

TM-0612

オブジェクトの形式化とスキーム

横田 一正

November, 1988

©1988, ICOT

ICOT

Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03) 456-3191~5
Telex ICOT J32964

Institute for New Generation Computer Technology

オブジェクトの形式化とスキーム

Formalization of Objects and its Scheme

横田 一正*

Kazumasa Yokota

(財) 新世代コンピュータ技術開発機構

Institute for New Generation Computer Technology

1 はじめに

知識情報処理を含む幅広い応用に対処するために、新しいデータベース技術が求められている。たとえば、工学データベースやマルチ・メディア・データベースや知識ベースなど、従来の関係モデルの枠内での対応だけでは不十分である。これらは、実世界のオブジェクトをより直接的に表現し、それらを操作(推論)することができるような、複雑なデータベース技術を必要としている。実際に ICOT の環境でも、定理証明支援のための数学知識ベース、自然言語処理のための辞書やシソーラスのデータベースや知識ベース、エキスパート・システムのためのルール・ベースなど、伝統的なデータベースで対処するのが難しい応用が多い。これらの要求には、単に、構造データの表現、そしてその格納、検索に留まらず、さまざまな推論機能を付加することが含まれている。たとえば、数学知識ベースは、項の単一化による検索、非巡回有向グラフとしての理論間の関係保持、証明の行間補間時の推論機能、などを必要としている。

伝統的なデータベースの拡張として、演繹データベース(DD)とオブジェクト指向データベース(OOD)がよく知られている。DDは、関係データベースを述語論理の枠内で形式理論と対応させたもので、証明論的アプローチとも呼ばれている。他方、OODは、オブジェクト指向プログラミング言語(OOP)の成功に対応したもので、その内容にまだコンセンサスは得られていないが、実世界の表現の有効な枠組と考えられている。DDとOODは、これまで互いに独立に研究開発が進められることが多かったが、それらの利点——とくに実世界のより直接的な表現能力とそれに対する推論機能——の融合が、今後のデータベースあるいは知識ベースにとって重要な位置を占めると考えられる。

本稿では、OODについて、その一つの側面と考えられる静的な面に着目する。そしてDDとOODの融合(DOOD)

に向けての一つの枠組として、オブジェクトの形式的表現とその質問処理のスキームについて議論したい。つまり、非正規関係や型階層や部品関係などのさまざまな構造データは、それらの各問題領域(つまり構造データの特定の性質によって抽象化された問題領域)が持っている意味をオブジェクトに付加したもので、その具体例(インスタンス)と考える。そして、領域間の媒体はオブジェクトであり、その質問評価は、オブジェクトに特定の意味を付加して具象化した各領域の評価系によって実行される。これは、各問題領域に応じて制約評価系を考える制約論理型言語スキームCLP(X)に対応している、と考えられる。つまり、オブジェクトに対する処理を、その性質によって限定された問題領域へと分散し、それぞれを限定された処理として実行し、それを組合せて処理結果とする、と言える。まず2節でDDとOODの枠組を検討し、3節ではオブジェクトと型の形式的表現を、そして4節でルールや無限構造を持つ型とオブジェクトの定義を検討する。さらに5節ではそれらの具体例としての構造データのいくつかを、6節では制約評価系について検討し、7節で今後の課題を述べる。

2 演えき vs. オブジェクト指向

関係モデルの拡張としてのDDの特徴は、一階述語での表現、形式理論としての扱い、その評価機構(定理証明)、データベースとしての制限、などである。述語表現が、現実の問題領域においては低い表現力しか持たないこと、つまり、引数の位置と個数の問題や構造表現の問題など、すでに数多く指摘されている[Ait86, BaKh85, Mukai87, Yokota88, 横田88b]。そこで、さまざまな構造データを、述語表現からレコード表現に拡張することが以下の理由で試みられている：

- 述語記法に比べて“高いレベル”の表現、つまり実世界により近い表現になっていること。

* e-mail: kyokota@icot.junet

- さらに複雑な構造、たとえば集合構成子によって集合のような構造も導入しやすいこと。
- 継承関係、部品関係などの特定の問題領域の意味を導入しやすいこと。

(伝統的な言葉としての)レコード表現とは、本稿でのオブジェクト表現に対応しており、以下でこの表現に基づいた考察を議論する。

一方、OOD は、モデリング能力、インピーダンス・ミスマッチの解消、型階層の利用など、大きな可能性を持ったモデルとして登場している。しかし、この言葉には多様・過剰の意味が含まれている状態で、まだ明確な定義は定まっていない。また、そのアプローチにはどちらかというと非形式的なものが多く、その基盤としての論理あるいは数学はまだ曖昧な状態である。“オブジェクト指向”(OO)という言葉に二種類の意味を考えることができる。つまり、複合構造、オブジェクト固定、データ抽象および型階層のような静的オブジェクトの側面と、メソッド、メッセージ・パッシング、情報隠蔽のような動的オブジェクトの側面である。OOD が動的な側面を強調するのに対し、伝統的なデータベースの視点での“オブジェクト”(非正規関係、複合オブジェクト、…)は静的な側面を主に考える傾向が強い。OOD の大部分はこの両者のほごまで考えられていると思われるが、本稿で“オブジェクト”という場合、その静的な側面に着目しており、非正規関係や複合オブジェクトを含むような構造データを考える。

DD の形式的なアプローチには、推論機能の付加や不確定情報の取扱いなどの質問処理能力の拡張の他に、意味論の明確化による将来のデータベース技術への拡張の見通しなど大きな利点が含まれている。したがって、オブジェクトを DD の枠組で考えること、あるいは OOD の形式化を DD の視点で考えることには、応用領域の拡大と共に、新しいデータベース技術としての利点が考えられる。これらの融合には否定的な意見もあるが、OOD を限定した形で、すでにいくつかの提案や議論も行われている。しかしそのオブジェクトへの意味の付与には不十分なことが多い。たとえば非正規関係という、特定のオブジェクトの形式化が、他の構造データへのアプローチのさまざまな文脈の中で、それらに含まれる、あるいは関連している、と指摘されている。たとえば、型継承[Ait86]、複合オブジェクト[BaKh85, AbGr88]、オブジェクト[Maier86]、集合を持った論理型言語[BNST87]、などがその例である。その原因には、従来の意味での非正規関係そのものを考えても、その集合構成子に対して必ずしも一意的な意味が付与されていないことにもあるが、これらの形式化の文脈で言及される非正規関係には、いずれもその一側面しか言及されていないことにある。

と考えられる。言い換えれば、非正規関係という特定の問題領域における固有の意味や操作を、それらは充分に表現しえていないと言うことができる。しかし一方で、このような共通の指摘は、さまざまな異なった問題領域での形式化の、ある側面での式述位を不変しているとも考えられる。

このような形式化は、データベース分野だけでなく、知識表現での型継承[Ait86]や、自然言語における状況の表現[Mukai87]などに見られるように、データベース言語、プログラミング言語、知識表現言語など多くの分野にまたがっており、それは、データ構造を基にしたプログラミング言語、“レコード・プログラミング言語”[横田88b]、あるいはデータベース・プログラミング言語といった、新しいパラダイムとして語ることができよう。本稿の目的は、このような認識を踏まえて、一般的なオブジェクトの表現と、各問題領域に固有な意味を持ったその具体例と質問処理、そしてそれら全体としての質問処理を考える、オブジェクトのスキームを議論することにある。

3 オブジェクトと型

3.1 型

まず、属性名の集合 $Attr$ と、原子型の集合 $Type$ と $\perp$ を仮定し、集合構成子 “[”, “)” および組構成子 “[”, “]” により、型を定義する：

- (1) 原子型 $\tau \in Type$, $\perp$, そして $\perp$ は型である。
- (2) τ が型ならば、 $\{\tau\}$ は型であり、集合型と言う。
- (3) $a_1, \dots, a_n \in Attr$ が異なった属性名で、 $\tau_1, \dots, \tau_n$ が型ならば、 $[a_1/\tau_1, \dots, a_n/\tau_n]$ は型である。これを組型と言い、 a_i/τ_i を組要素と言う。

つぎに、型の集合に半順序 $\leq$ を定義する：

- (1) 任意の型 τ に対して、 $\perp \leq \tau$ かつ $\tau \leq \perp$ 。
- (2) 集合型 $\tau_1 = \{\tau'_1\}$ と $\tau_2 = \{\tau'_2\}$ に対して、 $\tau'_1 \leq \tau'_2$ ならば、 $\tau_1 \leq \tau_2$ である。
- (3) 組型 $\tau = [a_1/\tau_1, \dots, a_n/\tau_n]$ に対して、 τ から $\tau_i = \perp$ なる組要素 a_i/τ_i を除外して得られる型を τ' とする。組型 τ_1 と τ_2 に対して、 $\tau'_1 = [a_{11}/\tau_{11}, \dots, a_{1n}/\tau_{1n}]$ として $\tau'_2 = [a_{21}/\tau_{21}, \dots, a_{2m}/\tau_{2m}]$ とする。もし $\tau_{1i} = \perp$ なる i が存在する、あるいは、 $\{a_{11}, \dots, a_{1n}\} \supseteq \{a_{21}, \dots, a_{2m}\}$ かつ $\forall j, \exists i (a_{1i} = a_{2j} \text{ かつ } \tau_{1i} \leq \tau_{2j})$ ならば、 $\tau_1 \leq \tau_2$ である。

この半順序の定義により、次の同値関係 $\approx$ を得ることができる：

$$\begin{aligned} [a_1/\tau_1, a_2/\tau_2, \dots, a_n/\tau_n] &\approx [a_2/\tau_2, \dots, a_n/\tau_n], \\ [a_1/\tau_1, a_2/\tau_2, \dots, a_n/\tau_n] &\approx [a_1/\tau_1], \text{ ただし } i \text{ は任意.} \end{aligned}$$

組型中の組要素は互いに可換であるので、この関係は容易に一般化できる。

一般性を失うことなく、型については $\approx$ に関する同値類を考えることにする。型 τ_1 と型 τ_2 が両立可能であるとは、 $\tau = \tau_1 \sqcup \tau_2$ かつ $\tau_1 \sqsubseteq \tau$ なる型 τ が存在することである。後述するが、非正規関係の場合、型はスキーマに対応する。そして、型の集合に対し、以下のように lub と glb を定義する：

(1) 任意の型 τ に対して、 $\tau \sqcup \top = \top$, $\tau \sqcap \top = \tau$, $\tau \sqcup \perp = \tau$, $\tau \sqcap \perp = \perp$.

(2) 集合型 $\tau_1 = \{\tau_1'\}$ と $\tau_2 = \{\tau_2'\}$ に対して、

$$\tau_1 \sqcup \tau_2 = \{\tau_1' \sqcup \tau_2'\}, \quad \tau_1 \sqcap \tau_2 = \{\tau_1' \sqcap \tau_2'\}.$$

(3) 2つの組型 $\tau_1 = [a/\tau_{11}, b/\tau_{12}]$ と $\tau_2 = [a/\tau_{21}, c/\tau_{22}]$ に対して、

$$\begin{aligned} \tau_1 \sqcup \tau_2 &= [a/(\tau_{11} \sqcup \tau_{21}), \\ \tau_1 \sqcap \tau_2 &= [a/(\tau_{11} \sqcap \tau_{21}), b/\tau_{12}, c/\tau_{22}]. \end{aligned}$$

(この定義が省略形であることに注意せよ。)

(4) (1), (2), (3) 以外、 lub は $\top$, glb は $\perp$.

これにより、任意の2つの型に対して lub と glb は一意に決まるので、 $\top$ と $\perp$ を含む型の集合から束が構成できる。

3.2 オブジェクト

空オブジェクト ω と、型に付随した原子オブジェクト o_i の集合 Obj を仮定し、属性名の集合 Attr 、集合構成子 “[$\cdot$, $\cdot$]” と組構成子 “[$\cdot$, $\cdot$]” によって、オブジェクトを以下のように定義する：

(1) 空オブジェクト ω 、任意の原子オブジェクト $o_i \in \text{Obj}$ はオブジェクトである。

(2) もし $o_1, \dots, o_n$ がオブジェクトなら、 $\{o_1, \dots, o_n\}$ はオブジェクトであり、これを集合オブジェクトと呼ぶ。

(3) もし $a_1, \dots, a_n \in \text{Attr}$ が異なった属性名で、 $o_1, \dots, o_n$ がオブジェクトならば、 $[a_1/o_1, \dots, a_n/o_n]$ はオブジェクトである。これを組オブジェクトと言い、 a_i/o_i を組要素と言う。

集合オブジェクトの要素がただ一つの要素しか持たないときは、誤解のない限り、集合構成子は無視することがある。また、組オブジェクト $O = [a_1/o_1, \dots, a_n/o_n]$ に対する操作として、属性名 a_i によって $O.a_i = o_i$ が定義される。

次に、オブジェクトと型の組合せ、つまり型付オブジェクトを定義する：

(1) 空オブジェクト ω と任意の原子オブジェクト $o_i \in \text{Type}$ に対し、 $\omega:\top$, $o_i:\tau_i$ は型付オブジェクトである。(以下それぞれを単に $\top$, τ_i と書く。)

(2) 任意の原子オブジェクト o_i と原子型 τ に対し、 $o_i:\tau$ は型付オブジェクトである。(以下これを単に原子オブジェクトと呼ぶ。)

(3) もし $o_1:\tau_1, \dots, o_n:\tau_n$ が型付オブジェクトならば、 $\{o_1:\tau_1, \dots, o_n:\tau_n\}:\{\tau\}$ は型付(集合)オブジェクトである。

(4) もし $a_1, \dots, a_n \in \text{Attr}$ が異なった属性名で、 $o_1:\tau_1, \dots, o_n:\tau_n$ が型付オブジェクトならば、

$$[a_1/o_1:\tau_1, \dots, a_n/o_n:\tau_n]:[a_1/\tau_1, \dots, a_n/\tau_n]$$

は型付(組)オブジェクトであり、 $a_i/o_i:\tau_i$ は型付組要素である。

(5) もし $o:\tau_1$ がオブジェクトで、かつ $\tau_1 \sqsubseteq \tau_2$ ならば、 $o:\tau_2$ は型付オブジェクトである。

型が明示的に付加されていなくても容易に推論することができるので、以下必ずしも型を明示していなくても、あるいは単にオブジェクトと言っても、型付オブジェクトを意味しているものとする。

そして、型間の半順序と同様に、オブジェクト間の半順序 $\preceq$ を定義する：

(1) 任意の原子オブジェクト o に対して、 $o \preceq \top$ ($o \preceq \omega:\top$)、かつ $\perp \preceq o$.

(2) 集合オブジェクト $o_1 = \{o_{11}, \dots, o_{1n}\}:\{\tau_1\}$ と $o_2 = \{o_{21}, \dots, o_{2m}\}:\{\tau_2\}$ に対して、

$$\forall i, j)(o_{1i} \preceq o_{2j}), \text{ かつ, } \tau_1 \preceq \tau_2$$

ならば、 $o_1 \preceq o_2$ である。

(3) 組オブジェクト $o = [a_1/\tau_{11}, \dots, a_n/\tau_{1n}]$ に対して、 o から $\tau_i \rightarrow \top$ なる組要素 $a_i/o_i:\tau_i$ を除外して得られるオブジェクトを τ' とする。組オブジェクト o_1 と o_2 に対して、

$$\begin{aligned} o_1' &= [a_{11}/\tau_{11}, \dots, a_{1n}/\tau_{1n}], \text{ そして} \\ o_2' &= [a_{21}/\tau_{21}, \dots, a_{2m}/\tau_{2m}] \end{aligned}$$

とする。もし (a) $\{a_{11}, \dots, a_{1n}\} \supseteq \{a_{21}, \dots, a_{2m}\}$, (b) $r_1 \leq r_2$, かつ (c) $r_1 = \perp$ なる r が存在する、あるいは $\forall j, \exists i (a_{1i} = a_{2j}, 0 \leq i \leq n, 0 \leq j \leq m \wedge r_{1i} \leq r_{2j})$ ならば、 $r_1' \leq r_2'$ である。

そして、型と同様に、オブジェクト間にも lub と glb を定義する：

(1) 任意のオブジェクト o に対して、 $o \sqcup \top = \top$, $o \sqcap \top = o$, $o \sqcup \perp = o$, $o \sqcap \perp = \perp$.

(2) 集合オブジェクト $o_1 = \{o_{11}, \dots, o_{1n}\} : \{r_1'\}$ と $o_2 = \{o_{21}, \dots, o_{2m}\} : \{r_2'\}$ に対して、

$$o_1 \sqcup o_2 = \{o_{11} \sqcup o_{21}, \dots, o_{1n} \sqcup o_{2m}\} : \{r_1' \sqcup r_2'\},$$

$$o_1 \sqcap o_2 = \{o_{11} \sqcap o_{21}, \dots, o_{1n} \sqcap o_{2m}\} : \{r_1' \sqcap r_2'\}.$$

(3) 組オブジェクト $o_1 = [a/o_{11}, b/o_{12}] : r_1$ と $o_2 = [a/o_{21}, c/o_{22}] : r_2$ に対して、

$$o_1 \sqcup o_2 = [a/(o_{11} \sqcup o_{21}), b/o_{12}] : (r_1 \sqcup r_2),$$

$$o_1 \sqcap o_2 = [a/(o_{11} \sqcap o_{21}), b/o_{12}, c/o_{22}] : (r_1 \sqcap r_2).$$

(この定義が省略形であることに注意せよ。)

(1), (2), (3) 以外、 lub は $\top$, glb は $\perp$.

この任意の2つのオブジェクトに対して、 lub と glb は一意に決定できるので、型と同様に、 $\top$ と $\perp$ を含むオブジェクトの集合から束を構成できる。

4 型とオブジェクトの定義

前節の型とオブジェクトは、原子型と原子オブジェクトから、集合構成子と組構成子によって構成されたが、それをさらに拡張しよう。一つはルールの集合オブジェクトであり、もう一つは無限構造を持つオブジェクトである。

そのために、新しい型の名前の集合 $Ntype$ と、オブジェクト名の集合 $Nobj$ を仮定する。前節のように構成された型 τ あるいはオブジェクト $O : \tau$ は、 $\tau_0 \in Ntype$ と $O_0 \in Nobj$ を使って以下のように定義される (名前づけされる)：

$$\tau_0 = \tau; O_0 : \tau = O : \tau.$$

ただし、 τ_0 , $O_0 : \tau$ はいずれも元の型あるいはオブジェクトと同じ性質を持っているとする。

さて、ある言語 L によって記述されたルール r を、その言語に依存した型 τ_L を持つ原子オブジェクト $r : \tau_L$ と考えよう。ルールの集合 $\{r_1, \dots, r_n\}$ は集合オブジェクトであり、 $O \in Nobj$ によって、それに名前を付けることができる：

$$O : \{\tau_L\} = \{r_1, \dots, r_n\} : \{\tau_L\}, \text{ あるいは}$$

$$O : \tau_0 = \{r_1, \dots, r_n\} : \{\tau_L\}, \text{ ただし } \tau_0 \in Ntype.$$

演繹データベース (DD) の IDB と EDB , あるいはその一部はこのようにして、オブジェクトとして定義する。原子オブジェクトとしての個々のルールを解釈するのは、各言語に対応した問題領域の評価系であり、原子オブジェクトの意味については、他の評価系はそれが原子オブジェクトであること以上には何も知らない。

さらに無限構造を持つオブジェクトのために、再帰的な定義を許すことにする。このような型は、(1) 項と同様に、

- 属性名の接合列の集合の部分集合から作られる木領域 Δ ,
- Δ から型の名前の集合 $Type \cup Ntype$ への部分関数 ϕ ,
- 集合型の定義、あるいは Δ から集合を識別する $\{0, 1\}$ への部分関数 ρ

の組として一般的に定義できる。このときの型間の順序は、(1) 項と同様に、たとえ無限構造を持っていたとしても、前節の定義をそのまま使用することができる。

再帰的な型に対応して、再帰的なオブジェクトは、関数型言語 ML [Harper86, 横田88a] の型宣言 (表現: 型) に対応させて考えることができる。たとえば、`letrec` を使用することによって、

$$\text{letrec } o = [id/o_1, name/o_2, age/o_3, \dots, dcs/\{o\}]$$

と定義できる。(この定義が、グラフ論的には有限のオブジェクトに限定していることに注意する必要がある。) ML との対応は [Ohori88] でも行われており、このようなオブジェクトの型推論についても、ML の型推論にならって行うことが考えられる [横田88a, Ohori88]。

具体的なオブジェクトは、オブジェクト識別子を用いて表現できる。たとえば

$$[id/10, name/\text{"taro"}, age/30, \dots, \\ dcs/\{[id/20], [id/30]\}] : \tau.$$

は、無限構造を持つ型 τ に対応したオブジェクトである。ただし、明らかな型は省略されており、また `id` はオブジェクト識別子の属性名となっている。

5 オブジェクトと具体例

前節の形式化に従って定義されるオブジェクトから、これにさまざまな条件を付加することによって、非正規関係、型階層、部品関係のような、さまざまな構造データを、その具体例として展開しよう。そのために、非正規関係における行ネストや行アンネストの操作や、型階層の `is_a` 関係や部品関係の `part_of` 関係や、継承機能を持った非正規関係のような特定の意味を、導入する必要がある。そして、それらに対応した意味論は、各問題領域に対応して与える。

5.1 非正規関係

非正規関係は直観的には、以下の関係 R と定義される：

$$R \subset E_1 \times \dots \times E_n \\ E_i := \text{この定義域 他は関係のべき集合.}$$

しかし、その形式や意味には、ネスト演算に対する構文と意味によって、いくつかの種類がある[三浦87, YKh88]。たとえば、以下の二つの組

$$[a/\{c_1, c_2\}, b/\{d_1\}], [a, c_1, b/\{c_1, d_1\}]$$

にアンネスト操作をいったん施すと元の組はネスト操作で復元できない。そこでネスト操作で可逆なサブクラスだけを対象とすることが考えられる。また、二つの組 $[a/c_1, b/\{c_2, c_3\}]$ と $[a/c_1, b/\{c_3, c_1\}]$ に対して、行ネストを施すと、次の二つの結果が考えられる：

$$[a/c_1, b/\{c_2, c_3, c_1\}], \text{ または } \\ [a/c_1, b/\{\{c_2, c_3\}, \{c_3, c_1\}\}].$$

それぞれの結果の組は、集合の合併とグルーピング操作に対応している。これは、ネスト演算あるいは集合構成子に対する意味の付与の違いによる。

一方、非正規関係のスキーマは、型に以下の構文上の制限を加えたものである：

- (1) 非正規組の型 τ からなる集合型 $\{\tau\}$ 、
- (2) τ の属性名の本領域 Δ は、以下の性質を持つ：
 $\forall t_1, t_2 \in \Delta (t_1 = s_1.l \wedge t_2 = s_2.l \supset s_1 = s_2)$.

ただし、(2)の制限を弛め、再帰的な構造を認めることも考えられる。そして、非正規関係データベースのスキーマは、関係名(属性名)と非正規関係の対を組要素とする組型である。

オブジェクト O に対して、上記非正規関係の意味を付加する関数 Λ^* を考えるそして、非正規関係に対する操作を λ 、操作対象のオブジェクトを $O_1, \dots, O_n$ とすると、 $\Lambda^*(\lambda, i, \{O_1, \dots, O_n\})$ と表現できる。ただし、 i は集合構成子の意味に対応させたパラメータである。結果は、非正規関係の構文上の制限に沿ったオブジェクトとなる。

5.2 型階層

型階層の目的は、型間の is-a 関係に基づいて、オブジェクト間の属性の継承を行うことにある。そのために、3節で定義した、型間の自然な半順序の他に、型間に新たな半順序の定義を付加する必要がある。この原子(定義)型間の半順序が構成型間の順序に反映され、たとえば、型 $\tau_1 =$

$[a/\rho_1, b/\rho_2]$ と $\tau_2 = [a/\rho_1, c/\rho_3]$ に対して、 $\tau_1 \preceq \tau_2$ の順序を付加すると、自然な順序が反映され、 $\tau_1 = [a/\rho_1, b/\rho_2, c/\rho_3]$ が仮定される。このような型間の順序を付加しても、型の集合が束を構成するのは容易にわかる。

この型階層は、型間の順序集合を付加するだけで、前節のオブジェクトの場合と本質的に変わりはない。これは、オブジェクト O に対し、ある特殊な問題領域に依存した順序集合 S を付加したオブジェクトを、 $T(o, S, O)$ と表現する(ただし o は無操作 nop である)。この結果もオブジェクトである。さらに、継承関係を持った非正規関係[中野88]についても、 $\Lambda^*(\delta, i, T(o, S, O))$ 、あるいは $O' = T(o, S, O) \wedge O'' = \Lambda^*(\delta, i, O')$ と拡張できる。

この型階層は、ルールのオブジェクト間に適用することによって、対象とする世界に応じた知識を表現できる。たとえば、確定節で表現されたルールのオブジェクトを以下のようにする：

$$O_1 : \tau_1 = \{sg(X, X) \leftarrow hu(X), \\ sg(X, Y) \leftarrow up(X, Z_1), sg(Z_1, Z_2), down(Z_2, Y)\} \\ O_2 : \tau_2 = \{sg(X, Y) \leftarrow up(X, Z_1), sg(Z_1, Z_2), flat(Z_2, Z_3), \\ sg(Z_3, Z_4), down(Z_4, Y)\}$$

のとき、 $\tau_1 \preceq \tau_2$ を定義することによって、 O_2 は O_1 のルールを継承することができる。したがって、ルール・オブジェクト(知識)の集合に対し、それらの間の半順序の集合を与えることによって、多くの世界に共通な知識(常識)の表現や、各個人に対応した知識の動的な設定が可能となる。

5.3 部品関係

部品関係 part.of についての形式的な定義は少ないが、[BaKh88]では次のように与えている：

- (1) 組オブジェクト O に対し、 $O.a_i$ は O の部品である。
- (2) 集合オブジェクト O に対し、 $o \in O$ ならば o は O の部品である。
- (3) O_1 が O_2 の部品で、 O_2 が O_3 の部品ならば、 O_1 は O_3 の部品である。

たとえば、この部品関係の定義に従えば、オブジェクト $O = [a/\{[c/o_1, d/o_2], [c/o_3, d/o_4]\}, \dot{q}/o_5]$ に対し、 $[c/o_1, d/o_2]$ および o_5 は O の部品である。この関係は型付オブジェクトにも問題なく拡張できる。

部品関係は、オブジェクトの自然な順序や型階層の順序と背反しないので、最初から組み込むことができる。しかし、部品関係固有の操作である展開(explosion)や統合(implosion)は、非正規関係のネスト操作同様に、この領域に

固有と考えられるが、これも他の操作と背反しない。しかし、共通部品の扱い、一部部品の取り替えのような操作を考えると、具体例の1つと考えたほうが扱いやすい。したがって、たとえば、オブジェクト O の部品 C を C' に取り替える操作 p を、 $P(p, \{C, C'\}, O)$ と表現しよう。

6 質問処理と制約評価系

オブジェクト O に対する質問処理を、それに特定の意味を付加した具体例 $A(\delta_{ij}, O)$, $T(o, S, O)$, $P(p, C, O)$ に対する質問処理に分解することを考える。このように、オブジェクトを問題領域に対応して具象化する関数を一般的に I_i とし、具象化するための情報を B_i 、そこで定義されている基本的な操作を δ_{ij} とすると、一般的に $I_i(\delta_{ij}, B_i, O)$ と表現できる。

6.1 オブジェクトの質問処理

各具体例に対する質問処理を、各問題領域における一種の制約評価系と考えることによって、制約論理型言語 (CLP) との対応を考える。

CLP の一般的な導出過程は、ゴール中のリテラルと制約の対の書換え過程と考えることができる。つまり、ゴールとして、リテラルの集合 $G_0 = \{L_1, \dots, L_n\}$ と制約の集合 $C_0 = \{C_{01}, \dots, C_{0m}\}$ が $\leftarrow C_0 \mid G_0$ で与えられたとき、 $L_1\theta = H\theta$ なる $H \leftarrow C' \mid B$ なるルールがあったとき、新しいゴール G_1 と制約 C_1 は、

$$C_1 = \text{solve}(C_0\theta \cup C'\theta), G_1 = \{L_2, \dots, L_n\}\theta \cup B\theta$$

となる。ただし、solve は該当の問題領域での制約評価系である。そして、

$$(G_0, C_0) \Rightarrow (G_1, C_1) \Rightarrow \dots \Rightarrow (G_n, C_n) = (\emptyset, C_n)$$

なる導出列があったとき、 C_n が解となる。

そして、オブジェクトの質問処理過程も、リテラルの集合 G と制約の集合 C の対の書換え過程と考えよう。そのために、オブジェクト変数の集合 Vobj 、述語記号の集合 Pred 、そして、型間の順序などの情報を伝搬する具象化変数の集合 Inf を与える (ここでの述語記法は単に便宜上のものであることに注意せよ)。まず、原子式は以下のように定義する：

$O_1, \dots, O_n$ がオブジェクト、オブジェクト変数、あるいは具象化変数で、 $p \in \text{Pred}$ ならば、 $p(O_1, \dots, O_n)$ は原子式である。

次に、オブジェクトの制約を定義する：

$O_1, O_2 \in \text{Vobj}$, O_i がオブジェクトまたはオブジェクト変数、 I_i, B_i, δ_{ij} をそれぞれ、具象化関数、具象化情報、そしてそれに対応する一つの基本操作とすると、 $O_1 = O_2$ 、または $O_1 = I_i(\delta_{ij}, B_i, O_2)$ は制約である。

この制約の評価は、 $=$ の右辺中の、オブジェクト変数および具象化変数がすべて具象化されるまで遅延させられる。そしてルールは、原子式 H と制約の集合 $C = \{C_1, \dots, C_m\}$ と原子式の集合 $B = \{B_1, \dots, B_n\}$ の組で定義され、

$$H \leftarrow C \mid B \text{ または } H \leftarrow C_1, \dots, C_m \mid B_1, \dots, B_n$$

と書くことにする。

オブジェクトの場合、Prolog での述語のような単一化は一般的には不可能なので、以下のように単一化に制限する。二つの述語 $p_1(O_{11}, \dots, O_{1n})$ と $p_2(O_{21}, \dots, O_{2m})$ に対し、

- 述語記号が等しい、つまり、 $p_1 = p_2$ 。
- 引数の個数が等しい、つまり、 $n = m$ 、かつ
- すべての i ($0 \leq i \leq n$) に対して、 O_{1i} または O_{2i} がオブジェクト変数、あるいは具象化情報と具象化変数。

三番目の条件を、オブジェクト (識別子) 同士に拡張することも考えられるが、ここでは考慮しない。

オブジェクトに対する操作は、ルールの集合 R として定義され、それは、原子式の集合 G と制約の集合 C の組からなるゴールを与えることによって実行される。制約としては、明示的に与えられたものの他に、解として求める必要のある G_0 中のオブジェクト変数も制約としよう。つまり G 中のオブジェクト変数 X に対し、新しい変数 $O \in \text{Vobj}$ により、制約 $X = O$ を構成する。質問処理は、上記の制約論理型言語の導出同様に、

$$(G, C) = (G_0, C_0) \Rightarrow \dots \Rightarrow (G_n, C_n)$$

の列で、 $G_n = \emptyset$ 時の C_n が処理結果となる。ただし、制約の評価で、 $=$ の左辺のオブジェクトが得られなかったとき、または型が $\perp$ になったとき、その評価は失敗し、他の導出列を求める。たとえば、

$$\begin{aligned} p(X_1, X_2) \leftarrow X_3 &= I_1(\delta_{13}, \dots, \{X_1.a, X_1.b\}) \\ X_4 &= I_2(\delta_{21}, \dots, X_3) \mid q(X_4, X_2), \\ q(X_1, X_2) \leftarrow X_2 &= I_3(\delta_{31}, \dots, X_1). \end{aligned}$$

を考える。これに対して、具体的なオブジェクト O を持ったゴール $\leftarrow p(O, X_0)$ を与えると、

$$\begin{aligned} &(\{p(O, X_0)\}, \{X_0 = O'\}) \\ &\Rightarrow (\{q(I_2(\delta_{21}, \dots, I_1(\delta_{13}, \dots, \{O.a, O.b\})), X_0)\}, \{X_0 = O'\}) \\ &\Rightarrow (\{\}, \{X_0 = I_3(\delta_{31}, \dots, I_2(\delta_{21}, \dots, I_1(\delta_{13}, \dots, \{O.a, O.b\})))\}) \end{aligned}$$

となって停止する。このような枠組は、Prolog に if 項を組み込むことにより、構造データを演繹データベースの枠組で扱う試み[BNTSS] をさらに複数の問題領域に一般化したものとも考えることができる。

6.2 制約評価の種類

それでは、各問題領域における制約評価系、つまり基本操作をいかに設定すべきだろうか。

[Ohori88] では、オブジェクトに対する結合、射影を束上の操作として与え、選択を ML のような関数として定義している。オブジェクトに対する操作と具体例に対する操作の組合せも考えられる。たとえば

$$O_2 = I.(b, B, O_1.a \sqcap O_1.b)$$

では、オブジェクト O_1 の部分オブジェクトの glb (結合) に対する具体例に対する操作を定義している。したがって、質問処理のルールは、オブジェクトと具象化情報の伝達と制御に限定でき、制約でデータベースに対する操作を行っていると考えることができる。

制約評価系として以下のように分類できる：

- (1) オブジェクト O に対する操作：
組要素を取り出す $O.a$ 、オブジェクトの lub と glb 、半順序に従った射影と結合、その他 ML のように定義された関数、...
- (2) 各問題領域に意味を与える関数：
非正規関係 I 、型階層 T 、部品関係 P 、... と、それらに対する具象化情報。
- (3) 各問題領域に固有の多種の操作群。

それらを具体的に設定する必要があるが、そのメタ情報をいかに表現するかはまた別の課題となろう。

6.3 演えきデータベース (DD) の組込

DD は、使用言語が確定節であれ CRL であれ、 EDB と IDB に分割できる。前述したように、 EDB も IDB も集合オブジェクトとして定義することが可能で、いくつかの細分化された単位で保持できる。そして、型階層を指定することにより、質問あるいは質問者に適した世界の EDB と IDB を構成することができる。このような DD に対する質問の処理結果は、集合オブジェクト (関係) と考えることができる。したがって、DD は、質問 (+ ルール) と、対象となるオブジェクト O 、そして EDB と IDB の指定 (型階層) によって、新しいオブジェクト O' を得る操作である、と考えられる。

確定節 DD の場合、各ルールは述語表現されているが、それは容易にオブジェクトのサブクラスとして表現できるので [Mukai87, Yokota88]、そのような表現に限ることにする (現実的にはそれらにインタフェースがあったほうがよいかもしれない)。それに対応して質問は、結果のオブジェクトの形を指定するもので、それもオブジェクトとして容易に拡張できる。たとえば、 $\text{--- } q(X, Y)$ に対し、 $\text{--- } \{[o/X, b/Y] : q\}$ と書き直すことができる。

したがって、制約部における DD の呼び出しは、 EDB と IDB の構成情報 T とその具象化関数 I 、そして結果のオブジェクトの型を指定する質問 q から、そして必要な DD の呼出し D から構成される： $D(o, q, T(o, T, O))$ 。

7 今後の課題

オブジェクトの、ある側面からの性質の抽象によって、非正規関係、型階層、あるいは部品関係など、さまざまな問題領域が設定できる。本稿では、各問題領域での処理を一種の制約評価系とみなし、それらを組み合わせた処理を制約論理型言語の枠組で考えた。今後さらに検討が必要なのは、以下の諸点である：

- 無限構造を含む型あるいはオブジェクトに対する形式的意味論の付与。おそらくこれには、ML のような意味論 [Harper86, Ohori88] がもっとも適しているように思える。
- 特定の問題領域での操作と意味論の明確化。領域を限定しているが、従来データベース分野では、あまり行われてこなかった点である。
- CLP の枠組でのオブジェクトに対する質問処理と、演繹データベースとの関係。本稿では、演繹データベースについても一つの具体例として、質問処理中に呼び出すことを考えたが、さらに質問処理の上層に持ってくることで、つまりルールと組み合わせることが検討されるべきだろう。
- 制約の表現として、右辺から左辺への書き換えと考えたが、必ずしも評価が終了しなくても、制約の集合を不確定解と考えるような拡張が必要だろう。
- オブジェクトに一定の制限を加えることによって、それを論理型言語あるいは形式理論として仕上げるができるが、各問題領域との関連でその制限を考える必要がある。これまた、オブジェクトの質問処理と演繹データベースの関係でもある。
- オブジェクトの表現をさらに一般的な知識表現 (ルール、意味ネットワーク、フレーム、...) といかに関連さ

せるか。つまり、単に論理型言語との関係だけでなく、実際の知識処理応用とオブジェクトとを関係させることも必要だろう。

- 制約論理型言語スキーム[JaLa87]とオブジェクト・スキームの関係の再整理。このように問題領域が縮み合ってきてくると、継承関係を持った非正規関係[中野88]のように、実際の応用分野の探究も必要だろう。
- “オブジェクト”の動的側面についてもすでに研究されているが[BNT88, LRV88]、さらに多くの側面との関連を探る必要があろう。ルールをオブジェクトとしたことにより、本稿の方法でも、動的側面についてある程度扱えるかも知れない。
- 単に質問応答だけでなく、更新問題[Maier86, Beeri88]など、さらに一般的な管理システムとしても考慮が必要だろう。これによって初めて、実際に、伝統的なデータベース管理システムの拡張となりうるだろう。

8 おわりに

“オブジェクト”あるいは“オブジェクト指向”という言葉は、使用する人あるいは使用する分野によって、現在はあまりに多義である。今後の複雑な応用の一層の進展は、それぞれの限定された領域についての研究から、それらを含む大きな領域での、（一般的な問題処理そのものではなく）問題解決の枠組の必要性が今後ますます求められるだろう。本稿では、そのような“多義”に対応した処理方式を提案した。

とくに、本稿では、データベースでのオブジェクトの取扱いという視点から、OODとDDの融合を目指した一般的な枠組として、特定の問題領域とは独立なスキームと、それに各領域固有の意味と操作を付加した具体例との関係を考え、またそれらに対する質問処理としての制約評価系を検討したが、さらに、実際の応用との関連を考える必要があろう。

参考文献

- [AbGr88] S. Abiteboul and S. Grumbach. “COL: A Logic-Based language for Complex Objects”. *EDBT*, in LNCS, 303, Springer, 1988.
- [Ait86] H. Ait-Kaci. “An Algebraic Semantics Approach to the Effective Resolution of Type Equations”. *TCS*, vol.45, 1986.
- [BaKh85] F. Bancilhon and S. Khoshahian. “A Calculus for Complex Objects”. *ACM PODS*, 1985.

- [Beeri88] C. Beeri. “Data Models and Languages for Databases *ICDT*, in LNCS, 326, 1988.
- [BNST87] C. Beeri, S. Naqvi, O. Shmueli and S. Tsur. “Sets and Negation in a Logic Database Language (LDL)”. *ACM PODS*, 1987.
- [BNT88] C. Beeri, R. Nasr and S. Tsur. “Embedding ω -term in a Horn-clause Logic”. in *Proc. of Third Int'l Conf. on Data and Knowledge Bases*, Jerusalem, 1988.
- [Harper86] R. Harper. “Introduction to Standard ML”. Univ. of Edinburgh, *ECS-LFCS-86-14*, 1986.
- [JaLa87] J. Jaffer and J.-L. Lassez. “Constraint Logic Programming”. *IEEE SLP*, 1987.
- [LRV88] C. Lecluse, P. Richard and F. Velez. “ O_2 , an Object-Oriented Data Model”. *EDBT*, in LNCS, 303, Springer, 1988.
- [Maier86] D. Maier. “A Logic for Objects”. in *Preprint of the Workshop on Foundations of Deductive Databases and Logic Programming*, 1986.
- [三浦87] 三浦, “非正規関係理論の動向”, 情報処理学会『アドバンスト・データベース・シンポジウム』, 12月, 1987.
- [Mukai87] K. Mukai, “Anadic Tuples in Prolog”, *ICOT-TR*, 239, 1987.
- [中野88] 中野, “フレーム束モデル”, 情報処理学会データベース研究会, 9月, 1988.
- [Ohori88] A. Ohori, “Semantics of Types for Database Objects”. *ICDT'88*, in LNCS, 326, 1988.
- [Yokota88] K. Yokota. “Deductive Approach for Nested Relations”. *ICOT-TR*, 1988.
- [横田88a] “型推論とML”. *bit*, 3月, 1988.
- [横田88b] “レコード・プログラミング”. *bit*, 12月, 1988.
- [YKK88] K. Yokota, M. Kawamura and A. Kanaegami. “Overview of the Knowledge Base Management System (Kappa)”. *FGCS'88*, 1988.