プログラミング-言語・基礎・実践- 21－10
(1995. 3. 9)

ＯＯＤＢのオブジェクトの論理中での型に対する高階論理からの一考察

海老原一郎
ebihara@etl.go.jp
電子技術総合研究所情報アーキテクチャ部
〒305 茨城県つくば市梅園 1-1-4

あらまし: 演繹型オブジェクト指向データベースの傾向として、一度オブジェクトを定義した後はそのオブジェクトＩＤの独立性を変更しないようにする仕様が大勢を占める。この仕様は高階論理風のオブジェクト指向の記述を一階述語論理で説明することを可能にしたが、オブジェクト同士の等価性を扱う際には使い勝手が良いシステムと言い切れない面を含んでいる。より柔軟な演繹型オブジェクト指向データベースを実現するためにはオブジェクトＩＤ間の等価性を宣言的に記述可能にすることが必要であると思われるが、そのためにはオブジェクトＩＤとメソッド・クラスの間で等価性を波及させる必要が生じる。その際、オブジェクト指向におけるオブジェクトはオブジェクトＩＤ・メソッド・クラスという三つの性質を同時に持つので、この三つの性質を統合して扱う必要が生じる。本論文では、データベースの基礎となる理論に Andrews による高階タイプ理論 $\mathcal{Q}_o$ を拡張した論理体系 $\mathcal{Q}_o^D$ を提案する。$\mathcal{Q}_o^D$ の上においては上述の三つの性質の間に必要とされる関係を簡潔な形で記述することが可能である。

和文キーワード: 演繹オブジェクト指向データベース、高階型理論、ブール代数

A Consideration of Types of an Object in Object-oriented Database from the Viewpoint of a Typed Higher-order Logic

Ichirou EBIHARA
Computer Science Division, Electrotechnical Laboratory
1-1-4 Umezono, Tsukuba, Ibaraki 305, JAPAN

Abstract: It has been proposed in field of object-oriented database systems that independence of an object-ID should not be changed after embodiment of the object. Which makes it able to explain the higher-order-logic-like description of object oriented database in first order logic,but which also makes it difficult to treat equality between objects. If we want to realize more flexible database systems, it is needed to treat declarations of equality between object-IDs, and it is needed to relate object-IDs to their methods and classes. In other words, when we try to use objects, we need to treat object-IDs , methods and classes at the same time. This paper proposes a typed higher-order logic $\mathcal{Q}_o^D$ which is an extension of Andrews' $\mathcal{Q}_o$. $\mathcal{Q}_o^D$ realizes relation of above three aspect of an object in brief description.

英文 **keywords:** Deductive object-oriented database, Typed higherorder logic, Boolean algebra

1 はじめに

演繹型オブジェクト指向データベースの傾向として、一度オブジェクトＩＤを定義した後はそのＩＤの独立性を変更しないようにする仕様が大勢を占める。これはこの分野の草分け的存在の O-logic・F-logic から採用されてきた方針であり [5, 6, 7, 8, 10, 11]、高階論理のように見えるオブジェクト指向の記述を一階述語論理で説明することを可能にしたが、オブジェクトＩＤやメソッドの等価性を考える際には使い勝手が良いシステムと言い切れない面を含んでいる。より柔軟な演繹型オブジェクト指向データベースを実現する際には、オブジェクトＩＤ間の等価性を宣言的に記述可能にすることが必要であると思われるが、そのためにはオブジェクトＩＤとメソッド・クラスの間で等価性を波及させる必要が生じる。つまり、一つのオブジェクトが同時に合わせ持つ別々の様相であるオブジェクトＩＤ・メソッド・集合という三つの性質を統合して扱う必要がある。従来のデータベースではこの三つの性質を統合して扱うことができなかったが、そのことを F-logic 風の記述をもとにして説明する。例として、日米で微生物が物質を分解する時の原料と生成物のデータベースを各々独立に作成したとする。

日本

$$\text{糖}[\text{酵母} \to \{alcohol\ ,\ carbon\ dioxide\}\ ,\ \text{大腸菌} \to \{methane\},\ ...]$$

$$\vdots$$

U.S.A.

$$sugar[yeast \to \{alcohol\}\ ,\ colitis\ germ \to \{methane\},\ ...]$$

$$\vdots$$

全てのデータを入力し終えた後で、二つのデータベースを統合することを考える。このとき、日本語の”糖”と英語の”sugar”が同じ概念を表すということが分かっても、各々に異なったオブジェクトＩＤを割り当ててあるので、データベース全体を書き直すか個々にルールを書く以外には”糖”と”sugar”に関する部分をデータベース上で統合することができない。また、上の記述で”酵母”と”yeast”に関する部分は普通の数式で表すと

$$\text{酵母 (糖)} \supset \{alcohol\ ,\ carbon\ dioxide\}$$
$$yeast(sugar) \supset \{alcohol\}$$

というように”酵母”と”yeast”は関数として働いているので、オブジェクトＩＤ同士の等価性が判明したとしても関数の等価性までは推論されない。その結果、人間が判断すれば”酵母”と”yeast”が同じものであるという事実から”yeat”は”carbon dioxide”も生成するということを判断できるが、従来の演繹データベースではこの推論ができない。

オブジェクトＩＤからメソッドやクラスの働きを推論するためには、この三つを常に統合して扱うことが必要とされる。そのためには、後に説明するように関数や述語を変数として扱えるような高階な論理が必要とされる。本研究では、Peter B. Andrews が提唱した $\mathcal{Q}_o$ を基礎に用いて上記の推論を行なうデータベースが実現可能であることを示す。

2 高階型理論 $\mathcal{Q}_0$

まず $\mathcal{Q}_o$ [1, 2, 3] の説明を簡単に行なう。$\mathcal{Q}_o$ では全ての変（定）数は型を持っていて、x_α のように記述される。この意味は、変数 x が型 α を持っているということを表している。型の構成法は以下の規則による。

(a) o は真理値の型を表す型記号である。
(b) ι は個体の型を表す型記号である。
(c) もし、α と β が型記号ならば、$(\alpha\beta)$ は β 型の要素から α 型の要素への関数の型を表す型記号である。

上記の型記号を使って、$\mathcal{Q}_o$ では関数も変（定）数として $x_{\alpha\beta}$ のように表現される。

$\mathcal{Q}_o$ で使用する記号は以下のものである。

(a) 一般的な数学記号：$[\ ,\ ]\ ,\ (\ ,\ )\ ,\ \lambda$
(b) 各型記号 α に対する α 型の変数の可算的リスト：$f_\alpha, g_\alpha, h_\alpha ... x_\alpha, y_\alpha, z_\alpha, f^1_\alpha, g^1_\alpha, h^1_\alpha ... x^1_\alpha, y^1_\alpha, z^1_\alpha, f^2_\alpha ...$
(c) 論理定数：α 型の要素間の恒等関係 $Q_{(o\alpha)\alpha}$ と 個体に対する選択演算子 $\iota_{\iota(o\iota)}$
(d) 様々な型に対する必要なだけの個数の個体定数。

これらの記号から α 型の整合論理式（wff_α と略記することにする。）を以下のように帰納的に定義する。

(a) α 型の原始定（変）数は wff_α である。
(b) $A_{\alpha\beta}, B_\beta$ が wff のとき、$[A_{\alpha\beta}B_\beta]$ は wff_α である。（ここで、$[A_{\alpha\beta}B_\beta]$ は B_β に関数 $A_{\alpha\beta}$ を作用させることを表す。）
(c) A_α が wff_α のとき、$[\lambda x_\beta A_\alpha]$ は $\mathrm{wff}_{\alpha\beta}$ である。

整合論理式を表す構文変数を A_α, B_α というように英大文字で表すことにすると $\mathcal{Q}_o$ の体系は以下の公理系で表せる。

(1) $g_{oo}T_o \wedge g_{oo}F_o = \forall x(g_{oo}x_o)$

(2^{α}) $(x_\alpha = y_\alpha) \supset (h_{o\alpha}x_\alpha = h_{o\alpha}y_\alpha)$
$(3^{\alpha\beta})$ $(f_{\alpha\beta} = g_{\alpha\beta}) = \forall x_\beta (f_{\alpha\beta}x_\beta = g_{\alpha\beta}x_\beta)$
(4_1) $[\lambda x_\alpha B_\beta]A_\alpha = B_\beta$、ここで B_β は原始定数または x_α と異なる変数である。
(4_2) $[\lambda x_\alpha x_\alpha]A_\alpha = A_\alpha$
(4_3) $[\lambda x_\alpha [B_{\beta\gamma}C_\gamma]]A_\alpha$
$= [[\lambda x_\alpha B_{\beta\gamma}]A_\alpha][[\lambda x_\alpha C_\gamma]A_\alpha]$
(4_4) $[\lambda x_\alpha [\lambda y_\gamma B_\beta]]A_\alpha = [\lambda y_\gamma [\lambda x_\alpha B_\beta]A_\alpha]$
ここで y_γ は x_α ではなく、かつ A_α の全ての変数とも異なる。
(4_5) $[\lambda x_\alpha [\lambda x_\alpha B_\gamma]]A_\alpha = [\lambda x_\alpha B_\gamma]$
(5) $\iota_{\iota(o\iota)}[Q_{o\iota\iota}y_\iota] = y_\iota$

(推論規則 R)

C における A_α の出現が λ の直後にくる変数の出現ではないとき、C および $Q_{((o)\alpha)\alpha}A_\alpha B_\alpha$ から C における A_α の一つの出現を、B_α の出現によって置き換えた結果を推論する。

また、$\mathcal{Q}_o$ では表記上の便宜のために以下の省略記号を用いる。

T_o : $Q_{o(((o)o)o)(((o)o)o)}Q_{((o)o)o}Q_{((o)o)o}$
命題論理の真理値の真に相当する。

$\wedge_{((o)o)o}$: $[\lambda x_o\, \lambda y_o\, ([\lambda g_{((o)o)o}\, (g_{((o)o)o}T_oT_o)] = [\lambda g_{((o)o)o}\, (g_{((o)o)o}x_oy_o)])]$
述語論理の $\wedge$ に相当する。

$\supset_{((o)o)o}$: $[\lambda x_o\, \lambda y_o\, (x_o = (x_o \wedge y_o))]$
述語論理の含意に相当する。

以上で準備が整ったので、実現しようとしている仮想的なデータベースの説明を行なう。

3 ＯＯＤＢ中でのオブジェクトの型

ＯＯＤＢの中のオブジェクトを眺めた時に、オブジェクトは三つの異なった性質を合わせ持つことは前にも述べた。その三つの性質はオブジェクトＩＤ・メソッド・クラスとしての性質であるが、これらを普通の数学の用語で表現すれば、オブジェクトＩＤはオブジェクト型の個体変（定）数として、メソッドはオブジェクト型からオブジェクト型への関数として、クラスはオブジェクト型の要素を持つ集合として表される。集合の定義の仕方を内包的に考えれば、α 型の要素を持つ集合は $\mathcal{Q}_o$ では $Set_{o\alpha}$ というように α 型から論理型への関数として表すことができる。$\mathcal{Q}_o$ でのオブジェクトを表す型を $_I$ で表すことにすると、a というオブジェクトが同時に合わせ持つ三つの様相は、a_I、$a_{(I)I}$、$a_{(o)I}$ となる。

一見したところ、$\mathcal{Q}_o$ は高階論理であるので一つの $_I$ 型の定数と二つの関数（$_{(I)I}$ 型と $_{(o)I}$ 型）を引数とする何か新しい三価の関数を定義して三つの引数 a_I、$a_{(I)I}$、$a_{(o)I}$ が同じオブジェクト a の三つの性質を表すならば真偽値の真を返すようにすれば、オブジェクトの三つの性質を関係づけられると思われるかも知れない。しかし、単純に三つの型の変数のみを関係づける関数を考えても、求めるデータベースは実現できない。例として、以下のような F-logic 風のデータベースを考える。

$bob[name \rightarrow'' Bob'',$
$\quad cooperation(george) \rightarrow ProjectA,$
$\quad work \rightarrow cs_1[dname \rightarrow \text{“}CS'',$
$\quad\quad mngr \rightarrow bob,\ assistants \rightarrow \{john\}]]$

これを $\mathcal{Q}_o$ の変数を用いて表すと、

$bob_{(o)I}\, name_I{=}T_o \wedge name_{(I)I}\, bob_I{=}''Bob''_I \wedge$
$bob_{(o)I}\, cooperation_I = T_o \wedge$
$cooperation_{((I)I)I}\, bob_I\, george_I = Project_I \wedge$
$bob_{(o)I}\, work_I = T_o \wedge work_{(I)I}\, bob_I = cs_{1I} \wedge$
$cs_{1(o)I}\, dname_I{=}T_o \wedge dname_{(I)I}\, cs_{1I}{=}''CS''_I \wedge$
$cs_{1(o)I}\, mngr_I = T_o \wedge mngr_{(I)I}\, cs_{1I} = bob_I \wedge$
$cs_{1(o)I}\, assistants_I = T_o \wedge$
$assistants_{((o)I)I}cs_{1I}john_I = T_o$

となる。上式の２行めは bob と $george$ が共同作業として $ProjectA$ に従事していることを表しているといった意味にとることができる。もし、ここで

$$cooperation_I = \text{共同作業}_I$$

というデータが他で定義されていたとしたら、上式の２行めより

$$\text{共同作業}_{((I)I)I}\, george_I\, bob_I = ProjectA_I$$

と推論することができるだろうか。$cooperation$ の型は $_{((I)I)I}$ であるので、$_I$ 型と $_{(I)I}$ 型と $_{(o)I}$ 型の間でのみ関係付けられたデータベースでは推論できない。同じことは６行めにも言えて、$assistants_{((o)I)I}$ の型は $_I$ でも $_{(I)I}$ でも $_{(o)I}$ でもない。F-logic の文法では上記の $cooperation$ や $assistants$ に相当する部分は更に価数の多い関数を用いることができるので、必要な推論を行なうためには全ての価数についてその引数の型の間での関係づけが必要になる。上記の他に、複合オブジェクトやクラスの包含関係等 F-logic の文法から要請される全ての型を満たそうとすると、結局

$$_I,\ _{(I)I},\ _{((I)I)I},\ _{(((I)I)I)I}, \cdots$$
$$_{(o)I},\ _{((o)I)I},\ _{(((o)I)I)I},\ _{((((o)I)I)I)I}, \cdots \quad (1)$$

というような型全てについて何かの関係を持たせることが必要になる。また、F-logic では関数の構成法についてはあまり詳しく言及されてい

ないが、ここで仮定しているデータベースでは関数の定義は既定義の関数と変数を使って構成される$\mathcal{Q}_o$の全ての関数を扱う予定である。すると、(1)式以外のさらに多くの型についての関係付けが必要になる。例えばあるメソッドの中で二つのオブジェクトがあるオブジェクトを同時に含むか含まないかを判断する関数を定義して使ったとする。すると、この関数は例えば

$$involve_{(((o)I))((o)I))((o)I)}$$

というような型を持つはずである。もし、

$$involve_I = \text{包含}_I$$

という定義が他でなされているならば、

$$involve_{(((o)I))((o)I))((o)I)} = \text{包含}_{(((o)I))((o)I))((o)I)}$$

というような関係が成り立つべきであるが、型$_{(((o)I))((o)I))((o)I)}$は(1)式には現れない。つまり、あらかじめ考慮しておかなければならない型のリストは

$$I\,,\ (I)I\,,\ I((o)I)\,,\ ((o)I)I\,,\ ((I)I)I\,,\ (I)((I)I)\,,\ (((o)I)I)I\,,\ ((I((o)I))I\,,\cdots \quad (2)$$

というようになる。

以上のようにして、一つのオブジェクトに必要とされる型を列挙したが、それらは$_I$型をもとにして帰納的に定義される。

(ＤＴ１)型αの中の$_I$を$_{(I)I}$で置き換える。

(ＤＴ２)型αの中の$_I$を$_{(o)I}$で置き換える。

上記の規則によって、オブジェクト型$_I$を含む型から導かれた型を、もとの型の派生型と呼ぶことにする。

4　ＯＯＤＢの事実と導出

本研究で仮定しているオブジェクト指向データベースに事実を登録する際の式の型について説明する。ここで考えているオブジェクト指向データベースで扱う事実は、$\mathcal{Q}_o$の二つの項を等号で結んだ形の閉整合論理式である。等号で二つの項を結んだ形というのは厳しい制約のように見えるが、$\mathcal{Q}_o$では全ての定理X_oについて$\vdash X_o = T_o$が成り立つので、等号で二項を結んだ形の式のみを用いた表現力は$\mathcal{Q}_o$の表現力と同じである。オブジェクトに対して派生型を決めたのと同様に、データベースの事実に対しても派生式を以下のように定める。

（ＤＥ）基礎の式を$A_\alpha = B_\alpha$としたとき、A_αの中のＩを含んだ型を持ついくつかの変（定）数についてそれらの変（定）数の型をその派生型で置き換える。B_αについても同様な操作をしたとき、両辺が整合論理式を成し、両辺を適当な型の等号で結ぶことができるならば、その式を$A_\alpha = B_\alpha$の派生式と呼ぶ。

データベースの結論とは、登録されたデータの全派生式を$\mathcal{Q}_o$の仮説としたとき導出される全ての閉整合論理式とする。これを前出の例を用いて説明すると次のようになる。

$$cooperation_I = \text{共同作業}_I$$

という事実が記述されているならば、それからの派生式

$$cooperation_{(I)I} = \text{共同作業}_{(I)I}$$
$$cooperation_{(o)I} = \text{共同作業}_{(o)I}$$
$$cooperation_{((I)I)I} = \text{共同作業}_{((I)I)I}$$
$$\vdots$$

も$\mathcal{Q}_o$に手渡されるとする。こうしておけば

$$cooperation_{((I)I)I}\, bob_I\, georg_I = Project_I$$

という事実が登録されている時、

$$\text{共同作業}_{((I)I)I}\, bob_I\, georg_I = ProjectA_I$$

という事実が導ける。もちろん、

$$cooperation_{((I)I)I}\, bob_I\, georg_I = ProjectA_I$$

に対してもその派生式

$$cooperation_{((I)I)I}\, bob_I\, georg_I = ProjectA_I$$
$$cooperation_{((I)((I)I))I}\, georg e_I\, bob_{(I)I} = ProjectA_I$$
$$cooperation_{((I)I)((I)I)}\, bob_{(I)I}\, georg_I = ProjectA_I$$
$$cooperation_{((I)((o)I))I}\, bob_I\, georg_{(o)I} = ProjectA_I$$
$$\vdots$$

も$\mathcal{Q}_o$の仮説として渡される。

このデータベースを直観的に説明すると、以下のようになる。

(1) オブジェクトとはオブジェクトＩＤ・メソッド・クラスを統合したものである。

(2) あるオブジェクトがあるオブジェクトに作用した結果があるオブジェクトになるということは、あるオブジェクトがあるオブ ジェクトに作用した結果がオブジェクトＩＤであり、メソッドであり、クラスでもあるということである。

(3) オブジェクトとはオブジェクトＩＤ・メソッド・クラスを統合したものであるとしたがこの三者を完全に同等に扱うのではなく、オブジェクトＩＤをオブジェクトを代表する顔として他の 二者に優先させる。

5 $\mathcal{Q}_o$ の拡張

前章で説明したように、$\mathcal{Q}_o$ を用いて導出を行なう際にデータベースに登録された事実とその全ての派生式を $\mathcal{Q}_o$ の仮説とすれば、その導出の結果がデータベースの結果である。一つの事実に対して派生式は無限個存在するので、$\mathcal{Q}_o$ を用いてオブジェクト型の変（定）数を含む事実の導出を行なう際には無限集合を仮説として用いなければならない。しかし、無限集合を直接扱うことは有限資源を用いた計算機上では実現不可能であり、また、可能だとしても繁雑である。そこで、$\mathcal{Q}_o$ を拡張して仮説に無限集合を含まなくても等価な導出を行なえるように改良する。そのために、$\mathcal{Q}_o$ を用いた仮説からの結論の導出を観察して、$\mathcal{Q}_o$ の拡張に必要とされる新しい公理を推測する。

まず、$\mathcal{Q}_o$ における導出 $\vdash$ の定義をする。$\mathcal{H}$ を論理型閉整合論理式の集合とし、個体定数は各型について有限個であるとする。（このとき一般モデルが標準モデルになり、$\mathcal{Q}_o$ の完全性が成り立つ。）

(a) $A \in \mathcal{H}$ ならば、$\mathcal{H} \vdash A$ である。

(b) もし A が $\mathcal{Q}_o$ の定理ならば、$\mathcal{H} \vdash A$ である。

(c) C における A_α の出現が λ の直後に来る変数 ではなく、C における A_α の出現が C の $[\lambda x_\beta E_\gamma]$ の整合部分にはないとする。そのときに、$\mathcal{H} \vdash A_\alpha = B_\alpha$ かつ $\mathcal{H} \vdash C$ であり、しかも、ある整合論理式 D が C の中の A_α の一つ以上の出現を B_α に置き換えることによって得られるならば $\mathcal{H} \vdash D$ である。

この $\vdash$ について二つの整合論理式が持つ等価関係についてみていく。$\mathcal{Q}_o$ は述語論理の全ての定理を含んでいるので、$\mathcal{Q}_o$ は次のような演繹定理を含んでいる。

演繹定理 もし、$\mathcal{H}, H \vdash P$ ならば、$\mathcal{H} \vdash H \supset P$ である。ただし、$\supset$ は集合論の包含関係ではなく $\mathcal{Q}_o$ の省略記号（含意）である。

次に、二つの論理型閉整合論理式 A_o と B_o との間の関係 $\stackrel{B}{=}$、$\stackrel{B}{\neq}$、$\stackrel{B}{\vee}$、$\stackrel{B}{\wedge}$ を以下に定める。

$A \stackrel{B}{=} B$ ： $\neg A \vdash \neg B$ であり、かつ $\neg B \vdash \neg A$

$\stackrel{B}{\neg} A$ ： $\neg A$

$A \stackrel{B}{\vee} B$ ： $A \vee B$

$A \stackrel{B}{\wedge} B$ ： $A \wedge B$

このとき、$\stackrel{B}{=}$、$\stackrel{B}{\neq}$、$\stackrel{B}{\vee}$、$\stackrel{B}{\wedge}$ は定義域を $\mathcal{Q}_o$ の全閉整合論理式としたときに次のブール代数の公理を満たすことが分かる。

0 を T_o、1 を F_o とすると、

(1) $0 \stackrel{B}{\neq} 1$

(2) $\stackrel{B}{=}0 \stackrel{B}{=} 1$, $\stackrel{B}{=}1 \stackrel{B}{=} 0$

(3) $b \stackrel{B}{\vee} 0 \stackrel{B}{=} 0$, $b \stackrel{B}{\wedge} 1 \stackrel{B}{=} b$

(4) $b \stackrel{B}{\vee} 1 \stackrel{B}{=} b$, $b \stackrel{B}{\wedge} 0 \stackrel{B}{=} b$

(5) $b \stackrel{B}{\vee} (\stackrel{B}{=} b) \stackrel{B}{=} 0$, $b \stackrel{B}{\wedge} (\stackrel{B}{=} b) \stackrel{B}{=} 1$

(6) $\stackrel{B}{=}(\stackrel{B}{=} b) \stackrel{B}{=} b$

(7) $b \stackrel{B}{\vee} b \stackrel{B}{=} b$, $b \stackrel{B}{\wedge} b \stackrel{B}{=} b$,

(8) $\stackrel{B}{=}(b_1 \stackrel{B}{\vee} b_2) \stackrel{B}{=} (\stackrel{B}{=} b_1) \stackrel{B}{\wedge} (\stackrel{B}{=} b_2)$
$\stackrel{B}{=}(b_1 \stackrel{B}{\wedge} b_2) \stackrel{B}{=} (\stackrel{B}{=} b_1) \stackrel{B}{\vee} (\stackrel{B}{=} b_2)$

(9) $b_1 \stackrel{B}{\vee} b_2 \stackrel{B}{=} b_2 \stackrel{B}{\vee} b_1$, $b_1 \stackrel{B}{\wedge} b_2 \stackrel{B}{=} b_2 \stackrel{B}{\wedge} b_1$

(10) $b_1 \stackrel{B}{\vee} (b_2 \stackrel{B}{\vee} b_3) \stackrel{B}{=} (b_1 \stackrel{B}{\vee} b_2) \stackrel{B}{\vee} b_3$
$b_1 \stackrel{B}{\wedge} (b_2 \stackrel{B}{\wedge} b_3) \stackrel{B}{=} (b_1 \stackrel{B}{\wedge} b_2) \stackrel{B}{\wedge} b_3$

(11) $b_1 \stackrel{B}{\vee} (b_2 \stackrel{B}{\wedge} b_3) \stackrel{B}{=} b_1 \stackrel{B}{\vee} b_2 \stackrel{B}{\wedge} b_1 \stackrel{B}{\vee} b_3$
$b_1 \stackrel{B}{\wedge} (b_2 \stackrel{B}{\vee} b_3) \stackrel{B}{=} (b_1 \stackrel{B}{\wedge} b_2) \stackrel{B}{\vee} (b_1 \stackrel{B}{\vee} b_3)$

このブール代数を用いてブール代数の要素間に順序を導入し位相空間を張る。導入する順序を $\leq$ とおけば、以下のように定義される。

$$b1 \leq b2 \iff b1 \stackrel{B}{\vee} b2 \stackrel{B}{=} b1$$

全空間を P（$\mathcal{Q}_o$ の全閉整合論理式）とし、$[\ ]_\leq$ を以下のように定義する。

$$[p]_\leq = \{q \mid q \leq p\}$$

上述の順序を用いて集合 G が開であることを

$$\forall p \in G([p]_\leq \subseteq G)$$

と定義する。$\leq$ ・ $[\ ]_\leq$ ・ 開の意味を説明すると、$A \leq B$ は A が B から導出されること（$B \vdash A$）を、$[p]_\leq$ は p から導出される全ての論理型閉整合論理式を、G が開であるということは G から始まる導出の枝がいくつかあるとき G が全てのその導出の結果を含んでいることを意味している。

さて、もとのブール代数の定義域 P のベキ集合 X を考えて、閉包・開核を定義する。

閉包 A の閉包 A^- を以下に定義する。

$$A^- = \{x \in X \mid \forall G((G \text{ は開 }) \wedge x \in G \Rightarrow G \cap A \neq \phi)\}$$

開核 A の開核 A° を以下に定義する。

$$A^\circ = X - (X - A)^-$$

ただし、$-$ は集合論の差集合を表す。

この閉包と開核を使って、正則開集合 A を $A = A^{-\circ}$ を満たす集合と定義し、また、0、1 と $-$、$\cdot$、$+$ を以下のように集合論をもとに定義する。

0 : ϕ

1 : 位相空間の全体（P）

$-$: $-p = (X - p)^\circ$

$\cdot$: $p \cdot q = p \cap q$

$+$: $p + q = (p \cup q)^{-\circ}$

上記のように定義すると、0、1、$-$、$\cdot$、$+$ はもとの P の上の 0、1、$\stackrel{B}{=}$、$\stackrel{B}{\vee}$、$\stackrel{B}{\wedge}$ と同型になる。つまり、

- $-[b] = [\overset{B}{-} b]$
- $[b_1] \cdot [b_2] = [b_1 \overset{B}{\cdot} b_2]$
- $[b_1] + [b_2] = [b_1 \overset{B}{+} b_2]$

が成り立つので、X 上の $[p]$ も P 上のブール代数と同型のブール代数を成す。

一般にある集合の巾集合の上の全正則開集合の集合演算は完備であることが知られているので[14, 15]、それと同型な P の上のブール代数も完備ブール代数になる。故に、このブール代数の前に何か別の完備なブール代数をつなげると、二つのブール代数を合成したものも完備性を保つはずである。ここで、始めにデータに登録された式は、$\mathcal{Q}_o$ に渡される時に各々についてその全ての派生式が生成されて渡されるので、派生式を派生させるやり方に従って順序を定めると一つのオブジェクトが一つの正則開集合を定めていることが分かる。つまり、全ての派生式を一つにまとめたものを一まとまりとしてそれらを全て集めるとそれは完備なブール代数を成す。そこで、その正則開集合を作るのに過不足のない以下の条件を $\mathcal{Q}_o$ の公理に加える。

規則D　規則DEによって整合論理式 B_α が A_α より生成されるとき、A_α から B_α を推論する。

この系を $\mathcal{Q}_o^D$ と呼ぶことにする。以上で、$\mathcal{Q}_o$ を拡張する際に必要になると思われる公理を見つけることができたが、規則Dは $\mathcal{Q}_o$ に渡されたデータと違って推論の各段で任意の深さだけ派生式を生成するので、$\mathcal{Q}_o$ に比べて余計な導出を行なう可能性を持っている。

$\mathcal{Q}_o$ と $\mathcal{Q}_o^D$ の導出が同じになることを証明するためには、全派生式の $\mathcal{Q}_o$ による導出の結果からもう一度派生式を作っても、もとの $\mathcal{Q}_o$ からの導出の結果と変わらないことを言えば充分である。いま、式の集合 A から式の集合 B が派生することを $B \ll A$ で表し、$[\ \ ]_\leq$ を定義したのと同様にして順序 $\ll$ について $[\ \ ]_\ll$ を定義する。すると、明らかに

$$[X+Y]_\ll = [X]_\ll + [Y]_\ll$$
$$[X \cdot Y]_\ll = [X]_\ll \cdot [Y]_\ll$$
$$[[X]_\ll]_\ll = [X]_\ll$$

が成り立つ。データベースに $x_1, ..., x_n$ が登録されているとすると、その時の全導出は $[[x_1]_\ll + ... + [x_n]_\ll]_\leq$ であるので（$[x_i]_\ll$ は P の巾集合 X の上の集合なので $[\ \ ]_\leq$ の定義域にない。そこで、一つの集合 $[x_i]_\ll$ の要素は $\wedge$ でつなぎ、集合間の $-, +, \cdot$ を $\neg, \vee, \wedge$ で置き換えて P の上で等価な式を得た後に $[\ \ ]_\leq$ をとるとする。このブール代数の完備性より、極限値においても等号が成り立つ。）、導出の結果からの派生式は

$$\begin{aligned}&[[[x_1]_\ll + ... + [x_n]_\ll]_\leq]_\ll \\ &= [[[x_1]_\ll]_\leq + ... + [[x_n]_\ll]_\leq]_\ll \\ &= [[[x_1]_\ll]_\leq]_\ll + ... + [[[x_n]_\ll]_\leq]_\ll \qquad \text{(a)}\end{aligned}$$

となる。このとき、データベースの一つの式については明らかに $[[x]_\leq]_\ll = [[x]_\ll]_\leq$ が成り立つので、(a) 式は

$$\begin{aligned}&[[[x_1]_\ll]_\leq]_\ll + ... + [[[x_n]_\ll]_\leq]_\ll \\ &= [[[x_1]_\leq]_\ll]_\ll + ... + [[[x_n]_\leq]_\ll]_\ll \\ &= [[x_1]_\leq]_\ll + ... + [[x_n]_\leq]_\ll \\ &= [[x_1]_\ll]_\leq + ... + [[x_n]_\ll]_\leq \\ &= [[x_1]_\ll + ... + [x_n]_\ll]_\leq\end{aligned}$$

となり、もとの導出の結果の式の集合と一致することが確認できる。

6　$\mathcal{Q}_o^D$ の健全性と完全性

次に $\mathcal{Q}_o^D$ の健全性と完全性について論じる。このことは $\mathcal{Q}_o$ の健全性と完全性を論じることと同値である。

健全性については、始めに登録する仮説が無矛盾ならば規則Dによって推論される派生式ともとの式は型が違うので明らかに無矛盾であり、$\mathcal{Q}_o^D$ の他の公理は $\mathcal{Q}_o$ と同じなので、健全性が成り立つ。ただし、派生式がデータベースの他の事実やその派生式と矛盾を起こさないように注意する必要がある。

完全性についても導出が $\mathcal{Q}_o$ と同じなので $\mathcal{Q}_o$ を基準に考える。$\mathcal{Q}_o$ では一つの型の個体定数の数が無限になると一般モデルが超準的になるので $\mathcal{Q}_o$ は完全でなくなり、始めに $\mathcal{Q}_o$ の上で定義したデータベース自体が完全ではなくなって存在意義がなくなる。そのとき、$\mathcal{Q}_o$ と $\mathcal{Q}_o^D$ の導出の一致は保証できなくなるが、始めに想定したデータベース自身が意味を持たなくなるので、$\mathcal{Q}_o^D$ の完全性が保証できないことは trivial な問題になる。個体定数が有限の時でも、$\mathcal{Q}_o^D$ では $\mathcal{Q}_o$ と違って、型に $_I$ を含むときは無限の式を持つことになるので考察を要する。派生式を作る際の一番基本になる規則（DT1）・（DT2）をみると、どちらの規則も一回適用されると型変数の文字数が1だけ増える。この文字数 n をもとに考えると、一つの定数に対して異なる型は $3^{(2n+1)}$ より小さい有限の数であることが分かる。故に、$\mathcal{Q}_o$ の有限個の仮説の各々の式について、式の中の各型の定数の型について長さが n の定数は有限であり、仮説の定数の数も有限であるので仮説全体でも一つの型の定数は有限個しか現れない。従って、これらの式に対する

一般モデルは標準的になる。故に、一般モデルが標準モデルになるときは $\mathcal{Q}_o$ が完全性を保つことになるので、同型で完備な導出を持つ $\mathcal{Q}_o^D$ も完全になる。

7 $\mathcal{Q}_o^D$ に基づいたＯＯＤＢ

まず、データベースの文法を定義する。

型　変数の型の構成法は $\mathcal{Q}_o$ と同じとする。

記号 $\{,\},\leftarrow,:$, 普通の関数記号以外は $\mathcal{Q}_o$ と同じとする。ただし、個体定数は有限個とする。

文法

項

（1）$\mathcal{Q}_o$ の整合論理式

（2）F が関数記号であり、$t_1,...,t_n$ が項であるとき $F(t_1,...,t_n)$

（3）$t_1,...,t_2$ が項であるとき $\{t_1,...,t_2\}$

式

（1）関数 → 定数　または　関数 → {...}

（2）$T_1,...,T_n$ を（1）の形の式とする時、関数 $[T_1,...,T_n]$

（3）項：項

（4）項=項

以上がデータベースの文法である。次にデータベースのセマンティクスについて説明する。このデータベースは $\mathcal{Q}_o^D$ に基づいているので詳しいセマンティクスは $\mathcal{Q}_o^D$（$\mathcal{Q}_o$）に譲り、ここではデータベースの式の $\mathcal{Q}_o^D$ の式への変換を例を用いて説明する。

関数、[ ]で囲まれた式、:で連結された式の中に現れる変数は全て $_I$ 型とする。F を関数としたとき、$F(t_1,...,t_n)$ は $\mathcal{Q}_o^D$ の式で表すと

$$F_{(((...(I)I)...I)}\, t_{1I}, t_{2I}, ...t_{nI}$$

というように多価関数は高階の一価関数で表す。

$F(t_1,...,t_n)[...]$ は式の先頭の関数 F が $t_1,...,t_n$ から構成されるオブジェクトであることを表し、... の部分がそのオブジェクトの性質を表している。... の部分には 関数 → 項 という式が並ぶ。

$$F(t_1,...,t_n)[M(s_1,...,s_n) \rightarrow C_1]$$

という形の式は

$$F_{((...(o)I)I)...I)}\, t_{1I}\, t_{2I} ... t_{nI} = T_o \wedge$$
$$M_{((...((I)I)...I)}\, F_I\, s_{1I} ... s_{nI} = C_{1I}$$

という式を表している。始めの項はＦというオブジェクトがＭというメソッドを表すオブジェクトを含んでいることを表していて、二番目の項は必要な他の引数 $s_1,...,s_n$ と共に M を F に作用させると C_{1I} になることを表している。

$$F(t_1,...,t_n)[M(s_1,...,s_n) \rightarrow \{C_1,...,C_n\}]$$

という形の式は

$$F_{((...((o)I)I)...I)}\, t_{1I}\, t_{2I} ... t_{nI}$$
$$M_{((...((I)I)...I)}\, s_{1I}\, s_{2I} ... s_{nI} = T_o \wedge$$
$$M_{((...((o)I)I)...I)}\, F_I(t_i)\, s_{1I} ... s_{nI}\, C_{1I} = T_0 \wedge$$
$$\vdots$$
$$M_{((...((o)I)I)...I)}\, F_I(t_i)\, s_{1I} ... s_{nI}\, C_{nI} = T_0$$

を表している。これは、F に M を作用させた結果の集合に $c_1,...c_n$ が要素として含まれているということを表している。

$Cp:Cc$ という式はクラスの包含関係を決める式であり、$\subseteq_{o((o)I)((o)I)}\ Cc_{(o)I} Cp_{(o)I} = T_o$ を表している。ここで、$\subseteq_{o((o)I)((o)I)}$ は集合の内包関係を表す $\mathcal{Q}_o$ の省略記号であり、

$$[\lambda x_{o\alpha} \lambda y_{o\alpha} \forall z_\alpha (x_{o\alpha} z_\alpha \supset y_{o\alpha} z_\alpha)]$$

と表される。クラス Cc が Cp に含まれていることを表している。

最後の項＝項という形の式は、オブジェクトＩＤやメソッドやクラスの恒等関係を表すための式であり、また、関数を書くための式である。メソッドやクラスを恒等関係で表して定義することは、特にメソッドを扱う場合に資源の節約のために望まれることであるが、派生型についてオブジェクトＩＤの方が優先しているのでオブジェクトＩＤを用いた定義と矛盾を起こさないように注意する必要がある。関数表現については、この仮想的なデータベースは $\mathcal{Q}_o$ の全ての関数表現を含んでいるので $\mathcal{Q}_o$ の表現力と同じ表現力を持っている。

このデータベースに対する質問には特殊な文字の集合を指定してその文字から構成された変数を使って求めたいデータを表し、その変数に適合する $\mathcal{Q}_o^D$ の導出結果を拾い出すというように指定する予定である。

8 結論

型付高階論理 $\mathcal{Q}_o$ を拡張してオブジェクト指向データベースの基礎となる $\mathcal{Q}_o^D$ を提案した。$\mathcal{Q}_o^D$ に基づいたデータベースでは定数と関数の間に関係を持たせて、従来のオブジェクト指向データベースでのオブジェクトよりも、オブジェクト指向の概念により忠実なオブジェクトとして扱

うことを可能にした。これにより、オブジェクト間・メソッド間・クラス間での等価関係を定義できるようになった。

今後の課題としては、以下のものが挙げられる。まず、$\mathcal{Q}_o^D$ に基づいてそのままデータベースを実現しようとすると、膨大な量の計算をしなければならなくなる。特に、公理として関数の外延的等価性を判断しなければならないものが存在するので、少しデータ量が増えただけでその上の関数の等価性を調べるための式が指数関数的に増える。表現に制約を加え、できるなら系自体をスマートにして計算量の問題を解決しなければならない。また、実際問題としては一階述語のＳＬＤ導出のような効率の良い導出方法も見つけなければならない。

$\mathcal{Q}_o^D$ では一つのオブジェクト型を定めると、自動的にその派生型にまで制約がかかってしまう。そのためにオブジェクト型に対するポリモルフィズムの実現が難しい。その必要性も含めて、検討したいと思っている。

現在の段階では、$\mathcal{Q}_o^D$ で実現されたオブジェクトの中にさらに細かい型を設けてはいない。データについて型を設けることに関する研究も活発に行なわれており [4, 8] そのことについても検討してみたい。

今回は、オブジェクト指向の基礎的な性質であるオブジェクトＩＤ・メソッド・クラスに着目して、オブジェクトの忠実な実現を目指した。オブジェクト指向の他の重要な性質であるカプセラレーションやインヘリタンスについては実現していない。しかし、$\mathcal{Q}_o^D$ にはそれらを実現するために必要とされる性質は持たせたつもりであるので、ルールを書くことにより解決されると思われる。これらを実現するための適切なルールを検討する予定である。

参考文献

[1] Peter B. Andrews, "An Introduction to Mathematical Logic and Type Theory : to Truth through Proof",Academic Press Inc., 1989.

[2] Peter B. Andrews, "A Transfinite Type Theory with Type Variables",North-Holland Series on Logic and the Foundations of Mathematics, North-Holland Publishing Company, 1965.

[3] Peter B. Andrews, "General Models, Descriptions, and ChIce in Type Theory",J. Symbolic Logic 37,pp 385-394, 1972.

[4] A.Ohori, "A Study of Semantics, Types, and Languages for Databases and Object-Oriented Programming," Ph D Thesis, University of Pensylvania, 1989.

[5] D.Maier : "A Logic for Objects", Proc. of the Workshop on Foundations of Deductive Databases and Logic Programming, pp.6-26, Washington DC, 1986.

[6] M.Kifer and J. Wu : "A Logic for Object-Oriented Logic Programming (Maier's O-Logic Revisited)", Proc. of 8th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, pp.379-393, 1989.

[7] M.Kifer and G.Lausen : "F-Logic: A Higher-Order Language for Reasoning about Objects, Inheritance, and Scheme", Proc. of the 1989 ACM SIGMOD Int'l Conf. on the Management of the Data, pp.134-146, 1989.

[8] M.Kifer, G.Lausen, J.Wu : "Logical Foundations of Object-Oriented and Frame-Based Languages", Technical Report 93/06, Department of Computer Science, SUNY at Stony Brook, April 1993.

[9] W. Chen and D.S.Warren : "C-Logic of Complex Objects", Proc. of the 1989 ACM SIGMOD Int'l Conf. on the Management of the Data, pp.369-378, 1989.

[10] S. Watari, Y. Honda and M. Tokoro : "Morphe: A Constraint-Based Object-Oriented Language Supporting Situated Knowledge", Proc. of the Int'l Conf. on Fifth Generation Computer Systerms, vol.2, pp.1044-1051, 1992.

[11] H. Yasukawa, H. Tsuda and K. Yokota : "Objects, Properties, and Modules in (Quixote)", Proc. of the Int'l Conf. on Fifth Generation Computer Systerms, vol.1, pp.257-268, 1992.

[12] Edit. Jack Minker : "Foundations of Deductive Databases and Logic Programming", Morgan Kaufmann Publishers Inc., 1988.

[13] M. Reinfrank, J. de Kleer, M. L. Ginsberg,E. Sandewall : "Non-Monotonic Reasoning", Lecture Notes in Artificial Intelligence (2nd Intl. Workshop Grassau, FRG, June 1988),Springer-Verlag, 1988.

[14] 西村、難波：“公理論的集合論”, 共立講座 現代の数学 2 , 共立出版,1985.

[15] 竹内外史：“現代集合論入門”, 日本評論社,1989.