 + x = x}\ (init)}{0 + x = x \vdash S(0 + x) = S(x)}}{0 + x = x \vdash 0 + S(x) = S(x)}\ (ind)}{\vdash \forall x.0 + x = x}$$

ここで (ind) を適用する際には $\varphi(x) \equiv 0 + x = x$ としている。

2. 同様にして、まず $\varphi(x) \equiv S(y) + x = S(y + x)$ について $\vdash \varphi(0)$ と $\varphi(x) \vdash \varphi(S(x))$ を示した上で (ind) を使えばよい。

3. 1 と 2 より $\vdash x + 0 = 0 + x$ および $x + y = y + x \vdash x + S(y) = S(y) + x$ は簡単に示せる。ゆえに (ind) により $\vdash \forall y.x + y = y + x$ が証明できる。 ■

かくして変数を用いた命題も自在に扱えるようになった。変数をうまく使えるかどうか、これは小学校の算数と中学校の数学の一番の違いであると言ってよい。また、数学的帰納

法自体も中学校の数学の要である。ゆえに **PA** はほぼ中学生レベルの数学者を理想化したものだ、ということができるかもしれない。

ただし、**PA** の証明能力は一見するよりもはるかに強力である。まず、初等数論に出てくる定理はほぼ全部証明できる。また、コード化を通して有理数、さらには代数的数についての基本的性質も証明できる。さらに驚くべきことに、近年 A. Macintyre は次のことを主張している。いま、

$$\mathsf{FLT} \quad \equiv \quad \forall x \forall y \forall z \forall n \geq 3 (x^n + y^n = z^n \to xyz = 0)$$

とすると $\vdash_{\mathbf{PA}} \mathsf{FLT}$ が成り立つ。すなわち **PA** はフェルマーの最終定理を証明できる！もちろんこのことは、現実の中学生がフェルマーの最終定理を自力で証明できるということを意味しない。また、ワイルズの証明を中学生が読んで理解できるということも意味しない。そうではなく、フェルマーの最終定理の証明は少なくとも原理上は算術の文だけを用いて書き下すことができ、おそらくそれは何万ページにも及ぶものになるだろうが、中学生レベルの知識さえあれば、少なくとも原理上はその 1 ステップ 1 ステップを追っていくことができるだろうということである。これだけでも十分に驚くべきことではないだろうか。

一方、**PA** にも不得手はある。既に示したように自然数の有限列は一つの自然数によりコード化することができるが、自然数の無限列についてはそのようなコード化はできない。自然数の無限列が取り扱えないということは、実数や無限集合をうまく取り扱えないということである。これらは **PA** の守備範囲を超え出ており、2 階算術なり集合論なりに移行する必要がある。

数列の極限や関数の微積分を含む実数の本格的な取り扱いが始まるのは高校数学である。その手の事柄が手に負えないという点でも、**PA** は中学生レベルの数学者の理想化であると考えることには一理あるように思われる。このような擬人化には危険もあるが、利点も多い。とくに第二不完全性定理を直感的に理解するには、理論を単に論理式の集合としてではなく、数学者の理想化として捉えることが大いに役立つはずだというのが持論である。

実数や無限集合についての諸命題はそもそも（一階）算術の言語で表現しえないものであり、そのような命題が **PA** で証明できないのはごく当然の話である。一方、仮に算術の言語で表現できる命題に限ったとしても、**PA** の証明能力には限界がある。具体例としてはヒドラゲームの必勝性等が挙げられる。（ヒドラゲームについては http://math.andrej.com/2008/02/02/the-hydra-game/ を参照。“Try the game online” をクリックすれば自分がヘラクレスになってヒドラと対戦できる。）

最後に **PA** の証明能力について若干捕捉しておく。

命題 3.24 任意の $\varphi(x)$ について **PA** は以下の論理式を証明できる（両者は対偶の関係にあることに注意）。

1. 強帰納法: $\forall x(\forall y < x.\varphi(y) \to \varphi(x)) \to \forall x.\varphi(x)$.
2. 最小値原理: $\exists x.\varphi(x) \to \exists x(\varphi(x) \wedge \forall y < x.\neg\varphi(y))$.

3.11 第一不完全性定理

理論 **Q** の証明能力は貧弱なので公理を追加する必要があった。そこで理論 **PA** を導入したのだが、**PA** の証明能力にもやはり限界がある。さらに証明能力を向上するためにはさらに公理を追加すればよいのだが、公理はどんなふうに追加してもよいわけではない。そこには自然な制約がある。まず第一に、公理を追加した結果矛盾が生じるようでは困る。また、無限に多くの公理を追加することが可能なのだが（実際、**PA** は無限に多くの公理からなる)、そもそもどんな公理が追加されたのかがわからないようでは困る。たとえば、TA は無限個の文の集合であり、これらを全部公理として認めれば最強の理論ができあがる。実際、理論 TA は真な文全てを証明できる。しかしこの理論の問題は、一体何が公理なのか我々にはわからないという点である。むしろどんな文 φ が TA に含まれるかを我々は知りたいのであり、そのための手段として公理系は存在するのである。これでは本末転倒といえよう。

そこでまずは理論の拡大について有用な概念をまとめておこう。

定義 3.25 (理論の拡大) 算術の文の集合を理論という。理論 T, U について $T \subseteq U$ が成り立つとき、U は T の**拡大** *(extension)* である、あるいは T は U の縮小であるという。集合 $\{\ulcorner\varphi\urcorner \mid \varphi \in T\}$ が再帰的なとき T は**再帰的** *(recursive)* であるという。また $\vdash_T \bot$ が成り立つとき T は**矛盾している** *(inconsistent)* という。さもなければ T は**無矛盾である** *(consistent)*。最後に次のことが成り立つとき T は **1-無矛盾である** *(1-consistent)* という：任意の Σ_1 文 φ について

$$\vdash_T \varphi \quad \Longrightarrow \quad \mathbf{N} \models \varphi.$$

理論 T が 1-無矛盾ならば無矛盾である。実際、$\mathbf{N} \not\models \bot$ だから 1-無矛盾性の対偶をとれば $\not\vdash_T \bot$ が言える。健全性（証明可能な文はすべて真であること）はもちろん 1-無矛盾性を含意する。それゆえ条件の強さの順序は次のようになる。

$$\text{無矛盾} \quad < \quad \text{1-無矛盾} \quad < \quad \text{健全}$$

Q や **PA** は健全であるが、そうでないような算術の理論もありうる。

再帰性について言えば、これは理論が公理系としての役割を果たすための最低限の要請であるといってよい。何が公理なのかがわからないようでは、とても公理系とはいえないからである。

定理 3.20 は 1-無矛盾性の仮定のもとで **Q** の拡大一般について成り立つ事実である。ゆえに定理 3.21 は次のように一般化できる。

定理 3.26 (Q の拡大の決定不能性) T を理論 **Q** の *1*-無矛盾な拡大とすると、$\mathsf{Prov}_T = \{\ulcorner\varphi\urcorner \mid \vdash_T \varphi\}$ は再帰的集合ではない。

つまり Prov_T は Δ_1 集合ではない。一方で次のことが再帰的理論一般について成り立つ。

定理 3.27 理論 T が再帰的ならば Prov_T は Σ_1 集合である。

証明　論理式 φ が与えられたとする。$\ulcorner\varphi\urcorner \in \mathsf{Prov}_T$ となるのは、ある有限集合 $\Gamma \subseteq T$ が存在して $\Gamma \vdash \varphi$ が証明図を持つときに限る。一方、証明図 π が与えられたとき、その下端に

現れるシークエント $\Gamma \vdash \psi$ は簡単に求めることができる。また T は再帰的だから、$\Gamma \subseteq T$ かどうかも決定することができる。

証明図は自然数を用いてコード化可能であるから $\pi_0, \pi_1, \pi_2, \ldots$ と枚挙していくことができる。これらのそれぞれについて下端が $\Gamma \vdash \varphi$ の形になっているかどうか、そして $\Gamma \subseteq T$ かどうかを調べていけばよいので、Prov_T は再帰的枚挙可能であり、それゆえ Σ_1 論理式により記述できる。 ■

注意 3.28 実をいうと、全ての再帰的枚挙可能理論（Σ_1 理論）にはそれと同等な再帰的理論が存在することが証明されている *(Craig)*。ゆえに、以下の諸定理は「再帰的」を「再帰的枚挙可能」と読み替えても全て成立する。

以上でゲーデルの第一不完全性定理を証明する準備が整った。ゲーデルの定理は後述のようにモデル論的な真という概念とは独立に述べられることが多いが、次の形で述べたほうが第二不完全性定理との関わりが見えやすい。

定理 3.29 (Π_1 不完全性) T を $\mathbf{Q}$ の再帰的拡大とする。もしも T が無矛盾ならば、真であるが T には証明できない Π_1 文が存在する。

証明 T は Π_1 健全である。すなわち T が証明できる Π_1 文は全て真である。実際、$\vdash_T \varphi$ かつ $\mathbf{N} \not\models \varphi$ となる Π_1 文 φ が存在するとしたら、$\neg\varphi$ は真な Σ_1 文（と論理的に同値）なので、Σ_1 完全性により $\vdash_{\mathbf{Q}} \neg\varphi$、そして T が Q の拡大であることから $\vdash_T \neg\varphi$ となる。よって T は矛盾するがこれは仮定に反する。

いま、さらに T が Π_1 完全であるとすると、任意の Π_1 文 φ について

$$\mathbf{N} \models \varphi \iff \vdash_T \varphi$$

となる。ゆえに任意の一変数 Π_1 論理式 $\psi(x)$ について

$$n \in X_\psi \iff \mathbf{N} \models \psi(\mathsf{n}) \iff \vdash_T \psi(\mathsf{n}) \iff \ulcorner \psi(\mathsf{n}) \urcorner \in \mathsf{Prov}_T \iff n \in f_\psi^{-1}(\mathsf{Prov}_T)$$

となる。定理 3.27 により Prov_T は Σ_1 集合であるから、補題 3.7 により $X_\psi = f_\psi^{-1}(\mathsf{Prov}_T)$ も Σ_1 集合になる。ゆえに $\Pi_1 \subseteq \Sigma_1$ が帰結するが、これは算術階層の厳密性に反する。 ■

次の定理が一般にゲーデルの第一不完全性定理と呼ばれるものである。

系 3.30 (第一不完全性) T を $\mathbf{Q}$ の再帰的拡大とする。もしも T が *1*-無矛盾ならば、φ も $\neg\varphi$ も T には証明できないような Π_1 文 φ が存在する。

証明 Π_1 不完全性により、真でありかつ証明不可能な Π_1 論理式 φ が存在する。$\neg\varphi$ は偽な Σ_1 論理式（と論理的に同値）であるから、1-無矛盾性の対偶により $\neg\varphi$ は T では証明不可能である。 ■

ただしゲーデルのオリジナルの定理は ω-無矛盾性という 1-無矛盾性よりも強い仮定に基づいている。なお本講義では取り上げないが、ロッサーのトリックをつかえば 1-無矛盾性ないし ω-無矛盾性という仮定は単なる無矛盾性に弱めることができる。

系 3.31 (第一不完全性（ロッサー）) T を $\mathbf{Q}$ の再帰的拡大とする。もしも T が無矛盾ならば、φ も $\neg\varphi$ も T には証明できないような Π_1 文 φ が存在する。

Π_1 不完全性は Σ_1 完全性と対比して理解するとよい。すなわち真な Σ_1 文を証明するには $\mathbf{Q}$ という非常に弱い理論だけで十分なのに対し、真な Π_1 文を全部証明することはどんなに多くの公理を $\mathbf{Q}$ に追加しても叶わない。数学的帰納法よりも強力な論法としては、たとえば順序数に関する**超限帰納法** (transfinite induction) が挙げられる。これはある程度算術の言語で記述可能なものであるが、どんなに強力な超限帰納法の公理を追加しても、決して全ての真な Π_1 文を証明し尽くすことはできない。

さらには土台となる枠組みを 2 階算術や集合論に拡大しても、不完全性は必ず生じる。一般に、自然数上の加減乗算を実現でき、「すべての自然数について $\cdots$」というような量化ができるような理論については、不完全性定理が必ず成り立つ。それゆえ、どんなに公理を加えても、またどんなに枠組みを拡大しても、全ての真理を覆い尽くすことは決してできない。この意味で数学は本質的に開かれているというのが第一不完全性定理の意義である。

一方で、ある程度豊かな理論でありながら完全であるような例も存在する。例えば**代数閉体** (algebraically closed field) の理論や**プレスバーガー算術** (Presburger arithmetic, **PA** から乗算記号および乗算に関する公理を取り除いたもの）がその例である。どちらも完全かつ決定可能である。なぜ不完全性が生じないかというと、前者は「すべての自然数について $\cdots$」という量化ができず、後者は乗算ができないからである。

ゲーデルの定理は 20 世紀の思想界に大きな影響を与えた。とくに人間とコンピュータの関係についての SF 的空想やインスピレーションの源となったようである。このように数学の定理から思想的洞察を引き出すのは大いに歓迎されるべきであるが、その際に定理の数学的内容を誤って理解していてはもともこもない。たとえば、ある人はこう主張する。

> 「コンピュータというのは、一定の規則に従って形式的に推論を行うものであるから、それは公理系 T のようなものだろう。そうするとゲーデルの定理により T には証明も反証もできない文 φ が存在する。一方で我々人間は意味的思考により φ が真であることを認識できる。ゆえに意味的思考能力を持つ我々人間は形式的推論しかできないコンピュータに勝る。」

これは完全に間違った主張であり、ゲーデルの定理を完全に誤解している。なぜ間違っているかは次節の第二不完全性定理の議論を見れば明らかだろう。

さすがに上のような議論を真顔で展開する人は今ではいないだろうが、それでも多少洗練された形でコンピュータの「形式的推論」と人間の「意味的思考」を対比しようとする傾向は少なからず存在する。そのような議論を有意義な形で展開するためには、まずゲーデルの定理の数学的内容を正確に理解することが肝要である。

3.12 第二不完全性定理

理論 T を $\mathbf{PA}$ の再帰的拡大とする。$\mathbf{Q}$ ではなく $\mathbf{PA}$ の拡大としている理由はすぐに明らかになる。このとき、Π_1 不完全性定理は次の性質を持つ文 φ が存在することを主張する：

1. T が無矛盾ならば φ は真である.

2. T が無矛盾ならば $\nvdash_T \varphi$.

Π_1 不完全性定理に至るまでの証明を注意深く検討すれば、この φ は具体的に構成することが可能である。そして、そのような具体的な φ に話を制限すれば、算術を超え出るような超越的な論法は一切用いずに Π_1 不完全性定理は証明可能である。実際、Π_1 不完全性を証明する際に本質的な役割を果たすのは

- 対角線論法
- $\mathbf{Q}$ の Σ_1 完全性
- Prov_T が Σ_1 集合であること

である。そして、ここが核心なのであるが、これらを証明する際に我々は中学生レベルの数学しか使ってこなかった。実際、論理式や証明図はコード化できるから自然数のことだと思ってよいし、論証には論理的推論と自然数の基本性質、そして数学的帰納法しか使っていない。特に実数の極限操作や微積などは一切使っていないし、無限集合についての量化（「すべての集合 $X \subseteq \mathbb{N}$ について $\cdots$ 」）も行っていない。だとすると、（理論 T は $\mathbf{PA}$ の拡大であることに注意すれば）**これまでの議論は全て T も行えるはずである!**

特に上記の 1 の論証を T にやらせることで、次のことが得られる。

1'. $\vdash_T \mathsf{Con}_T \to \varphi$

ただし Con_T は T の無矛盾性を表す文である。これは、集合 Prov_T を記述する論理式を同じ記号を用いて $\mathsf{Prov}_T(x)$ と書き、$\ulcorner \bot \urcorner = n$ とすれば、$\mathsf{Con}_T \equiv \neg \mathsf{Prov}_T(\mathsf{n})$ と定義することができる。上記 1' の帰結として、$\vdash_T \mathsf{Con}_T$ ならば $\vdash_T \varphi$ である。しかし後者は 2 に反するから、結局

3. T が無矛盾ならば $\nvdash_T \mathsf{Con}_T$.

すなわち、十分強力かつ無矛盾な理論は、自分自身の無矛盾性を証明することができない。これがゲーデルの第二不完全性定理である。

定理 3.32 (第二不完全性) 理論 T を $\mathbf{PA}$ の再帰的拡大とする。もしも T が無矛盾ならば $\nvdash_T \mathsf{Con}_T$。

以上の議論は大変面白い構造をしている。すなわち、我々がすでに証明した第一不完全性定理を、“理想化された数学者” たる T にもう一度証明させることによって第二不完全性定理が得られるのである。これはちょっと普通の数学者では思いつかない、数学基礎論に特有の発想であるといえる。

第一不完全性同様、第二不完全性も高度に普遍性を持つ現象である。すなわちこれまでの証明を遂行できる理論であれば、どんな理論についても第二不完全性定理は成り立つ。実際には $\mathbf{PA}$ ほど強力な必要はなく、せいぜい Σ_1 論理式について数学的帰納法が使えれば十分である。

第二不完全性定理はヒルベルトのプログラムに対するアンチテーゼとして重要である。ヒルベルトのプログラムの骨子は、理論 T の安全性（無矛盾性）を保証するために、T よりも具体的で確実な理論（有限主義数学）に Con_T を証明させるということに尽きる。T より確実ということは、T よりも弱いということでもある。しかし第二不完全性定理によれば T 自身ですら Con_T を証明できないのに、それよりも弱い理論に Con_T が証明できるはずはないのである。ゆえにヒルベルトのプログラムは深刻な問題点をはらむことが明らかになった。

ただし、第二不完全性定理が成り立つからといって、あまり悲観的になりすぎる必要もない。一方で、**PA** の無矛盾性に関してはいくつかの肯定的結果が知られているからである。いま、理論 T が**有限公理化可能** (finitely axiomatizable) であるとは、ある有限集合 T' が存在して、$\mathsf{Prov}_T = \mathsf{Prov}_{T'}$ が成り立つこととする。

定理 3.33 (局所的な無矛盾性の証明可能性) 理論 $T \subseteq PA$ が有限公理化可能ならば、$\vdash_{\mathbf{PA}} \mathsf{Con}_T$.

系 3.34 (PA の本質的無限性) **PA** は有限公理化不可能である。

次に、任意の $n \geq 1$ について、理論 $I\Sigma_n$ とは、**Q** に Σ_n 論理式に関する数学的帰納法の公理を全て付け加えることにより得られる理論のことであるとする。当然、$I\Sigma_n \subseteq \mathbf{PA}$ である。これらの **PA** の部分理論については、次のことが知られている。

定理 3.35 (各層ごとの有限公理化可能性) 任意の $n \geq 1$ について、$I\Sigma_n$ は有限公理化可能である。

系 3.36 (各層ごとの無矛盾性の証明可能性) 任意の $n \geq 1$ について、$\vdash_{\mathbf{PA}} \mathsf{Con}_{I\Sigma_n}$.

どのように大きな n をとったとしても、このことは成立する。特に、通常の数学において用いられる数学的帰納法を全て包含するほどに大きな Σ_n を取ったとしても、**PA** は、$I\Sigma_n$ の無矛盾性を証明できるのである。

以上のことから得られる教訓は、確かに **PA** については第二不完全性が成り立つが、それはもう本当に「紙一重」の話であり、実質上どんなに大きな有限部分についても無矛盾性証明は可能なのである。数学の確実な基礎を求めて無矛盾性証明を行うというパラダイムは、ゲーデルの第二不完全性定理により一見ついえたように見える。しかしここで大事なのは、悲観的になったり諦めたりせずに事実をあるがままに捉えること、すなわちこの「紙一重性」を正確に理解することであろう。

ゲーデルの不完全性定理に直面して、フォンノイマンをはじめとするヒルベルト学派の多くの研究者がヒルベルトのプログラムの遂行を断念した。しかしそれでも証明論的探究を諦めなかったのがゲンツェンである。彼は 1930 年代後半の一連の論文で **PA** の無矛盾性を証明する。それが数学の基礎づけにどの程度寄与するかについては議論の余地があるが、いずれにせよ彼の結果は **PA** の証明論的本質を明らかにするものであり、また彼の開発した技法はやがて 20 世紀後半の計算機科学に多大なる影響を与えていくことになるのである。

4　ラムダ計算 — パラドックスからプログラミングへ

数学者の重要な活動の一つに、関数を作ったり使ったりすることが挙げられる。たとえば式 $x^2+3\cdot x+5$ において x を自然数上を動く変数と考えることによって、関数

$$f(x)=x^2+3\cdot x+5$$

を作る。作られた関数を使うには x に値を代入して（たとえば $x=5$ として）

$$f(5) \quad \longrightarrow \quad 5^2+3\cdot 5+5 \quad \longrightarrow \quad 45$$

とすればよい。しかし式 $x^2+3\cdot x+5$ から関数を作る際にいちいち名前 f をつけるのは面倒であり、とくに関数を大量に作る場合には名前不足になりがちである。そこで式 $x^2+3\cdot x+5$ から作られる関数を f の代わりに $\lambda x.x^2+3\cdot x+5$ と書くことにする。そして関数を使うには単に $(\lambda x.x^2+3\cdot x+5)5$ のように、単に入力値 5 を関数の後に書くことにする。（このような書き方は例えば関数解析では標準的である。）すると

$$(\lambda x.x^2+3\cdot x+5)5 \quad \longrightarrow \quad 5^2+3\cdot 5+5 \quad \longrightarrow \quad 45$$

のように計算を実行することができる。これがラムダ計算の記法である。

関数の中には、関数を入力値としてとるような関数も存在する。たとえば微分演算子 $\frac{d}{dx}$ は $X\subseteq\mathbb{R}$ 上で微分可能な関数 $f:X\longrightarrow\mathbb{R}$ が与えられたとき、その導関数 $\frac{d}{dx}f:X\longrightarrow\mathbb{R}$ を返す、いわば “関数の関数” である。そうすると “関数の関数の関数” や “関数の関数の関数の関数” なども原理上考えることができる。では、いっそのこと数学的対象は全て関数だと思い、3 や 5 のような具体的な数も関数を使って構成することにしてはどうだろうか。

そのようにして得られるのがチャーチが 1930 年代に考案した**ラムダ計算** (lambda calculus) である。ラムダ計算は、関数を作ったり使ったりという操作を極限まで理想化したものである。具体的な数を第一義的には含まないにも関わらず、その表現力は計り知れず、実際、ラムダ計算は計算装置としてテューリング機械と同等のパワーを持っている。ラムダ計算のメカニズムはリスト処理言語 LISP に取り入れられており、また関数型言語 Scheme、ML、Haskell 等の中核となっている。本節では、このラムダ計算について概観する。

4.1　定義

述語論理やテューリング機械の各章を通して、そろそろ形式的な議論にも慣れてきたことと思うので、ここではあまり余計な説明はせずに形式的な定義を一気にしてしまうことにする。具体例は次の節で挙げる。

まず、述語論理の場合と同様に、変数 $x,y,z,\ldots$ が与えられているとする。ラムダ計算の基本表現は、変数からはじめてラムダ抽象、適用という二種類の操作を行うことにより構成される。

定義 4.1 (ラムダ項) 次の *BNF* 記法により定義される表現 M,N を**ラムダ項** *(lambda term)* という。

$$M,N \ ::= \ x \mid (\lambda x.M) \mid (MN).$$

$(\lambda x.M)$ の形のラムダ項を**ラムダ抽象** (lambda abstraction) といい、表現 M から関数を作る操作に対応する。また、(MN) の形のラムダ項を**適用** (application) といい、入力値 N 対して関数 M を使う操作に対応する。

不要なかっこは適宜省略してもよいとする。また、次の省略表現を用いる。

$$\begin{aligned} \lambda x_1 \cdots x_n.M &\equiv (\lambda x_1.\cdots(\lambda x_{n-1}.(\lambda x_n.M))\cdots) \\ MN_1\cdots N_n &\equiv (\cdots((MN_1)N_2)\cdots N_n) \end{aligned}$$

ここで $(\lambda x.M)$ は束縛表現であり、M の中に現れる x は先頭の λx により束縛される。このような変数を**束縛変数** (bound variable) という。どのような λx にも束縛されない変数を**自由変数** (free variable) という。要は λx は述語論理の場合の $\forall x$ や $\exists x$ と似たようなものと思ってくれればよい。自由変数を含まないラムダ項を**閉項** (closed term) という。ラムダ項全体の集合を Λ により表す。

項の代入　ラムダ項 M の中に現れる自由変数 x に注目するときには、$M \equiv M[x]$ のように書く。そして $M[x]$ の中に現れる自由変数 x 全てに N を代入して得られるラムダ項を $M[N]$ と書く。(述語論理の場合は普通のかっこを用いて $\varphi(x)$ と書いたが、ここでは四角いかっこを用いて $M[x]$ と書く。$M(x)$ と書くとラムダ項 M を変数 x に適用した表現と見分けがつかないからである。)

述語論理の場合と同様に、次の取り決めをしておく。

- 束縛変数の名前は、束縛関係を変えない限り自由に付け変えてよい。

それゆえ $\lambda x.M[x]$ と $\lambda y.M[y]$ は同じラムダ項を表す。また、代入を行う際に変数の衝突を避けるため、次の約束をしておく。

- ラムダ項をラムダ項に代入するときには、新たな束縛関係が生じないよう、事前に束縛変数の名前を付け替えておく。

ゆえに $M[y] \equiv (\lambda x.xy)$ のとき $M[(xz)]$ は $(\lambda x.x(xz))$ ではない。そうではなく、束縛変数 x を w と名前変えし、$M[y] \equiv (\lambda w.wy)$ とした上で、$M[(xz)] \equiv (\lambda w.w(xz))$ とするのが正しい。

β 簡約　テューリング機械と同様ににラムダ計算は一つの計算装置である。各ラムダ項は (関数型) プログラムを表す。そしてプログラムはどのようにして実行されるかというと、それは次の **β 簡約規則** (beta reduction rule) による。

$$(\lambda x.M[x])N \quad \longrightarrow_\beta \quad M[N].$$

ここで $(\lambda x.M[x])N$ の形のラムダ項を **β 基** (beta redex) という。すなわちラムダ計算における計算とは、β 基をみつけて上の β 簡約規則を使って次のラムダ項を求める、という操作を繰り返し行うことに他ならない。(なぜ "β" なのかというと、チャーチによるオリジナルの体系では束縛変数の名前のつけ変え $\lambda x.M[x] \equiv \lambda y.M[y]$ も計算規則と考えられており、α 規則と呼ばれていたからである。)

意味的に考えれば、β 簡約規則はごく当然のことを言っているにすぎない。なぜならば $(\lambda x.M[x])$ は「入力値 N が与えられたとき $M[N]$ を出力するような関数」を表し、そのよ

うな関数を値 N に適用するというのが $(\lambda x.M[x])N$ の意味であるから、その出力は当然 $M[N]$ となる。

計算は、

$$(\lambda x.x(xy))(\lambda z.wz) \longrightarrow_\beta (\lambda z.wz)((\lambda z.wz)y) \longrightarrow_\beta (\lambda z.wz)(wy) \longrightarrow_\beta w(wy)$$

というように進む。ところで上の例で二番目のラムダ項を見てみると、二つの β 基を含んでいることがわかる。実際、$(\lambda z.wz)((\lambda z.wz)y)$ 全体が一つの β 基であるし、その内部に含まれる $(\lambda z.wz)y$ も β 基である。上の例ではまず後者の小さいほうの β 基に β 簡約規則を用いて三番目のラムダ項を求めたが、順番を変えて前者に β 簡約規則を用いても構わない。すると計算は次のようになる。

$$(\lambda x.x(xy))(\lambda z.wz) \longrightarrow_\beta (\lambda z.wz)((\lambda z.wz)y) \longrightarrow_\beta w((\lambda z.wz)y) \longrightarrow_\beta w(wy).$$

いずれにせよ、計算結果 $w(wy)$ は変わらない。

最後に次の定義をしておく。

定義 4.2 (β 簡約、β 同値、正規形) *1.* ラムダ項 M に β 簡約規則を何回か用いることにより N に到達できるとき、$M \longrightarrow_\beta^* N$ と書く。もう少し詳しく言うと、ラムダ項の列 $M_0, \cdots, M_n$ *($n \geq 0$)* があり

$$M \equiv M_0 \longrightarrow_\beta M_1 \longrightarrow_\beta \cdots \longrightarrow_\beta M_n \equiv N$$

となるとき $M \longrightarrow_\beta^* N$ と書く。β 簡約規則を用いる回数は *0* 回でもいいので、$M \longrightarrow_\beta^* M$ は常に成り立つ。

2. ラムダ項 M に β 簡約規則とその逆を何回か用いることにより N に到達できるとき、$M =_\beta N$ と書く。すなわち、ラムダ項の列 $M_0, \cdots, M_n$ *($n \geq 0$)* があり

$$M \equiv M_0 \longleftrightarrow_0 M_1 \longleftrightarrow_1 \cdots \longleftrightarrow_{n-1} M_n \equiv N$$

となるとき $M =_\beta N$ と書く。ただし各 $\longleftrightarrow_i$ は $\longrightarrow_\beta$ か $_\beta \longleftarrow$ のどちらかを表す。

3. β 基を含まないラムダ項を**正規形** *(normal form)* という。

上の例により、$(\lambda x.x(xy))(\lambda z.wz) \longrightarrow_\beta^* w(wy)$ であり、また $w((\lambda z.wz)y) =_\beta (\lambda z.wz)(wy)$ である。$w(wy)$ は正規形であり、それ以上 β 簡約を続けることはできない。

4.2 ラムダ項を用いた関数表現

本節ではいくつかの重要なラムダ項を定義する。本節で挙げるラムダ項を組み合わせれば、だいたいどんな計算も表現できることがわかるだろう。

恒等関数と関数合成　まずは最も基本的な関数である恒等関数と、与えられた関数を合成する方法からはじめる。

$$\mathsf{id} := \lambda z.z, \qquad M_1 \circ M_2 := \lambda z.M_1(M_2 z), \qquad M_1 \circ \cdots \circ M_n := \lambda z.M_1(M_2 \cdots (M_n z) \cdots)$$

ただし z は $M_1, \ldots, M_n$ の中には現れない新しい変数とする。このように定義すれば

$$\mathsf{id}\ M \longrightarrow_\beta M, \qquad (M_1 \circ M_2)N \longrightarrow_\beta M_1(M_2 N)$$

となることは明らかだろう。さらに次のようなことも成り立つ：

$$\mathsf{id} \circ M =_\beta \lambda z.Mz =_\beta M \circ \mathsf{id}, \qquad M_1 \circ (M_2 \circ M_3) =_\beta M_1 \circ M_2 \circ M_3 =_\beta (M_1 \circ M_2) \circ M_3.$$

ブール値とブール関数　次にブール値（真、偽）と、それらを入出力値とするブール関数をラムダ計算の中で表現する方法を考える。

$$\begin{array}{rclrcl} \mathsf{true} & := & \lambda xy.x & \mathsf{false} & := & \lambda xy.y \\ \mathsf{neg} & := & \lambda bxy.byx & \mathsf{conj} & := & \lambda b_1 b_2 xy.b_1(b_2 xy)y \end{array}$$

λ が連続する場合の省略の仕方や名前の付け替えに関する約束に注意すると、

$$\mathsf{neg} \equiv \lambda b.\lambda x.\lambda y.(by)x, \qquad \mathsf{true} \equiv \lambda x'.\lambda y'.x'$$

と書ける。ゆえに次のような計算ができる：

$$\mathsf{neg\ true} \longrightarrow_\beta \lambda x.\lambda y.((\lambda x'.\lambda y'.x')y)x \longrightarrow_\beta \lambda x.\lambda y.(\lambda y'.y)x \longrightarrow_\beta \lambda x.\lambda y.y \equiv \mathsf{false}$$

同様にして $\mathsf{neg\ false} \longrightarrow_\beta^* \mathsf{true}$ も言えるので、neg は否定を表す関数であることが分かる。

練習問題 4.3

1. $\mathsf{conj\ true\ false} \longrightarrow_\beta^* \mathsf{false}$ となることを確かめよ。
2. conj は連言（かつ）を表す。同様にして、選言（または）を表すラムダ項を一つ求めよ。

順序対、条件文　次に、順序対の作り方を定める。

$$\langle M_1, M_2 \rangle := \lambda z.zM_1M_2, \qquad \mathsf{proj}_1 M := M\,\mathsf{true}, \qquad \mathsf{proj}_2 M := M\,\mathsf{false}.$$

このようにすれば

$$\mathsf{proj}_1\langle M_1, M_2 \rangle \longrightarrow_\beta^* M_1, \qquad \mathsf{proj}_2\langle M_1, M_2 \rangle \longrightarrow_\beta^* M_2$$

となる。また、

$$\mathsf{if}\ N\ \mathsf{then}\ M_1\ \mathsf{else}\ M_2 \quad := \quad NM_1M_2$$

とすれば、

$$\mathsf{if\ true\ then}\ M_1\ \mathsf{else}\ M_2 \longrightarrow_\beta^* M_1, \qquad \mathsf{if\ false\ then}\ M_1\ \mathsf{else}\ M_2 \longrightarrow_\beta^* M_2$$

が成り立つ。

チャーチ数項と繰り返し: 自然数 $n \geq 1$ を表すラムダ項を

$$\mathsf{n} := \lambda f.\underbrace{f \circ \cdots \circ f}_{n}$$

と定義する。すなわち「与えられた関数を n 回合成したものを返す関数」をもって自然数 n と定義するのである。ただし $\mathsf{0} := \lambda f.\mathsf{id}$ とする。これは**チャーチ数項** (Church numeral) と呼ばれる。繰り返し関数を

$$\mathsf{iter} := \lambda fgm.mfg.$$

と定義すれば、次が成り立つ。

$$\mathsf{iter}\ M\ N\ \mathsf{n} \longrightarrow_\beta^* \mathsf{n}\ M\ N \longrightarrow_\beta (\underbrace{M \circ \cdots \circ M}_{n})N \longrightarrow_\beta \underbrace{M(M \cdots (M}_{n}\ N)\cdots).$$

つまり、$\mathsf{iter}\ M\ N\ \mathsf{n}$ は値 N に対して関数 M を n 回適用した結果を返す。

後続関数、足し算、掛け算、べき乗 チャーチ数項による自然数の表現は、自然数上の演算を定義するのに大変都合がよい。実際、チャーチ数項上の後続関数（$+1$)、足し算、掛け算、べき乗は次のように簡単に定義できる。

$$\begin{array}{rcllrcl} \mathsf{suc} & := & \lambda mf.f \circ (mf) & & \mathsf{add} & := & \lambda nmf.(nf) \circ (mf) \\ \mathsf{mult} & := & \lambda nm.n \circ m & & \mathsf{exp} & := & \lambda nm.mn. \end{array}$$

例えば足し算について見てみると

$$\mathsf{add}\ \mathsf{3}\ \mathsf{2} \longrightarrow_\beta^* \lambda f.(\mathsf{3}\ f) \circ (\mathsf{2}\ f) \longrightarrow_\beta^* \lambda f.(f \circ f \circ f) \circ (f \circ f) \longrightarrow_\beta^* \lambda f.f \circ f \circ f \circ f \circ f \equiv \mathsf{5}$$

となる。掛け算は単にチャーチ数項の合成にすぎないことに注意しよう。簡単な場合について計算してみると次のようになる。

$$\begin{array}{rcl} \mathsf{mult}\ \mathsf{3}\ \mathsf{2} & \longrightarrow_\beta^* & \mathsf{3} \circ \mathsf{2} \\ & \equiv & \lambda z.\mathsf{3}(\mathsf{2}z) \\ & \equiv & \lambda z.(\lambda f.f \circ f \circ f)((\lambda f.f \circ f)z) \\ & \longrightarrow_\beta & \lambda z.(\lambda f.f \circ f \circ f)(z \circ z) \\ & \longrightarrow_\beta & \lambda z.(z \circ z) \circ (z \circ z) \circ (z \circ z) \\ & \longrightarrow_\beta^* & \lambda z.z \circ z \circ z \circ z \circ z \circ z \\ & \equiv & \mathsf{6}. \end{array}$$

練習問題 4.4 $\mathsf{exp}\ \mathsf{2}\ \mathsf{3} \longrightarrow_\beta^* \mathsf{8}$ となることを確かめよ。

原始再帰法 本節の終わりに重要なラムダ項を定義する。

$$\mathsf{rec} := \lambda fgn.\mathsf{proj}_2(\mathsf{iter}\ \lambda\langle m,h\rangle.\langle \mathsf{suc}\ m, fmh\rangle\ \langle \mathsf{0}, g\rangle\ n)$$

ここで $\lambda\langle m,h\rangle.M(m,h)$ と書いたのは $\lambda x.M(\mathsf{proj}_1 x, \mathsf{proj}_2 x)$ の省略である。

命題 4.5 (原始再帰法) M, N をラムダ項とするとき、次のことが成り立つ。

$$\begin{aligned} \mathsf{rec}MN0 &=_\beta N, \\ \mathsf{rec}MN\mathsf{n}+1 &=_\beta M\mathsf{n}(\mathsf{rec}MN\mathsf{n}). \end{aligned}$$

証明は煩雑なのでここでは省略する。上の命題では $\longrightarrow_\beta^*$ ではなく $=_\beta$ が使われていることに注意してほしい。つまり β 簡約規則を一方向的に使うだけでは、$\mathsf{rec}MN\mathsf{n}+1$ から $M\mathsf{n}(\mathsf{rec}MN\mathsf{n})$ へは到達できない。しかしこのことはプログラムを実行する上で問題にはならない（後で述べるチャーチ・ロッサーの定理を参照)。

原始再帰法を使えば、さまざまな関数をプログラミングできる。例えば階乗関数を考える。

$$\begin{aligned} 0! &:= 1 \\ (n+1)! &:= (n+1)n! \end{aligned}$$

この関数をラムダ項で表すには、$N := \mathsf{1}$, $M := \lambda xy.\mathsf{mult}(\mathsf{suc}\ x)y$ としたうえで、$\mathsf{fact} := \mathsf{rec}MN$ とすればよい。そうすれば上の命題により

$$\begin{aligned} \mathsf{fact}\ 0 &=_\beta \mathsf{1} \\ \mathsf{fact}\ \mathsf{n}+1 &=_\beta M\mathsf{n}(\mathsf{fact}\ \mathsf{n}) =_\beta \mathsf{mult}(\mathsf{suc}\ \mathsf{n})(\mathsf{fact}\ \mathsf{n}) =_\beta \mathsf{mult}(\mathsf{n}+\mathsf{1})(\mathsf{fact}\ \mathsf{n}) \end{aligned}$$

となって期待通りの結果が得られる。このようにして n について帰納的に定義できる関数は、ほぼ全部原始再帰法を用いて定義できる。

最後に、次の節で用いる関数をいくつか定義しておく。

$$\begin{aligned} \mathsf{zero?} &:= \mathsf{rec}(\lambda xy.\mathsf{false})\mathsf{true} \\ \mathsf{one?} &:= \lambda x.\mathsf{conj}\ \mathsf{neg}(\mathsf{zero?}\ x)\ \mathsf{zero?}(\mathsf{pred}\ x) \\ \mathsf{even?} &:= \mathsf{iter}\ \mathsf{neg}\ \mathsf{true} \\ \mathsf{half} &:= \mathsf{proj}_1(\mathsf{iter}M\langle \mathsf{0}, \mathsf{false}\rangle) \end{aligned}$$

ただし $M := \lambda\langle y, b\rangle.b\langle \mathsf{suc}\ y,\ \mathsf{neg}\ b\rangle\langle y, \mathsf{neg}\ b\rangle$ とする。

練習問題 4.6 上で定義したラムダ項の働きを調べよ。

4.3 ラッセルのパラドックス再訪

本講義の最初に素朴集合論におけるラッセルのパラドックスについて言及したことを思い出してほしい。素朴集合論の基本原理は次の包括原理であった。

包括原理 “性質”$P(x)$ が与えられたとき、それを満たすもの全てからなる集合 $\{x|P(x)\}$ が存在する。つまり、

$$a \in \{x|P(x)\} \quad \Longleftrightarrow \quad P(a).$$

この原理を仮定した途端、ラッセルのパラドックスが生じてしまう。すなわち包括原理を $\neg(x \in x)$ に適用すると、集合 $R = \{x \mid \neg(x \in x)\}$ が得られる。しかし

$$R \in R \quad \Longleftrightarrow \quad \neg(R \in R)$$

となって矛盾が発生する。これがラッセルのパラドックスであった。

素朴集合論の言語は、次のようにしてラムダ計算の言語に翻訳できる。

$$\begin{array}{rcl} \{x|P(x)\} & \mapsto & \lambda x.P[x] \\ a \in X & \mapsto & X\ a \\ \neg P & \mapsto & \mathsf{neg}\ P. \end{array}$$

すると包括原理とは単に β 簡約のことに他ならないことがわかる。

$$(\lambda x.P[x])a \quad \longrightarrow_\beta \quad P[a].$$

このようにして、純粋な素朴集合論はラムダ計算に埋め込んでしまえるのである。(実際、チャーチがラムダ計算を考案した当初の意図はそのような普遍的な論理体系を構築することにあった。)

それでは、ラッセルのパラドックスはラムダ計算の文脈ではどのような現象として現れるのだろうか？まず、ラッセル集合 R は $\lambda x.\mathsf{neg}(xx)$ と表すことができ、言明 $R \in R$ は RR と表すことができる。そして RR を β 簡約すると次のようになる。

$$RR \longrightarrow_\beta \mathsf{neg}(RR) \longrightarrow_\beta \mathsf{neg\ neg}(RR) \longrightarrow_\beta \mathsf{neg\ neg\ neg}(RR) \longrightarrow_\beta \cdots$$

このようにして、パラドクシカルなプログラム RR を実行すると、計算は止まらない。このことから得られる教訓は、非常に大雑把に言えば、論理と計算の間には次のような対応関係があるということである。

論理の矛盾　≈　計算の非停止.

現段階では単なるアナロジーにすぎないが、この対応関係は本格的な検討に値する。実際、ヒルベルトのプログラム、そしてゲンツェンの証明論のエッセンスはまさにこの対応関係にあるといっても過言ではないのである。

このあたりの事情を追及するのは次章以降にまわすことにして、本節では別の方向に話を進める。パラドックスは、論理な観点からは絶対悪であり避けるべきものであるが、計算の観点からは必ずしもそうとは言えず有用性すら兼ね備えている。

このことを見るために、まずはラッセルのパラドックスを一般化しよう。M をラムダ項とするとき、

$$\mathsf{fix}M \quad := \quad (\lambda x.M(xx))(\lambda x.M(xx))$$

と定義する。すると $\mathsf{fix}M \longrightarrow_\beta M(\mathsf{fix}M)$ が成り立つ。すなわち $\mathsf{fix}M$ は方程式 $x =_\beta Mx$ の解を与える。このような解を一般に関数 M の**不動点** (fixed point) という。

さて、関数の不動点はプログラミングにおいて重要な役割を果たす。たとえばコラッツ関数を考える。

$$\begin{array}{rcll} col(n) & = & 1 & (n = 1) \\ & = & col(n/2) & (n \text{ は偶数}) \\ & = & col(3n+1) & (n \text{ は } 1 \text{ 以外の奇数}) \end{array}$$

この定義の難点は $col(n)$ を定義するのに、n よりも大きい数 $3n+1$ について col を用いていることである。このような場合には原始再帰法は使えない。ではどのようにすればラムダ項を使ってこの関数を定義することができるだろうか？

まず、

$$M \quad := \quad \lambda Fx.\mathsf{if\ one?}(x)\ \mathsf{then}\ x\ \mathsf{else}\ (\mathsf{if\ even?}(x)\ \mathsf{then}\ F(x/2)\ \mathsf{else}\ F(3x+1))$$

とする。ただし $x/2, 3x+1$ はそれぞれ half x と add(mult 3 x)1 の略である。その上で、M の不動点をとって col $:=$ fix$M$ とする。

このラムダ項が確かにコラッツ関数を表すことをみるために、n を 1 以外の奇数としよう。すると、

$$\begin{aligned}
\mathsf{col\ n} &\longrightarrow_\beta M\ \mathsf{col\ n} \\
&\longrightarrow_\beta^* \mathsf{if\ one?(n)\ then\ n\ else\ (if\ even?(n)\ then\ col}(\mathsf{n}/2)\ \mathsf{else\ col}(3\mathsf{n}+1)) \\
&\longrightarrow_\beta^* \mathsf{if\ even?(n)\ then\ col}(\mathsf{n}/2)\ \mathsf{else\ col}(3\mathsf{n}+1) \\
&\longrightarrow_\beta^* \mathsf{col}(3\mathsf{n}+1)
\end{aligned}$$

となって、意図したとおりに働くことがわかる。このように不動点はプログラミングにおける強力な武器であり、それは特に停止するかどうかわからないようなプログラムを作成するときに効力を発揮するのである。ラッセルのパラドックスは、歴史上のある一点において否定的な役割をになったものの、計算機科学の台等とともに一気に名誉挽回して肯定的な役割を果たすことになったのである。そういうことが歴史上ままある。

注意 4.7 col 0 を β 簡約しても計算は停止しない。しかし 0 についても計算が停止するように col の定義を修正するのは簡単である。

一方、任意の自然数 $n \geq 1$ について col n の計算が停止するかどうかは未解決の問題である（コラッツの予想）。*Wikipedia* によれば、少なくとも $3 \cdot 2^{53}$ までの数については col n $\longrightarrow_\beta^*$ 1 となることが確かめられているそうである。

4.4 ラムダ計算の性質

ラムダ計算の表現能力　最初に述べたとおり、ラムダ計算とは関数を作ったり使ったりする操作を抽象化したものである。それゆえ数や四則演算はラムダ項の定義には含まれない。それにもかかわらず、数はチャーチ数項として、四則演算は適当なラムダ項として表現できる。では、ラムダ計算の枠組みで一体どこまで複雑な関数が表現可能なのだろうか？

この問いに正確に答えるために、まずは限定論理式をラムダ項で表す方法について考えよう。ラムダ項を用いれば、算術の言語の $0, S, +, \cdot$ がうまく表現できることはすでに見た通りである。等号 $=$ を表すには次の一連のラムダ項を考える。

$$\begin{aligned} \mathsf{pred} &:= \mathsf{rec}(\lambda xy.x)\mathsf{0} \\ \mathsf{sub} &:= \lambda nm.\mathsf{iter\ pred}\ n\ m \\ \mathsf{eq} &:= \lambda nm.\mathsf{conj\ zero?(sub}\ n\ m)\ \mathsf{zero?(sub}\ m\ n) \end{aligned}$$

pred は与えられた数から 1 を引く関数を表す。実際、

$$\mathsf{pred\ n} + 1 =_\beta \mathsf{n}, \qquad \mathsf{pred\ 0} =_\beta \mathsf{0}$$

が成り立つ。これを必要回繰り返すことにより引き算を表すラムダ項 sub が得られる。ただし $n < m$ のとき $\mathsf{sub\ n\ m} =_\beta \mathsf{0}$ である。これとゼロテスト zero? を組み合わせることにより等号を表すラムダ項 eq が得られる。実際、eq は

$$\begin{aligned} \mathsf{eq\ n\ m} &=_\beta \mathsf{true} \quad (n = m \text{ のとき}) \\ &=_\beta \mathsf{false} \quad (n \neq m \text{ のとき}) \end{aligned}$$

を満たす。

否定 $\neg$ と連言 $\wedge$ はラムダ項 neg と conj を使って表すことができる。

次に限定量化 $\forall z < n.\varphi(z)$ について考える（ただし話を簡単にするために $\forall z \leq n$ ではなく $\forall z < n$ を考える）。いま、ラムダ項 M_φ が一変数論理式 $\varphi(z)$ を表すとする。すなわち

$$\begin{aligned} M_\varphi\ \mathsf{n} &=_\beta \mathsf{true} \quad (\mathbf{N} \models \varphi(n) \text{ のとき}) \\ &=_\beta \mathsf{false} \quad (\text{それ以外のとき}) \end{aligned}$$

このとき $\forall x < n.\varphi(x)$ を表すラムダ項は次のように定義できる。

$$N := \mathsf{rec}(\lambda xy.\mathsf{conj}(M_\varphi\ x)y)\mathsf{true}.$$

するとたとえば $n = 3$ のとき

$$\begin{aligned} N\ 3 &=_\beta \mathsf{conj}(M_\varphi\ 2)(N\ 2) \\ &=_\beta \mathsf{conj}(M_\varphi\ 2)(\mathsf{conj}(M_\varphi\ 1)(N\ 1)) \\ &=_\beta \mathsf{conj}(M_\varphi\ 2)(\mathsf{conj}(M_\varphi\ 1)(\mathsf{conj}(M_\varphi\ 0)(N\ 0))) \\ &=_\beta \mathsf{conj}(M_\varphi\ 2)(\mathsf{conj}(M_\varphi\ 1)(\mathsf{conj}(M_\varphi\ 0)(\mathsf{true}))) \end{aligned}$$

となって得られる結果は $\varphi(2) \wedge \varphi(1) \wedge \varphi(0) \wedge \mathsf{T}$ の真理値と一致することがわかる。これは $\forall x < 3.\varphi(x)$ の真理値に他ならない。

最後に $\exists x.\varphi(x)$ について考える。この形の文を表すには

$$N\ \mathsf{n} =_\beta \mathsf{if}\ (M_\varphi\ \mathsf{n})\ \mathsf{then\ true\ else}\ N(\mathsf{suc\ n})$$

を満たすラムダ項 N をひとつ求めて $N\ \mathsf{0}$ とすればよい。実際、

$$\mathbf{N} \models \exists x.\varphi(x) \quad \Longleftrightarrow \quad N\ \mathsf{0} =_\beta \mathsf{true}$$

が成り立つ。そのような N は不動点を用いれば簡単に定義できる。

$$N\ :=\ \mathsf{fix}(\lambda F x.\mathsf{if}\ (M_\varphi\ x)\ \mathsf{then\ true\ else}\ F(\mathsf{suc}\ x)).$$

以上のことから次の定理が成り立つ。

定理 4.8 任意の Σ_1 集合 $X \subseteq \mathbb{N}$ に対してラムダ項 M_X が存在し、全ての $n \in \mathbb{N}$ について次のことが成り立つ：

$$n \in X \quad \Longleftrightarrow \quad M_X\ \mathsf{n} =_\beta \mathsf{true}.$$

ここで $=_\beta$ は $\longrightarrow_\beta^*$ で置き換えてもよい。そのようにしてもよいことは後で系 4.13 で示す。

上の同値性が成り立つとき、M_X は X を**弱ラムダ定義する** (weakly lambda define) という。多少の工夫をすれば、次の定理も証明できる。

定理 4.9 任意の Δ_1 集合 $X \subseteq \mathbb{N}$ に対してラムダ項 N_X が存在し、全ての $n \in \mathbb{N}$ について次のことが成り立つ：

$$\begin{aligned} n \in X \quad &\Longrightarrow \quad N_X\ \mathsf{n} =_\beta \mathsf{true} \\ n \notin X \quad &\Longrightarrow \quad N_X\ \mathsf{n} =_\beta \mathsf{false} \end{aligned}$$

上のことが成り立つとき、N_X は X を**強ラムダ定義する** (strongly lambda define) という。

次にテューリング機械を用いてラムダ計算をシミュレートすることについて考える。ラムダ項は文字列であるから、適当なアルファベット Σ を選べば、ラムダ項はテューリング機械のテープ上に書き下すことができる。そして β 簡約をシミュレートするテューリング機械は簡単に構成することができる。それゆえ、結局のところ次の同値性が成り立つことになる。

系 4.10 任意の $X \subseteq \mathbb{N}$ について次の各同値性が成り立つ。

$$\begin{array}{lllll} X \in \Sigma_1 & \Longleftrightarrow & X \text{ は再帰的枚挙可能} & \Longleftrightarrow & X \text{ は弱ラムダ定義可能} \\ X \in \Delta_1 & \Longleftrightarrow & X \text{ は再帰的} & \Longleftrightarrow & X \text{ は強ラムダ定義可能} \end{array}$$

M をラムダ項とする。もしもある正規形 N が存在し $M \longrightarrow_\beta^* N$ となるとき、M は**正規化可能** (normalizable) であるという（正確には、後に導入する強正規化可能性と区別して**弱正規化可能** (weakly normalizable) という）。例えば、$\mathsf{mult\ 3\ 5} \longrightarrow_\beta^* \mathsf{15}$ でありチャーチ数はすべて正規形だから、$\mathsf{mult\ 3\ 5}$ は正規化可能である。他方でラムダ項 $\omega := (\lambda x.xx)(\lambda x.xx)$ を考えると、$\omega \longrightarrow_\beta \omega \longrightarrow_\beta \omega \cdots$ となって正規形には絶対に到達できないから、ω は正規化可能ではない。

テューリング機械の停止問題の非決定性と同様に、与えられたラムダ項が正規化可能かどうかも決定不能である。すなわち、各ラムダ項 M を適切なゲーデル数 $\lceil M \rceil$ によりコード化すれば次が成り立つ。

系 4.11 集合 $\mathsf{WN} := \{\lceil M \rceil \mid M \text{ は正規化可能}\}$ は再帰的ではない。

計算の順序　一つのラムダ項は複数の β 基を含みうる。それゆえラムダ項が与えられたとき、β 簡約を適用する順序は一通りには定まらない。しかし実際上問題が起こらないことは、次の定理が保証してくれる。

定理 4.12 (チャーチ・ロッサー) ラムダ項 M_0, M_1, M_2 について

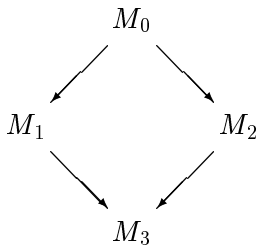
$$M_0 \longrightarrow_\beta^* M_1, \quad M_0 \longrightarrow_\beta^* M_2$$

が成り立つならば、あるラムダ項 M_3 が存在して

$$M_1 \longrightarrow_\beta^* M_3, \quad M_2 \longrightarrow_\beta^* M_3$$

が成り立つ。

証明は煩雑なので省略する。チャーチ・ロッサーの定理の意味は次のような図を考えるとわかりやすい。ただし $\longrightarrow$ は $\longrightarrow_\beta^*$ を表すものとする。



チャーチ・ロッサーの定理からは、解の一意性をはじめとする重要な性質が帰結する。

系 4.13

1. 各ラムダ項 M について、$M \longrightarrow_\beta^* N$ となるような正規形 N は高々一つしか存在しない。

2. $M_1 =_\beta M_2$ ならば、あるラムダ項 M_3 が存在して

   $$M_1 \longrightarrow_\beta^* M_3, \quad M_2 \longrightarrow_\beta^* M_3$$

   が成り立つ。

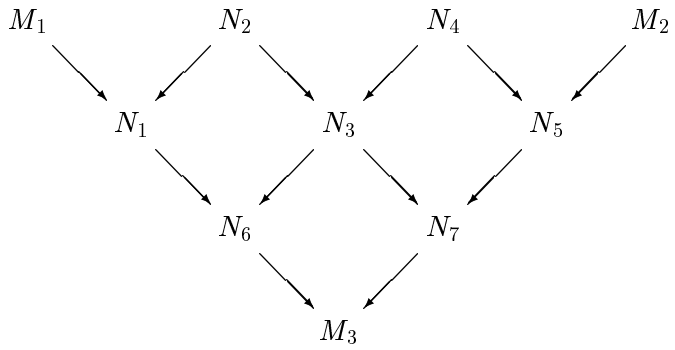
3. $M_1 =_\beta M_2$ かつ M_2 が正規形ならば、$M_1 \longrightarrow_\beta^* M_2$ が成り立つ。

証明　1. 二つの正規形 N_1, N_2 について $M \longrightarrow_\beta^* N_1$ と $M \longrightarrow_\beta^* N_2$ が成り立つとすると、チャーチ・ロッサーの定理によりあるラムダ項 N_3 が存在し、$N_1 \longrightarrow_\beta^* N_3$ と $N_2 \longrightarrow_\beta^* N_3$ が成り立つはずである。しかし正規形のラムダ項はそれ以上 β 簡約することはできないから、実際には $N_1 \equiv N_3 \equiv N_2$ が成り立つ。すなわち正規形は高々一つしか存在しない。

2. $M_1 =_\beta M_2$ となるのは、β 簡約とその逆を用いて M_1 から M_2 に到達できるときである。たとえば

$$M_1 \longrightarrow N_1 \longleftarrow N_2 \longrightarrow N_3 \longleftarrow N_4 \longrightarrow N_5 \longleftarrow M_2$$

とする。ただし $\longrightarrow$ はすべて $\longrightarrow_\beta^*$ を表すものとする。このときチャーチ・ロッサーの定理を用いて次のような図を作れば、適切な M_3 を求めることができる。



3. 上の 2 により、あるラムダ項 M_3 が存在して $M_1 \longrightarrow_\beta^* M_3$ と $M_2 \longrightarrow_\beta^* M_3$ が成り立つはずである。しかし M_2 は正規形であるから、実際には $M_2 \equiv M_3$ である。ゆえに $M_1 \longrightarrow_\beta^* M_2$ が成り立つ。 ■

1 により、計算の順序（すなわち β 簡約を用いる順序）によって出力が変わることはない。また 3 により、β 簡約とその逆を用いて正規形に到達できるときには、β 簡約のみを用いて正規形に到達できる。定理 4.8 の後で $=_\beta$ は $\longrightarrow_\beta^*$ で置き換えてもよいといったのは、この性質を念頭においてのことである。

ただし注意しなければならないのは、あるラムダ項から出発して正規形に到達できるかどうかは、β 基の選び方に大いに依存するという点である。たとえて言えば、仮にある迷路があって、うまく道順を選べば入口から出口に到達できるとわかっていたとしても、分かれ道で誤った選択をつづければ永遠に出られない可能性があるというのと同様である。

簡単な例として、$\omega := (\lambda x.xx)(\lambda x.xx)$ とし、$M := (\lambda y.z)\omega$ とする。M は二つの β 基を含む。すなわち M 全体と ω である。M を選んで β 簡約を実行すれば $M \longrightarrow_\beta z$ となり正規形に到達する。一方、ω を選んで β 簡約を続けると、

$$M \equiv (\lambda y.z)\omega \longrightarrow_\beta (\lambda y.z)\omega \longrightarrow_\beta (\lambda y.z)\omega \longrightarrow_\beta \cdots$$

となり計算は永遠に止まらない。このようにラムダ項を正規化できるかどうかは、計算の順序に依存する。しかし幸いなことに次の定理がある。

定理 4.14 正規化可能な M が与えられたとする。このとき、常にもっとも左側にある β 基を選んで β 簡約を続ければ、必ず正規形に到達することができる。

先ほどのたとえで言えば、迷路の入り口に立ったときには、左右どちらか一方の壁に沿って進めば必ず出口に到達できるというのと同様である。（もちろん入り口から出口への経路が存在するということが絶対条件である。）

5　カリー・ハワード対応 — 証明とはプログラムである

論理と計算の関係は多岐にわたる。たとえば 3 章では、計算論的性質である再帰性や再帰的枚挙可能性が論理的性質である Δ_1 や Σ_1 と対応することを説明した。後者は論理的であるが、もっと厳密にいえば算術の論理式の記述能力に関する性質である。一方本章では、論理的性質の中でも「証明」に関するものに注目する。「証明」に計算の側で対応するのは「プログラム」である。ゆえに本章の主題は「証明」と「プログラム」の対応関係である。

数学者は証明を書くことによって定理を正当化する。プログラマはプログラムを書くによって課題を遂行する。両者は一見すると全くの別物に見えるかもしれないが、よく考えてみれば密接な対応関係のあることがわかる。

たとえば具体例として「素数は無限に多く存在する」というユークリッドの定理を考えてみよう。これは「どんな自然数 n が与えられたとしても、それより大きな素数 p が存在する」と言いかえることができる。この定理を証明するにはどうしたらよいだろうか？もっとも自然な方法は、「入力として自然数 n が与えられたとき、それよりも大きな素数 p を出力する」プログラム（より正確にはその背後にある計算手順、すなわち**アルゴリズム** (algorithm)）を記述し、その正しさを検証することである。たとえば、次のような簡単なプログラムを考える。

$$p := n! + 1 \text{ の最小の非自明な約数}$$

すると、p が n より大きい素数であることは容易に検証できる。このプログラム（および正しさの検証）は確かにユークリッドの定理に対する証明となっている。

一般に、「$\cdots$ を満たす対象が存在する」ということを示すのに、そのような対象の構成方法を具体的に記述するような証明を**構成的証明** (constructive proof) という。プログラムは、構成的証明の最たるものである。一方、背理法を不用意に用いると、証明は構成的でなくなることがあるので注意が必要である。実際、背理法を用いると議論は次のように進む。

> 仮に $\cdots$ を満たす対象が存在しないとしよう。すると $\times\times\times$ となり矛盾する。
> ゆえに $\cdots$ を満たす対象が存在するはずである。

これは正しい論法であるが、このようにして論証を進めると「$\cdots$ を満たす対象」を具体的に構成せずとも証明が完遂されてしまうのである。

背理法と並んで危険なのは、**排中律** (excluded middle) の使用である。次の例を思い出そう。

命題 5.1　a^b が有理数となるような二つの無理数 a, b が存在する。

証明　排中律により、$\sqrt{2}^{\sqrt{2}}$ は有理数であるか無理数であるかのどちらかである。もしも有理数ならば、$a = b = \sqrt{2}$ とすれば命題が成り立つ。一方無理数の場合も、$a = \sqrt{2}^{\sqrt{2}}$, $b = \sqrt{2}$ とすれば $a^b = \sqrt{2}^2 = 2$ となって命題が成り立つ。 ■

これは非構成的証明の典型例である。この証明によれば「$\cdots$ を満たす対象が存在する」ことはわかるものの、具体的にどんな対象が $\cdots$ を満たすのかはわからずじまいである。このような場合には、「証明」と「プログラム」の間の関係は必ずしも明確ではない。

さて、背理法や排中律などの非構成的論法を認めないというのはまさに直観主義の根本原理に他ならない。実際、直観主義論理で妥当な証明はすべて構成的である。ゆえに「証明」と「プログラム」の対応関係を調べるにあたっては、まずその関係性が明確な直観主義論理にのっとって話を進めるのが理にかなっている。その際、「証明」に対応する「プログラム」はラムダ計算におけるプログラム、すなわちラムダ項により与えられる。両者の対応関係は**カリー・ハワード対応** (Curry-Howard correspondence) として知られている。これは 1920 年代後半のカリーの論文にその萌芽がみられるものの、明確に意識されたのは 1960 年代のハワードの論文がきっかけである。

5.1 二階直観主義命題論理

カリー・ハワード同型対応をもっとも印象的な形で述べるため、本節では直観主義論理の一形態である**二階直観主義命題論理** (second order intuitionistic propositional logic) **IL2** を導入する。

定義 5.2 (IL2 の論理式) 命題変数 *(propositional variable)* $\alpha, \beta, \cdots$ が与えられているとする。このとき **IL2** の論理式 A, B は次の *BNF* 記法により定義される。

$$A, B \ ::= \ \alpha \mid (A \Rightarrow B) \mid \forall \alpha.A$$

論理式 $A \Rightarrow B$ は「A ならば B」を表す。以前とは異なる記号を用いているが、とくに意味はない。一方、$\forall \alpha.A(\alpha)$ は「すべての α について $A(\alpha)$」を表す。ただし α に代入されるのは A, B などの論理式である。ここが **IL** と本質的に異なる点である。(**IL** における $\forall x.\varphi(x)$ の場合は、x に代入されるのは t, u などの項であった。)

前と同じく論理式の有限集合を Γ, Δ 等により表す。**IL2** の推論規則は次の通りである。

$$\frac{}{\Gamma \vdash A} \ (\mathit{init}) \quad \text{ただし } A \in \Gamma$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \Rightarrow B} \ (\Rightarrow I) \qquad \frac{\Gamma \vdash A \Rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B} \ (\Rightarrow E)$$

$$\frac{\Gamma \vdash A}{\Gamma \vdash \forall \alpha.A} \ (\forall I) \qquad \frac{\Gamma \vdash \forall \alpha.A(\alpha)}{\Gamma \vdash A(B)} \ (\forall E)$$

ただし $(\forall I)$ 規則を使う際には、Γ の論理式の中で α が自由変数として用いられていてはならない。上の規則を使ってシークエント $\Gamma \vdash A$ が導出できるときには、$\Gamma \vdash_{\mathbf{IL2}} A$ と書く。

その他の論理結合子の定義 **IL2** には「ならば」と「すべて」しか論理的な言葉が含まれていない。しかし「かつ」や「または」などその他の言葉は全部「ならば」と「すべて」を使って定義することができる。これが二階論理の威力である。

$$
\begin{aligned}
\top &:= \forall\alpha.\alpha \Rightarrow \alpha \\
\bot &:= \forall\alpha.\alpha \\
\neg A &:= A \Rightarrow \bot \\
A \wedge B &:= \forall\alpha.(A \Rightarrow B \Rightarrow \alpha) \Rightarrow \alpha \\
A \vee B &:= \forall\alpha.(A \Rightarrow \alpha) \Rightarrow (B \Rightarrow \alpha) \Rightarrow \alpha \\
\exists\beta.A &:= \forall\alpha.(\forall\beta.(A \Rightarrow \alpha)) \Rightarrow \alpha
\end{aligned}
$$

ただし α は A や B の中では自由変数として用いられていない新しい変数である。

このように定義すれば、第 1 章で挙げた直観主義論理の推論規則は全て **IL2** の中でシミュレートすることができる。たとえば

$$\frac{\Gamma \vdash \bot}{\Gamma \vdash A}\ (\bot E) \qquad \text{は} \qquad \frac{\Gamma \vdash \forall\alpha.\alpha}{\Gamma \vdash A}\ (\forall E)$$

に他ならない。また、$\wedge$ についての推論規則

$$\frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B}\ (\wedge I) \qquad \text{は} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma, A \Rightarrow B \Rightarrow \alpha \vdash A \Rightarrow B \Rightarrow \alpha}\ (init) \quad \Gamma \vdash A}{\Gamma, A \Rightarrow B \Rightarrow \alpha \vdash B \Rightarrow \alpha}\ (\Rightarrow E) \quad \Gamma \vdash B}{\Gamma, A \Rightarrow B \Rightarrow \alpha \vdash \alpha}\ (\Rightarrow E)}{\Gamma \vdash (A \Rightarrow B \Rightarrow \alpha) \Rightarrow \alpha}\ (\Rightarrow I)}{\Gamma \vdash \forall\alpha.(A \Rightarrow B \Rightarrow \alpha) \Rightarrow \alpha}\ (\forall I)$$

とシミュレートできる。(ただし $\Gamma \vdash A$ が成り立つとき $\Gamma, A \Rightarrow B \Rightarrow \alpha \vdash A$ も成り立つので、正確には後者を書くべきところを単に $\Gamma \vdash A$ と書いている。$\Gamma \vdash B$ についても同様。)
また、

$$\frac{\Gamma \vdash A \wedge B}{\Gamma \vdash A}\ (\wedge E) \qquad \text{も} \qquad \dfrac{\dfrac{\Gamma \vdash \forall\alpha.(A \Rightarrow B \Rightarrow \alpha) \Rightarrow \alpha}{\Gamma \vdash (A \Rightarrow B \Rightarrow A) \Rightarrow A}\ (\forall E) \quad \dfrac{\dfrac{\dfrac{}{\Gamma, A, B \vdash A}\ (init)}{\Gamma, A \vdash B \Rightarrow A}\ (\Rightarrow I)}{\Gamma \vdash A \Rightarrow B \Rightarrow A}\ (\Rightarrow I)}{\Gamma \vdash A}\ (\Rightarrow E)$$

というふうにシミュレートできる。

練習問題 5.3 同様にして $(\vee I), (\vee E), (\exists I), (\exists E)$ 規則がシミュレートできることを示せ。

このように **IL2** は見かけによらず大きな表現力を持っている。実際、(Urzyczyn 1997) によれば次の定理が成り立つ。

定理 5.4 (IL2 の決定不能性)

1. **IL** の論理式 A を **IL2** の論理式 A^* に移す再帰的関数が存在し、

$$\vdash_{\mathbf{IL}} A \iff \vdash_{\mathbf{IL2}} A^*$$

が成り立つ。

2. 論理式 A が与えられたとき、$\vdash_{\mathbf{IL2}} A$ が成り立つかどうかは決定不能である。

しかし二階命題論理が表現力を持つのは、直観主義の場合に限られる。古典論理に場合には、表現力は非常に弱い。いま、**IL2** に背理法の推論規則

$$\frac{\Gamma, \neg A \vdash \bot}{\Gamma \vdash A}\ (abs)$$

を加えて得られる論理体系を **CL2** と書くことにする。すると次のことが成り立ってしまう。

$$\vdash_{\mathbf{CL2}} \forall\alpha.A(\alpha) \ \Leftrightarrow\ A(\top) \wedge A(\bot).$$

ゆえに **CL2** における $\forall\alpha$ は簡単に除去できてしまい、**CL2** の論理式は古典命題論理の論理式に翻訳できてしまう。古典命題論理の論理式が証明可能かどうかを調べるには、真理値表を書いてトートロジーかどうかを調べればよい（完全性定理）から、結局次のことが成り立つ。

定理 5.5 (CL2 の決定可能性) 論理式 A が与えられたとき、$\vdash_{\mathbf{CL2}} A$ が成り立つかどうかは決定可能である。

計算量理論の用語を用いれば、上の問題は決定可能であるどころか PSPACE 完全である。(PSPACE 完全というのは、大雑把にいって、囲碁の与えられた局面で必勝手をみつけるのと同程度の計算量である。)

5.2 証明をラムダ項で表す

本章のはじめで「証明」と「プログラム」の間には密接な関係があると述べたが、その際に与えたのは、証明やプログラム全体を考察の対象とするマクロな説明であった。ここではもう少しミクロな説明を与えてみよう。

関数型プログラミングの発想によれば、プログラムの作成は関数を作ったり使ったりすることを基本とする。ここでは使う方について考える。関数 $f : X \longrightarrow Y$ を値 $a \in X$ に対して使う操作は

$$\frac{f : X \longrightarrow Y \quad a \in X}{f(a) \in Y}$$

と書ける。ラムダ項を使って書き方を少しアレンジすれば、

$$\frac{M : X \Rightarrow Y \quad N : X}{MN : Y}\ (*)$$

となる。この図が表しているのは、「X から Y への関数 M を対象 $N \in X$ に対して使うと、対象 $MN \in Y$ が得られる」ということである。しかし図のように書いてみると、関数を使うことは modus ponens の推論（つまり $(\Rightarrow E)$ 規則）に似ていることに気がつく。そこで論理的観点にたって、$M : A$ と書いたら「M は A の証明である」と読むことにしてみよう。すると上の図は次のように解釈することができる。すなわち、「M が $X \Rightarrow Y$

の証明で、N が X の証明ならば、MN は Y の証明である。」これは modus ponens を使った証明の構成法に他ならない。このアイデアを展開すると

ラムダ項　　≈　　IL2 の証明

という対応が得られる。

より詳しく説明するために、$\Gamma \vdash B$ を **IL2** で証明可能なシークエントとする。まず、左側に現れる各論理式 $A \in \Gamma$ に適当な変数 x を割り当てて $x : A$ とし、シークエントの右側に現れる論理式 B には適当なラムダ項 M を割り当てて $M : B$ とする。たとえば

$$A_1, \ldots, A_n \vdash B \qquad \mapsto \qquad x_1 : A_1, \ldots, x_n : A_n \vdash M : B$$

となる。ここで変数 $x_1, \ldots, x_n$ の選び方は完全に任意である。ただしこれらの変数は互いに異なっていなければならない。一方で、M がどんなラムダ項であるかは、$\Gamma \vdash B$ の証明図がどのようであるかに依存する。具体的には、**IL2** の各推論規則に対して、次のようなラムダ項の割り当て方を定める。

$$\frac{}{\Gamma \vdash x : A}\ (init) \quad \text{ただし } x : A \in \Gamma$$

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : A \Rightarrow B}\ (\Rightarrow I) \qquad \frac{\Gamma \vdash M : A \Rightarrow B \quad \Gamma \vdash N : A}{\Gamma \vdash MN : B}\ (\Rightarrow E)$$

$$\frac{\Gamma \vdash M : A}{\Gamma \vdash M : \forall\alpha.A}\ (\forall I) \qquad \frac{\Gamma \vdash M : \forall\alpha.A(\alpha)}{\Gamma \vdash M : A(B)}\ (\forall E)$$

つまり $(init)$ 規則を使うときには仮定 A に割り当てられた変数を用い、$(\Rightarrow I)$ 規則を使うときにはラムダ抽象を行い、$(\Rightarrow E)$ 規則を使うときには適用を行う。$(\forall I)$ 規則と $(\forall E)$ 規則を使うときにはラムダ項を変えない。

たとえば $\forall\alpha.\forall\beta.\forall\gamma.(\alpha \Rightarrow \beta) \Rightarrow (\beta \Rightarrow \gamma) \Rightarrow (\alpha \Rightarrow \gamma)$ の証明

$$\left.\begin{array}{l}
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash \beta \Rightarrow \gamma}\ (init) \quad \dfrac{\dfrac{}{\Gamma \vdash \alpha \Rightarrow \beta}\ (init) \quad \dfrac{}{\Gamma \vdash \alpha}\ (init)}{\Gamma \vdash \beta}\ (\Rightarrow E)}{\Gamma \vdash \gamma}\ (\Rightarrow E)}{\alpha \Rightarrow \beta, \beta \Rightarrow \gamma \vdash \alpha \Rightarrow \gamma}\ (\Rightarrow I)}{\alpha \Rightarrow \beta \vdash (\beta \Rightarrow \gamma) \Rightarrow (\alpha \Rightarrow \gamma)}\ (\Rightarrow I)}{\vdash (\alpha \Rightarrow \beta) \Rightarrow (\beta \Rightarrow \gamma) \Rightarrow (\alpha \Rightarrow \gamma)}\ (\Rightarrow I)}{\vdash \forall\gamma.(\alpha \Rightarrow \beta) \Rightarrow (\beta \Rightarrow \gamma) \Rightarrow (\alpha \Rightarrow \gamma)}\ (\forall I)}{\vdash \forall\beta.\forall\gamma.(\alpha \Rightarrow \beta) \Rightarrow (\beta \Rightarrow \gamma) \Rightarrow (\alpha \Rightarrow \gamma)}\ (\forall I)}{\vdash \forall\alpha.\forall\beta.\forall\gamma.(\alpha \Rightarrow \beta) \Rightarrow (\beta \Rightarrow \gamma) \Rightarrow (\alpha \Rightarrow \gamma)}\ (\forall I)
\end{array}\right\}\ (*)$$

を考えよう。（ただし $\Gamma = \{\alpha \Rightarrow \beta, \beta \Rightarrow \gamma, \alpha\}$。）この証明 (*) に現れる各シークエントに対して上から順にラムダ項を割り当てていくと次のようになる。

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash g : \beta \Rightarrow \gamma}\ (init) \quad \dfrac{\dfrac{}{\Gamma \vdash f : \alpha \Rightarrow \beta}\ (init) \quad \dfrac{}{\Gamma \vdash x : \alpha}\ (init)}{\Gamma \vdash fx : \beta}\ (\Rightarrow E)}{\Gamma \vdash g(fx) : \gamma}\ (\Rightarrow E)}{f : \alpha \Rightarrow \beta, g : \beta \Rightarrow \gamma \vdash \lambda x.g(fx) : \alpha \Rightarrow \gamma}\ (\Rightarrow I)}{f : \alpha \Rightarrow \beta \vdash \lambda gx.g(fx) : (\beta \Rightarrow \gamma) \Rightarrow (\alpha \Rightarrow \gamma)}\ (\Rightarrow I)}{\vdash \lambda fgx.g(fx) : (\alpha \Rightarrow \beta) \Rightarrow (\beta \Rightarrow \gamma) \Rightarrow (\alpha \Rightarrow \gamma)}\ (\Rightarrow I)}{\vdash \lambda fgx.g(fx) : \forall\gamma.(\alpha \Rightarrow \beta) \Rightarrow (\beta \Rightarrow \gamma) \Rightarrow (\alpha \Rightarrow \gamma)}\ (\forall I)}{\vdash \lambda fgx.g(fx) : \forall\beta.\forall\gamma.(\alpha \Rightarrow \beta) \Rightarrow (\beta \Rightarrow \gamma) \Rightarrow (\alpha \Rightarrow \gamma)}\ (\forall I)}{\vdash \lambda fgx.g(fx) : \forall\alpha.\forall\beta.\forall\gamma.(\alpha \Rightarrow \beta) \Rightarrow (\beta \Rightarrow \gamma) \Rightarrow (\alpha \Rightarrow \gamma)}\ (\forall I)$$

(ただし $\Gamma = \{f : \alpha \Rightarrow \beta,\ g : \beta \Rightarrow \gamma,\ x : \alpha\}$。) このようにして最終的には $\lambda fgx.g(fx)$ が得られる。これが証明 $(*)$ に対応するラムダ項である。

$\lambda fgx.g(fx)$ は証明 (*) の構造を表していると言ってよい。実際、変数 f, g, x はそれぞれ途中の仮定 $\alpha \Rightarrow \beta$, $\beta \Rightarrow \gamma$, α を表している。また $(\Rightarrow I)$ 規則が使われるたびにラムダ抽象が行われ、$(\Rightarrow E)$ 規則が使われるたびに適用が行われることは既に述べた通りである。ただしラムダ項は $(\forall I)$ 規則、$(\forall E)$ 規則の使われ方は反映しないので、より正確には「ラムダ項は証明のスケルトンを表す」というべきだろう。細かい点は後回しにして、ここでは

$$\begin{array}{rcl} (init) \text{ 規則} & \approx & \text{変数} \\ (\Rightarrow I) \text{ 規則} & \approx & \text{ラムダ抽象} \\ (\Rightarrow E) \text{ 規則} & \approx & \text{適用} \end{array}$$

という対応関係にだけ注目してほしい。

注意 5.6 $(\forall I)$ 規則、$(\forall E)$ 規則がラムダ項に反映されないのは、本講義ではラムダ計算と論理の対応関係をカリーの流儀で説明しているためである。実をいうともう一つチャーチの流儀というのもあり、それに従えば $(\forall I)$ 規則、$(\forall E)$ 規則はラムダ項にきちんと反映される。カリー・ハワード同型対応について語るときには後者のほうが標準的であるが、本講義では説明のしやすさを考慮にいれて前者の流儀を採用することにした。

さて、それではその他の規則、たとえば $(\wedge I)$ 規則や $(\wedge E)$ 規則はどうなるのだろうか？そこで先ほどのシミュレーションをもう一度見直して、ラムダ項を割り当ててみよう。まず $(\wedge I)$ 規則についてであるが、$\Gamma \vdash M : A$ および $\Gamma \vdash N : B$ から出発するとラムダ項の割り当ては次のようになる。

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma, z : A \Rightarrow B \Rightarrow \alpha \vdash z : A \Rightarrow B \Rightarrow \alpha}\,(init) \quad \Gamma \vdash M : A}{\Gamma, z : A \Rightarrow B \Rightarrow \alpha \vdash zM : B \Rightarrow \alpha}\,(\Rightarrow E) \quad \Gamma \vdash N : B}{\Gamma, z : A \Rightarrow B \Rightarrow \alpha \vdash zMN : \alpha}\,(\Rightarrow E)}{\Gamma \vdash \lambda z.zMN : (A \Rightarrow B \Rightarrow \alpha) \Rightarrow \alpha}\,(\Rightarrow I)}{\Gamma \vdash \lambda z.zMN : \forall\alpha.(A \Rightarrow B \Rightarrow \alpha) \Rightarrow \alpha}\,(\forall I)$$

結果として得られたラムダ項 $\lambda z.zMN$ は、4.2 で導入した順序対 $\langle M, N\rangle$ に他ならないことに注意しよう。次に $(\wedge E)$ 規則については、$\Gamma \vdash M : A \wedge B$ から出発すると次のようになる。

$$\dfrac{\dfrac{\Gamma \vdash M : \forall\alpha.(A \Rightarrow B \Rightarrow \alpha) \Rightarrow \alpha}{\Gamma \vdash M : (A \Rightarrow B \Rightarrow A) \Rightarrow A}\,(\forall E) \quad \dfrac{\dfrac{\dfrac{}{\Gamma, x : A, y : B \vdash x : A}\,(init)}{\Gamma, x : A \vdash \lambda y.x : B \Rightarrow A}\,(\Rightarrow I)}{\Gamma \vdash \lambda xy.x : A \Rightarrow B \Rightarrow A}\,(\Rightarrow I)}{\Gamma \vdash M(\lambda xy.x) : A}\,(\Rightarrow E)$$

結果として得られたラムダ項 $M(\lambda xy.x)$ は 4.2 節では $\mathsf{proj}_1 M$ と書いたものである。一方、もう一つの $(\wedge E)$ 規則

$$\frac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}$$

にラムダ項を割り当てると $\mathsf{proj}_2 M$ が得られる。ゆえに次の対応が成り立つ。

$$\begin{array}{rcl} (\wedge I) \text{ 規則} & \approx & \text{順序対} \\ (\wedge E) \text{ 規則} & \approx & \text{射影} \end{array}$$

このようにして論理推論規則はプログラムの構成法として自然に解釈できるのである。

5.3 プログラムに型を割り当てる

前節では論理の側から出発し、計算の側に到達した。すなわち証明図にはラムダ項が自然に対応し、推論規則には関数を作ったり使ったり、または順序対を作ったり使ったりというプログラムの基本構成が自然に対応することを見た。一言でいえば、証明はプログラムと見なすことができる。

今度は逆に計算の側から出発してみよう。まず注目するのは型の概念である。普通プログラミング言語においては、データは様々な種類に分類されている。たとえば “234” は整数を表し、“true” は真理値（ブール値）を表し、“love” は文字列を表す。整数、真理値、文字列のようなデータの種類のことを**型** (type) という。いま仮に、整数型を **Int** と書き、ブール型を **Bool**、文字列型を **Str** と書くことにし、そして “234” が整数型を持つことを $234 : \mathbf{Int}$ と書くことにする。

型の概念が有用なのは、プログラムのエラーをプログラム実行以前の段階で検出できる点にある。例えば足し算を考える。二つの整数についての足し算、たとえば “234 + 567” を実行せよといわれれば、コンピュータはこれを難なく実行することができる。しかし、ブール値や文字列を含む足し算、たとえば “234 + true” や “234 + love” を実行せよといわれても、コンピュータは戸惑うだけであろう。このようなプログラムを与えると、言語処理系は**型エラー** (type error) を返してくる。すなわち、「足し算をできるのは整数型のデータだけであり、文字列型のデータは足し算できませんよ」と指摘してくるわけである。（もっと微妙なのは、異なるタイプの数の型、たとえば整数型と浮動小数点型の間の演算を許すかどうかであるが、このあたりはプログラム言語によって変わってくる。）

型エラーの検出を行うには、変数の使用にも注意を払う必要がある。たとえば “$x+y$” は一見無害に思えるが、もしも x に整数が、y に文字列が代入されたなら型エラーが発生する。このように、型エラーはプログラムそのものを見ただけでは検出できないこともある。

ではどうすれば型エラーを効果的に検出できるだろうか？基本的なアイデアは型の概念を拡張し、“+” のような関数にも型を与えることである。たとえば “+” は「二つの整数が与えられると一つの整数を返す関数である」ことをあらわすために、次のように**関数型** (function type) を割り当てる。

$$+ : \mathbf{Int} \Rightarrow \mathbf{Int} \Rightarrow \mathbf{Int}$$

そしてプログラムを構成するときには、常に型が合っていなければならないものとする。「型が合っている」ことを規則の形で書けば

$$\frac{M : A \Rightarrow B \quad N : A}{MN : B}$$

となる。このように取り決めをしておけば、正しい型を持つプログラムは問題なく構成することができる。

$$\dfrac{\dfrac{+ : \mathbf{Int} \Rightarrow \mathbf{Int} \Rightarrow \mathbf{Int} \quad 234 : \mathbf{Int}}{+\ 234 : \mathbf{Int} \Rightarrow \mathbf{Int}} \qquad 567 : \mathbf{Int}}{+\ 234\ 567 : \mathbf{Int}}$$

（ただし “234 + 567” のことを “+ 234 567” と順番を変えて書いている。）一方で “234 + love” のように型エラーを含むプログラムは構成することができない。

変数を含むプログラムを正しく構成するには、変数にも型を割りふっておく必要がある。たとえば「変数 x は整数型のデータを格納するのに用いますよ」と宣言するときは、$x : \mathbf{Int}$ と書いて表す。これを変数の**型宣言** (type declaration) という。型宣言の集合を Γ, Δ などの記号を使って表す。そして「Γ のもとでプログラム M は型 A を持つ」ということを $\Gamma \vdash M : A$ と表すことにする。そうすると、正しい型を持つプログラムを構成するための規則として、次のものを自然に考えることができる。

$$\frac{}{\Gamma \vdash x : A}\ (init) \quad \text{ただし } x : A \in \Gamma$$

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : A \Rightarrow B}\ (\Rightarrow I) \qquad \frac{\Gamma \vdash M : A \Rightarrow B \quad \Gamma \vdash N : A}{\Gamma \vdash MN : B}\ (\Rightarrow E)$$

これらの規則に従えば、$\Gamma = \{x : \mathbf{Int}, y : \mathbf{Int}\}$ のもとで $x + y$ は正しい型を持つことが確かめられる。

$$\dfrac{\dfrac{\Gamma \vdash + : \mathbf{Int} \Rightarrow \mathbf{Int} \Rightarrow \mathbf{Int} \quad \dfrac{}{\Gamma \vdash x : \mathbf{Int}}\ (init)}{\Gamma \vdash + x : \mathbf{Int} \Rightarrow \mathbf{Int}}\ (\Rightarrow E) \quad \dfrac{}{\Gamma \vdash y : \mathbf{Int}}\ (init)}{\Gamma \vdash + x\ y : \mathbf{Int}}\ (\Rightarrow E)$$

また、「与えられた整数を倍にして返す」プログラム $(\lambda x. +\ x\ x)$ が型 $\mathbf{Int} \Rightarrow \mathbf{Int}$ を持つことも確かめられる。

$$\dfrac{\dfrac{\dfrac{x : \mathbf{Int} \vdash + : \mathbf{Int} \Rightarrow \mathbf{Int} \Rightarrow \mathbf{Int} \quad \dfrac{}{x : \mathbf{Int} \vdash x : \mathbf{Int}}\ (init)}{x : \mathbf{Int} \vdash + x : \mathbf{Int} \Rightarrow \mathbf{Int}}\ (\Rightarrow E) \quad \dfrac{}{x : \mathbf{Int} \vdash x : \mathbf{Int}}\ (init)}{x : \mathbf{Int} \vdash + x\ x : \mathbf{Int}}\ (\Rightarrow E)}{\vdash \lambda x. +\ x\ x : \mathbf{Int} \Rightarrow \mathbf{Int}}\ (\Rightarrow I)$$

ところで、上で挙げた規則はすでに前節で登場したものである。前節では「証明図にプログラム（ラムダ項）を対応させる」規則として紹介したが、全く同じ規則が「正しい型を持つプログラム（ラムダ項）を構成する」規則としても解釈できるというのが面白い点である。では、上で挙げなかった $\forall$ はプログラミングの文脈ではいったい何に対応するのだろうか？

足し算は数にしか適用できず、ブール値や文字列には適用できない。一方、ある種のプログラム構成はどんな型のデータに対しても適用することができる。典型的なのは順序対 $\langle M, N \rangle$ の構成である。これは M, N の型が何であっても適用することができる。

$$\langle 234, 567 \rangle : \mathbf{Int} \wedge \mathbf{Int}, \qquad \langle 234, \mathsf{true} \rangle : \mathbf{Int} \wedge \mathbf{Bool}, \qquad \langle 234, \mathrm{love} \rangle : \mathbf{Int} \wedge \mathbf{Str}.$$

ただしここで $\mathbf{Int} \wedge \mathbf{Int}$ は二つの整数の順序対を表す型であり、$\mathbf{Int} \wedge \mathbf{Bool}$ は整数とブール値の順序対を表す型である。このような型を**直積型** (product type) という。そこで「与えられた二つのデータの順序対を作る」プログラム $\lambda xy.\langle x, y \rangle$ を考えると、このプログラムは様々な型を持ちうることがわかるだろう。

$$\begin{aligned} \lambda xy.\langle x, y \rangle \quad &: \mathbf{Int} \Rightarrow \mathbf{Int} \Rightarrow \mathbf{Int} \wedge \mathbf{Int} \\ &: \mathbf{Int} \Rightarrow \mathbf{Bool} \Rightarrow \mathbf{Int} \wedge \mathbf{Bool} \\ &: \mathbf{Int} \Rightarrow \mathbf{Str} \Rightarrow \mathbf{Int} \wedge \mathbf{Str} \end{aligned}$$

では、このように「様々な型を持ちうる」ことを一つの型を使って表す方法はないだろうか？ここで登場するのが $\forall$ である。$\forall$ を使えば上の様々な型は単一の型を使って表すことができる。

$$\lambda xy.\langle x, y\rangle : \forall\alpha.\forall\beta.\alpha \Rightarrow \beta \Rightarrow \alpha \wedge \beta$$

この型が意味しているのは、「どんな型 A, B についてもプログラム $\lambda xy.\langle x, y\rangle$ は型 $A \Rightarrow B \Rightarrow A \wedge B$ を持つ」ということである。このような型を**多相型** (polymorphic type) という。正しい多相型を持つプログラムを導出し、そして多相型のプログラムを正しく使うために必要なのが、まさに前節で登場した $(\forall I)$ 規則と $(\forall E)$ 規則なのである。

$$\frac{\Gamma \vdash M : A}{\Gamma \vdash M : \forall\alpha.A}\ (\forall I) \qquad \frac{\Gamma \vdash M : \forall\alpha.A(\alpha)}{\Gamma \vdash M : A(B)}\ (\forall E)$$

さらには、前節で説明したとおり $\forall$ があれば $\wedge$ や $\vee$ などの論理結合子が定義できる。たとえば

$$A \wedge B := \forall\alpha.(A \Rightarrow B \Rightarrow \alpha) \Rightarrow \alpha$$

と定義すれば $(\wedge I)$ 規則と $(\wedge E)$ 規則がシミュレートできる。4.2 節の記法を用いれば次のようになる。

$$\frac{\Gamma \vdash M : A \quad \Gamma \vdash N : B}{\Gamma \vdash \langle M, N\rangle : A \wedge B}\ (\wedge I) \qquad \frac{\Gamma \vdash M : A \wedge B}{\Gamma \vdash \mathsf{proj}_1 M : A}\ (\wedge E) \qquad \frac{\Gamma \vdash M : A \wedge B}{\Gamma \vdash \mathsf{proj}_2 M : B}\ (\wedge E)$$

以上の規則を用いれば、プログラム $\lambda xy.\langle x, y\rangle$ が型 $\forall\alpha.\forall\beta.\alpha \Rightarrow \beta \Rightarrow \alpha \wedge \beta$ を持つことを確かめることができる。（ここで $\Gamma = \{x : \alpha, y : \beta\}$ とする。）

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash x : \alpha}\ (init) \quad \dfrac{}{\Gamma \vdash y : \beta}\ (init)}{\Gamma \vdash \langle x, y\rangle : \alpha \wedge \beta}\ (\wedge I)}{x : \alpha \vdash \lambda y.\langle x, y\rangle : \beta \Rightarrow \alpha \wedge \beta}\ (\Rightarrow I)}{\vdash \lambda xy.\langle x, y\rangle : \alpha \Rightarrow \beta \Rightarrow \alpha \wedge \beta}\ (\Rightarrow I)}{\vdash \lambda xy.\langle x, y\rangle : \forall\beta.\alpha \Rightarrow \beta \Rightarrow \alpha \wedge \beta}\ (\forall I)}{\vdash \lambda xy.\langle x, y\rangle : \forall\alpha.\forall\beta.\alpha \Rightarrow \beta \Rightarrow \alpha \wedge \beta}\ (\forall I)$$

具体的な型を持つデータに対してこのプログラムを用いるには $(\forall E)$ 規則を用いればよい。（ここで $M = \lambda xy.\langle x, y\rangle$ とする。）

$$\dfrac{\dfrac{\dfrac{\dfrac{\vdash M : \forall\alpha.\forall\beta.\alpha \Rightarrow \beta \Rightarrow \alpha \wedge \beta}{\vdash M : \forall\beta.\mathbf{Int} \Rightarrow \beta \Rightarrow \mathbf{Int} \wedge \beta}\ (\forall E)}{\vdash M : \mathbf{Int} \Rightarrow \mathbf{Bool} \Rightarrow \mathbf{Int} \wedge \mathbf{Bool}}\ (\forall E) \quad \vdash 234 : \mathbf{Int}}{\vdash M\ 234 : \mathbf{Bool} \Rightarrow \mathbf{Int} \wedge \mathbf{Bool}}\ (\Rightarrow E) \quad \vdash \mathsf{true} : \mathbf{Bool}}{\vdash M\ 234\ \mathsf{true} : \mathbf{Int} \wedge \mathbf{Bool}}\ (\Rightarrow E)$$

$M\ 234\ \mathsf{true} \equiv (\lambda xy.\langle x, y\rangle)\ 234\ \mathsf{true} \longrightarrow_\beta^* \langle 234, \mathsf{true}\rangle$ となることはすでに見た通りである。

5.4 システム F

一般に、プログラムに型を与えるシステムのことを**型システム** (type system) という。型システムには二つの流儀がある。型をプログラムの一部と考えるかそうでないかによりチャーチ流、カリー流などと呼ばれる。本講義で考える型システムはカリー流であり、型はプログラムの一部ではない。

さて、二階直観主義命題論理 **IL**2 は型システムと考えられることを今まで見てきたわけだが、実は **IL**2 を型システムとみなすときには別の名前が付けられており、**システム F**(System F) と呼ばれるのが普通である。これは**多相型ラムダ計算** (polymorphic lambda calculus) の代表例である。システム **F** は 1970 年代の初めにジラールとレイノルズにより別々に考案された。お互いに相手の研究のことを全く知らずに独自に考えたものが、後になって同じだとわかったのである。前者は証明論の大家、後者はプログラミング言語理論の大家であるという事実は、いかに論理と計算が密接に関わりあっているかを示しており、大変象徴的である。

ジラールがシステム **F** を導入した背景には、ヒルベルト・ゲンツェンの証明論の延長線上で日本の数学者・竹内外史が 1950 年代に立てた予想、いわゆる**竹内の基本予想** (Takeuti's fundamental conjecture) がある。これはヒルベルトのプログラムの発展形だと思ってよい。一方、レイノルズの強い影響下で、多相型という考え方はプログラミングの基本原理として、多くのプログラミング言語に採用されるようになる。代表的なのは関数型プログラミング言語 ML である。

ついでに言っておくと、型の概念を最初に系統立てて考察したのは 20 世紀初頭の論理主義者ラッセルである。現代における型システムの研究は一般に**型理論** (type theory) と呼ばれているが、これは元をたどればラッセルの型理論に由来するものである。以上のことを総合的に勘案すれば、システム **F** とは現代的な型システムの雛型であると同時に、ヒルベルトの形式主義、ブラウアーの直観主義、ラッセルの論理主義の集大成であるともいえる。こんなふうにして伝統的な数学基礎論は現代の計算機科学につながっているのである。

IL2 とシステム **F** は実質上同じものなのだが、前者は論理の体系、後者は計算の体系として考えられているので、使われる用語はかなり異なる。この節では計算の文脈で用いられる用語を意識してシステム **F** を再定義しておくことにする。

定義 5.7 (型、型判断) システム **F** の**型** *(type)* は次の *BNF* 記法により定義される。

$$A, B ::= \alpha \mid (A \Rightarrow B) \mid \forall \alpha.A$$

x をラムダ計算の変数とするとき、$x : A$ を**型宣言** *(type declaration)* という。型宣言の有限集合 $\{x_1 : A_1, \ldots, x_n : A_n\}$ を**型環境** *(type environment)* という。ただしここで $x_1, \ldots, x_n$ は互いに異なる変数とする。($A_1, \ldots, A_n$ は同一であってもよい。) 型環境を表すのに Γ, Δ 等の記号を用いる。$\Gamma = \{x_1 : A_1, \ldots, x_n : A_n\}$ のとき、外側のかっこを省略して単に $x_1 : A_1, \ldots, x_n : A_n$ と書く。また集合 $\Gamma \cup \{x : A\}$ のことを単に $\Gamma, x : A$ と書く。M をラムダ項とするとき、$\Gamma \vdash M : A$ の形の表現を**型判断** *(type judgment)* という。Γ が空集合のときには単に $\vdash M : A$ と書く。

システム **F** の**型推論規則** (type inference rule) は以下の 5 つである。

$$\frac{}{\Gamma \vdash x : A}\ (init) \quad \text{ただし } x : A \in \Gamma$$

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : A \Rightarrow B}\ (\Rightarrow I) \qquad \frac{\Gamma \vdash M : A \Rightarrow B \quad \Gamma \vdash N : A}{\Gamma \vdash MN : B}\ (\Rightarrow E)$$

$$\frac{\Gamma \vdash M : A}{\Gamma \vdash M : \forall\alpha.A}\ (\forall I) \qquad \frac{\Gamma \vdash M : \forall\alpha.A(\alpha)}{\Gamma \vdash M : A(B)}\ (\forall E)$$

ただし $(\forall I)$ 規則を使う際には、Γ の中で α が自由変数として用いられていてはならない (**固有変数条件**, eigenvariable condition)。

上の型推論規則を使って型判断 $\Gamma \vdash M : A$ が導出できるとき、M は型環境 Γ のもとで型 A を持つといい $\Gamma \vdash_{\mathbf{F}} M : A$ と書く。Γ が空集合のときは単に M は型 A を持つという。ラムダ項 M が与えられたとき、何らかの Γ, A について $\Gamma \vdash_{\mathbf{F}} M : A$ となるとき、M は**型付け可能** (typable) であるという。

明らかに次の定理が成り立つ。

定理 5.8 $\Gamma = \{A_1, \ldots, A_n\}$, $\Gamma' = \{x_1 : A_1, \ldots, x_n : A_n\}$ とするとき、以下の二つは同値である。

1. $\Gamma \vdash_{\mathbf{IL2}} B$
2. $\Gamma' \vdash_{\mathbf{F}} M : B$ となるラムダ項 M が存在する。

とくに **IL2** で論理式 B が証明できることとシステム **F** で型 B を持つラムダ項 M が存在することは同値である。

最後の主張は、実際には次のことを意味する。まず、**IL2** で論理式 B が証明できるということは、シークエント $\vdash B$ の証明図が存在するということである。5.2 節で説明したとおり、証明図が一つ与えられれば、それに従ってラムダ項 M を構成することができる。このとき M は与えられた証明図の構造を何らかの意味で反映しているものとみなすことができる。証明図のスケルトンであるといってもよい。

一般に、一つの証明可能なシークエントに対して複数の証明図が存在しうるから、どのような証明図を考えるかによって M は異なってくる。本当のことをいえば、ここに見られる証明図（の集合）とラムダ項（の集合）の間の対応関係は “同型的” だと主張したいのだが、残念ながらこのことを正確に述べるにはカリー流ではなく、チャーチ流の型システムを採用しなければならない。ゆえに本講義では

型 B のラムダ項はシークエント $\vdash B$ の証明図の構造を反映する

という弱い主張で満足して話を進めることにする。

5.5 データ型の定義

システム **F** では、$\forall$ と $\Rightarrow$ を使うことにより様々な型が定義できる。論理に対応するものとしては次のものが挙げられる。

$$\begin{array}{rcll} A \Rightarrow B & & & \text{関数型} \\ A \wedge B & := & \forall\alpha.(A \Rightarrow B \Rightarrow \alpha) \Rightarrow \alpha & \text{直積型} \\ A \vee B & := & \forall\alpha.(A \Rightarrow \alpha) \Rightarrow (B \Rightarrow \alpha) \Rightarrow \alpha & \text{直和型} \end{array}$$

関数型と直積型についてはすでに述べた。論理における選言 $A \vee B$ は計算においては型 A と型 B の直和に対応する。このことは、連言 $A \wedge B$ と直積の対応関係を調べたときと同じようにすればわかる。すなわち、$(\vee I)$ 規則と $(\vee E)$ 規則を **IL2** でシミュレートし、それをシステム **F** でラムダ項に翻訳してみて、その働きを調べてみればわかるのであるが、ここでは省略する。

次にブール値を表す**ブール型** (Boolean type)、自然数を表す**自然数型** (Natural number type) を以下で定義する。

$$\begin{array}{rcll} \mathbf{Bool} & := & \forall\alpha.\alpha \Rightarrow \alpha \Rightarrow \alpha & \text{ブール型} \\ \mathbf{Nat} & := & \forall\alpha.(\alpha \Rightarrow \alpha) \Rightarrow (\alpha \Rightarrow \alpha) & \text{自然数型} \end{array}$$

このようにデータを表す型を一般に**データ型** (data type) という。これらが確かにブール値、自然数を表すことは次の命題により保証できる。

命題 5.9

1. ブール型 **Bool** を持つ正規形のラムダ項は true と false に限られる。
2. 自然数型 **Nat** を持つ正規形のラムダ項はチャーチ数項 n $(n \in \mathbb{N})$ および $\mathsf{id} \equiv \lambda f.f$ に限られる。

念のため、証明の前にチャーチ数項とは何だったかを思い出しておこう。

$$\mathsf{0} \equiv \lambda fz.z, \qquad \mathsf{1} \equiv \lambda fz.fz, \qquad \mathsf{2} \equiv \lambda fz.f(fz), \qquad \mathsf{3} \equiv \lambda fz.f(f(fz)), \qquad \cdots$$

証明 1. $\mathsf{true} \equiv \lambda xy.x$ が **Bool** 型を持つことは次のようにして確かめることができる。

$$\dfrac{\dfrac{\dfrac{\dfrac{}{x:\alpha, y:\alpha \vdash x:\alpha}\,(init)}{x:\alpha \vdash \lambda y.x:\alpha \Rightarrow \alpha}\,(\Rightarrow I)}{\vdash \lambda xy.x:\alpha \Rightarrow \alpha \Rightarrow \alpha}\,(\Rightarrow I)}{\vdash \lambda xy.x:\forall\alpha.\alpha \Rightarrow \alpha \Rightarrow \alpha}\,(\forall I)$$

$\vdash_{\mathbf{F}} \mathsf{false} : \mathbf{Bool}$ が成り立つことも同様にして確かめることができる。

2. たとえば $2 \equiv \lambda fz.f(fz)$ が **Nat** 型を持つことは次のとおりである。（ここで $\Gamma = \{f : \alpha \Rightarrow \alpha, z : \alpha\}$ とする。）

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash f : \alpha \Rightarrow \alpha}\,(init) \quad \dfrac{\dfrac{}{\Gamma \vdash f : \alpha \Rightarrow \alpha}\,(init) \quad \dfrac{}{\Gamma \vdash z : \alpha}\,(init)}{\Gamma \vdash fz : \alpha}\,(\Rightarrow E)}{\Gamma \vdash f(fz) : \alpha}\,(\Rightarrow E)}{f : \alpha \Rightarrow \alpha \vdash \lambda z.f(fz) : \alpha \Rightarrow \alpha}\,(\Rightarrow I)}{\vdash \lambda fz.f(fz) : (\alpha \Rightarrow \alpha) \Rightarrow (\alpha \Rightarrow \alpha)}\,(\Rightarrow I)}{\vdash \lambda fz.f(fz) : \forall\alpha.(\alpha \Rightarrow \alpha) \Rightarrow (\alpha \Rightarrow \alpha)}\,(\forall I)$$

逆に **Bool** 型、**Nat** 型を持つ正規形のラムダ項は上のものに限られることを示さなければならないが、それには多少の準備が必要なので後にまわすことにする。 ■

注意 5.10 基本的に **Nat** 型を持つものはチャーチ数に限られるが、例外として id がある。例外があることは気持ち悪く感じられるかもしれないが、実用上問題が生じることはほとんどない。それは次の理由による。

M を変数 z を含まないラムダ項とするとき、意味的に考えると M と $\lambda z.Mz$ は同一視することができる。実際、任意のラムダ項 N について $(\lambda z.Mz)N =_\beta MN$ であるから、M と $\lambda z.Mz$ が関数として用いられるときには両者は同じ働きをすると言ってよい。このことを M と $\lambda z.Mz$ は η **同値** *(eta equivalent)* であるという。

話を元に戻すと、$\mathsf{id} \equiv \lambda f.f$ はチャーチ数項 $1 \equiv \lambda f.\lambda z.fz$ と η 同値である。ゆえに意味的に考えれば両者は同じものを表すとして差し支えないのである。

もしもどうしても id を排除したければ、チャーチ数項 n の定義を $\lambda zf.\underbrace{f(f(\cdots(f}_{n} z)\cdots))$ と変え、**Nat** 型の定義を $\forall\alpha.\alpha \Rightarrow (\alpha \Rightarrow \alpha) \Rightarrow \alpha$ に変えるという手もある。こうすれば id は **Nat** 型を持たなくなる。しかし美的観点からいってあまりよろしくないので、このようなことはせず id という例外の存在に甘んじるのが慣例である。

4.2 節で挙げたブール値や自然数上の関数にも自然の型を与えることができる。たとえば次のことが成り立つ。

$$\vdash_{\mathbf{F}} \mathsf{neg} : \mathbf{Bool} \Rightarrow \mathbf{Bool} \quad \vdash_{\mathbf{F}} \mathsf{conj} : \mathbf{Bool} \Rightarrow \mathbf{Bool} \Rightarrow \mathbf{Bool}$$
$$\vdash_{\mathbf{F}} \mathsf{suc} : \mathbf{Nat} \Rightarrow \mathbf{Nat} \quad \vdash_{\mathbf{F}} \mathsf{add} : \mathbf{Nat} \Rightarrow \mathbf{Nat} \Rightarrow \mathbf{Nat}$$

そして重要なのが次の事実である。

命題 5.11 *4.2* 節で定義したラムダ項 rec について

$$\vdash_{\mathbf{F}} \mathsf{rec} : \forall\beta.(\beta \Rightarrow \beta) \Rightarrow \beta \Rightarrow \mathbf{Nat} \Rightarrow \beta$$

が成り立つ。

基本的な関数が型を持ち、しかも原始再帰法も使えるので多くの関数がシステム **F** の中で型を持つことになる。たとえば階乗を計算するプログラム fact が型 $\mathbf{Nat} \Rightarrow \mathbf{Nat}$ を持つことがわかる。

それでは不動点 fix$M$ は一体どのような型を持つだろうか？これは後で検討する重要なテーマであるが、結論を先に言ってしまうと fix$M$ はシステム **F** では型を持たない。どんなプログラムも型を持つというわけではないのである。

5.6 システム F の性質

本節ではシステム **F** が満たすいくつかの性質を証明する。中でも最も重要なのは、β 簡約により型は変わらないという性質（定理 5.14）である。

補題 5.12 (代入) $\Gamma, x : A \vdash_{\mathbf{F}} M[x] : B$ かつ $\Gamma \vdash_{\mathbf{F}} N : A$ ならば $\Gamma \vdash_{\mathbf{F}} M[N] : B$ である。

話を単純にするため、仮に $\Gamma = \emptyset$ としよう。するとこの補題は論理的には次のことを意味する。「仮定 A から B が帰結し、なおかつ A が成り立つならば、B は仮定なしで成り立つ。」これは直感的には明らかだろう。

証明 $\Gamma, x : A \vdash_{\mathbf{F}} M[x] : B$ の証明の構造に関する帰納法による。型判断 $\Gamma, x : A \vdash M[x] : B$ が $(init)$ 規則により導かれるときには、$M[x] \equiv x$ であるから $M[N] \equiv N$ であり、また $A \equiv B$ であるから、示すべき事柄 $\Gamma \vdash_{\mathbf{F}} M[N] : B$ はもう一つの仮定 $\Gamma \vdash_{\mathbf{F}} N : A$ そのものである。

$\Gamma, x : A \vdash_{\mathbf{F}} M[x] : B$ がその他の規則により導かれるときには、帰納法の仮定を用いれば簡単に結論を示すことができる。 ■

次の補題は、型推論規則 $(\forall I)$ と $(\Rightarrow I)$ 規則は逆方向に用いてもよいことを示している。

補題 5.13 (反転)

1. $\Gamma \vdash_{\mathbf{F}} M : \forall\alpha.A$ ならば $\Gamma \vdash_{\mathbf{F}} M : A$.
2. $\Gamma \vdash_{\mathbf{F}} \lambda x.M : A \Rightarrow B$ ならば $\Gamma, x : A \vdash_{\mathbf{F}} M : B$.

証明 1.$(\forall E)$ 規則を用いればよい。

2. まず次のことに注意する。ある証明図の中で $(\forall I)$ 規則と $(\forall E)$ 規則が連続して使われているときには、次のような書き換えをして証明を簡略化することができる。

$$\dfrac{\dfrac{\dfrac{\begin{array}{c}\vdots\ \pi(\alpha)\\ \Gamma \vdash M : A(\alpha)\end{array}}{\Gamma \vdash M : \forall\alpha.A(\alpha)}\ (\forall I)}{\Gamma \vdash M : A(B)}\ (\forall E)}{\begin{array}{c}\vdots\ \tau\end{array}} \quad \Longrightarrow \quad \begin{array}{c}\vdots\ \pi(B)\\ \Gamma \vdash M : A(B)\\ \vdots\ \tau\end{array}$$

ただし τ と $\pi(\alpha)$ は証明図の部分を表し、$\pi(B)$ は証明図 $\pi(\alpha)$ の中に出てくる α のうち適切なものを B に置き換えて得られる証明図とする。そのような置き換えをしても Γ が変わらないことは $(\forall I)$ 規則に関する固有変数条件による（Γ の中で α が自由変数として用いられていてはならない）。

さて、上の書き換えをすると証明図は必ず小さくなる。それゆえ上の書き換えを無限に繰り返すことはできず、いつかは $(\forall I)$ 規則と $(\forall E)$ 規則が連続して使われていないような証明図に到達するはずである。そのような証明図を仮に $\forall$ **正規** ($\forall$-normal) であるということにしよう。

2 の証明に戻り、$\Gamma \vdash \lambda x.M : A \Rightarrow B$ が証明図 π を持つとする。上で述べたことにより、π は $\forall$ 正規であると仮定して差し支えない。では、π において一番最後に用いられる規則は何だろうか？$(init)$ や $(\Rightarrow E)$ は $\lambda x.M$ の部分で合致しないし、$(\forall I)$ は $A \Rightarrow B$ の部分で合致しない。ゆえに最後に用いられる規則は $(\Rightarrow I)$ か $(\forall E)$ のどちらかのはずである。前者の場合、π の最後の部分は

$$\dfrac{\begin{array}{c}\vdots\\ \Gamma, x : A \vdash M : B\end{array}}{\Gamma \vdash \lambda x.M : A \Rightarrow B}\ (\Rightarrow I)$$

となっているので、一つ前の段階で確かに $\Gamma, x : A \vdash M : B$ が証明されていることわかる。後者の場合、π の最後の部分は

$$\dfrac{\begin{array}{c}\vdots\\ \Gamma \vdash \lambda x.M : \forall\alpha.A'(\alpha) \Rightarrow B'(\alpha)\end{array}}{\Gamma \vdash \lambda x.M : A \Rightarrow B}\ (\forall E)$$

となっているはずである。ここでさらに一つ前で用いられる規則は何だろうかと考えてみると、$(\forall I)$ 規則か $(\forall E)$ 規則しかありえないことがわかる。しかし前者を選ぶと $\forall$ 正規性の仮定に反するし、後者を選ぶと同様の議論が繰り返せて無限後退に陥ってしまう。 ■

先に進む前に、後回しにしておいた命題 5.9 の証明をこの当たりで完結させておこう。

命題 5.9 の証明の続き 示さなければならないのは次のことであった。もしも M が正規形であり、$\vdash_{\mathbf{F}} M : \mathbf{Bool}$ が成り立つならば、M は true か false のどちらかかである。

まず、$\vdash_{\mathbf{F}} M : \mathbf{Bool}$ に補題 5.13 の 1 を用いれば $\vdash_{\mathbf{F}} M : \alpha \Rightarrow \alpha \Rightarrow \alpha$ が成り立つことがわかる。ここで M は変数ではありえない（なぜならば型環境 Γ が空集合だからである）。仮に M が適用 $M_0 M_1$ の形だとすると、同じ理由により M_0 は変数ではありえず、またラムダ抽象 $\lambda x.M_0'$ でもありえない（なぜならば M は正規形だからである）。すると M_0 自体も適用 $M_{00} M_{01}$ ということになる。すると M_{00} は変数ではありえず、ラムダ抽象でもありえない。ゆえに M_{00} は適用 $M_{000} M_{001}$ であり $\cdots$ この議論はいつまでも繰り返すことができて無限後退となってしまう。

ゆえに M はラムダ抽象でしかありえない。同様の議論（今度はもう少し注意深い論証が必要である）をもう一度行うと、結局 $M \equiv \lambda xy.N$ であることがわかる。すると補題 5.13 の 2 により、$x : \alpha, y : \alpha \vdash_{\mathbf{F}} N : \alpha$ であるが、型推論規則をようく吟味することにより、N は x か y のどちらかしかありえないことがわかる。ゆえに M は true または false のどちらかである。

Nat 型についての主張も同様にして証明できる。 ■

本筋に戻って、次の型保存定理に進もう。これは型システムが満たすべき性質のうちでもっとも基本的なものである。（英語では subject reduction theorem と呼ばれるが、しっくりといく訳語が見つからなかったので型保存定理と呼ぶことにする。）

定理 5.14 (型保存) $\Delta \vdash_{\mathbf{F}} M_0 : C$ かつ $M_0 \longrightarrow_\beta M_1$ ならば、$\Delta \vdash_{\mathbf{F}} M_1 : C$ である。

証明　$M_0 \longrightarrow_\beta M_1$ ということは、M_0 の中に β 基 $(\lambda x.M[x])N$ があって、M_1 はそれを $M[N]$ で置き換えたものだということである。ということは $\Gamma \vdash M_0 : A$ の証明図の中にも $(\lambda x.M[x])N$ が出てくるはずである。そこで $(\Rightarrow E)$ 規則を使って $(\lambda x.M[x])N$ を構成している部分に着目すると、次のようになっているはずである。

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \Gamma, x : A \vdash M[x] : B\end{array}}{\Gamma \vdash \lambda x.M[x] : A \Rightarrow B}\,(\Rightarrow I) \qquad \begin{array}{c}\vdots\\ \Gamma \vdash N : A\end{array}}{\Gamma \vdash (\lambda x.M[x])N : B}\,(\Rightarrow E)$$

ここで $(\Rightarrow E)$ 規則の直前で $(\Rightarrow I)$ 規則が使われていると仮定してよいのは、補題 5.13 の 2 による。よって $\Gamma, x : A \vdash_{\mathbf{F}} M[x] : B$ と $\Gamma \vdash_{\mathbf{F}} N : A$ が成り立つので、補題 5.12 により $\Gamma \vdash M[N] : B$ が証明図を持つことがわかる。$\Delta \vdash M_0 : C$ の証明図のうち上の部分をこの新しい証明図で置き換えれば、$\Delta \vdash M_1 : C$ の証明図を得ることができる。 ■

型保存定理の意義を考えてみよう。いま、プログラム M が型 $\mathbf{Nat} \Rightarrow \mathbf{Bool}$ を持つとする。このプログラムを実行するには、入力 $\mathsf{n} : \mathbf{Nat}$ を与えて $M\ \mathsf{n} : \mathbf{Bool}$ とした上で β 簡約をすればよい。すると

$$M\ \mathsf{n} \longrightarrow_\beta M_1 \longrightarrow_\beta M_2 \longrightarrow_\beta M_3 \longrightarrow_\beta \cdots$$

というふうに計算は進むが、型保存定理によれば、$M\ \mathsf{n}$ のみならず、すべての M_i がうまく型付けができるはずである。すなわち、プログラムの実行過程で決して型エラーは生じず、“234 + love” のような項が出てくることは決してないということがわかる。ゆえに $\vdash_{\mathbf{F}} M : \mathbf{Nat} \Rightarrow \mathbf{Bool}$ が成り立つことさえ示すことができれば、どんな入力が与えられても決して実行時に型エラーが発生しないことを理論的に保証できるのである。プログラムを実行せずともエラーが生じないとわかる。これはすごいことである。

また、$M\ \mathsf{n}$ を実行した結果、何らかの正規形 N に到達したとしよう（後で示すように、システム $\mathbf{F}$ の場合はこのことは常に成り立つ)。すると型保存定理により N もやはり $\mathbf{Bool}$ 型を持つことがわかる。ゆえに命題 5.9 により N は true か false のどちらかである。すなわちプログラム実行の結果は常に真理値であり、得体のしれない何かが出てくることは決してないということも理論的に保証できるのである。

このようにしてみると、実用上決定的に重要なのは**型推論** (type inference)、すなわちプログラム M が与えられたときに適切な型を割り当てる作業であることがわかる。しかし型推論を手作業で行うのがいかに煩雑であるかは、これまでの諸例を見てもらえれば十分体感できるはずである。ゆえに型推論はある程度機械的に行われることが望ましいが、ここには限界のあることが (Wells 1994) により証明されている。

定理 5.15 次の二つの問題はいずれも決定不能である。

1. ラムダ項 M と型 A が与えられたとき、$\vdash_{\mathbf{F}} M : A$ は成り立つか？(**型判定問題**, *type checking problem*)
2. ラムダ項 M が与えられたとき、$\vdash_{\mathbf{F}} M : A$ となるような型 A は存在するか？(**型付け可能性問題**, *typability problem*)

これはシステム **F** で採用されている多相型があまりにも複雑であり、実用にはそぐわないことに起因する。そこで、多相型の有用な部分は維持しつつももう少し単純な型だけに制限することにより、上の問題を決定可能にしようという方向性が自然にでてくる。そうすれば、プログラマは煩雑な型のことはあまり気にせずに自由にプログラムを書くことができ、型推論は言語処理系のほうである程度自動的にやってくれるという理想的な状況が得られるだろう。そのような方向性のもとで生まれたプログラミング言語の代表例が ML である。

プログラムに型を与えることによって、プログラムが意図通りに動き、実行時にエラーが発生しないことを事前に保証することができる。すなわち型システムは**プログラム検証** (program verification) のための強力な手段を与えてくれるのである。システム **F** で検証できるのは、せいぜい型エラーが発生しないことと、意図通りの型の実行結果が得られること（および次章で示すようにプログラム実行が常に停止すること）くらいであるが、より複雑な型システムを考えることによって、様々な性質が検証できるようになる。型によるプログラム検証の理論は、実用性があるのみならず理論的にも奥深い分野であり、今なお世界中で精力的に研究が進められている。20 世紀初頭に数学の基礎付けのためにラッセルが考案した型理論は、姿を変え形を変え、21 世紀になった現代でも活き続けているのである。

5.7 カリー・ハワード対応

これまでにラムダ計算における型とラムダ項は、それぞれ論理における論理式と証明図に対応することを見てきた。では、ラムダ計算における β 簡約は、論理においては一体何に対応するのだろうか？

定理 5.14 の証明でかいま見たことであるが、β 簡約に対応するのは次のような証明図の書き換えである。

$$\dfrac{\dfrac{\dfrac{}{\Gamma', A \vdash A}\,(init)}{\vdots\,\pi}}{\dfrac{\dfrac{\Gamma, A \vdash B}{\Gamma \vdash A \Rightarrow B}\,(\Rightarrow I) \qquad \begin{matrix}\vdots\,\tau\\ \Gamma \vdash A\end{matrix}}{\Gamma \vdash B}\,(\Rightarrow E)} \quad \Longrightarrow \quad \begin{matrix}\vdots\,\tau\\ \Gamma \vdash A\\ \vdots\,\pi'\\ \Gamma \vdash B\end{matrix}$$

左側の証明図はある意味で無駄なことをしている。Γ を省略して説明すると、次のようになる。まず A を仮定して B を示し (π)、それにより $A \Rightarrow B$ と主張した直後に、A が成り立つこと (τ) を根拠に B を主張している。しかしそのようにまわりくどいことをするくらいなら、まず A を示し (τ)、それから B を示したほうが (π') はるかに手っ取り早い。これが右側の証明図である。

ゆえに論理の観点からいって β 簡約に相当するのは、「まわり道を含む証明を単純な証明に書き換える」操作であると言ってよい。ただし注意が必要である。このように述べたからといって、右側の単純な証明さえあれば左側のまわり道を含む証明は不要であるかのような印象を受けたとしたら、それは誤解である。まわり道を含む証明は数学において絶対不可欠なものだからである。

数学である程度大きな定理を証明するときには、まずはいくつかの補題を先に示し、それらを組み合わせることによって定理を証明するということが頻繁に行われる。そのようにするで議論の筋道がはっきりとし、またくりかえし使われる論法や性質を補題としてまとめて証明することで証明のコストを大幅に節約できる。そのような補題を用いた証明が、まわり道を含む証明の典型例なのである。上図左の証明図は “補題”$A \Rightarrow B$ および “補題”A を先に示した上で “定理”B を証明する過程を示したものとみることができる。一方、右の証明図は補題を用いずに、B を直接証明する過程とみることができる。ゆえに β 簡約とは、「補題を用いた間接的な証明を補題を用いない直接的な証明に書き換えること」ともいえる。しかしだからといって補題が不要になるわけではない。事実、間接的な証明から補題を除去して直接的な証明に直すと、証明が天文学的に長くなってしまう可能性がある。断言するが、補題は絶対に必要である。

補題を用いることの利点は、単に証明コストの軽減のみではない。そのことを実感してもらうため、もう少し複雑な状況を考えてみる。（簡単にするため、$\Gamma = \emptyset$ とする。）

$$\dfrac{\dfrac{\dfrac{\dfrac{\vdots\ \pi}{A(\alpha) \vdash B(\alpha)}}{\vdash A(\alpha) \Rightarrow B(\alpha)}\ (\Rightarrow I)}{\vdash \forall\alpha.A(\alpha) \Rightarrow B(\alpha)}\ (\forall I)}{\vdash A(C) \Rightarrow B(C)}\ (\forall E) \qquad \dfrac{\vdots\ \tau}{\vdash A(C)}}{\vdash B(C)}\ (\Rightarrow E) \quad \Longrightarrow \quad \begin{array}{c}\vdots\ \tau\\ \vdash A(C)\\ \vdots\ \pi'\\ \vdash B(C)\end{array}$$

左の証明図では、一般的かつ抽象的な “補題” $\forall\alpha.A(\alpha) \Rightarrow B(\alpha)$ を示した上で、それを $\alpha = C$ という特別な場合に適用している。この証明図は、右の証明図に簡略化できる。右では、一般的・抽象的な補題は用いず、$A(C)$ や $B(C)$ といった具体的な命題のみを用いて論証を進めている。

この状況に当てはまる典型例は、群・環・体といった代数構造の使用であろう。右はたとえば、数論的な定理を純粋に初等的な方法のみを用いて証明する過程を表すものと思ってほしい。左はたとえば、全ての “環”α について成り立つ一般的な補題 $\forall\alpha.A(\alpha) \Rightarrow B(\alpha)$ を示した上でそれを “整数環”C という具体的な状況にあてはめるような代数的証明を表すものとみることができる。このような状況でいえば、β 簡約とは「抽象的概念や構造を用いた証明を、初等的証明に書き換えること」とも考えることができる。しかし仮にこのような書き換えが原理上可能だとしても、だからといって抽象的概念・構造は不要であると主張する人は誰もいないだろう。事実、抽象的代数構造の発見により諸理論の統合が進められ、数学が飛躍的に深化してきたことは誰にも否定しえないからである。

最後に、これまでに論じてきたカリー・ハワード対応についてまとめた表を挙げておく。

論理	計算
二階直観主義命題論理 $\mathbf{IL2}$	システム $\mathbf{F}$
論理式	型
証明図	ラムダ項
証明の単純化	ラムダ項の β 簡約
補題を用いない直接的証明	正規形
含意 $A \Rightarrow B$	関数型
連言 $A \wedge B$	直積型
選言 $A \vee B$	直和型
全称量化 $\forall \alpha.A$	多相型

6　論理的述語 — 命題の意味は証明にあり

「証明可能な論理式は真である。」意味論の健全性と呼ばれる性質は、古典論理について成り立つ最も重要な性質の一つである。本章ではこのアイデアを直観主義的な方向に改変したものを取り上げる。ブラウアー本来の直観主義においては、人間による証明とは独立して命題が真であったり偽であったりということは意味をなさない。数学とは主体による構成であり、主体とは独立した神秘的な宇宙で真偽が事前に定まっているといった類のものではないというのが直観主義の根本思想だからである。直観主義において真偽に代わる役割を果たすのはむしろ証明の概念である。

古典論理においては各論理記号の "意味" は命題が成り立つための条件（**真理条件**, truth condition）によって与えられる。たとえば、$A \to B$ が真となるのは、A が真なときには常に B も真となるとき、言いかえれば A が偽であるか B が真であるかのどちらかのときである。この真理条件をもって含意記号 $\to$ の "意味" を定めるのである。

一方、直観主義論理においては、各論理記号の "意味" は命題の証明（証拠）とは何かを規定することによって与えられる。たとえば、標準的なブラウアー・ハイティング・コルモゴロフ解釈（**BHK 解釈**, BHK interpretation）によれば、含意記号 $\Rightarrow$ の意味は次の条件をもって与えられる。

- $A \Rightarrow B$ の証明（証拠）とは、A の証明が与えられたときにそれを B の証明へと変換する構成法である。

もっともここでいう "構成法" とは一体何なのかはあまり明確ではなく、何をもって "構成法" とみなすかによって様々な思想的立場が存在しうる。しかし本講義ではそのような哲学的問題には立ち入らない。そのかわり、この BHK 解釈の計算機科学版といってもよい**論理的述語** (logical predicate) の理論を取り上げて説明する。これはもとをたどれば BHK 解釈、クリーニの実現可能性解釈、クライゼルの実現可能性解釈・改などにいきつくものであるが、本章で述べることに直接関係するのは、テイトの簡約可能性論法、ジラールの簡約可能性候補、およびプロトキンの論理的関係などである。論理的関係の 1 項版なので、論理的述語と呼ぶことにする。

6.1　論理的述語の基本補題

ここで取り上げるのは論理的述語の特別な場合であり、その基本的アイデアは、型 A をラムダ項の集合 $[\![A]\!]$ により解釈することにある。しかしラムダ項の集合なら何でもよいというわけではなく、最低限の条件を満たしている必要がある。いま、ラムダ項の集合 X が、$M \in X$ かつ $M' \longrightarrow_\beta M \in X$ ならば $M' \in X$ を満たすとき、X は β **拡張について閉じている** (closed under beta expansion) ということにする。β 拡張について閉じている X 全体の集合を $\mathcal{E}_0$ とする。

さて、型を $\mathcal{E}_0$ の要素により解釈するのが目的であるが、そのためにまずは型変数 $\alpha, \beta, \ldots$ に $\mathcal{E}_0$ の要素 $X, Y, \ldots$ を適当に割り当てる。

その上で、含意記号 $\Rightarrow$ の解釈は次のようにする。$X, Y \in \mathcal{E}_0$ のとき

$$X \Rightarrow Y := \{M \mid \forall N \in X.\, MN \in Y\}.$$

$M \in X$ を「M は X の証明である」と読むことにすれば、上の定義は次のように読める。「$X \Rightarrow Y$ の証明とは、X の証明が与えられたときにそれを Y の証明へと変換するラムダ項である。」“構成法” = ラムダ項とすれば、これはまさに含意の BHK 解釈に他ならない。

最後に量化子 $\forall$ について考える。$A(\alpha)$ を型とすると、$A(\alpha)$ は $\mathcal{E}_0$ の何らかの要素により解釈されるわけであるが、具体的にどのような要素となるかは α（およびその他の型変数）にどのような $\mathcal{E}_0$ の要素を割り当てるかに依存する。すなわち $A(\alpha)$ は関数 $\Phi : \mathcal{E}_0 \longrightarrow \mathcal{E}_0$ を表すものと思ってよい。このとき $\forall$ の解釈を次のように与える。

$$\forall\alpha.\Phi(\alpha) := \bigcap_{X \in \mathcal{E}_0} \Phi(X).$$

念のため次のことを確認しておこう。

補題 6.1

1. $X, Y \in \mathcal{E}_0$ ならば $X \Rightarrow Y \in \mathcal{E}_0$.
2. $\Phi : \mathcal{E}_0 \longrightarrow \mathcal{E}_0$ ならば $\bigcap_{X \in \mathcal{E}_0} \Phi(X) \in \mathcal{E}_0$.

証明 1. $M \in X \Rightarrow Y$、$M' \longrightarrow_\beta M$ とする。$M' \in X \Rightarrow Y$ を示すために、$N \in X$ を任意にとる。すると $MN \in Y$、$M'N \longrightarrow_\beta MN$ であるから、$M'N \in Y$ である。ゆえに $X \Rightarrow Y$ は β 拡張について閉じている。

2. は自明である。 ■

以上により、自由変数を含まない型 A には $\mathcal{E}_0$ の要素 $[\![A]\!]$ を一意に割り当てることができる。たとえば

$$[\![\mathbf{Bool}]\!] = \bigcap_{X \in \mathcal{E}_0} X \Rightarrow X \Rightarrow X, \qquad [\![\mathbf{Nat}]\!] = \bigcap_{X \in \mathcal{E}_0} (X \Rightarrow X) \Rightarrow (X \Rightarrow X)$$

となる。A が自由変数を含む場合、たとえば $A \equiv \alpha \Rightarrow \beta \Rightarrow \alpha$ の場合には、A の解釈は α と β の解釈に依存する。一般に A が自由変数 $\alpha, \beta_1, \ldots, \beta_n$ を含むとする。いま、α に注目して $A \equiv A(\alpha)$ と考えるときには、$\beta_1, \ldots, \beta_n$ の解釈は固定し、α の解釈だけ $\mathcal{E}_0$ 上を動かす。すると $A \equiv A(\alpha)$ には関数

$$[\![A]\!] : \mathcal{E}_0 \longrightarrow \mathcal{E}_0$$

を自然に対応させることができる。これは α の解釈に依存して $A(\alpha)$ の解釈を返す関数である。

以上をまとめると、自由型変数に $\mathcal{E}_0$ の要素を適当に割り当てれば、次のようにして型の解釈が定まる。

$$[\![A \Rightarrow B]\!] := [\![A]\!] \Rightarrow [\![B]\!], \qquad [\![\forall\alpha.A(\alpha)]\!] := \bigcap_{X \in \mathcal{E}_0} [\![A]\!](X).$$

ただし右辺の $\Rightarrow$ は単なる記号ではなく、上で定義した集合論的操作を表すので混同しないようにしてほしい。

古典論理のときの意味論の健全性に相当するのは次の定理である。これはしばしば論理的関係の**基本補題** (basic lemma) と呼ばれる重要な性質である。

定理 6.2 $\Gamma = \{x_1 : A_1, \ldots, x_n : A_n\}$ について $\Gamma \vdash_{\mathbf{F}} M_0[x_1, \ldots, x_n] : A_0$ が成り立つとする。自由型変数に適当な $\mathcal{E}_0$ の要素を割り当てることにより $[\![A_0]\!], \ldots, [\![A_n]\!] \in \mathcal{E}_0$ が定められるが、これについて次のことが成り立つ。

$$K_1 \in [\![A_1]\!], \ldots, K_n \in [\![A_n]\!] \quad \Longrightarrow \quad M_0[K_1, \ldots, K_n] \in [\![A_0]\!].$$

とくに $\vdash_{\mathbf{F}} M_0 : A_0$ ならば $M_0 \in [\![A_0]\!]$ が成り立つ。

証明 証明図の構成に関する帰納法による。

1. 証明図が

$$\frac{}{\Gamma \vdash x_i : A_i}\ (init)$$

の形のとき。$x_i[K_1, \ldots, K_n] \equiv K_i$ であるから、この場合定理は自明である。

2. 証明図の最後の部分が

$$\frac{\Gamma \vdash M : A \Rightarrow B \quad \Gamma \vdash N : A}{\Gamma \vdash MN : B}\ (\Rightarrow E)$$

の形の形のとき。M, N の中に現れる変数 $x_1, \ldots, x_n$ を明記して

$$MN \equiv (MN)[x_1, \ldots, x_n] \equiv (M[x_1, \ldots, x_n])(N[x_1, \ldots, x_n])$$

と書く。また、$K_1 \in [\![A_1]\!], \ldots, K_n \in [\![A_n]\!]$ とする。帰納法の仮定により $M[K_1, \ldots, K_n] \in [\![A \Rightarrow B]\!] = [\![A]\!] \Rightarrow [\![B]\!]$ かつ $N[K_1, \ldots, K_n] \in [\![A]\!]$ であるから、$\Rightarrow$ の解釈により $(MN)[K_1, \ldots, K_n] \equiv (M[K_1, \ldots, K_n])(N[K_1, \ldots, K_n]) \in [\![B]\!]$ である。

その他の場合でも型環境 Γ の扱いは同様なので、簡単にするために以下では $\Gamma = \emptyset$ と仮定する。

3. 証明図の最後の部分が

$$\frac{y : A \vdash M[y] : B}{\vdash \lambda y.M[y] : A \Rightarrow B}\ (\Rightarrow I)$$

のとき。$\lambda y.M[y] \in [\![A \Rightarrow B]\!]$ を示すために、$K \in [\![A]\!]$ とする。すると帰納法の仮定により $M[K] \in [\![B]\!]$. $[\![B]\!]$ は β 拡張について閉じているから、$(\lambda y.M[y])K \in [\![B]\!]$. よって $\lambda y.M[y] \in [\![A]\!] \Rightarrow [\![B]\!] = [\![A \Rightarrow B]\!]$.

4. 証明図の最後の部分が

$$\frac{\vdash M : \forall \alpha.A(\alpha)}{\vdash M : A(B)}\ (\forall E)$$

の形のとき。帰納法の仮定により $M \in [\![\forall \alpha.A(\alpha)]\!] = \bigcap_{X \in \mathcal{E}_0} [\![A]\!](X)$ であり、補題 6.1 より $[\![B]\!] \in \mathcal{E}_0$ であるから、$M \in [\![A]\!]([\![B]\!]) = [\![A(B)]\!]$ がいえる。

5. 証明図の最後の部分が

$$\frac{\vdash M : A(\alpha)}{\vdash M : \forall \alpha.A(\alpha)}\ (\forall I)$$

の形のとき。$X \in \mathcal{E}_0$ が与えられたとき、割り当て $\alpha \mapsto X$ のもとで帰納法の仮定を用いると $M \in [\![A]\!](X)$. これが任意の $X \in \mathcal{E}_0$ について成り立つから $M \in \bigcap_{X\in\mathcal{E}_0}[\![A]\!](X) = [\![\forall\alpha.A]\!]$. ■

この定理により実に多くのことがわかる。まずは **IL2** の無矛盾性が帰結する。

系 6.3 $\vdash_{\mathbf{IL2}} \bot$ は成り立たない。すなわち $\vdash_{\mathbf{F}} M : \bot$ となるラムダ項は存在しない。

証明 仮に $\vdash_{\mathbf{F}} M : \bot$ とすると $M \in [\![\bot]\!] = \bigcap_{X\in\mathcal{E}_0} X$ となるはずだが、$\emptyset \in \mathcal{E}_0$ であるから $\bigcap_{X\in\mathcal{E}_0} X = \emptyset$ であり、不合理である。 ■

次に命題 5.9 の別証明を与える。

命題 5.9 の証明の続き 1. $\vdash_{\mathbf{F}} M : \mathbf{Bool}$ とすると、定理 6.2 により $M \in [\![\mathbf{Bool}]\!] = \bigcap_{X\in\mathcal{E}_0} X \Rightarrow X \Rightarrow X$ が成り立つ。

z, w を M の中に現れない変数とし、$X := \{N : N \longrightarrow_\beta^* z \text{ または } N \longrightarrow_\beta^* w\}$ とする。すると $X \in \mathcal{E}_0$ であるから、$M \in X \Rightarrow X \Rightarrow X$. また、$z, w \in X$ なので $Mzw \in X$. ゆえに $Mzw \longrightarrow_\beta^* z$ または $Mzw \longrightarrow_\beta^* w$. これを満たす正規形は $M \equiv \lambda xy.x$ または $\lambda xy.y$ に限られる。

2. $\vdash_{\mathbf{F}} M : \mathbf{Nat}$ とすると、定理 6.2 により $M \in [\![\mathbf{Nat}]\!] = \bigcap_{X\in\mathcal{E}_0}(X \Rightarrow X) \Rightarrow (X \Rightarrow X)$ が成り立つ。

g, w を M の中に現れない変数とし、$X := \{N : N \longrightarrow_\beta^* \underbrace{g(g(\cdots(g}_{n}\, w)\cdots)),\ n \in \mathbb{N}\}$ とする。すると $g \in X \Rightarrow X$, $w \in X$ がわかる。よって $Mgw \in X$、つまり $Mgw \longrightarrow_\beta^* \underbrace{g(g(\cdots(g}_{n}\, w)\cdots))$. これを満たす正規形は $M \equiv \mathsf{n}$ か id に限られる。 ■

最後に直観主義論理に特有の性質である選言特性と存在特性を証明する。まず、

$$\mathsf{inj}_1 M := \lambda xy.xM, \qquad \mathsf{inj}_2 M := \lambda xy.yM, \qquad \mathsf{inj} M := \lambda x.xM.$$

と定義する。ただし x, y は M に含まれない新しい変数である。すると

$$\frac{\Gamma \vdash M : A}{\Gamma \vdash \mathsf{inj}_1 M : A \vee B}\ (\vee I) \qquad \frac{\Gamma \vdash M : B}{\Gamma \vdash \mathsf{inj}_2 M : A \vee B}\ (\vee I) \qquad \frac{\Gamma \vdash M : A(B)}{\Gamma \vdash \mathsf{inj} M : \exists\beta.A(\beta)}\ (\exists I)$$

が成り立つことは容易に確かめることができる。ここでさらに Γ が空集合のときにはある意味で逆も成り立つというのが、古典論理にはない直観主義論理の大きな特徴である。

定理 6.4 (選言特性、存在特性)

1. $\vdash_{\mathbf{F}} N : A \vee B$ かつ N は正規形であるとする。このとき $N \equiv \mathsf{inj}_1 M$ かつ $\vdash_{\mathbf{F}} M : A$、または $N \equiv \mathsf{inj}_2 M$ かつ $\vdash_{\mathbf{F}} M : B$ のいずれか一方が成り立つ。

2. $\vdash_{\mathbf{F}} N : \exists\beta.A(\beta)$ かつ N は正規形であるとする。このときある型 B が存在して $N \equiv \mathsf{inj} M$ かつ $\vdash_{\mathbf{F}} M : A(B)$ が成り立つ。

証明 1. A, B に現れる自由変数に適当な $\mathcal{E}_0$ の要素を割り当てることにより、$N \in [\![A \vee B]\!] = \bigcap_{X \in \mathcal{E}_0}([\![A]\!] \Rightarrow X) \Rightarrow ([\![B]\!] \Rightarrow X) \Rightarrow X$ が成り立つ。z, w を新しい自由変数とし、

$$X := \{K : K \longrightarrow_\beta^* zM,\ M \in [\![A]\!]\} \cup \{K : K \longrightarrow_\beta^* wM,\ M \in [\![B]\!]\}$$

と定義すると $X \in \mathcal{E}_0$ であるから、$N \in ([\![A]\!] \Rightarrow X) \Rightarrow ([\![B]\!] \Rightarrow X) \Rightarrow X$ が成り立つ。また $z \in [\![A]\!] \Rightarrow X$, $w \in [\![B]\!] \Rightarrow X$ は明らかに成り立つので、$Nzw \in X$、すなわち $Nzw \longrightarrow_\beta^* zM$ かつ $M \in [\![A]\!]$、または $Nzw \longrightarrow_\beta^* wM$ かつ $M \in [\![B]\!]$ が成り立つ。これを満たす正規形の N は $\mathsf{inj}_1 M$ または $\mathsf{inj}_2 M$ の形に限られる。前者の場合、$\vdash_{\mathbf{F}} N : A \vee B$ の証明図の最後の部分は

$$\dfrac{\dfrac{\dfrac{\dfrac{}{x : A \Rightarrow \alpha, y : B \Rightarrow \alpha \vdash x : A \Rightarrow \alpha}\,(init) \quad \begin{array}{c}\vdots\\ x : A \Rightarrow \alpha, y : B \Rightarrow \alpha \vdash M : A\end{array}}{x : A \Rightarrow \alpha, y : B \Rightarrow \alpha \vdash xM : \alpha}\,(\Rightarrow E)}{x : A \Rightarrow \alpha \vdash \lambda y.xM : (B \Rightarrow \alpha) \Rightarrow \alpha}\,(\Rightarrow I)}{\vdash \lambda xy.xM : (A \Rightarrow \alpha) \Rightarrow (B \Rightarrow \alpha) \Rightarrow \alpha}\,(\Rightarrow I)}{\vdash \lambda xy.xM : \forall\alpha.(A \Rightarrow \alpha) \Rightarrow (B \Rightarrow \alpha) \Rightarrow \alpha}\,(\forall I)$$

の形であるとしてよい。x, y は M に含まれない変数だから、結局 $\vdash_{\mathbf{F}} M : A$ が成り立つ。$N \equiv \mathsf{inj}_2 M$ の場合も同様。

2 の証明も同様である。 ■

上の命題により、排中律 $A \vee \neg A$ や酒場の法則 $\exists\alpha(A(\alpha) \Rightarrow \forall\beta.A(\beta))$ が $\mathbf{IL}2$ において証明できないことが確認できる。実際、仮に $\vdash_{\mathbf{IL}2} \alpha \vee \neg\alpha$ とすると $\vdash_{\mathbf{IL}2} \alpha$ か $\vdash_{\mathbf{IL}2} \neg\alpha$ のどちらかが成り立つはずだが、どちらも成り立ちえないことは明らかである。酒場の法則についても同様に議論ができる。

選言特性と存在特性は直観主義の体系一般について成り立つ性質である。もちろん一階直観主義述語論理 $\mathbf{IL}$ についても成り立つ。そこで存在特性のほうについて考えてみると、これは「$\exists x.\varphi(x)$ を示すときには、$\varphi(t)$ を満たす対象 t の構成法を与えなければならない」という構成主義の思想と合致する。このようにして直観主義論理の証明は構成的であることが厳密に確かめられるのである。

6.2 正規化定理

前節でとりあげた命題5.9の証明を詳しく見てみると、実際にはもっと強い主張が証明されていることがわかる。すなわち、正規形とは限らないラムダ項 M についても、$\vdash_{\mathbf{F}} M : \mathbf{Bool}$ ならば、Mzw は正規化可能であり、その正規形が z か w になることが証明されているのである。このことから M 自体が正規化可能であることを導くのはそれほど難しくない。

正規化可能性は、何も $\mathbf{Bool}$ 型のラムダ項に限ったことではなく、一般にシステム $\mathbf{F}$ で型を持つ全てのラムダ項について成り立つ性質である。このことを示すには、前節の論理的述語の技法をやや拡張する必要がある。以下で述べる技法は、1960 年代にテイトにより考案されジラールによりシステム $\mathbf{F}$ へと拡張された技法（簡約可能性候補などと呼ばれる）の簡易版である。

まず、次の二つの集合を定義する。

$$\begin{aligned}\mathbf{WN} &:= \{M \mid M \text{ は正規化可能 }\}\\ \mathbf{HN} &:= \{xM_1 \ldots M_n \mid M_1, \ldots, M_n \in \mathbf{WN},\ n \in \mathbf{N}\}\end{aligned}$$

とくに **HN** は全ての変数を含む（$n = 0$ とすればよい）。

前節では、型を $\mathcal{E}_0$ の任意の要素、すなわち β 拡張についている任意の集合として解釈した。以下ではこれを相対化し、$\mathcal{E}_0$ の部分集合 $\mathcal{E}_1$ を考える。

$$\mathcal{E}_1 := \{X \in \mathcal{E}_0 \mid \mathbf{HN} \subseteq X \subseteq \mathbf{WN}\}.$$

すなわち、以下では型を (i)β 拡張について閉じており、(ii)**HN** を含み、(iii)**WN** に含まれる集合により解釈するのである。なお、$\mathbf{HN} \subseteq \mathbf{WN}$ が成り立ち、**WN** は β 拡張について閉じている。ゆえに $\mathbf{WN} \in \mathcal{E}_1$ であり、$\mathcal{E}_1$ は空ではない。

当然 $\forall$ の定義も $\mathcal{E}_1$ に相対化する必要がある。

$$\forall\alpha.\Phi(\alpha) := \bigcap_{X \in \mathcal{E}_1} \Phi(X).$$

次に、この相対化によって補題 6.1 が保たれることを示す。

補題 6.5

1. $X, Y \in \mathcal{E}_1$ ならば $X \Rightarrow Y \in \mathcal{E}_1$.
2. $\Phi : \mathcal{E}_0 \longrightarrow \mathcal{E}_1$ ならば $\bigcap_{X \in \mathcal{E}_1} \Phi(X) \in \mathcal{E}_1$.

証明　$X \Rightarrow Y$, $\forall\alpha.\Phi(\alpha)$ が β 拡張について閉じていることは補題 6.1 から言えるので、$X \Rightarrow Y$ と $\forall\alpha.\Phi(\alpha)$ が **HN** を含み **WN** に含まれることを示せばよい。

(a) $\mathbf{HN} \subseteq X \Rightarrow Y$. これを示すために、$xM_1 \cdots M_n \in \mathbf{HN}$ とする。つまり $M_1, \ldots, M_n \in \mathbf{WN}$. $N \in X$ とすると仮定より $N \in \mathbf{WN}$ であるから $xM_1 \cdots M_n N \in \mathbf{HN}$. ふたたび仮定より $xM_1 \cdots M_n N \in Y$. 以上から $xM_1 \cdots M_n \in X \Rightarrow Y$ が成り立つ。

(b) $X \Rightarrow Y \subseteq \mathbf{WN}$. これを示すために $M \in X \Rightarrow Y$ とする。x を M に含まれない新しい変数とすると、仮定より $x \in \mathbf{HN} \subseteq X$ となり、ふたたび仮定により $Mx \in Y \subseteq \mathbf{WN}$ である。すなわち Mx は正規化可能である。ここで M 自身も正規化可能であるといいたいのだが、それには次のように考えればいい。Mx から出発して正規形に到達する β 簡約列を考えると、x のポジションはずっと変わらずに正規形 Kx まで到達するか、あるいはある時点で $(\lambda y.N[y])x$ の形に到達し、次のステップで $N[x]$ に β 簡約されるかのどちらかである。前者の場合は M から出発して正規形 K に至る β 簡約列を簡単に構成することができる。

$$\begin{array}{llllll} Mx & \longrightarrow_\beta & M'x & \longrightarrow_\beta & \cdots & \longrightarrow_\beta \quad Kx \\ M & \longrightarrow_\beta & M' & \longrightarrow_\beta & \cdots & \longrightarrow_\beta \quad K. \end{array}$$

後者の場合、β 簡約列は次の図の上側のようになっているはずであり、これを分析することにより M から出発する下側の β 簡約列を構成することができる。

$$\begin{array}{lllllllllllll} Mx & \longrightarrow_\beta & M'x & \longrightarrow_\beta & \cdots & \longrightarrow_\beta & (\lambda y.N[y])x & \longrightarrow_\beta & N[x] & \longrightarrow_\beta & N'[x] & \cdots & \longrightarrow_\beta & K[x]. \\ M & \longrightarrow_\beta & M' & \longrightarrow_\beta & \cdots & \longrightarrow_\beta & \lambda y.N[y] & & & \longrightarrow_\beta & \lambda y.N'[y] & \cdots & \longrightarrow_\beta & \lambda y.K[y]. \end{array}$$

これで $M \in \mathbf{WN}$ が言えた。

(c) $\mathbf{HN} \subseteq \bigcap_{X\in\mathcal{E}_1} \Phi(X)$ は自明である。

(d) $\bigcap_{X\in\mathcal{E}_1} \Phi(X) \subseteq \mathbf{WN}$. これを示すには、$\mathbf{WN} \in \mathcal{E}_1$ に注意して、$\bigcap_{X\in\mathcal{E}_1} \Phi(X) \subseteq \Phi(\mathbf{WN}) \subseteq \mathbf{WN}$ とすればよい。 ■

補題 6.1 の相対化版がいえたので、定理 6.2 の相対化版もいえる。証明は前と全く同じである。

定理 6.6 $\Gamma = \{x_1 : A_1, \ldots, x_n : A_n\}$ について $\Gamma \vdash_{\mathbf{F}} M_0[x_1, \ldots, x_n] : A_0$ が成り立つとする。自由型変数に適当な $\mathcal{E}_1$ の要素を割り当てることにより $[\![A_0]\!], \ldots, [\![A_n]\!] \in \mathcal{E}_1$ が定められるが、これについて次のことが成り立つ。

$$K_1 \in [\![A_1]\!], \ldots, K_n \in [\![A_n]\!] \quad \Longrightarrow \quad M_0[K_1, \ldots, K_n] \in [\![A_0]\!].$$

とくに $\vdash_{\mathbf{F}} M_0 : A_0$ ならば $M_0 \in [\![A_0]\!]$ が成り立つ。

これで正規化定理が証明できた。

定理 6.7 (正規化定理) M がシステム $\mathbf{F}$ で型を持つならば、M は正規化可能である。

証明 $x_1 : A_1, \ldots, x_n : A_n \vdash_{\mathbf{F}} M : A_0$ とする。自由型変数に適当な $\mathcal{E}_1$ の要素を割り当てることにより $[\![A_0]\!], \ldots, [\![A_n]\!] \in \mathcal{E}_1$ となり、また $\mathcal{E}_1$ の定義より $x_1 \in [\![A_1]\!], \ldots, x_n \in [\![A_n]\!]$ である。定理 6.6 より $M \in [\![A_0]\!]$. $[\![A_0]\!] \in \mathcal{E}_1$ であるから、とくに $M \in \mathbf{WN}$ である。 ■

上の定理は**弱正規化定理** (weak normalization theorem) と呼ばれることも多い。なぜならば M から出発して正規形に到達する β 簡約列が**存在**することしか主張していないからである。これに対して**強正規化定理** (strong normalization theorem) は、M から出発する**全ての** β 簡約列が正規形に到達することを主張する。実をいうとシステム $\mathbf{F}$ についてはこの強い意味での正規化定理が成り立つ。

定理 6.8 (強正規化定理) M がシステム $\mathbf{F}$ で型を持つならば、M から出発して無限に続く β 簡約列

$$M \longrightarrow_\beta M' \longrightarrow_\beta M'' \longrightarrow_\beta \cdots$$

は存在しない。どんな仕方で β 簡約を行っても、必ず有限回で正規形に到達する。

証明の方針は弱正規化定理の場合と同様であるが、もう少し複雑になるのでここでは割愛する。

強正規化定理により不動点 $\mathsf{fix}M$ はシステム $\mathbf{F}$ で型を持たないことがわかる。システム $\mathbf{F}$ における再帰呼び出しには制限があり、停止するかどうかわからないようなプログラムは絶対に書けないのである。

7　算術とラムダ計算

第 5 章では、論理と計算の密接な対応関係（カリー・ハワード対応）について説明した。本章ではこの対応関係を論理から算術へと拡大し、直観主義算術の体系（ハイティング算術）がラムダ計算と密接に対応することを述べる。主要な結果は、「算術の証明が与えられたらそこからプログラムを抽出することができる」ことを主張するプログラム抽出定理である。この定理を証明するために、前章で説明した論理的述語のアイデアを再び用いる。

7.1　古典論理と直観主義論理

まずは準備として、一階古典述語論理 **CL** と一階直観主義述語論理 **IL** の関係を整理しておく。以下考える論理式は **CL** や **IL** のものであり、**IL2** のものではないことに注意。

補題 7.1 次の論理式は古典論理 **CL** で証明可能である。

1. $\varphi \vee \psi \leftrightarrow \neg(\neg\varphi \wedge \neg\psi)$
2. $\exists x.\varphi \leftrightarrow \neg\forall x.\neg\varphi$

この結果によれば、$\wedge, \to, \bot, \forall$ があれば $\vee, \exists$ は定義可能なことがわかる。そこで次の論理式のクラスを定義する。

定義 7.2 負の論理式 *(negative formula)* を以下のように定義する。

$$\varphi, \psi ::= \neg p(t_1, \ldots, t_n) \mid \top \mid \bot \mid \varphi \wedge \psi \mid \varphi \to \psi \mid \forall x.\varphi.$$

すなわち、原子論理式の否定と $\top, \bot$ から始めて、$\wedge, \to, \forall$ を用いて構成できる論理式が負の論理式である。

論理式 φ が与えられたとき、

$$p(t_1, \ldots, t_n) \mapsto \neg\neg p(t_1, \ldots, t_n), \qquad \psi \vee \xi \mapsto \neg(\neg\psi \wedge \neg\xi), \qquad \exists x.\psi \mapsto \neg(\forall x.\neg\psi)$$

という置き換えをほどこして得られる負の論理式を φ^N と書く。上の補題から次のことが言える。

補題 7.3 φ を論理式とするとき、

$$\vdash_{\mathbf{CL}} \varphi \leftrightarrow \varphi^N$$

が成り立つ。

負の論理式の特徴は、直観主義論理においても自分自身の二重否定と等しいという点にある。すなわち次のことが言える。

補題 7.4 φ を負の論理式とするとき、

$$\vdash_{\mathbf{IL}} \neg\neg\varphi \to \varphi$$

が成り立つ。

証明 φ の構成についての帰納法による。 ■

さて、古典論理と直観主義論理の唯一の違いは背理法、すなわち (abs) 規則が使えるかどうかである。しかし上の補題により、負の論理式については (abs) 規則を使わなくても同じことが推論できる：

$$\frac{\Gamma, \neg\varphi \vdash \bot}{\Gamma \vdash \varphi}\,(abs) \quad \Longrightarrow \quad \frac{\dfrac{}{\Gamma \vdash \neg\neg\varphi \to \varphi} \quad \dfrac{\Gamma, \neg\varphi \vdash \bot}{\Gamma \vdash \neg\neg\varphi}\,(\to I)}{\Gamma \vdash \varphi}\,(\to E)$$

ゆえに負の論理式に制限すれば、古典論理と直観主義論理の区別は消滅してしまう。それゆえ次の定理が成り立つ。

定理 7.5 任意の φ について

$$\vdash_{\mathbf{CL}} \varphi \quad \Longleftrightarrow \quad \vdash_{\mathbf{CL}} \varphi^N \quad \Longleftrightarrow \quad \vdash_{\mathbf{IL}} \varphi^N.$$

上の定理によれば、

- 直観主義者は、古典主義者の言っていることを完璧に理解することができる。実際、古典主義者が「$\varphi \vee \psi$ だ」と口にするたびに、直観主義者は「ああ、奴は $\neg(\neg\varphi \wedge \neg\psi)$ と言っているんだな」と脳内変換してあげればよい。$\varphi \mapsto \varphi^N$ という脳内変換のもとで直観主義者は古典主義者と対等の証明能力を持つ。

- 直観主義者は「$\varphi \vee \psi$」と「$\neg(\neg\varphi \wedge \neg\psi)$」ということと完璧に区別して話すが、古典主義者にはその区別ができない。イギリス人のブラックジョークを日本人がなかなか理解できないのと同じように、直観主義者の微妙な言い回しを古典主義者は必ずしも理解できないのである。

- もちろん、そのような繊細な区別をすることが数学においてどの程度有用かは別問題である。ブラックジョーク同様、単なる自己満足にすぎないのかもしれない。（いやそうではない、というのが構成主義の考え方である。）

ところで、負の論理式についてはもう一つ大事な性質がある。

補題 7.6 φ を負の論理式とするとき、$\bot \to \varphi$ は $(\bot E)$ 規則を用いずに **IL** で証明可能である。

証明 φ の構成に関する帰納法による。φ が $\neg p(\overline{t})$ の形のときには、証明すべき論理式は $\bot \to (p(\overline{t}) \to \bot)$ であるが、これは $(init)$ 規則と $(\to I)$ 規則を用いれば簡単に証明できる。その他の場合も帰納法の仮定から明らかである。 ■

この補題により負の論理式についての $(\bot E)$ 規則は他の規則を用いてシミュレートできることがわかる。

$$\frac{\Gamma \vdash \bot}{\Gamma \vdash \varphi}\,(\bot E) \qquad \Longrightarrow \qquad \frac{\overline{\Gamma \vdash \bot \to \varphi} \quad \Gamma \vdash \bot}{\Gamma \vdash \varphi}\,(\to E)$$

さて、$\bot$ と他のあまたの論理式の違いはまさに $(\bot E)$ 規則が使えるかどうかという点に尽きる。$(\bot E)$ 規則が必要ないということは、別に $\bot$ でなくても、他のどんな論理式を用いてもこれまでの議論は成立するということである。そこで ξ を任意の論理式とするとき、φ^N の中の $\bot$ をすべて ξ で置き換えて得られる論理式を φ^ξ と書くことにする。すると次のことが成り立つ。

系 7.7 任意の論理式 φ, ξ について次のことが成り立つ。

$$\vdash_{\mathbf{CL}} \varphi \qquad \Longrightarrow \qquad \vdash_{\mathbf{IL}} \varphi^\xi$$

このアイデアは次の節で重要な役割を果たす。

7.2 ハイティング算術

ペアノ算術 **PA** は一階古典述語論理 **CL** をベースにしていた。ここでベース論理を直観主義論理 **IL** に変えてできる体系を**ハイティング算術** (Heyting arithmetic)**HA** という。別の言い方をすれば、**PA** から (abs) 規則を取り除いたものが **HA** である。もっと大雑把にいえば “直観主義中学生”、それが **HA** である。**HA** が証明できる文はもちろん **PA** も証明できる。一方、簡単な文については逆も成り立つ。

定理 7.8 φ を限定論理式とするとき、**HA** は次のことを証明できる。

1. $\varphi \vee \neg\varphi$
2. $\neg\neg\varphi \leftrightarrow \varphi$

証明 1. φ の構成に関する帰納法による。その際 **HA** が数学的帰納法の公理もつことが本質的である。

2. 一般に $\varphi \vee \psi, \neg\psi \vdash_{\mathbf{IL}} \varphi$ が成り立つ。そこで $\psi \equiv \neg\varphi$ とおいて 1 と組み合わせればよい。 ■

よって **HA** は限定論理式についての背理法が使えるので、こと限定論理式に関する限り、**HA** と **PA** は何も変わらないことがわかる。特に定理 3.19（Σ_1 完全性）は **HA** でも成り立つ。実際には、**HA** と **PA** の間にはもっと強い対応関係がある。

定理 7.9 (Π_2 保存拡大) φ を Π_2 文とすると、

$$\vdash_{\mathbf{PA}} \varphi \quad \Longleftrightarrow \quad \vdash_{\mathbf{HA}} \varphi.$$

このことは、結論とする文が Π_2 文という比較的単純な文である限り、たとえ証明の中で背理法を使用したとしても、それらは原理上取り除くことができ、構成的な証明を復元できることを表している。

証明　**PA** においても **HA** においても $\forall x.\varphi(x)$ が証明できることと $\varphi(x)$ が証明できることは一致するから、Σ_1 論理式 $\exists y.\psi(y)$ （ただし $\psi(y)$ は限定論理式）について定理を証明すればよい。($\exists y.\psi(y)$ は文であるとは限らないから、ここで Σ_1 完全性定理を使うことはできない。)

前節で論理式 $\varphi \mapsto \varphi^\xi$ という翻訳を与えた。これを少し変えて、次のような翻訳を考える。まず、φ が限定論理式のときには $\varphi^\xi := \varphi$ とする（前の定理により、この場合は翻訳の必要がない)。そうでないときには、前節と同じ翻訳をする。

$$\begin{aligned}
(\varphi_1 \wedge \varphi_2)^\xi &:= \varphi_1^\xi \wedge \varphi_2^\xi \\
(\varphi_1 \to \varphi_2)^\xi &:= \varphi_1^\xi \to \varphi_2^\xi \\
(\varphi_1 \vee \varphi_2)^\xi &:= (\varphi_1^\xi \to \xi \wedge \varphi_2^\xi \to \xi) \to \xi \\
(\forall x.\varphi_0)^\xi &:= \forall x.\varphi_0^\xi \\
(\exists x.\varphi_0)^\xi &:= (\forall x.(\varphi_0^\xi \to \xi)) \to \xi
\end{aligned}$$

この翻訳のもとでも系 7.7 と同様のことが成り立つ。すなわち

$$\vdash_{\mathbf{PA}} \varphi \qquad \Longrightarrow \qquad \vdash_{\mathbf{HA}} \varphi^\xi.$$

(**PA** は公理を含むが、公理も適切に扱うことができるので問題ない。)

ここで $\varphi \equiv \exists y.\psi(y)$ とすると、$\vdash_{\mathbf{HA}} (\forall y.(\psi(y) \to \xi)) \to \xi$ が成り立つが、これは $\vdash_{\mathbf{HA}} ((\exists y.\psi(y)) \to \xi) \to \xi$ と同値である。ξ は任意の論理式でよかったから $\xi := \exists y.\psi(y)$ とすれば

$$\vdash_{\mathbf{HA}} (\exists y.\psi(y) \to \exists y.\psi(y)) \to \exists y.\psi(y).$$

ゆえに $\vdash_{\mathbf{HA}} \exists y.\psi(y)$ が成り立つ。 ■

上の証明で与えた巧妙な翻訳法はフリードマンの A **翻訳** (A translation) として広く知られている。

さて、以上で Π_2 文に関する限り **PA** は **HA** に還元できることがわかったので、これからはもっぱら **HA** について論じていく。話を簡単にするため、以下では $\top$, $\bot$, $\vee$ を除いて考える。これらの論理記号は次のように定義できる。

$$\begin{aligned}
\top &:= \mathsf{0} = \mathsf{0} \\
\bot &:= \mathsf{0} = \mathsf{1} \\
\varphi \vee \psi &:= \exists x \le \mathsf{1}.(x = \mathsf{0} \to \varphi) \wedge (x = \mathsf{1} \to \psi)
\end{aligned}$$

補題 7.10 **HA** の論理式 φ に対して上のような置き換えを施して得られる論理式を φ' とする。また **HA** の公理 φ を φ' で置き換え、記号 $\top$, $\bot$, $\vee$（およびそれらに関する推論規則）はないものとして考え、さらに特別な推論規則

$$\frac{\Gamma \vdash \mathsf{0} = \mathsf{1}}{\Gamma \vdash \psi}\ (\mathsf{0} = \mathsf{1}E)$$

を加えて得られる体系を $\mathbf{HA}'$ とする。このとき、次のことが成り立つ。

$$\vdash_{\mathbf{HA}} \varphi \qquad \Longleftrightarrow \qquad \vdash_{\mathbf{HA}'} \varphi'.$$

以下では $\mathbf{HA}$ とは $\mathbf{HA}'$ のことであると改めて定義しなおして話を進める。さらにもう一つ変更をする。

φ を算術の文とするとき、$\mathbf{HA}$ が φ を証明できるとは、元の定義によれば公理 $\psi_1, \ldots, \psi_n$ が存在して $\psi_1, \ldots, \psi_n \vdash_{\mathbf{IL}} \varphi$ が成り立つことであった。しかし同じことは推論規則

$$\frac{}{\Gamma \vdash \varphi}\ (axiom) \quad \varphi \text{ は } \mathbf{HA} \text{ の公理}$$

を $\mathbf{IL}$ に追加することによっても実現できる。ゆえに、以下では上の規則を $\mathbf{IL}$ に追加した体系を $\mathbf{HA}$ と呼び、この新しい体系で $\vdash \varphi$ が導出できるとき、$\vdash_{\mathbf{HA}} \varphi$ と書くことにする。以下の事柄に注意しておく。

- $\mathbf{Q}$ の公理は 3.8 節で挙げたものであるが、正式にはその全称閉包（自由変数を $\forall$ で束縛すること）をとる必要があった。しかしこの新しい立場によれば全称閉包をとる必要はない。
- $\bot$ は $\mathsf{0} = \mathsf{1}$ で置き換えることにしたので、公理 $S(x) \neq \mathsf{0}$ は $S(x) = \mathsf{0} \to \mathsf{0} = \mathsf{1}$ となる。
- 公理 $x \neq 0 \to \exists y(x = S(y))$ は、数学的帰納法を用いることにより他の公理から導出できる。よって省略してもよい。
- 3.8 節では等号に関する推論規則も挙げたが、これらは次の公理と同等である。

  $$x = x \qquad x = z \to y = z \to x = y \qquad x = y \to S(x) = S(y)$$
  $$x = y \to z = w \to x + z = y + w \quad x = y \to z = w \to x \cdot z = y \cdot w$$

  以下ではこの公理の形で取り扱うことにする。
- 以下では $\nvdash_{\mathbf{HA}} \mathsf{0} = \mathsf{1}$ を仮定する。このことは集合論を用いれば簡単に証明できる。

7.3 実現可能性解釈

第 6 章に引き続き、本節では算術の論理式の “意味” を考える。基本的なアイデアは論理的述語とおなじであるが、算術の文脈では**実現可能性解釈** (realizability interpretation) という用語がよく用いられるので、それに従うことにする。実際には以下で展開する議論は**真理概念をともなう実現可能性解釈・改** (modified realizability interpretation with truth) と呼ばれるものに近い。

ここでの目標は、算術の論理式をラムダ項の集合により解釈することである。そのための準備として算術の項 t に対してラムダ項 M_t を割り当てる。算術の演算 $0, S, +, \cdot$ はすべてラムダ項で表現できるから、この割り当ては自然にできる。たとえば $t \equiv x + (S(0) \cdot y)$ のとき、$M_t := \mathsf{add}x(\mathsf{mult}(\mathsf{suc}\ \mathsf{0})y)$ である。

次に、$\mathbf{IL}$ の推論規則に従って証明図にラムダ項を割り当てていく仕方を定める。まず、以下では二種類の変数を用いることを断っておく。

- ラムダ項の中でのみ用いられる変数 $a, b, c, \ldots$
- 算術の項・ラムダ項両方で用いられる変数 $x, y, z, \ldots$

後者はラムダ項の中でも用いられるが、その際には常に何らかの自然数を表すのに用いられる。以下では Γ は $\{a_1 : \varphi_1, \cdots, a_n : \varphi_n\}$ の形の型環境を表すが、ここで $a_1, \ldots, a_n$ はラムダ項の中でのみ用いられる変数であるから、$\varphi_1, \cdots, \varphi_n$ の中には現れないものする。

$$\frac{\Gamma \vdash M : \mathsf{0} = \mathsf{1}}{\Gamma \vdash N : \varphi}\ (\mathsf{0} = \mathsf{1}E) \qquad \frac{}{\Gamma \vdash a : \varphi}\ (init) \quad \text{ただし } a : \varphi \in \Gamma$$

$$\frac{\Gamma \vdash M : \varphi \quad \Gamma \vdash N : \psi}{\Gamma \vdash \langle M, N \rangle : \varphi \wedge \psi}\ (\wedge I) \quad \frac{\Gamma \vdash M : \varphi \wedge \psi}{\Gamma \vdash \mathsf{proj}_1 M : \varphi}\ (\wedge E) \qquad \frac{\Gamma \vdash M : \varphi \wedge \psi}{\Gamma \vdash \mathsf{proj}_2 M : \psi}\ (\wedge E)$$

$$\frac{\Gamma, a : \varphi \vdash M : \psi}{\Gamma \vdash \lambda a.M : \varphi \to \psi}\ (\to I) \qquad \frac{\Gamma \vdash M : \varphi \to \psi \quad \Gamma \vdash N : \varphi}{\Gamma \vdash MN : \psi}\ (\to E)$$

$$\frac{\Gamma \vdash M : \varphi(x)}{\Gamma \vdash \lambda x.M : \forall x.\varphi(x)}\ (\forall I) \qquad \frac{\Gamma \vdash M : \forall x.\varphi(x)}{\Gamma \vdash M M_t : \varphi(t)}\ (\forall E)$$

$$\frac{\Gamma \vdash M : \varphi(t)}{\Gamma \vdash \langle M_t, M \rangle : \exists x.\varphi(x)}\ (\exists I) \quad \frac{\Gamma \vdash M : \exists x.\varphi(x) \quad \Gamma, a : \varphi(x) \vdash N[x, a] : \xi}{\Gamma \vdash N[\mathsf{proj}_1 M, \mathsf{proj}_2 M] : \xi}\ (\exists E)$$

ここで $(\mathsf{0} = \mathsf{1}E)$ 規則の中の N はどんなラムダ項でもよい。ただし後の定理 7.20 を成り立たせるためには、$N \in \varphi^{\mathbf{T}}$ となるようにとる必要がある（後述）。

最後に $(axiom)$ 規則にラムダ項を割り当てる。

$$\frac{}{\Gamma \vdash M : \varphi}\ (axiom) \quad \varphi \text{ は } \mathbf{HA} \text{ の公理}$$

ここで M は公理 φ により異なり、次の表により定められる。

φ	M
$x = x$	id
$x = z \to y = z \to x = y$	$\lambda ab.\mathsf{id}$
$x = y \to S(x) = S(y)$	$\lambda a.\mathsf{id}$
$x = y \to z = w \to x + z = y + w$	$\lambda ab.\mathsf{id}$
$x = y \to z = w \to x \cdot z = y \cdot w$	$\lambda ab.\mathsf{id}$
$S(x) = 0 \to \mathsf{0} = \mathsf{1}$	$\lambda a.\mathsf{id}$
$S(x) = S(y) \to x = y$	$\lambda a.\mathsf{id}$
$x + 0 = x$	id
$x + S(y) = S(x + y)$	id
$x \cdot 0 = 0$	id
$x \cdot S(y) = x \cdot y + x$	id
$(\forall x.\psi(x) \to \psi(S(x))) \to \psi(0) \to \forall x.\psi(x)$	rec

ここで数学的帰納法の公理が原始再帰法 rec により解釈されているところが本質的である。以上の割り当て方により $\Gamma \vdash M : \varphi$ が導出できるとき、$\Gamma \vdash_{\mathbf{HA}} M : \varphi$ と書く。

次に算術の文に対して実現可能性解釈を与える。

定義 7.11 (実現可能性解釈) 算術の文 φ に対してラムダ項の集合 $[\![\varphi]\!]$ を次のように割り当てる。まず、$\vdash_{\mathbf{HA}} \varphi$ が成り立たないときには、$[\![\varphi]\!] := \emptyset$ とする（$\not\vdash_{\mathbf{HA}} \mathsf{0} = \mathsf{1}$ を仮定しているので、特に $[\![\mathsf{0} = \mathsf{1}]\!] = \emptyset$ である）。さもなければ、次のようにする。

$$\begin{aligned}
[\![t = u]\!] &:= \{M \mid M \text{ はラムダ項 }\} \\
[\![\varphi \wedge \psi]\!] &:= \{M \mid M =_\beta \langle N_1, N_2\rangle \text{ かつ } N_1 \in [\![\varphi]\!],\ N_2 \in [\![\psi]\!]\} \\
[\![\varphi \to \psi]\!] &:= \{M \mid \forall N \in [\![\varphi]\!].\ MN \in [\![\psi]\!]\} \\
[\![\forall x.\varphi(x)]\!] &:= \{M \mid \forall n \in \mathbb{N}.\ M\mathsf{n} \in [\![\varphi(\mathsf{n})]\!]\} \\
[\![\exists x.\varphi(x)]\!] &:= \{M \mid M =_\beta \langle \mathsf{n}, N\rangle \text{ かつ } N \in [\![\varphi(\mathsf{n})]\!]\}
\end{aligned}$$

ここで、$\to$ の定義は論理的述語の場合と同じである。また $\forall$ と $\exists$ の定義は BHK 解釈に由来する。

- $\forall x.\varphi(x)$ の証明とは、自然数 n が与えられたときに $\varphi(n)$ の証明を創り出す構成法である。
- $\exists x.\varphi(x)$ の証明とは、自然数 n と $\varphi(n)$ の証明の組である。このような n は、しばしば存在文 $\exists x.\varphi(x)$ の**証拠** (witness) と呼ばれる。

論理的述語の場合と同様、各 $[\![\varphi]\!]$ は β 拡張について閉じてさえいればよいのだが、上ではさらに $=_\beta$ について閉じているように定義してある。

補題 7.12 任意の算術の文 φ について、$[\![\varphi]\!]$ は $=_\beta$ について閉じている。

証明　φ の複雑さに関する帰納法による。$\not\vdash_{\mathbf{HA}} \varphi$ のときには $[\![\varphi]\!] = \emptyset$ だから明らか。また、そうでなくとも $[\![t = u]\!]$, $[\![\varphi \wedge \psi]\!]$ と $[\![\exists x.\varphi(x)]\!]$ が $=_\beta$ について閉じていることは定義から明らか。残るのは $[\![\varphi \to \psi]\!]$ と $[\![\forall x.\varphi(x)]\!]$ であるが、どちらの場合も証明は同様なので、後者の場合のみ示す。

$M \in [\![\forall x.\varphi(x)]\!]$、$M' =_\beta M$ とすると、任意の $n \in \mathbb{N}$ について $M\mathsf{n} \in [\![\varphi(\mathsf{n})]\!]$ である。帰納法の仮定により $[\![\varphi(\mathsf{n})]\!]$ は $=_\beta$ について閉じているので $M'\mathsf{n} \in [\![\varphi(\mathsf{n})]\!]$. これが任意の $n \in \mathbb{N}$ について成り立つから、結局 $M' \in [\![\forall x.\varphi(x)]\!]$ が言える。 ■

補題 7.13 t を変数を含まない算術の項とし、その値を n とする。つまり $\mathbb{N} \models t = \mathsf{n}$. このとき任意の一変数論理式 $\varphi(x)$ について

$$[\![\varphi(t)]\!] = [\![\varphi(\mathsf{n})]\!].$$

証明　$\varphi(x)$ の複雑さに関する帰納法による。$\varphi(x)$ が $u(x) = v(x)$ の形のとき、$\mathbf{N} \models u(t) = v(t)$ となるのは $\mathbf{N} \models u(\mathsf{n}) = v(\mathsf{n})$ となるときに限る。よって $[\![\varphi(t)]\!] = [\![\varphi(\mathsf{n})]\!]$ が成り立つ。その他の場合は明らか。 ■

定理 7.14 $\Gamma = \{a_1 : \varphi_1(\overline{x}), \ldots, a_k : \varphi_k(\overline{x})\}$ のもとで $\Gamma \vdash_{\mathbf{HA}} M[\overline{a}, \overline{x}] : \psi(\overline{x})$ が成り立つとする。ただし $\overline{a}$, $\overline{x}$ はそれぞれ変数の列 $a_1, \ldots, a_k$ および $x_1, \ldots, x_l$ を表す。このとき、任意の自然数の列 $\overline{n} \equiv n_1, \ldots, n_l$ について

$$K_1 \in [\![\varphi_1(\overline{\mathsf{n}})]\!], \ldots, K_k \in [\![\varphi_k(\overline{\mathsf{n}})]\!] \quad \Longrightarrow \quad M[\overline{K}, \overline{\mathsf{n}}] : \psi(\overline{\mathsf{n}})$$

が成り立つ。ただし $\overline{K} \equiv K_1, \ldots, K_k$. とくに ψ を文として $\vdash_{\mathbf{HA}} M : \psi$ が成り立つときには、$M \in [\![\psi]\!]$ が成り立つ。

証明　証明図の構成に関する帰納法による。

1. 証明図が

$$\frac{}{\Gamma \vdash a_i : \varphi_i(\overline{x})}\ (init)$$

の形のとき。$a_i[\overline{K}, \overline{\mathsf{n}}] \equiv K_i$ であるから、この場合定理は自明である。

2. 証明図が

$$\frac{}{\Gamma \vdash M : \varphi(\overline{x})}\ (axiom) \quad \varphi(\overline{x}) \text{ は } \mathbf{HA} \text{ の公理}$$

の形のとき。このとき $\vdash_{\mathbf{HA}} \varphi(\overline{\mathsf{n}})$ であるから $[\![\varphi(\overline{\mathsf{n}})]\!]$ は $\varphi(\overline{\mathsf{n}})$ の構造により定まる。数学的帰納法の公理以外の場合は明らかだから、

$$\varphi(\overline{\mathsf{n}}) \equiv \forall x.(\psi(x) \to \psi(S(x))) \to \psi(0) \to \forall x.\psi(x), \qquad M \equiv \mathsf{rec}$$

の場合を考える。示すべきことは、$L \in [\![\forall x.\psi(x) \to \psi(S(x))]\!]$, $N \in [\![\psi(0)]\!]$, $n \in \mathbb{N}$ としたときに $\mathsf{rec}LN\mathsf{n} \in [\![\psi(\mathsf{n})]\!]$ となることである。このことを n に関する帰納法により示す。

$n = 0$ のとき、

$$\mathsf{rec}LN\mathsf{0} =_\beta N \in [\![\psi(0)]\!]$$

であるから、$\mathsf{rec}LN\mathsf{0} \in [\![\psi(\mathsf{n})]\!]$ が成り立つ。

$n = m + 1$ のとき、$\mathsf{rec}LN\mathsf{m}+\mathsf{1} =_\beta L\mathsf{m}(\mathsf{rec}LN\mathsf{m})$ であるが、帰納法の仮定により $\mathsf{rec}LN\mathsf{m} \in [\![\psi(\mathsf{m})]\!]$. また、仮定より $L\mathsf{m} \in [\![\psi(\mathsf{m}) \to \psi(S(\mathsf{m}))]\!]$. よって

$$\mathsf{rec}LN\mathsf{m}+\mathsf{1} =_\beta L\mathsf{m}(\mathsf{rec}LN\mathsf{m}) \in [\![\psi(S(\mathsf{m}))]\!] = [\![\psi(\mathsf{m}+\mathsf{1})]\!].$$

ここで最後の等号は補題 7.13 による。以上により $\mathsf{rec}LN\mathsf{n} \in [\![\psi(\mathsf{n})]\!]$ が成り立つ。

3. 最後に用いられる推論が $(\to I)$, $(\to E)$ のとき。この場合は定理 6.2 の証明と同様なので割愛する。

4. 証明図の最後の部分が

$$\frac{\Gamma \vdash M[\overline{a}, \overline{x}] : \varphi(\overline{x}) \quad \Gamma \vdash N[\overline{a}, \overline{x}] : \psi(\overline{x})}{\Gamma \vdash \langle M[\overline{a}, \overline{x}], N[\overline{a}, \overline{x}]\rangle : \varphi(\overline{x}) \wedge \psi(\overline{x})}\ (\wedge I)$$

の形のとき。いま、$\Gamma = \{a_1 : \varphi_1(\overline{x}), \ldots, a_k : \varphi(\overline{x})\}$ であるが、どれか一つでも $\nvdash_{\mathbf{HA}} \varphi_i(\overline{\mathsf{n}})$ となる $1 \leq i \leq k$ がある場合には $[\![\varphi_i(\overline{\mathsf{n}})]\!] = \emptyset$ となり前件は成り立ちえないので、定理は自明に成り立つ。さもなければ全ての i について $\vdash_{\mathbf{HA}} \varphi_i(\overline{\mathsf{n}})$ であるから、$\vdash_{\mathbf{HA}} \varphi(\overline{\mathsf{n}})$ および $\vdash_{\mathbf{HA}} \psi(\overline{\mathsf{n}})$ が成り立つ。

そこで $K_1 \in [\![\varphi_1(\overline{\mathsf{n}})]\!]$, $\ldots$, $K_k \in [\![\varphi_k(\overline{\mathsf{n}})]\!]$ とすると、帰納法の仮定により

$$M[\overline{K}, \overline{\mathsf{n}}] \in [\![\varphi(\overline{\mathsf{n}})]\!], \qquad N[\overline{K}, \overline{\mathsf{n}}] \in [\![\psi(\overline{\mathsf{n}})]\!].$$

よって $\langle M[\overline{K}, \overline{\mathsf{n}}], N[\overline{K}, \overline{\mathsf{n}}]\rangle \in [\![\varphi(\overline{\mathsf{n}}) \wedge \psi(\overline{\mathsf{n}})]\!]$ が従う。

以後、簡単にするために $\Gamma = \emptyset$ として話を進める。

5. 証明図の最後の部分が

$$\frac{\vdash M : \varphi \wedge \psi}{\vdash \mathsf{proj}_1 M : \varphi}\ (\wedge E)$$

の形のとき。帰納法の仮定により $M \in [\![\varphi \wedge \psi]\!]$. すなわちある $N_1 \in [\![\varphi]\!], N_2 \in [\![\psi]\!]$ が存在して $M =_\beta \langle N_1, N_2 \rangle$. このとき $\mathsf{proj}_1 M =_\beta N_1$ だから $\mathsf{proj}_1 M \in [\![\varphi]\!]$ が成り立つ。

6. 証明図の最後の部分が

$$\frac{\vdash M[x] : \varphi(x)}{\vdash \lambda x.M[x] : \forall x.\varphi(x)}\ (\forall I)$$

の形のとき。証明すべきことは、任意の $n \in \mathbf{N}$ について $(\lambda x.M)\mathsf{n} \in [\![\varphi(\mathsf{n})]\!]$ となることである。しかし帰納法の仮定により $M[\mathsf{n}] \in [\![\varphi(\mathsf{n})]\!]$ が成り立つので、このことは明らかである。

7. 証明図の最後の部分が

$$\frac{\Gamma \vdash M : \forall x.\varphi(x)}{\vdash M M_t : \varphi(t)}\ (\forall E)$$

の形のとき。t の値を n とすると、補題 7.13 より $[\![\varphi(t)]\!] = [\![\varphi(\mathsf{n})]\!]$ が成り立つ。また、帰納法の仮定により $M \in [\![\forall x.\varphi(x)]\!]$ であるから、$M\mathsf{n} \in [\![\varphi(\mathsf{n})]\!]$ が成り立つ。$M_t =_\beta \mathsf{n}$ でるから $M M_t \in [\![\varphi(t)]\!]$ が従う。

8. 証明図の最後の部分が

$$\frac{\vdash M : \varphi(t)}{\vdash \langle M_t, M \rangle : \exists x.\varphi(x)}\ (\exists I)$$

の形のとき。t の値を n とすると、補題 7.13 より $[\![\varphi(t)]\!] = [\![\varphi(\mathsf{n})]\!]$ が成り立つ。また、帰納法の仮定により $M \in [\![\varphi(t)]\!]$ であるから、$\langle \mathsf{n}, M \rangle \in [\![\exists x.\varphi(x)]\!]$ が成り立つ。$M_t =_\beta \mathsf{n}$ でるから $\langle M_t, M \rangle \in [\![\exists x.\varphi(x)]\!]$ が従う。

9. 証明図の最後の部分が

$$\frac{\vdash M : \exists x.\varphi(x) \quad a : \varphi(x) \vdash N[x,a] : \xi}{\vdash N[\mathsf{proj}_1 M, \mathsf{proj}_2 M] : \xi}\ (\exists E)$$

の形のとき。帰納法の仮定により $M \in [\![\exists x.\varphi(x)]\!]$ である。すなわちある $n \in \mathbf{N}$ と $K \in [\![\varphi(\mathsf{n})]\!]$ について $M =_\beta \langle \mathsf{n}, K \rangle$ である。ふたたび帰納法の仮定により $N[\mathsf{n}, K] \in [\![\xi]\!]$ となるが、

$$\mathsf{proj}_1 M =_\beta \mathsf{proj}_1 \langle \mathsf{n}, K \rangle =_\beta \mathsf{n}, \qquad \mathsf{proj}_2 M =_\beta \mathsf{proj}_2 \langle \mathsf{n}, K \rangle =_\beta K$$

であるから、結局 $N[\mathsf{proj}_1 M, \mathsf{proj}_2 M] \in [\![\xi]\!]$ が言える。 ■

7.4 プログラム抽出定理

論理的述語の場合と同様、実現可能性解釈から多くのことがわかる。まず自明なのは次のことである。

系 7.15 (完全性) φ を算術の文とするとき、

$$\vdash_{\mathbf{HA}} \varphi \qquad \Longleftrightarrow \qquad [\![\varphi]\!] \neq \emptyset.$$

次に、直観主義的な体系一般の特徴である存在特性が証明できる。

系 7.16 (存在特性) 論理式 $\varphi(x)$ について $\vdash_{\mathbf{HA}} \exists x.\varphi(x)$ が成り立つとする。このときある $n \in \mathbb{N}$ が存在して $\vdash_{\mathbf{HA}} \varphi(\mathsf{n})$ が成り立つ。

証明 あるラムダ項 M について $\vdash_{\mathbf{HA}} M : \exists x.\varphi(x)$ が成り立つ。定理 7.14 によれば $M \in [\![\exists x.\varphi(x)]\!]$ である。すなわちある $n \in \mathbb{N}$ および $N \in [\![\varphi(\mathsf{n})]\!]$ について $M =_\beta \langle \mathsf{n}, N\rangle$. よって $[\![\varphi(\mathsf{n})]\!]$ は空でないから、$\vdash_{\mathbf{HA}} \varphi(\mathsf{n})$ が成り立つ。 ■

最後に本章の本題であるプログラム抽出定理を証明する。この定理は「証明とはプログラムである」というパラダイムをもっとも端的に表すものである。

系 7.17 (プログラム抽出) 二変数論理式 $\varphi(x, y)$ について $\vdash_{\mathbf{HA}} \forall x \exists y.\varphi(x, y)$ が成り立つとする。このときあるラムダ項 M_φ が存在し、任意の $m \in \mathbb{N}$ について

$$M_\varphi \mathsf{m} \longrightarrow_\beta^* \mathsf{n} \qquad \text{かつ} \qquad \vdash_{\mathbf{HA}} \varphi(\mathsf{m}, \mathsf{n})$$

が成り立つ。

証明 あるラムダ項 M について $\vdash_{\mathbf{HA}} M : \forall x \exists y.\varphi(x, y)$ が成り立つ。定理 7.14 によれば $M \in [\![\forall x \exists y.\varphi(x, y)]\!]$ である。すなわち任意の $m \in \mathbb{N}$ について $M\mathsf{m} \in [\![\exists y.\varphi(\mathsf{m}, y)]\!]$. すると上と同様にして、ある $n \in \mathbb{N}$ とラムダ項 $N \in [\![\varphi(\mathsf{m}, \mathsf{n})]\!]$ が存在して $M\mathsf{m} =_\beta \langle \mathsf{n}, N\rangle$ となる。ゆえに $M_\varphi := \lambda z.\mathsf{proj}_1(Mz)$ とおけば、

$$M_\varphi \mathsf{m} =_\beta \mathsf{proj}_1(M\mathsf{m}) =_\beta \mathsf{proj}_1\langle \mathsf{n}, N\rangle =_\beta \mathsf{n}$$

が言える。系 4.13(3) により等号 $=_\beta$ は $\longrightarrow_\beta^*$ で置き換えることができるので $M\mathsf{m} \longrightarrow_\beta^* \mathsf{n}$ が成り立つ。最後に、$[\![\varphi(\mathsf{m}, \mathsf{n})]\!]$ は空でないので、$\vdash_{\mathbf{HA}} \varphi(\mathsf{m}, \mathsf{n})$ が言える。 ■

このことと定理 7.9 を組み合わせると、一定の条件下では実は **PA** の証明図からもプログラムが抽出できることがわかる。背理法や排中律をふんだんに用いた古典的な証明も、実はプログラムと見ることができるのである。

系 7.18 (プログラム抽出) 二変数の Σ_1 論理式 $\varphi(x, y)$ について $\vdash_{\mathbf{PA}} \forall x \exists y.\varphi(x, y)$ が成り立つとする。このときあるラムダ項 M が存在し、任意の $n \in \mathbb{N}$ について

$$M\mathsf{n} \longrightarrow_\beta^* \mathsf{m} \qquad \text{かつ} \qquad \vdash_{\mathbf{PA}} \varphi(\mathsf{n}, \mathsf{m})$$

が成り立つ。

たとえば $\varphi(x,y) := x < y \wedge \mathsf{prime}(y)$ とおくとこれは限定論理式であり、$\forall x \exists y.\varphi(x,y)$ は「無限に多くの素数が存在する」ことを主張するユークリッドの定理となる。さて、この定理の **PA** における証明が与えられたとしよう。その証明はどんなに回りくどくともよいし、背理法をはじめとする非構成的論法を用いていてもよいとする。すると A 翻訳を通して **HA** における証明が得られる（定理 7.9）。その証明にラムダ項を割り当てると、「与えられた自然数よりも大きな素数を出力する」プログラム M_φ が自動的に得られる。

このようにして、証明さえ与えられればそこからプログラムが抽出できる。興味深いのはこの抽出は完全に自動的に行えるという点である。しかも、デバッグやテストの必要は全くない。得られたプログラムが正しく働くということは 100 パーセント理論的に保証されているのである。上の例でいえば、出力は常に素数となることが保証されており、単に出力が自然数となることしか保証することのできないシステム **F** を完全に凌駕する。この観点からすると、**HA** をシステム **F** よりも精密な型システムとみなすこともできるだろう。

M_φ がプログラムだとしたら、$\forall x \exists y.\varphi(x,y)$ はプログラムが満たすべき**仕様** (specification) である。たとえば上の例の場合、論理式 $\forall x \exists y.\varphi(x,y)$ は「入力として自然数が与えられたらそれよりも大きな素数を出力する」という仕様として見ることができる。ソフトウェア開発の理想的な状況では、まず満たされるべき仕様、すなわち「こんな働きをするソフトウェアが欲しい」というクライアントの願いがあって、それを実現するためにソフトウェアが開発される（実装）。そして開発されたソフトウェアが仕様にあっているかどうかが最後に検証される。この仕様-実装-検証という 3 ステップをもってソフトウェア開発の 1 サイクルが構成される—これが普通の考え方である。

一方、プログラム抽出定理に触発されて次のようなソフトウェア開発の方法論が取りざたされるようになった。すなわち仕様-実装-検証という 3 ステップを経るのでなく、仕様を形式的体系の定理だと思って直接証明してしまおう！というのである。そうすれば証明からプログラムを自動的に抽出することができるから実装は不要であるし、そのようにして得られたプログラムは絶対に正しいから検証も不要である。しかも証明を**証明支援系** (proof assistant system) で行うことにより、ルーティン的な部分はコンピュータにより自動的に証明できるものと期待できるから、証明コスト＝プログラミングコストも削減できるであろう。このような方法論を**構成的プログラミング** (constructive programming) という。

もちろん世の中そんなにうまくものではなく、一見素晴らしく見えるアイデアにも必ずデメリットが存在するものである。構成的プログラミングの場合、もちろん何よりも問題なのは、証明するのはプログラミングよりもはるかに大変だ、という単純なるがゆえに深刻な事実である。その他諸々の問題ははらみつつも、構成的プログラミングという大胆なアイデアには抗しがたい魅力があり、現在でも多くの試みがなされている。証明コスト削減、プログラムの効率化はいうまでもなく、ある種の数学的証明から抽出されたプログラムは大変効率的である（要は速い）という事例も報告されている。

最後にもう一点指摘しておく。系 7.18 が示唆するのは、一定の制限はあるものの、少なくとも原理上は古典的な証明もプログラムとして見ることができるということである。すなわちカリー・ハワード対応は古典論理へも拡張できる！

このように古典論理の証明を計算論的に取り扱う試みが 1990 年代に盛んに行われた。鍵となるのは次の洞察である。古典論理と直観主義論理を分ける論理式の一つに**パースの法**

則 (Peirce's law)

$$\varphi := ((\psi \to \xi) \to \psi) \to \psi$$

が挙げられる。この論理式は古典論理では証明可能だが直観主義論理では証明可能ではない。そしてこの論理式を公理として直観主義論理に加えると古典論理に一致する。1990年代初頭にグリフィンは、パースの法則の計算論的な意味は一体何だろうかと考えた。彼の結論は、ある種の例外処理に有用な**制御演算子** (control operator) に相当するというものである。それゆえ、直観主義論理の証明が純粋な関数プログラムを表すとすれば、古典論理の証明は関数的ではあるものの例外処理ができるようなプログラムとみなすことができる。

グリフィンの洞察をブレークスルーとして、1990年代には多くの試みがなされた。中でも重要なのは、古典論理の双対原理（定理1.32）である。すなわち、古典論理においては $\wedge$ と $\vee$、$\forall$ と $\exists$ はそれぞれ双対的な関係にある。ではこの双対原理をカリー・ハワード対応を通して計算の文脈に移し替えたら、一体どのような性質として現れるのだろうか？

ラムダ計算において関数 $\lambda x.M[x]$ に入力 N を与えるときには、二通りの計算の進め方（**方略**, strategy）がある。第一に、まずは N を計算してその値 V（正規形）を求めた上で関数に与えるという方略で、この場合

$$(\lambda x.M[x])N \longrightarrow_\beta^* (\lambda x.M[x])V \longrightarrow_\beta M[V] \longrightarrow_\beta \cdots$$

という風に計算は進むことになる。この方略を関数の**値呼び出し** (call-by-value) という。もう一つの方略は、N を値の "名前" だと思って N そのものを関数に与えるというものである。この場合は計算は

$$(\lambda x.M[x])N \longrightarrow_\beta \lambda x.M[N]) \longrightarrow_\beta \cdots$$

という風に進む。これを関数の**名前呼び出し** (call-by-name) という。

1990年代から2000年代にわたって確立された考え方を非常に大雑把にいえば、この値呼び出しと名前呼び出しの間には計算論的な意味での双対性があり、この双対性が古典論理の論理的な意味での双対性に対応するというのである。

$$\text{論理的双対性} \qquad \approx \qquad \text{計算論的双対性}$$

このように、カリー・ハワード対応は単にそれ自身に留まるものではなく、それを通してさまざまな論理的現象と計算論的現象が対応づけられることにより、新しいパラダイムを次々に生み出していく、そういう創造的な側面をも併せ持つのである。

7.5 システム T

前々節で示したとおり、**HA** の証明図にはラムダ項を割り当てることができる。別の言い方をすれば、**HA** はラムダ項に対する型システムと見ることができる。そこで証明図に対応するラムダ項をよく注意してみると、それらは全てシステム **F** で型付けできることがわかる。さらに詳しく見れば、システム **F** の中でもごく限られた形の型しか用いる必要がないことがわかる。以下ではシステム **F** の部分型システムとして**システム T**(System T)

を導入する。これはゲンツェンの仕事を分析することでゲーデルが考案したシステムで、歴史的にはシステム $\mathbf{F}$ にはるかに先立つものである。

システム $\mathbf{T}$ の型は次のように定義される。

$$A, B ::= \mathbf{Nat} \mid \top \mid A \wedge B \mid A \Rightarrow B.$$

つまり自然数型と $\top$ から $\Rightarrow$ と $\wedge$ を使って作ることができる型のみがシステム $\mathbf{T}$ の型であり、変数型や多相型は一切含まない。

システム $\mathbf{T}$ では、$\mathsf{0}$, suc, rec, $\langle\ ,\ \rangle$, proj_1, proj_2 を基本的なラムダ項と考え、次の型推論規則を用いる。

$$\frac{}{\Gamma \vdash x : A}\ (init) \quad \text{ただし } x : A \in \Gamma \qquad\qquad \frac{}{\Gamma \vdash \mathsf{id} : \top}\ (\top I)$$

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x.M : A \Rightarrow B}\ (\Rightarrow I) \qquad \frac{\Gamma \vdash M : A \Rightarrow B \quad \Gamma \vdash N : A}{\Gamma \vdash MN : B}\ (\Rightarrow E)$$

$$\frac{\Gamma \vdash M : A \quad \Gamma \vdash N : B}{\Gamma \vdash \langle M, N\rangle : A \wedge B}\ (\wedge I) \qquad \frac{\Gamma \vdash M : A \wedge B}{\Gamma \vdash \mathsf{proj}_1 M : A}\ (\wedge E) \qquad \frac{\Gamma \vdash M : A \wedge B}{\Gamma \vdash \mathsf{proj}_2 M : B}\ (\wedge E)$$

$$\frac{}{\Gamma \vdash \mathsf{0} : \mathbf{Nat}}\ (\mathsf{0}) \qquad \frac{\Gamma \vdash M : \mathbf{Nat}}{\Gamma \vdash \mathsf{suc} M : \mathbf{Nat}}\ (\mathsf{suc})$$

$$\frac{\Gamma \vdash M : A \Rightarrow A \quad \Gamma \vdash N : A \quad \Gamma \vdash K : \mathbf{Nat}}{\Gamma \vdash \mathsf{rec} MNK : A}\ (\mathsf{rec})$$

これらの型推論規則を用いて $\Gamma \vdash M : A$ が導出できるとき、$\Gamma \vdash_{\mathbf{T}} M : A$ と書く。

第 5.5 節（とくに命題 5.11）を思い出せば、次のことは明らかだろう。

定理 7.19 A をシステム $\mathbf{T}$ の型とし、型環境 Γ はシステム $\mathbf{T}$ の型からなるものとする。このとき

$$\Gamma \vdash_{\mathbf{T}} M : A \quad \Longrightarrow \quad \Gamma \vdash_{\mathbf{F}} M : A.$$

次に $\mathbf{HA}$ とシステム $\mathbf{T}$ の関係をみるために、算術の論理式 φ に対してシステム $\mathbf{T}$ の型 $\varphi^{\mathbf{T}}$ を割り当てる。

$$\begin{aligned}
(t = u)^{\mathbf{T}} &:= \top \\
(\varphi \wedge \psi)^{\mathbf{T}} &:= \varphi^{\mathbf{T}} \wedge \psi^{\mathbf{T}} \\
(\varphi \to \psi)^{\mathbf{T}} &:= \varphi^{\mathbf{T}} \Rightarrow \psi^{\mathbf{T}} \\
(\forall x.\varphi)^{\mathbf{T}} &:= \mathbf{Nat} \Rightarrow \varphi^{\mathbf{T}} \\
(\exists x.\varphi)^{\mathbf{T}} &:= \mathbf{Nat} \wedge \varphi^{\mathbf{T}}
\end{aligned}$$

たとえば

$$(\forall x.\exists y.\varphi(x,y))^{\mathbf{T}} \quad = \quad \mathbf{Nat} \Rightarrow (\mathbf{Nat} \wedge \varphi(x,y)^{\mathbf{T}})$$

である。すると次のことが成り立つ。

定理 7.20 $\Gamma = \{a_1 : \varphi_1, \ldots, a_n : \varphi_n\}$ のもとで $\Gamma \vdash_{\mathbf{HA}} M : \varphi$ が成り立つならば、

$$a_1 : \varphi_1^{\mathbf{T}}, \ldots, a_n : \varphi_n^{\mathbf{T}} \vdash_{\mathbf{T}} M : \varphi^{\mathbf{T}}$$

が成り立つ。とくに $\vdash_{\mathbf{HA}} M : \varphi$ ならば $\vdash_{\mathbf{T}} M : \varphi^{\mathbf{T}}$ である。

それゆえ系 7.17 で抽出されたラムダ項 M_φ はシステム $\mathbf{T}$ で型付け可能であり、$\vdash_{\mathbf{T}} M_\varphi : \mathbf{Nat} \Rightarrow \mathbf{Nat}$ が成り立つ。

次に **PA** の証明能力とシステム $\mathbf{T}$ の計算能力を関連づける定理を述べる。そのために第 3 章の定義 3.6 を思い出そう。

定義 7.21 関数 $f : \mathbb{N} \longrightarrow \mathbb{N}$ に対して次のような Σ_1 論理式 $\varphi_f(x, y)$ が存在するとき、f は Δ_1 **関数**であるという：任意の $n \in \mathbb{N}$ について、

$$f(n) = m \quad \Longleftrightarrow \quad \mathbf{N} \models \varphi_f(\mathsf{n}, \mathsf{m}).$$

このとき明らかに $\mathbb{N} \models \forall x \exists y.\varphi_f(x, y)$ が成り立つが、**PA** がこのことを証明できるとき、すなわち $\vdash_{\mathbf{PA}} \forall x \exists y.\varphi_f(x, y)$ が成り立つとき、f **の全域性は PA で証明可能である**という。

もう一つ次の定義をしておく。

定義 7.22 関数 $f : \mathbb{N} \longrightarrow \mathbb{N}$ に対して次のようなラムダ項 M_f が存在するとき、f は**ラムダ定義可能**であるという：任意の $n \in \mathbb{N}$ について、

$$f(n) = m \quad \Longleftrightarrow \quad M_f\mathsf{n} \longrightarrow_\beta^* \mathsf{m}.$$

さらに上のラムダ項 M_f について $\vdash_{\mathbf{T}} M_f : \mathbf{Nat} \Rightarrow \mathbf{Nat}$ が成り立つとき、f は**システム T で定義可能**であるという。

関数 f が Δ_1 関数であること（すなわち計算可能であること）とラムダ定義可能であることは一致する（系 4.10 参照）。この対応関係は、**PA** とシステム $\mathbf{T}$ の対応関係へと精密化することができる。

定理 7.23 任意の関数 $f : \mathbb{N} \longrightarrow \mathbb{N}$ について、f の全域性が **PA** で証明可能であることと f がシステム $\mathbf{T}$ で定義可能であることは一致する。

証明　f の全域性が **PA** で証明可能ならば、ある Σ_1 論理式 $\varphi(x, y)$ が存在し、$\vdash_{\mathbf{PA}} \forall x \exists y.\varphi(x, y)$ が成り立つ。$\forall x \exists y.\varphi(x, y)$ は Π_2 文であるから、同じことは **HA** でも成り立つ。ゆえにあるラムダ項 M が存在し、$\vdash_{\mathbf{HA}} M : \forall x \exists y.\varphi(x, y)$ となる。

系 7.17 の証明を参考にして $M_\varphi := \lambda z.\mathsf{proj}_1(Mz)$ とすれば、任意の $m \in \mathbb{N}$ について $M\mathsf{m} \longrightarrow_\beta^* \mathsf{n}$ かつ $\vdash_{\mathbf{HA}} \varphi(\mathsf{m}, \mathsf{n})$ が成り立つ。後者は $\mathbf{N} \models \varphi(\mathsf{m}, \mathsf{n})$ と同値であり、それゆえ $f(m) = n$ と同値である。よって M_φ は関数 f をラムダ定義する。また、

$$(\forall x \exists y.\varphi(x, y))^{\mathbf{T}} = \mathbf{Nat} \Rightarrow (\mathbf{Nat} \wedge \varphi(x, y)^{\mathbf{T}})$$

であるから、定理 7.20 を用いれば $\vdash_{\mathbf{T}} M : \mathbf{Nat} \Rightarrow (\mathbf{Nat} \wedge \varphi(x, y)^{\mathbf{T}})$ が成り立つ。このことから $\vdash_{\mathbf{T}} M_\varphi : \mathbf{Nat} \Rightarrow \mathbf{Nat}$ は容易に示すことができる。

逆に f がシステム $\mathbf{T}$ で定義可能であれば f の全域性が **PA** で証明可能となるが、このことの証明は省略する。 ■

PA が全域性を証明できるような関数に限ってシステム $\mathbf{T}$ は取り扱うことができる。このように、**PA** の証明能力とシステム $\mathbf{T}$ の計算能力は正確に対応するのである。