

Martin-Löf の型理論における オブジェクトおよび型についての開放性

塚田 恭章

NTT 基礎研究所

Martin-Löf の型理論のもつ自己充足性は、新しいオブジェクトや型の導入を前提とするセマンティクスの実現を導いた。本論文はオブジェクトおよび型についての開放性を定式化したものである。未解決問題である型についての開放性に焦点をあてながら、理論の根底に潜む言語をメソッド計算系として規定する。型の一様な構成法は、メソッド計算系上で構成される型システムの帰納的定義を可能にする。定義の整合性を確認するとともに、決定的かつ正則なメソッド計算系の拡張列が任意に与えられたとき、その上で構成される型システムは常に型理論のモデルになること、すなわちオブジェクトおよび型についての開放性がとらえられたことを示す。

Open-Endedness of Objects and Types in Martin-Löf's Type Theory

Yasuyuki TSUKADA

NTT Basic Research Laboratories

Self-contained description of Martin-Löf's type theory produced a semantical framework that anticipates the introduction of new objects and types, and this paper presents a comprehensive formulation of open-endedness of types as well as objects in the theory. A notion of method calculus is introduced to define an open-ended body of language underlying the theory, and then types are uniformly built as a type system inductively defined on this calculus. The main theorem in this formulation is that a model of type theory is provided by any type system built on a deterministic and regular extension of the original method calculus.

1 はじめに

Martin-Löf は、構成的数学の基礎を明らかにしたいという哲学的動機にもとづいて、1970 年代より、直観主義型理論 (Intuitionistic Type Theory, ITT) のいくつかのバージョンを設計し、発表してきた。彼は、自身の理論に、メタ数学的な研究対象としての意義だけでなく、独立した思想的価値をも与えたいと考えた。その結果、独自のセマンティクスをもつ型理論が、哲学的議論とともに誕生した [9] [10]。

ITT を特徴づける基本的原理のひとつとして、命題と型の同一性を主張する Curry-Howard 対応 (Curry-Howard Isomorphism) は有名である。この原理により、プログラムの仕様を型を使って記述することが可能となった。型推論と仕様検証とが、型理論というひとつの枠組みの中で融合されたのである。ITT を基礎にした多くの証明開発システム (proof development system) が計算機上に実現され [4] [13]、また各方面へ応用されている背景に、Curry-Howard 対応によるこの豊かな表現力があることは明らかであるが、はたしてそれだけなのだろうか？

ITT のもつ次のふたつの性質は注目に値する：

- 自己充足性 (self-containedness)
- 開放性 (open-endedness)。

一般に、言語の意味はどのように与えられるのだろうか。数学やプログラミングにおいて用いられる形式的言語の意味はどのようにして説明されるのだろうか。たとえば 1 階述語論理については、よく知られているように、モデルを与えることによってその意味が説明される。しかし、モデルを与えるという行為は言語と無縁には存在しないことは明らかである。つまり、今の場合、集合論の言語を使ってモデル化したのだと。すなわち、1 階述語論理を集合論の言語に翻訳したのだとみなすことも可能なのである。形式的言語に意味を与えるこのふたつの方法 — モデル化 (モデル理論的) と他の言語への翻訳 (証明論的) — は、方法論の違いだけで区分けされているのであって、じつは全く同じものである。

さて、われわれはこのような意味の説明をどこまでくりかえせばよいのだろうか。われわれが最終的に獲得したいのは、モデル化や他の言語への翻訳によって意味を説明する必要のない理論である。それは、他の概念によって説明されることのない最も基本的な概念とともに提示されるべきものである。このような理論を得るには、今までと全く異なった、純粋にセマンティカルな洞察から始めなければならないであろう。ITT は、このような自己充足的な理論のひとつの例として、すなわち、算術・解析・プログラミング等の言語の意味を統一的な枠組みの中で説明することのできる最も基本的な理論の一例として提示されたものである [11]。

Martin-Löf はこの理論に対し首尾一貫した説明を与えることで彼の哲学を完成させようとした。ITT 特有のこのセマンティクスを含む、いくつかの哲学的

議論は、理論の厳密な理解を困難にさせる原因にもなっているが、すべて、上で述べたような、チャレンジな問題を解決する過程で生まれてくるものなのである。

自己充足性を獲得しようとする姿勢は、ITT の設計に多大な影響を与えた。選ばれた基本概念 — 型とそのオブジェクト — は、それ以上還元することのできないプリミティブな概念 — メソッド (method) — によって説明された。メソッドをプリミティブに扱うことは、ITT の第二の性質 — 型とオブジェクトについての開放性 — を自然に導く¹。すなわち、静的に閉じた従来の理論とは異なり、将来新しい型やオブジェクト、あるいは数学的概念や対象が言語の中に導入されることを前提とし、またそれを予期した理論として、ITT は設計されたのである。

ITT の開放性は、新しい型やオブジェクトの導入を排除しないという単純な側面でもとらえられるものではない。その進取性は、新しい型やオブジェクトの導入を前提として組み立てられたセマンティクスを完備しているという点にある。次のような例を考えよう。プログラマが、自然数の型と、その上で定義されるある手続きを、新たに導入する必要に迫られたとする。手続きは任意の自然数に対して定義されなければならない。彼は、その時点で、すべての自然数からなる型をどのように定義したらよいのだろうか。0 は自然数であり、 a が自然数ならば $S(a)$ も自然数である。この伝統的な定義は明らかに不十分である。なぜなら、後にさまざまなオペレータが導入されることで、新たに構成される自然数 — たとえば 10^{10} 、 $10 + 10$ 、 $10000!$ など — が含まれていないからである。同様に、その後導入されるさまざまな型を考慮して、型の全体であるユニバース — 矛盾を引き起こさぬよう注意が必要だが — をある時点で定義するにはどうしたらよいのだろうか。

一般に、ある時点で得られた言語 L_0 に対し、その意味 $M(L_0)$ が “effective” 写像 M によって定められたとする。このとき、 L_0 を始点とする任意の拡張言語列 $L_0 \subseteq L_1 \subseteq L_2 \subseteq \dots$ に対し、次の 2 条件：

- $M(L_i)$ はそれぞれ L_i ($i \geq 1$) の意味を定める
- L_0 での推論は、すべての $M(L_i)$ ($i \geq 0$) における充足性を考慮して行なわれる

がともに満たされているならば、この理論²は開放性をもつという (*)。Martin-Löf は、型とは何か、そのオブジェクトとは何か、メソッドの概念をもとに明らかにすることを通じて、そのような開放性を ITT にもたらしたのである。

もちろん、数学やプログラミングのための理論が、必ずしも開放的である必要はないかもしれない。実際、多くの汎用言語は、ある程度の表現能力を備えた上で、

¹同様の姿勢は BHK (Brouwer-Heyting-Kolmogorov) 解釈においても特徴的である。実際、いくつかの基本的論理記号から形成される命題とその証明が、メソッドというプリミティブな概念によって説明される。これにより “証明についての開放性” が導かれる。

²言語とその意味の組をここでは理論と呼ぶ。

さまざまな数学的概念や型を内部で定義できるようにしている。しかし、そのような定義は、しばしば巧妙かつ複雑な符合化を伴う。新しい数学的概念や型は、それぞれにもっとも適切な形で定義され、導入されることが望ましい。

すなわち、自己充足のかつ開放的な ITT は、数学やプログラミングといった知的活動を自然に定式化するのに適した理論であることがわかる。まず、人間の思考と直結し、理論の中で活動できること。しかも、開放的な環境の中で、生生発展していく概念や知識を動的にとらえられること。もちろん、ITT は、証明開発システムとして計算機上に実現されると、形式的かつ部分的な存在となる。ましてや、Martin-Löf が自身の哲学を完成させるために形成したセマンティクスなどは、実際のシステムと無縁のようにも思える。しかし、形と意味とを完全に分離することが不可能なことは明らかであり、少なくとも数学者やプログラマの思考はこのようなセマンティクスと深く結びついている。したがって、自己充足性と開放性——従来の理論にはみられない、しかし自然な性質——をもつ ITT のセマンティクスが、人間の思考をとらえる同種の理論を設計する際に、大きな参考となることが期待される。たとえば、型とオブジェクトにかんする知識論的な主張である判定 (judgement) の形を拡張あるいは変更することにより、新しい自己充足のかつ開放的な理論を確立することも可能なはずである。

本論文の主題は、哲学的に記述された ITT における開放性を、数学的にとらえること——すなわち、(*) を数学的に記述し、証明すること——である。型とそのオブジェクトが基本概念として選択されたことに対応して、2 種類の開放性を ITT に見いだすことができる：

- オブジェクトについての開放性
- 型についての開放性。

前者は、新しいオブジェクトの導入を前提として、個々の型が定義されている性質をさす。後者は、さまざまな型の導入を前提として、ユニバースが定義されている性質をさす。

計算論的 (computational) 開放性とも呼ばれる前者については、次のような定式化が得られている。まず、ITT の背景に潜む言語——型のないラムダ計算系と本質的に同じもの——を、計算システム (computation system) として規定する。計算システムは、項の全体と項の間の (遅延) 評価関係の組である。次に、特定の基本的型構成子から、その言語上で帰納的に形成される型の世界を、計算システム上の型システム (type system) τ として規定する [2]。 τ は、項の全体から項の間の部分同値関係 (partial equivalence relation) の全体への部分関数であり、ITT のセマンティクスを数学的に表現したものと考えられる。実際、この τ を用いて、ITT の判定

$$T \text{ type}, t \in T, t = t' \in T, T = T'$$

は、それぞれ

$$T \in \text{dom}(\tau), (t, t) \in \tau(T), (t, t') \in \tau(T), \tau(T) = \tau(T')$$

と解釈される。そのとき、オブジェクトについての開放性を、次のような性質として数学的に規定し、証明することができる：適当な条件のもとで、計算システムの拡張列が任意に与えられたとき、それらの上で構成される型システムは常に ITT のモデルとなる。実際、計算システムをじゅうぶん大きく拡張することにより、ITT の古典的モデル——その中で排中律が成り立つ——も構成される³ [7]。

しかし、型の導入を分析するための枠組みはこの定式化からは得られない。型システムが、特定の基本的型構成子に依存して定義されているからである。すなわち、後者の開放性は重要な未解決問題である。われわれの目標は、オブジェクトだけでなく、型についての開放性をもとらえることの可能な枠組みを提供することである。そのためには、特定の基本的型構成子に依存した型システムの定義を、より一般的に記述しなおさなければならない。そこでまず、型構成にかんする情報を有する、拡張された計算システムとして、メソッド計算系 (method calculus) の概念を導入する。本質的な貢献は、型構成子の候補となるオペレータに種 (kind) と呼ばれる特別な情報を付加したことにある。ITT における型構成子は、ある型の族から別の型を新たに構成する写像と解釈できるが、種はそのような写像の定義域を一般化したものである。種をもとにした型の一様な構成法は、メソッド計算系上で構成される型システムの帰納的定義を可能にする。帰納的定義の整合性を確認するとともに、決定的かつ正則なメソッド計算系 (前者はメソッドの評価法についての条件、後者は各型に定義される“正規的オブジェクト”間の等号についての条件) の拡張列が任意に与えられたとき、それらの上で構成される型システムは常に ITT のモデルになること、すなわちオブジェクトおよび型についての開放性がとらえられたことを示す。

2 メソッド計算系

ITT⁴においては、メソッドはプリミティブな概念である。すなわち、メソッドの全体は開放的である。さらに、型——メソッドのある種の集まり——の定義に強い制限がないことから、メソッド間のある種の関係の全体もまた開放的である。これらの開放性を統一的に扱うために、メソッド計算系の概念を導入する。

メソッド計算系 (method calculus) $C = (O, K, \alpha, R, \kappa, \delta, \varphi)$ を次のように定義する。

O をオペレータの集合、 $K \subset O$ を正規的な (canonical) オペレータの集合とし、

$$\alpha : O \rightarrow \{(k_1, \dots, k_n) \mid n, k_i \geq 0\}$$

とする。 $\alpha(p)$ は、オペレータ $p \in O$ のアリティ (arity) と呼ばれ、引数の数と変数の束縛構造を規定する。

³メソッドを集合論的な関数とみなせば BHK 解釈は古典的セマンティクスに一致する、という事実に対応する。

⁴本論文で扱う ITT は、[9] で述べられているものと実質的に同じバージョンである。

各 $i \geq 0$ に対し、アリティ i の 2 階の変数を無限個用意する。2 階の変数は、アリティが 0 のとき変数、アリティが 1 以上のときメタ変数と呼ばれる。アリティ m の 2 階の項スキーマを次のように帰納的に定義する。アリティ m の 2 階の変数は、アリティ m の 2 階の項スキーマである。 ρ がアリティ $(k_1, \dots, k_n)$ のオペレータで、 $b_1, \dots, b_n$ がそれぞれアリティ $k_1, \dots, k_n$ の 2 階の項スキーマならば、 m 個の変数の列 $\bar{x}$ に対して、 $\bar{x}.\rho(b_1, \dots, b_n)$ はアリティ m の 2 階の項スキーマである。 s がアリティ n の 2 階の項スキーマで、 $a_1, \dots, a_n$ がアリティ 0 の 2 階の項スキーマならば、 m 個の変数の列 $\bar{x}$ に対して、 $\bar{x}.s[a_1, \dots, a_n]$ はアリティ m の 2 階の項スキーマである。メタ変数を含まない 2 階の項スキーマを、2 階の項と呼ぶ。アリティ 0 の 2 階の項および項スキーマは、それぞれ項および項スキーマと呼ばれる。 ρ をオペレータ、 $\bar{i}$ を異なる 2 階の変数の列とすると、 $\rho(\bar{i})$ なる項スキーマを、単純項スキーマと呼ぶ。

アリティ m の 2 階の項スキーマ $x_1, \dots, x_m.t$ において、各変数 x_i は t の対応する変数を束縛する。2 階の項スキーマが束縛変数の交換によって同じものになるとき、それらを同一視する。アリティ m の閉じた 2 階の項の全体を、 $\hat{T}^m(C)$ と書く。閉じた項の全体 $\hat{T}^0(C)$ を、とくに $\hat{T}(C)$ と書く。

2 階の項 t に対し、その自由変数 $x_1, \dots, x_n$ に項 $a_1, \dots, a_n$ をそれぞれ同時に代入したものを、 $t[a_1, \dots, a_n/x_1, \dots, x_n]$ と書く。文脈から自由変数の列が明らかなきとき、単に $t[a_1, \dots, a_n]$ と書く。また、アリティ m の 2 階の項 $x_1, \dots, x_m.t$ および項 $a_1, \dots, a_m$ に対し、 $(x_1, \dots, x_m.t)[a_1, \dots, a_m]$ と $t[a_1, \dots, a_m]$ を同一視する。

項または単純項スキーマ $\rho(\bar{i})$ は、 ρ が正規的であるとき正規的、そうでないとき非正規的 (noncanonical) であるという。閉じた項はプログラムに、閉じた正規的項はプログラムの評価結果に対応する。

$R = \{r_i\}_{i \in I}$ は評価規則の集合である。各 r_i は、

$$a \rightarrow b \Leftarrow a_1 \rightarrow b_1 \ \& \ \dots \ \& \ a_n \rightarrow b_n \ (n \geq 1)$$

という形で与えられ、次の条件をみたすものとする：

1. a は非正規的な単純項スキーマである
2. 各 b_i ($1 \leq i \leq n$) は変数または正規的な単純項スキーマで、 a および b_j ($j \neq i$) とは自由変数およびメタ変数を共有しない
3. 各 a_i ($1 \leq i \leq n$) は項スキーマで、 a_i に出現する自由変数およびメタ変数は a または b_j ($j < i$) に出現するものに限る
4. b は b_n と同じ変数である。

アリティ m のメタ変数を $\hat{T}^m(C)$ 上で、自由変数を $\hat{T}(C)$ 上で、それぞれ解釈することにより、評価規則の集合 $\{a \rightarrow a \mid a \text{ は任意の正規的な単純項スキーマ}\} \cup R$ は、評価関係 $\rightarrow_C \subset \hat{T}(C) \times \hat{T}(C)$ の帰納的定義を導く。 $\rightarrow_C$ は次の条件をみたす：

1. $a \rightarrow_C b$ ならば、 b は正規的である
2. a が正規的ならば、 $a \rightarrow_C b$ と $a \equiv b$ は同値である。

これらの条件は、評価関係 $\rightarrow_C$ が遅延的 (lazy) であることを主張する。とくに、 $a \rightarrow_C b$ かつ $a \rightarrow_C c$ ならば $b \equiv c$ となる評価関係 $\rightarrow_C$ は、決定的 (deterministic) であるという。決定的評価関係を生成するメソッド計算系を、決定的であるという。

型についての開放性をとらえるため、型構成子 (ただしユニバースを除く) の候補となるオペレータに、必要な情報をあらかじめ付加しておくことにしよう。部分関数⁵

$$\kappa: (K - \{U_n \mid n \geq 0\}) \rightarrow \left(\bigcup_{m \geq 1} 1(\square)^* 2(\square)^* 3(\square)^* \dots m(\square)^* \right)^*$$

は、正規的オペレータ $\Delta \in \text{dom}(\kappa)$ に対し、 Δ の種 (kind) $\kappa(\Delta)$ を与える。また、

$$\kappa(\Delta) = \overbrace{1\square \dots \square}^{\square \text{ が } k_{1,1} \text{ 個}} \dots \overbrace{m_1\square \dots \square}^{\square \text{ が } k_{1,m_1} \text{ 個}} \dots \overbrace{1\square \dots \square}^{\square \text{ が } k_{n,1} \text{ 個}} \dots \overbrace{m_n\square \dots \square}^{\square \text{ が } k_{n,m_n} \text{ 個}}$$

に対し、次の演算を導入する。

$$|\kappa(\Delta)| = n$$

$$\kappa(\Delta)_i = m_i$$

$$\kappa(\Delta)_{i,j} = k_{i,j}.$$

ただし、 $1 \leq i \leq |\kappa(\Delta)|$ 、 $1 \leq j \leq \kappa(\Delta)_i$ とする。

ITT における型構成子は、ある型の族から別の型を新たに構成する写像と解釈することができる。種は、そのような写像の定義域を一般化したものである。種をもとにした型およびユニバースの一般的な構成法 (後述) は、ITT が述語的 (predicative) 理論であることを含意する。ITT における基本的な型構成子の種は、次の表で与えられる。

種	ε	$1\square\square$	11	12
型構成子	N_n, N	I	$+$	Π, Σ, W

さて、正規的オペレータ $\Delta \in \text{dom}(\kappa)$ は型構成子の候補である。実際に型を構成するには、その型の (等しい 2 つの) 正規的オブジェクトを規定しなければならない。すなわち、正規的オブジェクトを形成する正規的オペレータの集合と、その正規的オペレータの引数がみたす条件を規定しなければならない。前者は δ によって、後者は φ によって、実質的に定められる。関数

$\delta: \text{dom}(\kappa) \times \omega \rightarrow (K - (\text{dom}(\kappa) \cup \{U_n \mid n \geq 0\}))^*$ (ただし ω は自然数の全体) は次の条件をみたすものとする：

1. 任意の $\Delta \in \text{dom}(\kappa)$ 、 $k \in \omega$ 、 j_1, j_2 ($0 \leq j_1 < j_2 \leq |\delta(\Delta, k)|$) について $\delta(\Delta, k)_{j_1}$ と $\delta(\Delta, k)_{j_2}$ は異なる
2. 任意の $\Delta, \Delta' \in \text{dom}(\kappa)$ について $\bigcup \{\delta(\Delta, 0)_j \mid 0 \leq j \leq |\delta(\Delta, 0)|\} \cap \bigcup \{\delta(\Delta', 0)_j \mid 0 \leq j \leq |\delta(\Delta', 0)|\} = \emptyset$.

⁵ われわれは、ユニバースを表す正規的オペレータの集合 $\{U_n \mid n \geq 0\}$ を固定して考える。

ただし、 $\delta(\Delta, k) = \theta_0 \dots \theta_l$ に対し、 $|\delta(\Delta, k)| = l$ 、 $\delta(\Delta, k)_j = \theta_j$ ($0 \leq j \leq |\delta(\Delta, k)|$) と定義する。

一方、関数 φ は、 $\Delta \in \text{dom}(\kappa)$ に対して、式の集合 $\mathcal{E}(\Delta)$ の適当な要素を割り当てる。 $\mathcal{E}(\Delta)$ を定義するために、2 階の変数の集合 V 、 $k \in \omega$ に対して、 $\mathcal{E}(\Delta)_V^k$ をまず定義することにしよう。 $\mathcal{E}(\Delta)_V^k$ の要素は、

$$\begin{aligned} \mu P.(t, t'). \exists \theta(s_1, \dots, s_n) \exists \theta(s'_1, \dots, s'_n). \\ t \rightarrow \theta(s_1, \dots, s_n) \& t' \rightarrow \theta(s'_1, \dots, s'_n) \\ \& [\theta = \delta(\Delta, k)_0 \& A_0] \vee \dots \vee \\ [\theta = \delta(\Delta, k)_{|\delta(\Delta, k)|} \& A_{|\delta(\Delta, k)|}] \end{aligned}$$

という形で与えられる。とくに、 $\mu P.(t, t')$ の後続く部分を本体と呼ぶことにする。ここで、 t および t' は変数、 $\theta(s_1, \dots, s_n)$ および $\theta(s'_1, \dots, s'_n)$ は正規的な単純項スキームである。また、各部分論理式 A_l ($0 \leq l \leq |\delta(\Delta, k)|$) は、 $\mu P.(s_i, s'_i) \cdot \Phi \in \mathcal{E}(\Delta)_{V \cup \{s_1, \dots, s_n, s'_1, \dots, s'_n\} - \{s_i, s'_i\}}^{k+1}$ の本体 Φ (ただし $1 \leq i \leq n$) または $\mathcal{E}(\Delta)_V^k$ 式であるとする。 $\mathcal{E}(\Delta)_V^k$ 式は、次のように帰納的に定義される $\mathcal{E}(\Delta)_V^k$ 項:

1. 項は $\mathcal{E}(\Delta)_V^k$ 項である
2. $s \in \{s_1, \dots, s_n, s'_1, \dots, s'_n\} \cup V$ がアリティ m の 2 階の変数で、 $a_1, \dots, a_m$ が $\mathcal{E}(\Delta)_V^k$ 項ならば、 $s[a_1, \dots, a_m]$ は $\mathcal{E}(\Delta)_V^k$ 項である
3. $a_1, \dots, a_{j-1}$ が $\mathcal{E}(\Delta)_V^k$ 項ならば、 $t_{i,j,1}[a_1, \dots, a_{j-1}], \dots, t_{i,j,\kappa(\Delta)_{i,j}}[a_1, \dots, a_{j-1}]$ (ただし $1 \leq i \leq |\kappa(\Delta)|$ 、 $1 \leq j \leq \kappa(\Delta)_i$) はすべて $\mathcal{E}(\Delta)_V^k$ 項である

をもとに

1. a_1, a_2 が $\mathcal{E}(\Delta)_V^k$ 項ならば、 $P(a_1, a_2)$ は $\mathcal{E}(\Delta)_V^k$ 式である
2. $a_1, \dots, a_{j+1}$ が $\mathcal{E}(\Delta)_V^k$ 項ならば、 $\gamma_{i,j}(a_1, \dots, a_{j+1})$ (ただし $1 \leq i \leq |\kappa(\Delta)|$ 、 $1 \leq j \leq \kappa(\Delta)_i$) は $\mathcal{E}(\Delta)_V^k$ 式である
3. A, B が $\mathcal{E}(\Delta)_V^k$ 式ならば、 $A \& B$ 、 $A \vee B$ 、 $A \Rightarrow B$ は $\mathcal{E}(\Delta)_V^k$ 式である
4. a が変数で、 A が $\mathcal{E}(\Delta)_V^k$ 式ならば、 $\forall a.A$ 、 $\exists a.A$ は $\mathcal{E}(\Delta)_V^k$ 式である

と定義され、かつ次の条件を満たすものとする:

1. 自由な変数の出現はない (ただし、 $\{s_1, \dots, s_n, s'_1, \dots, s'_n\} \cup V$ に属する、アリティ 0 の 2 階の変数については、その出現を許す)
2. P の出現は、すべて厳密な意味で陽 (strictly positive)⁶ である。

$\mathcal{E}(\Delta) = \mathcal{E}(\Delta)_\emptyset^0$ とする。われわれは、 $\mathcal{E}(\Delta)$ に属する式 $\mu P.(t, t') \cdot \Phi[P, t, t']$ を、

$$P(t, t') \Leftarrow \Phi[P, t, t']$$

によって再帰的に定義される最小の述語 P として解釈する。

⁶たとえば [14] を参照せよ。

3 メソッド計算系の拡張

ITT [9] における基本的な型を網羅したメソッド計算系 $C_0 = (O_0, K_0, \alpha_0, R_0, \kappa_0, \delta_0, \varphi_0)$ を、次のように定義する。

$$\begin{aligned} O_0 \ni N_n, m_n (m \in \{0, 1, \dots, n-1\}), R_n, \\ N, 0, S, R, I, r, J, +, i, j, D, \\ \Pi, \lambda, \text{Ap}, \Sigma, (\cdot, \cdot), E, W, \text{sup}, \text{Tr}, U_n \end{aligned}$$

$$\begin{aligned} K_0 \ni N_n, m_n (m \in \{0, 1, \dots, n-1\}), \\ N, 0, S, I, r, +, i, j, \Pi, \lambda, \Sigma, (\cdot, \cdot), W, \text{sup}, U_n \end{aligned}$$

$$\begin{aligned} () &= \alpha_0(N_n) = \alpha_0(m_n) (m \in \{0, 1, \dots, n-1\}) \\ &= \alpha_0(N) = \alpha_0(0) = \alpha_0(r) = \alpha_0(U_n), \\ (0) &= \alpha_0(S) = \alpha_0(i) = \alpha_0(j), \\ (1) &= \alpha_0(\lambda), \\ (0, 0) &= \alpha_0(\text{Ap}) = \alpha_0((\cdot, \cdot)) \\ &= \alpha_0(J) = \alpha_0(+), \alpha_0(\text{sup}), \\ (0, 1) &= \alpha_0(\Pi) = \alpha_0(\Sigma) = \alpha_0(W), \\ (0, 2) &= \alpha_0(E), \\ (0, 3) &= \alpha_0(\text{Tr}), \\ (0, 0, 0) &= \alpha_0(I), \\ (0, 0, 2) &= \alpha_0(R), \\ (0, 1, 1) &= \alpha_0(D), \\ (0, \underbrace{0, \dots, 0}_n) &= \alpha_0(R_n) \end{aligned}$$

R_0 は次の評価規則からなる。

$$\begin{aligned} \bullet R_n(c, c_0, \dots, c_{n-1}) \rightarrow d &\Leftarrow c \rightarrow m_n \& c_m \rightarrow d \\ &\text{(ただし } m \in \{0, 1, \dots, n-1\}) \\ \bullet R(c, d, e) \rightarrow f &\Leftarrow c \rightarrow 0 \& d \rightarrow f \\ \bullet R(c, d, e) \rightarrow f &\Leftarrow c \rightarrow S(a) \& e[a, R(a, d, e)] \rightarrow f \\ \bullet J(c, d) \rightarrow e &\Leftarrow c \rightarrow r \& d \rightarrow e \\ \bullet D(c, d, e) \rightarrow f &\Leftarrow c \rightarrow i(a) \& d[a] \rightarrow f \\ \bullet D(c, d, e) \rightarrow f &\Leftarrow c \rightarrow j(b) \& e[b] \rightarrow f \\ \bullet \text{Ap}(c, a) \rightarrow d &\Leftarrow c \rightarrow \lambda(b) \& b[a] \rightarrow d \\ \bullet E(c, d) \rightarrow e &\Leftarrow c \rightarrow (a, b) \& d[a, b] \rightarrow e \\ \bullet \text{Tr}(c, d) \rightarrow e &\Leftarrow c \rightarrow \text{sup}(a, b) \\ &\& d[a, b, \lambda(v. \text{Tr}(\text{Ap}(b, v), d))] \rightarrow e \end{aligned}$$

$$\begin{aligned} N_n, N, I, +, \Pi, \Sigma, W &\in \text{dom}(\kappa_0) \\ \kappa_0(N_n) &= \kappa_0(N) = \varepsilon, \\ \kappa_0(I) &= 1 \square \square, \\ \kappa_0(+), \kappa_0(\Pi) &= 11, \\ \kappa_0(\Sigma) &= \kappa_0(W) = 12 \end{aligned}$$

$$\begin{aligned} \delta_0(N_n, i) &= 0_n 1_n \dots n-1_n \quad (i \geq 0), \\ \delta_0(N, i) &= 0S \quad (i \geq 0), \\ \delta_0(I, i) &= r \quad (i \geq 0), \\ \delta_0(+, i) &= ij \quad (i \geq 0), \\ \delta_0(\Pi, i) &= \lambda \quad (i \geq 0), \\ \delta_0(\Sigma, i) &= (\cdot, \cdot) \quad (i \geq 0), \\ \delta_0(W, 0) &= \text{sup}, \\ \delta_0(W, i) &= \lambda \quad (i \geq 1) \end{aligned}$$

$\varphi_0(N_n), \varphi_0(N), \varphi_0(I), \varphi_0(+), \varphi_0(\Pi), \varphi_0(\Sigma), \varphi_0(W)$ は、それぞれ次のように与えられる。

$$\begin{aligned}
& \mu P.(t, t'). \exists \theta(s_1, \dots, s_n) \exists \theta(s'_1, \dots, s'_n). \\
& \quad t \rightarrow \theta(s_1, \dots, s_n) \& t' \rightarrow \theta(s'_1, \dots, s'_n) \\
& \quad \& [\theta = \delta_0(N_n, 0)_0] \vee \dots \vee \\
& \quad [\theta = \delta_0(N_n, 0)_{n-1}] \\
& \mu P.(t, t'). \exists \theta(s_1, \dots, s_n) \exists \theta(s'_1, \dots, s'_n). \\
& \quad t \rightarrow \theta(s_1, \dots, s_n) \& t' \rightarrow \theta(s'_1, \dots, s'_n) \\
& \quad \& [\theta = \delta_0(N, 0)_0] \vee \\
& \quad [\theta = \delta_0(N, 0)_1 \& P(s_1, s'_1)] \\
& \mu P.(t, t'). \exists \theta(s_1, \dots, s_n) \exists \theta(s'_1, \dots, s'_n). \\
& \quad t \rightarrow \theta(s_1, \dots, s_n) \& t' \rightarrow \theta(s'_1, \dots, s'_n) \\
& \quad \& [\theta = \delta_0(I, 0)_0 \& \gamma_{1,1}(t_{1,1,1}, t_{1,1,2})] \\
& \mu P.(t, t'). \exists \theta(s_1, \dots, s_n) \exists \theta(s'_1, \dots, s'_n). \\
& \quad t \rightarrow \theta(s_1, \dots, s_n) \& t' \rightarrow \theta(s'_1, \dots, s'_n) \\
& \quad \& [\theta = \delta_0(+, 0)_0 \& \gamma_{1,1}(s_1, s'_1)] \vee \\
& \quad [\theta = \delta_0(+, 0)_1 \& \gamma_{2,1}(s_1, s'_1)] \\
& \mu P.(t, t'). \exists \theta(s_1, \dots, s_n) \exists \theta(s'_1, \dots, s'_n). \\
& \quad t \rightarrow \theta(s_1, \dots, s_n) \& t' \rightarrow \theta(s'_1, \dots, s'_n) \\
& \quad \& [\theta = \delta_0(\Pi, 0)_0 \\
& \quad \& [\forall a a'. \gamma_{1,1}(a, a') \\
& \quad \Rightarrow \gamma_{1,2}(a, s_1[a], s'_1[a'])]] \\
& \mu P.(t, t'). \exists \theta(s_1, \dots, s_n) \exists \theta(s'_1, \dots, s'_n). \\
& \quad t \rightarrow \theta(s_1, \dots, s_n) \& t' \rightarrow \theta(s'_1, \dots, s'_n) \\
& \quad \& [\theta = \delta_0(\Sigma, 0)_0 \\
& \quad \& \gamma_{1,1}(s_1, s'_1) \& \gamma_{1,2}(s_1, s_2, s'_2)] \\
& \mu P.(t, t'). \exists \theta(s_1, \dots, s_n) \exists \theta(s'_1, \dots, s'_n). \\
& \quad t \rightarrow \theta(s_1, \dots, s_n) \& t' \rightarrow \theta(s'_1, \dots, s'_n) \\
& \quad \& [\theta = \delta_0(W, 0)_0 \\
& \quad \& \exists \eta(v_1, \dots, v_m) \exists \eta(v'_1, \dots, v'_m). \\
& \quad \quad s_2 \rightarrow \eta(v_1, \dots, v_m) \\
& \quad \quad \& s'_2 \rightarrow \eta(v'_1, \dots, v'_m) \\
& \quad \quad \& [\eta = \delta_0(W, 1)_0 \\
& \quad \quad \& \gamma_{1,1}(s_1, s'_1) \\
& \quad \quad \& [\forall u u'. \gamma_{1,2}(s_1, u, u') \\
& \quad \quad \Rightarrow P(v_1[u], v'_1[u'])]]]
\end{aligned}$$

補題 1 C_0 は決定的メソッド計算系である。

メソッド計算系間に半順序 $\sqsubseteq$ を定義する。メソッド計算系 $C = (O, K, \alpha, R, \kappa, \delta, \varphi)$ および $C' = (O', K', \alpha', R', \kappa', \delta', \varphi')$ に対して、次の条件が成り立つとき、 $C \sqsubseteq C'$ であるとする。

$$\begin{aligned}
& O \subset O' \\
& K \subset K' \\
& (K' - K) \cap O = \emptyset \\
& \alpha = \alpha' \upharpoonright O \\
& \forall r [r \in R' \& o(r) \in O \Rightarrow r \in R] \\
& (\text{ただし、} r \text{ が } \rho(i) \rightarrow b \Leftarrow a_1 \rightarrow b_1 \& \dots \& a_n \rightarrow b_n \\
& \text{のとき } o(r) = \rho) \\
& \kappa = \kappa' \upharpoonright \text{dom}(\kappa) \\
& \delta = \delta' \upharpoonright \text{dom}(\kappa) \times \omega \\
& \bigcup \{ \delta(\Delta, 0)_j \mid 0 \leq j \leq |\delta(\Delta, 0)| \} \cap \text{dom}(\kappa') = \emptyset \\
& \Delta \in \text{dom}(\kappa) \\
& \varphi = \varphi' \upharpoonright \text{dom}(\kappa)
\end{aligned}$$

ただし、関数 $f: X \rightarrow Y$ と $X' \subset X$ に対し、 $f \upharpoonright X'$ は f の X' への制限を表す。 $C \sqsubseteq C'$ のとき、 C' は

C の拡張であるという。定義から明らかなように、 C の拡張 C' は、 C のもつ“意味”を変えない。

拡張の例を示す。リスト型を導入することで形成される C_0 の拡張 $C'_0 = (O'_0, K'_0, \alpha'_0, R'_0, \kappa'_0, \delta'_0, \varphi'_0)$ は、次のように定義される。

$$O'_0 = O_0 \cup \{\text{List}, \text{nil}, (\cdot), \text{listrec}\}$$

$$K'_0 = K_0 \cup \{\text{List}, \text{nil}, (\cdot)\}$$

α'_0 の定義は、 α_0 の定義に次の節を加えたものである。

$$\begin{aligned}
& () = \alpha'_0(\text{nil}), \\
& (0) = \alpha'_0(\text{List}), \\
& (0, 0) = \alpha'_0((\cdot)), \\
& (0, 0, 3) = \alpha'_0(\text{listrec})
\end{aligned}$$

R'_0 は、 R_0 に次の評価規則を加えたものである。

$$\begin{aligned}
& \bullet \text{listrec}(c, d, e) \rightarrow f \Leftarrow c \rightarrow \text{nil} \& d \rightarrow f \\
& \bullet \text{listrec}(c, d, e) \rightarrow f \\
& \quad \Leftarrow c \rightarrow (a.b) \& e \llbracket a, b, \text{listrec}(b, d, e) \rrbracket \rightarrow f
\end{aligned}$$

κ'_0 の定義は、 κ_0 の定義に次の節を加えたものである。

$$\begin{aligned}
& \text{List} \in \text{dom}(\kappa'_0) \\
& \kappa'_0(\text{List}) = 1
\end{aligned}$$

δ'_0 の定義は、 δ_0 の定義に次の節を加えたものである。

$$\delta'_0(\text{List}, i) = \text{nil}(\cdot) \quad (i \geq 0)$$

φ'_0 の定義は、次に示す $\varphi'_0(\text{List})$ を φ_0 の定義に加えたものである。

$$\begin{aligned}
& \mu P.(t, t'). \exists \theta(s_1, \dots, s_n) \exists \theta(s'_1, \dots, s'_n). \\
& \quad t \rightarrow \theta(s_1, \dots, s_n) \& t' \rightarrow \theta(s'_1, \dots, s'_n) \\
& \quad \& [\theta = \delta'_0(\text{List}, 0)_0] \vee \\
& \quad [\theta = \delta'_0(\text{List}, 0)_1 \& \gamma_{1,1}(s_1, s'_1) \\
& \quad \& P(s_2, s'_2)]
\end{aligned}$$

決定的な C'_0 に対し、その拡張 $C''_0 = (O''_0, K''_0, \alpha''_0, R''_0, \kappa''_0, \delta''_0, \varphi''_0)$ は非決定的になる。 C''_0 は、 C'_0 の定義に次の項目を新たに加えることによって定義される。

$$O''_0 = O'_0 \cup \{\text{amb}\}$$

$$(0, 0) = \alpha''_0(\text{amb})$$

$$\begin{aligned}
& \bullet \text{amb}(a, b) \rightarrow c \Leftarrow a \rightarrow c \\
& \bullet \text{amb}(a, b) \rightarrow c \Leftarrow b \rightarrow c
\end{aligned}$$

4 メソッド計算系上の型システム

ITT のモデルは、計算システム上の型システムによって提供された[2]。しかし、この型システムの定義は、特定の基本的型構成子に依存しており、いわば閉じた構成法によるものであった。型についての開放性をとらえるためには、これを開いた定義に発展させなければならない。その実現が、メソッド計算系上の型システムである。

メソッド計算系 $C = (O, K, \alpha, R, \kappa, \delta, \varphi)$ をひとつ固定する。今後われわれは、型システム (type system) と呼ばれる 2 項関係

$$\tau \in \widehat{T}(C) \times 2^{\widehat{T}(C) \times \widehat{T}(C)}$$

の帰納的構成をすすめていく。型システム τ は、ITT のモデルとして提供されるために、いくつかの性質を満たしている必要がある。

$$\begin{aligned}\text{Fun}(\tau) &\Leftrightarrow \forall T \phi \phi' [\tau T \phi \& \tau T \phi' \Rightarrow \phi = \phi']^7 \\ \text{TyVal}(\tau) &\Leftrightarrow \forall T \phi [\tau T \phi \Leftrightarrow \exists T'. T \rightarrow_C T' \& \tau T' \phi] \\ \text{TrSy}(\tau) &\Leftrightarrow \forall T \phi [\tau T \phi \Rightarrow \phi \text{ は対称的かつ推移的}] \\ \text{Val}(\tau) &\Leftrightarrow \forall T \phi [\tau T \phi \Rightarrow \forall ts [t \phi s \Leftrightarrow \exists t'. t \rightarrow_C t' \& t' \phi s]]\end{aligned}$$

型システム τ が上記 4 条件をともにみたすとき、 τ は外延的 (extensional) であるといい、ETS(τ) と書く。

われわれがめざす型システムの帰納的構成は、型システム上の 2 変数関数 I^C を用いて行なわれる。実際、目標とする型システムは、 $I^C(\sigma, \cdot)$ の最小不動点 (σ は適当な基底) として実現されるであろう。 $I^C(\sigma, \tau)$ は、“ σ 内の型” と “ τ 内の型から C の型構成子 (ただしユニバースを除く) によって 1 ステップで構成される型” からなる型システムとして、次のように定義される。

$$I^C(\sigma, \tau) T \phi \Leftrightarrow \sigma T \phi \vee K^C(\tau) T \phi$$

ただし、

$$\begin{aligned}K^C(\tau) T \phi &\Leftrightarrow \exists \Delta(\Omega) \exists \Gamma. T \rightarrow_C \Delta(\Omega) \\ &\quad \& \Delta \in \text{dom}(\kappa) \\ &\quad \& \text{Fam}_\Delta(\tau, \Omega, \Gamma) \\ &\quad \& \forall tt' [t \phi t' \Leftrightarrow [\varphi(\Delta)](t, t')]\end{aligned}$$

$$\Omega = (A_{i,j}, t_{i,j,1}, \dots, t_{i,j,\kappa(\Delta)_{i,j}})_{\substack{1 \leq i \leq |\kappa(\Delta)| \\ 1 \leq j \leq \kappa(\Delta)_i}}$$

$$\Gamma = (\gamma_{i,j})_{\substack{1 \leq i \leq |\kappa(\Delta)| \\ 1 \leq j \leq \kappa(\Delta)_i}}$$

$$\text{Fam}_\Delta(\tau, \Omega, \Gamma) = \bigotimes_{\substack{1 \leq i \leq |\kappa(\Delta)| \\ 1 \leq j \leq \kappa(\Delta)_i}}$$

$$\left[\begin{aligned} &\forall a_1 a'_1 \dots \forall a_{j-1} a'_{j-1}. \\ &\gamma_{i,1}(a_1, a'_1) \& \dots \& \gamma_{i,j-1}(\bar{a}, a_{j-1}, a'_{j-1}) \\ &\Rightarrow \tau A_{i,j}[\bar{a}, a_{j-1}](\lambda uv'. \gamma_{i,j}(\bar{a}, a_{j-1}, u, u')) \\ &\& \tau A_{i,j}[\bar{a}, a'_{j-1}](\lambda uv'. \gamma_{i,j}(\bar{a}, a'_{j-1}, u, u')) \\ &\& \forall uu' [\gamma_{i,j}(\bar{a}, a_{j-1}, u, u') \Leftrightarrow \gamma_{i,j}(\bar{a}, a'_{j-1}, u, u')] \\ &\& \gamma_{i,j}(\bar{a}, a_{j-1}, t_{i,j,1}[\bar{a}, a_{j-1}], t_{i,j,1}[\bar{a}, a'_{j-1}]) \\ &\dots \\ &\& \gamma_{i,j}(\bar{a}, a_{j-1}, t_{i,j,\kappa(\Delta)_{i,j}}[\bar{a}, a_{j-1}], t_{i,j,\kappa(\Delta)_{i,j}}[\bar{a}, a'_{j-1}]) \end{aligned} \right]$$

(ここで、 $\bigotimes$ は式の連言を、 $\bar{a}$ は $a_1, \dots, a_{j-2}$ の省略を、 λ はメタな関数抽象を表す) とする。また、 $[\varphi(\Delta)]$ は、 $\hat{T}(C)$ を領域とした $\varphi(\Delta)$ の標準的な解釈を表す。ただし、述語記号 $\gamma_{i,j}$ と関数記号 $t_{i,j,1}, \dots, t_{i,j,\kappa(\Delta)_{i,j}}$ (ただし $1 \leq i \leq |\kappa(\Delta)|$ 、 $1 \leq j \leq \kappa(\Delta)_i$) は、それぞれ Γ および Ω 中の対応する要素として解釈する。また、 $\rightarrow$ は $\rightarrow_C$ として解釈する。

定義から明らかのように、 $I^C(\sigma, \tau)$ は τ について単調な (monotone) 関数である。したがって、

$$\mu^C(\sigma) T \phi \Leftrightarrow \forall \tau [\forall S \psi [I^C(\sigma, \tau) S \psi \Rightarrow \tau S \psi] \Rightarrow \tau T \phi]$$

と定義される $\mu^C(\sigma)$ が $I^C(\sigma, \cdot)$ の最小不動点になることが保証される。換言すれば、 $\mu^C(\sigma)$ の帰納的定義は整合的である。

⁷この性質は、 τ が $\hat{T}(C)$ から $2^{\hat{T}(C) \times \hat{T}(C)}$ への部分関数であることを主張する。

メソッド計算系 C から構成される型システム $\mathcal{M}^C$ を次のように定義する:

$$\begin{aligned}\sigma_n^C &= \mu^C(\text{spine}_n^C) \\ \text{spine}_n^C &= \{(T, =_{\sigma_n^C}) \mid 0 \leq m < n \& T \rightarrow_C \cup_m \\ &\quad \& \forall S S' [S =_{\sigma_n^C} S' \Leftrightarrow \exists \phi. \sigma_m^C S \phi \& \sigma_m^C S' \phi]\} \\ \mathcal{M}^C &= \mu^C \left(\bigcup_{n \geq 0} \text{spine}_n^C \right)\end{aligned}$$

$\text{TrSy}(\tau)$ ならば $\text{TrSy}(\mathcal{M}^C(\tau))$ が任意の型システム τ について成り立つとき、メソッド計算系 C は正則 (regular) であるという。

補題 2 C を任意のメソッド計算系とする。 C が決定的かつ正則ならば、任意の n について $\text{ETS}(\sigma_n^C)$ かつ $\text{ETS}(\mathcal{M}^C)$ が成り立つ。

証明 σ_n^C と $\mathcal{M}^C$ の構成にかんする帰納法による。 ■

補題 3 C を任意のメソッド計算系とする。任意の n について、 $\sigma_n^C \subseteq \sigma_{n+1}^C$ かつ $\sigma_n^C \subseteq \mathcal{M}^C$ が成り立つ。

証明 $\sigma \subseteq \sigma'$ ならば $\mu^C(\sigma) \subseteq \mu^C(\sigma')$ であることから明らか。 ■

$\mathcal{M}^C$ は ITT のセマンティクスの数学的表現を提供する。実際、 $x_1, \dots, x_n$ を変数、 $T, T', t, t', T_1, \dots, T_n$ を C_0 におけるアリティ $n, n, n, n, 0, \dots, n-1$ の閉じた 2 階の項、 Ψ を

$$x_1 \in T_1, \dots, x_n \in T_n \llbracket x_1, \dots, x_{n-1} \rrbracket$$

としたとき

$$\begin{aligned}T \llbracket x_1, \dots, x_n \rrbracket &\text{ type} & (\Psi) \\ T \llbracket x_1, \dots, x_n \rrbracket &= T' \llbracket x_1, \dots, x_n \rrbracket & (\Psi) \\ t \llbracket x_1, \dots, x_n \rrbracket &\in T \llbracket x_1, \dots, x_n \rrbracket & (\Psi) \\ t \llbracket x_1, \dots, x_n \rrbracket &= t' \llbracket x_1, \dots, x_n \rrbracket \in T \llbracket x_1, \dots, x_n \rrbracket & (\Psi)\end{aligned}$$

と定義される C_0 の判定式 (statement、以後単に判定式) Θ に対し、 $\mathcal{M}^C \models \Theta$ 、すなわち $\mathcal{M}^C$ における Θ の充足可能性を、判定式の仮定の数にかんする帰納法で自然に定義することができる。とくに、仮定がない場合は、

$$\begin{aligned}\mathcal{M}^C \models T \text{ type} &\Leftrightarrow \exists \phi. \mathcal{M}^C T \phi \\ \mathcal{M}^C \models T = T' &\Leftrightarrow \exists \phi. \mathcal{M}^C T \phi \& \mathcal{M}^C T' \phi \\ \mathcal{M}^C \models t \in T &\Leftrightarrow \exists \phi. \mathcal{M}^C T \phi \& t \phi t \\ \mathcal{M}^C \models t = t' \in T &\Leftrightarrow \exists \phi. \mathcal{M}^C T \phi \& t \phi t'\end{aligned}$$

となる。

ITT [9] を形式化して得られる体系⁸において、判定式 Θ が導出可能とき、ITT $\vdash \Theta$ と書くことにする。

定理 4 [ITT における開放性] Θ を ITT $\vdash \Theta$ を満たす任意の判定式とする。このとき、任意のメソッド計算系 $C \supseteq C_0$ に対し、 C が決定的かつ正則⁹ならば、 $\mathcal{M}^C \models \Theta$ が成り立つ。

証明 Θ の導出の長さにかんする帰納法による。 ■

⁸ C_0 を言語として定義される。たとえば [14] を参照せよ。

⁹Martin-Löf によれば、正規的オブジェクト間の等号が常に反射的、対称的、推移的でないといけないという条件を除いて、型を定義する規定に制限はない。メソッド計算系の正則性はこの条件を表現したものである。

メソッド計算系上の型システムの定義は、Martin-Löfによって与えられたセマンティクスを自然に反映させつつ、その開放性を明示的に記述したものと考える。Martin-Löfにしたがって型を定義するには、正規的オブジェクトはどのように構成されるのか、2つの正規的オブジェクトはいつ等しくなるのか、この2点を規定することが必要である。それゆえこのセマンティクスは、 U_n typeを主張する際、固定された型構成子に限定してユニバースを形成することを強要する。その代償は、ユニバース上で定義される関数の導入—ユニバース消去規則—であるが、[9]はこの消去規則を含んでいない。なぜなら、真にユニバースを固定して型についての開放性を放棄するかわりに、背景に潜む言語を各時点で考慮し、その都度言語内の型構成子をもとにユニバースを形成すると考えれば、先ほどのセマンティクスを維持しつつ開放性を実現できるからである。われわれのメソッド計算系とそれ上の型システムは、このアイデアを具現化したものである。実際、定理4は、“メソッド計算系の拡張列 $\{C_i\}$ を添字集合とする型システムの族 $\{M^{C_i}\}$ ”をモデルとして、ITT [9]の推論規則が体系化されたことを物語っている。

5 おわりに

本論文は、[2] [6] [7] [3]で展開された枠組みを基盤にそれを発展させ、重要な未解決問題である型についての開放性に対しひとつの定式化を与えたものである。実際、適当な条件のもとで、任意のメソッド計算系 C から構成される型システム M^C が、ITTを形式化した体系のモデルになることが示された。

固定された言語ではなく、言語とその意味の拡張列についての分析が、[12] [1]に見られる。しかし、言語の拡張に伴う意味の拡張手続きが“effective”には与えられていない。本論文における定式化はこの問題を解決した。

われわれの枠組みは、型の導入を一般的に扱う手法を必要とする。この点で[5] [13] [8]は参考になるように思われる。しかし、各オペレータに型をつけることから議論を開始するこれらの手法が、より広範な言語を扱うわれわれの枠組みに適用できるかどうかは不明である。

本論文で展開された議論は、ITT—メソッドというプリミティブな概念をもとに構築された理論—における開放性が、

- メソッドの評価の遅延性
- メソッドの評価の決定性
- 外延性
- 述語性

といったさまざまな条件によって実現されていることを浮き彫りにしている。新たな開放的理論を構築するために、これらの条件を整理することは、今後の課題でもある。

謝辞

本研究を進めるにあたり、NTT基礎研究所の勝野裕文氏、小川瑞史氏、NTTソフトウェア研究所の後藤滋樹博士、電子技術総合研究所の木下佳樹博士、北陸先端科学技術大学院大学の白須裕之氏ほか多数の方々のご協力をいただきました。ここに感謝いたします。

参考文献

- [1] P. Aczel, D. P. Carlisle, and N. Mendler. Two frameworks of theories and their implementation in Isabelle. In G. Huet and G. Plotkin, editors, *Logical Frameworks*, pp. 3–39. Cambridge University Press, 1991.
- [2] S. F. Allen. A non-type-theoretic definition of Martin-Löf's types. In *Proceedings of the Second Annual Symposium on Logic in Computer Science*, pp. 215–221. IEEE, 1987.
- [3] D. A. Basin and D. J. Howe. Some normalization properties of Martin-Löf's type theory, and applications. In *Proceedings of the First International Conference on Theoretical Aspects of Computer Software*, volume 526 of *Lecture Notes in Computer Science*, pp. 475–494. Springer-Verlag, 1991.
- [4] R. L. Constable, S. F. Allen, H. M. Bromley, W. R. Cleaveland, J. F. Cremer, R. W. Harper, D. J. Howe, T. B. Knoblock, N. P. Mendler, P. Panangaden, J. T. Sasaki, and S. F. Smith. *Implementing Mathematics with the Nuprl Proof Development System*. Prentice-Hall, 1986.
- [5] P. Dybjer. Inductive sets and families in Martin-Löf's type theory and their set-theoretic semantics. In G. Huet and G. Plotkin, editors, *Logical Frameworks*, pp. 280–306. Cambridge University Press, 1991.
- [6] D. J. Howe. Equality in lazy computation systems. In *Proceedings of the Fourth Annual Symposium on Logic in Computer Science*, pp. 198–203. IEEE, 1989.
- [7] D. J. Howe. On computational open-endedness in Martin-Löf's type theory. In *Proceedings of the Sixth Annual Symposium on Logic in Computer Science*, pp. 162–172. IEEE, 1991.
- [8] 木下佳樹. Martin-Löfの集合定数の導入と λ -cube. 日本ソフトウェア科学会プログラム合成変換研究会予稿集, 1992.
- [9] P. Martin-Löf. Constructive mathematics and computer programming. In L. J. Cohen, J. Los, H. Pfeiffer, and K. P. Podewsky, editors, *Logic, Methodology and Philosophy of Science*, pp. 153–175. North-Holland, 1982.
- [10] P. Martin-Löf. *Intuitionistic Type Theory*. Bibliopolis, 1984.
- [11] P. Martin-Löf. Truth of a proposition, evidence of a judgement, validity of a proof. *Synthese*, 73:407–420, 1987.
- [12] P. F. Mendler and P. Aczel. The notion of a framework and a framework for LTC. In *Proceedings of the Third Annual Symposium on Logic in Computer Science*, pp. 392–399. IEEE, 1988.
- [13] B. Nordström, K. Petersson, and J. M. Smith. *Programming in Martin-Löf's Type Theory: An Introduction*. Oxford University Press, 1990.
- [14] A. S. Troelstra and D. van Dalen. *Constructivism in Mathematics*, volume I, II. North-Holland, 1988.