解　説

3. オブジェクト指向データベースの技術的諸問題

3.1 オブジェクト指向データベースの形式化†

大　堀　　淳††

1. はじめに

オブジェクト指向データベース (OODB) で提唱されている「オブジェクト同一性」(object identity) や「インヘリタンス」(inheritance) などの概念は，実世界の複雑な実体を自然に表現する上で有用性が高く，それらを取り入れた OODB は，データベースシステムの使いやすさ及び応用範囲を飛躍的に高めると期待されている．しかしながら，それらの概念の多くは曖昧であり，何をもって OODB と呼ぶかといった基本的なことについてさえコンセンサスが得られているとはいえない．このような状況にある OODB が，関係データベースで経験したような着実な実用化技術の蓄積を享受し，次世代のデータベースの基礎と成り得るためには，そこに含まれる種々の直感的概念を形式化し，それらを一つの整合性のある理論システムの中に統合することが必要である．この認識のもとに，OODB に形式的基礎を与えようとする多数の試みが始められている．

現状では，それらの研究は，それぞれ独自の解釈と方法論に基づく新しい提案と受けとるべきものである．また，それらが唯一のデータモデルへ収斂される動きはない．したがって「OODB の形式化の体系的な解説」といったものはいまだ存在しえない．しかしながら，それらの研究を通じて，幾つかの一般性をもった方法や技術は生み出されつつある．そこで本稿では，OODB を，オブジェクト指向プログラミングや意味データモデルなどの分野で提唱された一連の望ましい機能を実現するデータベースの総称ととらえ，その形式化の基本的な枠組の説明と最近の形式化の研究の幾つかを解説する．

† Formalization of Object-Oriented Databases by Atsushi OHORI (OKI Electric Industry, Kansai Laboratory).

†† 沖電気工業(株)関西総合研究所

2. OODB の形式化の構造

OODB は，その名が示すとおり，“オブジェクト”すなわち実世界の実体を分かりやすい形でデータベースの中に表現できるようにすることを目指している．そのための最も基本的な要件の一つは，データベースのデータとして，整数などの原子データ (atomic data) のみならず内部構造をもつ複雑なデータをも許すことである．データベースの文献では，そのようなデータは複合オブジェクト (complex object) と呼ばれる．OODB の基本は，複合オブジェクトに対して問合せなどのデータベース操作を行うシステムである．OODB は，複合オブジェクト操作システムに「インヘリタンス」，「クラス」，「メソッド」，「オブジェクト同一性」などの機能を採り入れたデータベースと理解できる．したがって OODB の形式化の目的は，これら直感的概念にデータベースシステムの枠組の中で一つの解釈を与え，複合オブジェクト操作システムと統合することと理解できる．

オブジェクト指向プログラミングはもともとプログラム言語であり，上記の諸概念の多くはプログラミング・パラダイムに深くかかわっている．したがって，それら諸概念と複合オブジェクトの操作システムを統合し OODB に形式的基礎を与える枠組としては，データモデルとプログラミング・パラダイムを融合したものが必要となる．この融合はまた，データベースとプログラム言語間のインピーダンス・ミスマッチの解消を意味し，実用上重要な課題でもある．この点に関しては，本特集号の解説「オブジェクト指向データベース・プログラミング言語」を参照されたい．

形式化の枠組となり得るプログラミング・パラダイムとしては，述語論理に基づく論理型計算と，ラムダ計算に基づく関数型計算が有力であ

る．前者は主にデータベースの研究者の間で，後者は主にプログラム言語の研究者の間で，オブジェクト指向の諸概念の形式化の枠組として使われている．

以上の理解のもとに，まず **3.** で複合オブジェクトを含んだデータ操作システムの研究を紹介する．次に複合オブジェクトの操作システムにOODB の機能を統合しようとする研究を紹介する．OODB にとっての望ましい機能については数多くの議論と提案がなされている．この点に関しては本特集の第1部を参照されたい．紙面の制約上本稿では，多くの研究者によって共通にあげられているインヘリタンスとオブジェクト同一性の二つに焦点を絞ることにする．**4.** では，これらの概念を論理型計算のパラダイムに基づいて形式化しようとする研究を，**5.** では関数型計算のパラダイムに基づいた形式化の研究を紹介する．

3. 複合オブジェクトの操作システム

複合オブジェクトの基本的な構造は，原子データからレコード構成子 (record constructor) と集合構成子 (set constructor) を任意回繰り返し用いて構成される複合項である．レコードはラベル（属性）とその値の組の集合である．本稿では，ラベル l_i の値が v_i であるレコードを $[l_1=v_1,\cdots,l_n=v_n]$ と書くことにする．ここでラベル $l_1,\cdots,l_n$ はすべて異なる．また，D を与えられた集合とするとき，D からレコード構成子と集合構成子を用いて生成される複合項の集合を $Obj(D)$ と書くことにする．これら複合オブジェクトをデータベースとして使用するための操作を定義する研究が多数なされている[1),4),11),20),25),26)]．またデータベース以外の分野でも関連ある研究がある[3),23)]．本章ではこれらの中から，複合オブジェクトの操作の研究では先駆的な複合オブジェクト計算[4)]，インヘリタンスとの融合を試みた ψ-項[3)]，関数型言語の型理論との融合を試みた複合オブジェクトの型システム[20)]の三つを紹介する．

3.1 複合オブジェクト計算

Bancilhon と Khoshafian は，複合オブジェクト集合に対する強力で一般的な操作を体系的に定義する意図のもとに，複合オブジェクトに対する操作言語を提案した[4)]．彼らのシステムの対象とする複合項は，原始定数の集合を C とすると $Obj(C\cup\{\top,\bot\})$ である．彼らの操作言語の基本要素である式 (formula) の集合は，変数の集合を V として $Obj(C\cup\{\top,\bot\}\cup V)$ で与えられる．

彼らのアイデアの要点は，式 E を，その中の変数をオブジェクトで置き換えることによって得られるすべてのオブジェクトの集合の表現と考えたことである．この集合自体もオブジェクトであることに注目すると，式 E は，与えられたオブジェクト（データベース）DB から E が表すオブジェクトと DB との交わりを取り出す，単純で一般的な操作と解釈できる．この操作の結果を，式 E のオブジェクト DB の下での解釈と呼び，$E(DB)$ と書く．彼らはこの直感的なアイデアを，複合オブジェクト集合上に束構造を定義することによって形式化した．以下その概要をスケッチする．

集合 $Obj(C\cup\{\top,\bot\})$ 上の関係 $O\sqsubseteq O'$ を，以下がなり立つ最小のものと定義する．

$$c\sqsubseteq c \text{ for all } c\in C$$
$$\bot\sqsubseteq O \text{ for all } O$$
$$O\sqsubseteq\top \text{ for all } O$$
$$[a_1:O_1,\cdots,a_n:O_n]\sqsubseteq[a_1:O'_1,\cdots,a_n:O'_n,\cdots] \quad \text{if } O_i\sqsubseteq O'_i(1\leq i\leq n)$$
$$\{O_1,\cdots,O_n\}\sqsubseteq\{O'_1,\cdots,O'_m\} \quad \text{if } \forall O\in\{O_1,\cdots,O_n\}\ \exists O'\in\{O'_1,\cdots,O'_m\}\ O\sqsubseteq O'$$

オブジェクトの同値関係を $O\sqsubseteq O'$ かつ $O'\sqsubseteq O$ と定めると，関係$\sqsubseteq$は，この同値関係が導出する同値類の集合上の半順序関係である．さらに同値類の集合はその半順序に関して束を構成する．Bancilhon と Koshafian は，この束上の最小上界を求める操作を和集合演算の一般化と解釈し，式 E のオブジェクト DB の元での解釈 $E(DB)$ に，以下のような単純でエレガントな定義を与えた．

$$E(DB)=\bigsqcup\{\sigma(E)\mid\sigma(E)\sqsubseteq DB \text{ for some } \sigma\}$$

ここに σ は変数に値づけをする関数，$\bigsqcup$は最小上界演算子である．たとえば，式 {[XCord=X, YCord=X, Color="Green"]} は，データベースの中から，XCord, YCord, Color フィールドを含み，XCord と YCord の値が同じで Color の値が "Green" であるオブジェクトをすべて取り出す問合せとして使うことができる．彼らは以上の定義を基本とし，式を含んだ規則で構成される論

理型言語を定義し，その形式的な意味論を与えている．

3.2 ψ-項の階層

Aït-Kaci は，is-a 関係を形式化しようとの意図のもとに，順序関係を用いた階層構造をもつデータの計算モデルを提案した[3]．データの集合上に定められた束上の最小上界演算子を用いてデータ操作を定義するというアイデアは，Bancilhon らの提案と共通しているが，彼のシステムには以下の三つの新しいアイデアが含まれている．第一は，内包的意味を盛り込むため，オブジェクトにユーザの意図を反映するオブジェクト名* を付加し，オブジェクト名の間のあらかじめ定められた従属関係の拡張としてオブジェクトの順序関係を定義したこと，第二は，オブジェクトの定義を任意の相互再帰的構造も扱えるように拡張したこと，第三は，部分構造の共有関係を表現するタグを付加したことである．彼の主要な貢献は，以上すべてのアイデアを，正規木 (regular tree)[8] を基にした一つのエレガントな体系として提示したことである．以下にその概略を示す．

オブジェクト名をもちレコード構成子から作られる複合オブジェクトは，属性名を枝としオブジェクト名を節点とする木で表現できる．オブジェクト名（定数もオブジェクト名として扱われる）の集合を D とし，木の根から枝をたどって得られる属性名のストリングの集合を Δ とすると，木は，関数 $\mu: \Delta \to D$ で表現できる．この表現を用いれば，複合項としては表現できない無限の構造も厳密に取り扱うことができる．正規木は，高高有限個の異なった部分木しかもたない（一般に無限な）木である．ここで，木 μ の節点 α における部分木 μ/α は以下のように定義される．

$$dom(\mu/\alpha) = \{\beta \mid \alpha \cdot \beta \in \Delta\}$$
$$\mu/\alpha(\beta) = \mu(\alpha \cdot \beta)$$

一般の無限木と違い，正規木の性質はよく知られており，正規木間のほとんどの関係は計算可能である．無限な正規木は，再帰的な構造の表現として使うことができる．たとえば，再帰構造

(rec S. stream[head=char, rest=S])

は図-1 の無限木で表現できる．これは以下の関数 μ で表現される正規木である．

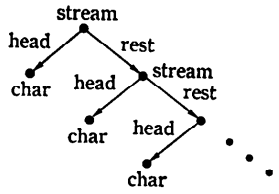

図-1　無限な木の例

$$dom(\mu) = \{(\text{rest})^n \mid n \geq 0\} \cup \{(\text{rest})^n \cdot \text{head} \mid n \geq 0\}$$
$$\mu(\alpha) = \begin{cases} \text{stream} & \text{if } \alpha = (\text{rest})^n \\ \text{char} & \text{if } \alpha = (\text{rest})^n \cdot \text{head} \end{cases}$$

Aït-Kaci の ψ-項は，正規木を，部分木の共有関係を表すタグを導入して拡張したものである．T をタグ名の集合とすると，ψ-項は (Δ, μ, τ) と定義される．ここに $\mu: \Delta \to D$ は正規木であり，$\tau: \Delta \to T$ は制約

$$\forall \alpha, \beta \in \Delta,\ \text{if } \tau(\alpha) = \tau(\beta) \text{ then } \mu/\alpha = \mu/\beta$$

を課すタグ付け関数である．彼はさらに，オブジェクト名の集合 D 上にその内包的意味を反映して順序関係が定義されていると仮定し，この順序関係を ψ-項に拡張した．この順序関係は，Bancilhon らのレコードに対する順序関係の無限構造への一般化である．さらに彼は，Huet によって開発された正規木の単一化 (unification) アルゴリズム[10]を拡張することによって，この束上の最小上界と最大下界を求めるアルゴリズムを開発し，これを基に知識ベース用の言語を定義した．

3.3 複合オブジェクトの型理論

本稿の冒頭で述べたように，データベースシステムとプログラミング言語の統合は OODB 実現上の一つの課題である．複合オブジェクトの操作システムも，プログラム言語が認識可能なデータ構造として実現されるのが望ましい．文献 20) では，この統合を意図して複合オブジェクトの操作システムのための多相型理論の構築を試みている．複合オブジェクトの操作に関しては，Bancilhon らの複合オブジェクトの順序関係と Aït-Kaci の正規木のアイデアを受け継ぎ，さらに操作システム全体を関係データベースの関係代数の一般化となるように定義している．扱われている複合オブジェクトは，レコード，集合，及びバリアント (variant) から構成可能な正規木であるが，本稿では簡単のために，レコードと集合のみを含みかつ有限な場合について説明する．

---

* 文献 3) では型とよばれているが，プログラム言語の型との混乱を避けるために言い換えた．

Bancilhon らのシステムのオブジェクト操作言語は，オブジェクト束上の最小上界演算子を和集合演算の一般化と解釈することを基礎としていた．この性質は，彼らの集合に対する以下の順序づけによる．

$S_1 \sqsubseteq S_2$ iff $\forall x \in S_1 \exists y \in S_2 x \sqsubseteq y$

これに対し，関係

$S_1 \sqsubseteq S_2$ iff $\forall y \in S_2 \exists x \in S_1 x \sqsubseteq y$

が導出する束上の最小上界演算子は集合の交わりの一般化に対応する．さらに，この順序関係をレコードの順序関係と組み合わせて得られる束上の最小上界演算は，関係代数の自然結合の操作的定義の一般化とみなすことができる．以下の性質が証明できる[5]．

複合オブジェクト O_1, O_2 が共に第一正規関係であるなら，$O_1 \sqcup O_2$ は O_1 と O_2 の自然結合である．

文献 20) では，この性質を複合オブジェクトの型の性質と組み合わせることにより，自然結合と射影を多相型をもつ複合オブジェクト集合上の演算に一般化している．オブジェクトの構造を表現する型は，原子型からレコード型 ($[l_1 : \tau_1, \cdots, l_n : \tau_n]$) と集合型 ($\{\tau\}$) によって生成される複合型である．オブジェクトの順序関係に対応して，型にも以下のような半順序関係が定義できる．

$$b \ll b$$

$$[l_1 : \tau_1, \cdots, l_n : \tau_n] \ll [l_1 : \tau_1', \cdots, l_n : \tau_n', \cdots] \quad \text{if } \tau_i \ll \tau_i' \ (1 \leq i \leq n)$$

$$\{\tau\} \ll \{\tau'\} \quad \text{if } \tau \ll \tau'$$

オブジェクトと型のこれら二つの半順序関係を用いると，関係代数の自然結合と射影は，多相型をもつ以下のような関数として，複合オブジェクトに一般化できる．

$$join(o_1, o_2) = o_1 \sqcup o_2$$

$$join : \tau_1 \times \tau_2 \to (\tau_1 \sqcup \tau_2)$$

$$project_\tau(o) = \bigsqcup \{o' \mid o' : \tau, o' \sqsubseteq o\}$$

$$project_\tau : \tau' \to \tau \quad \tau \ll \tau'$$

型に関する情報は，実際に演算を実行する以前に決定可能である．この性質により，上記の 2 操作を含む複合オブジェクトの操作システムをプログラム言語の静的型システムの中に統合することができる．

4. 論理型言語を基礎とした OODB の実現

Maier の O-logic の提案[14]以来，述語論理をプログラミング・パラダイムとして OODB を実現しようとする研究が盛んに行われている．この分野の研究の動向については解説[27]があるので，一般的な動向と参考文献についてはそちらを参照されたい．

4.1 インヘリタンスの表現

OODB がオブジェクト指向プログラム言語から継承したアイデアの中で最も重要なものの一つは，おのおののオブジェクトは一つのクラスに属し，クラスの集合は継承関係（インヘリタンス）で結ばれた階層構造を成すというものである．オブジェクト指向プログラミングでは，クラス間で継承されるものはクラスの属性及びクラスに定義されたメソッドである．OODB の形式化の研究では，以上の継承関係に加えて，意味データモデルの is-a 関係の考えを受け継ぎ，クラスに属するすべてのオブジェクトの集合間の包含関係をも意味する場合が多い．たとえば，関係 Employee *is-a* Person は Employee の集合が Person の集合に含まれることをも意味する．OODB の文献では，クラスに属するオブジェクトの集合を，論理学の用語を用いてクラスの外延 (extent) と呼ぶことが多い．OODB における以上のインヘリタンスの概念を，演繹データベースの枠組の中で最もきれいに形式化しているのは，Kifer と Lausen の提案した F-logic[13] であろう．

F-logic で扱われる複合オブジェクトは，オブジェクト名の集合 $\mathcal{O}$ からオブジェクト生成子の集合によって構成される複合項の集合 $\mathcal{O}^*$ であり，束を構成する．この束構造は **3.2** で紹介した ψ-項の集合の特殊な場合と考えることができる．

F-logic では，クラスとインスタンスの区別をせず，すべてをこの束の要素として扱う．逆にまたこの $\mathcal{O}^*$ の要素は，クラスとオブジェクトの両方に解釈できる．束上の順序関係 $o_1 \preceq_{\mathcal{O}} o_2$ は，o_2 をクラスとみるか実体とみるかにより，「o_2 は o_1 のサブクラスである」という関係と「o_2 はクラス o_1 のインスタンスである」という関係の両方に解釈可能である．したがって，与えられた要素 o に対して，集合 $\{o' \mid o \preceq_{\mathcal{O}} o'\}$ は，クラス o のサブクラスの集合という意味と，クラス o のインスタ

ンスの集合という意味の二つをもつ．この取り扱いにより，クラス外延の包含関係が自然にサブクラスの関係と融合されている．すなわち，o_2 が o_1 のサブクラスであれば，o_2 の外延は o_1 の外延に含まれるという関係が一般的に成立する．

次に，F-logic での論理式の取り扱いを検討することによって，メソッドの継承がどのように実現されているかをみることにしよう．F-logic の規則の基本構成要素である F-項は，一般に

$P: Q[f_1 \rightarrow O_1, \cdots, f_n \rightarrow O_n,$

$s_1 \rightarrow \{O_{1,1}, \cdots, O_{k,1}\}, \cdots, s_m \rightarrow \{O_{1,m}, \cdots, O_{l,m}\}]$

の形をした複合項である．ここに，P, Q は，集合 $\mathcal{O}^*$ を変数を含めて拡張した集合の要素であり，f_i は単一の値を取る属性名，s_j は集合値を取る属性名である．上記の項は直感的には，「クラス P に属し，指定された属性をもつようなオブジェクト Q」を意味する．F-logic の論理式は，この F-項から通常の論理結合子によって構成される式である．

F-logic の意味論は，$\mathcal{O}^*$ を解釈するための領域及び解釈関数によって定まる，関数を含んだ一階述語論理の一般的な意味論である．F-logic では，クラス間のメソッドの継承をこの意味論に組み込むために，「$\mathcal{O}^*$ の解釈領域が束を成し，各解釈関数は $\mathcal{O}^*$ の順序構造を保存する」との仮定をする．Kifer と Lausen の意味論の優れている点は，上記の，言ってみればごくわずかな拡張によって，メソッドの継承という高次元の概念を一階の述語論理の意味論の中で形式化することに成功したことである．$\mathbf{D}$ を F-式の集合，p, v を $\mathcal{O}^*$ の要素，Q, R を F-式とする．また $\mathbf{D}$ が Q を論理的に含意するとき，$\mathbf{D} \models Q$ と書く．彼らは以下の性質を証明した．

もし $\mathbf{D} \models p[f \rightarrow Q, s \rightarrow \{R\}]$ であり

$p \leq_{\mathcal{O}} q$ ならば $\mathbf{D} \models q[f \rightarrow Q, s \rightarrow \{R\}]$

も成り立つ．

つまり，F-logic においては，ある性質が p について成り立てば，その性質は p の任意のサブクラス（または p のインスタンス）q についても成り立つことが保証されている．この性質はメソッドの継承の述語論理の枠組の中での形式化の一つの満足ゆく回答と言えよう．

4.2 オブジェクト同一性の表現

実世界の実体は属性値の変化にかかわらずその「同一性」を保っている．オブジェクト同一性は，そのような実体の性質を表現するために提案された概念である．この概念はオブジェクトの更新や共有を簡単に表現する上できわめて有用である．オブジェクト同一性を表現するためには，オブジェクトを，その属性値とは独立に同定するためのオブジェクト識別子 (OID) の集合と，OID を含んだ複合オブジェクトに対応させる機構が必要である．これらの機構を演繹データベースの枠組みの中に取り入れることに最も成功していると考えられるのは Abiteboul と Kannelakis の提案[2]であろう．

彼らの提案の静的側面であるデータ構造の定義は，型を定義するスキーマとそのインスタンスに分けられる．彼らは OID の集合をクラスと呼び，一般の値の集合をリレーションと呼ぶ．データベース内のリレーション名の集合を R，クラス名の集合を P とする．型の集合は，P の要素及び原子型から構成される複合型である．この集合を $type(P)$ と書くことにする．スキーマは (R, P, T) の三組で表現される．ここに T は以下のような関数である．

$T: R \cup P \rightarrow type\ (P)$

P が関数 T の領域と値域の両方に現れるこの構造により，オブジェクト識別子の型付けと相互再帰的な型定義の両方が巧みに実現されている．スキーマ (R, P, T) のインスタンスは R, P, T の各要素に対応した三組 (ρ, π, ν) である．ここに ρ, π は R, P の要素に値づけをする以下のような関数である．

$\rho: R \rightarrow 2^{Obj(C \cup OID)}$

$\pi: P \rightarrow 2^{OID}$

ここで 2^X は X の冪集合である．ν は，インスタンスに含まれる各オブジェクト識別子に，その属性である複合オブジェクトを対応させる以下のような関数である．

$\nu: \cup \{\pi(p) \mid p \in P\} \rightarrow Obj(C \cup OID)$

さらにこれら三つの関数はスキーマ T によって規定された型制約を満たさねばならない．すなわち R の各要素 r にたいして，$\rho(r)$ の各要素は型 $T(R)$ をもち，P の各要素 p にたいして，$\cup \{\nu(\pi(o)) \mid o \in p\}$ の各要素は型 $T(P)$ をもつ．この構

造により，型付き複合オブジェクトとオブジェクト同一性との統合がうまく実現されている．

Abiteboul と Kanellakis の提案の動的側面は，以上のデータ構造を操作するための規則に基づく論理型言語 IQL である．IQL の基本構成要素である項は，型づけられた変数及び R, P の要素を含み，集合構成子，レコード構成子及び OID の属性値取り出し演算子 ($\hat{}$) によって作られる複合項である．IQL のルールは，否定 (¬) 及び同値関係 (=) を含んで以上の複合項から構成される拡張 Horn 節である．IQL のプログラムはそれらルールの有限な集合である．彼らの提案の最も大きな貢献は，以上の IQL 言語に形式的意味を与えることにより演繹データベースの枠組みの中でのオブジェクト同一性の種々の性質を明らかにしたことである．意味論の基本的な戦略は，否定を含んだ Datalog プログラムの意味論で用いられる infrationary fixed point semantics である．彼らはこの意味論に基づき，IQL プログラムの意味と，インスタンスの計算可能な変換との対応関係を確立した．

5. 関数型言語を基礎とした OODB の実現

述語論理に基づく演繹データベースは OODB を実現する上での一つの有力な枠組みであり，データベースの研究者の間では現在最も盛んに研究されている分野である．しかしながらこの事実によって，演繹データベースが OODB を実現する上で最も適した枠組みであると結論するのは危険である．述語論理に基づく論理型言語は計算パラダイムの中の一つであり，利点と同時にそれ固有の弱点をもっている．OODB との関連で特に問題となると考えられる弱点は，データ型の理論が未発達であることと関数などの高階のデータの取り扱いが難しいことであろう．本章ではこれらの点に優れている関数型プログラミングに基づく OODB の実現を検討する．

関数型プログラミングの基本的なモデルは，単純な型付きラムダ計算 (the simply typed lambda calculus) である．このモデルでは，プログラムもデータも共に以下のラムダ式で表現される．

$e ::= c^{\tau} | x | \lambda x : \tau . e | e(e)$

ここに c^{τ} は型付き定数，$\lambda x : \tau . e$ は関数定義，$e(e)$ は関数呼び出しである．型付きラムダ計算の解説としては，文献 17) が基礎的事項から最近の研究動向までカバーしていて推薦できる．

この枠組みの中に，複合オブジェクト及びそれに対する代数的演算子を取り込むのは論理的にはそう困難なことではない．OODB を実現する上での課題は，インヘリタンスとオブジェクト同一性（及び本稿では検討しないその他の機能）をうまく統合することである．本章では簡単のために，複合オブジェクトの構成子としてレコードのみを扱う．基本的な考えは集合やバリアントなどにも適応可能である．単純な型付きラムダ計算にレコードを追加するにはラムダ式の定義を以下のように拡張すれば良い．

$e ::= c^{\tau} | x | \lambda x : \tau . e | e(e) | [l = e, \cdots, l = e] | e \cdot l$

$e \cdot l$ はレコードからラベル l の値を取り出す演算である．

5.1 インヘリタンスの表現

メソッドの継承としてのインヘリタンスの形式化の研究は，プログラム言語の型理論の研究の中で現在盛んに研究されているテーマである．その出発点となったものは，以下に紹介する Cardelli の構造的サブタイプ理論[6]である．

5.1.1 サブタイプを用いた表現

型 point を [X : num, Y : num] で表現したとして，原点からの仰角のタンジェントの値を計算する関数

$\tan = \lambda o : \text{point}. (o \cdot Y / o \cdot X)$

を考えてみよう．古典的な型付きラムダ計算の理論では tan には point→num の型が与えられ，point 型にしか適用することができない．しかしこの関数本体は，point 型のみならず X : num, Y : num フィールドを含む種々の型，たとえば circle = [X : num, Y : num, R : num] にも適応可能な構造をしている．オブジェクト指向プログラムでは，この柔軟性を得るため，circle *is-a* object と明示的に宣言することにより tan の circle への適応を可能にしている．

Cardelli はこの柔軟性を組織的にラムダ計算の型システムの中に取り入れるため，まず *is-a* 関係を以下のサブタイプ関係で表現した[6]．

$[l_1 : \tau_1, \cdots, l_n : \tau_n, \cdots] \leq [l_1 : \tau_1', \cdots, l_n : \tau_n']$ if $\tau_i \leq \tau_i'$ $(1 \leq i \leq n)$

$\tau_1 \to \tau_2 \leq \tau_1' \to \tau_2'$ if $\tau_1' \leq \tau_1, \tau_2 \leq \tau_2'$

さらに，型システムを以下の推論規則が成り立つ

ように拡張した.

$$(\text{sub})\quad \frac{\mathcal{A} \triangleright e : \tau_1}{\mathcal{A} \triangleright e : \tau_2} \text{ if } \tau_1 \leq \tau_2$$

この規則により，型 τ をもつオブジェクトは同時に τ のすべてのスーパタイプももち，したがって望ましい柔軟性が自動的に達成される．たとえば circle≤point であるからすべての circle オブジェクトは point オブジェクトとしても扱うことができ，point に対して定義された関数を適用することができる．この理論にはいろいろな拡張が提案されている．代表的なものには2階ラムダ計算に拡張した FUN 言語[7]がある．

5.1.2 多相型による表現

Cardelli のサブタイプ理論は，オブジェクトに複数の型をもたせることによって，単純な型をもつ関数の適用に柔軟性をもたせるものであった．これに対して Wand は，ある関数が一つ以上の型に適用できるという性質そのものを関数の型の中に表現するため，ML の多相型 (polymorphic type)[15] を複合オブジェクトへ拡張することを試みた[28].

彼の提案を理解するために ML の多相型システムを簡単に振り返ってみよう．ML では，ラムダ式は明示的な型指定を含まない．たとえば同一関数は id=λx. x と書かれる．そのような ML プログラムは一般に複数の型をもつが，ML の型推論システムは，可能なすべての型を表現する型スキームを推論する．たとえば id に対しては，型スキーム $\alpha \to \alpha$ が推論される．ここに α は型集合上を動く型変数である．この型スキームは，多相型 $\forall t.\, t \to t$ の表現とみなすことができる．

以上の考えを複合オブジェクトに拡張するために，Wand は，通常の型変数に加えてレコードのフィールドの集合を動く列変数 (row variable) を導入した．これによって，たとえば関数 tan の取り得る型すべての集合を

tan : [ρ X : num, Y : num]→num

と表現しようと試みた．ここに ρ は列変数である．このアイデアは，Cardelli のサブタイプシステムより正確に関数の型を表現しようとする画期的なものであり，オブジェクト指向プログラムの形式化の研究に大きな影響を与えた．その後，Wand のシステムを改良した幾つかのシステムが提案された[12),21),22),30)☆]．Cardelli のサブタイプの理論との融合も試みられている[16),24)]．以上の基本的な考えは，レコード型のみならず，**3.3** で紹介した複合オブジェクト上のデータベース用の代数演算にも拡張でき[21)]，したがって OODB の一つの基礎になり得るものである．

☆ 残念ながら Wand の原論文[30]には致命的な誤りがあり，後の研究にそのまま利用されることはなかった．21), 29) に誤りが分析されている．

5.2 オブジェクト同一性の表現

純粋な関数型言語では，すべての式は数学的な値を表現しており，更新可能な属性やオブジェクトの共有といった概念は存在しない．しかしこれらの概念を関数型言語につけ加えることは可能である．最もよく知られた試みは，Standard ML における参照型 (reference type) であろう．これは，ラムダ式の定義に，値 v の参照を生成する演算 (new (v))，参照の値を取り出す演算 ($!r$) 及び参照の値を変更する演算 ($r := v$) を追加することによって実現される．参照間の同値関係は，それが new 演算子の同一の起動の結果であるときのみ成り立つ．たとえば同値関係をテストする演算子を eq とすると，(λx. eq (x, x)) (new (1)) の結果は true であるが eq (new (1), new (1)) の結果は false となる．この性質によって関数型言語でもオブジェクト同一性の実現が可能となる．

このアプローチの問題点は，参照型を操作する上記三つの演算子が，従来の関数型言語の理論の枠内ではうまく説明ができないことである．たとえば以下の二つのラムダ関数を考えてみよう．

f1 = λf. eq(f(1), f(1))

f2 = λf.(λx. eq(x, x))(f(1))

関数型理論の枠組みではこれら二つの関数は同一の意味をもつが，参照型を含む言語ではこの同一性はもはや成り立たない．たとえば f1 (new) は false の値をもつが f2 (new) は true の値をもつ．したがって参照型によりオブジェクト同一性の概念を関数型プログラムの枠組みの中で形式化し，複合オブジェクトやインヘリタンスなどの概念との整合の取れた融合を達成するためには，参照型演算子を含んだ言語を理解するための理論を確立する必要がある．これを目指した研究が最近始められている．それらの研究の内容の紹介は詳細にわたるラムダ計算の理論を必要とし，残念ながら本稿の範囲を越えるので，ここでは研究の方向を簡単に説明するにとどめる．一つのアプローチは，

参照型を取り込むため，従来の関数型理論の枠組みを拡張しようとする試みである．たとえば文献9）などがある．もう一つのアプローチは，参照型を含む言語を参照型を含まない純粋な関数型言語に翻訳することによって，参照型を含む言語の意味を確立しようとする試みである．この方向で有力なのは Moggi によって最近提案されたモナド理論[18]であろう．これはプログラム言語の種々の新しい機能を，古典的な理論の中で解釈する一般的な方法論を目指した研究である．文献19）では，この理論に基づきオブジェクト同一性の形式化を試みている．

6. おわりに

本稿では，論理型計算及び関数型計算を基本的な枠組みとして OODB に形式的基礎を与えようとする研究を紹介した．冒頭で指摘したように，それらの研究が一つのデータモデルに統合される動きはない．しかしこれは，必ずしも OODB 研究の否定的要素ではないと思われる．本稿でみてきたように，OODB の形式化の研究は数多くの新しい技術を生み出している．文献27）で指摘されているように，それら技術により，応用分野に応じた種々の OODB が実用化されることが期待できる．

しかしそのためには，解決しなければならない多くの課題が残されている．OODB が提供すべき機能としては，本稿で検討した，複合オブジェクト，インヘリタンス，オブジェクト同一性以外にも，たとえば，データの永続性，動的オブジェクト，メソッドの多重定義，メソッドのカプセル化などが提唱されている．これらの有用性は直観的に明らかであるが，それらを形式化し OODB のその他の基本的要素と統合する研究は，まだあまり進んでいないようである．今後の形式化の研究によって，これらの概念に対しても，それらを安全かつ効率的にデータベースシステムに取り入れる技術が生み出されることを期待したい．

謝辞 草稿に対し貴重な助言をくださった査読者と丁寧なコメントをくださった横田一正氏(ICOT) に深謝いたします．

参考文献

1) Abiteboul, S. and Hull, R.: Restructuring Hierarchical Database Objects, *Theor. Comput. Sci.*, Vol. 62, pp. 3-38 (1988). (Special issue devoted to International Conference on Database Theory, 1986).
2) Abiteboul, S. and Kanellakis, P.: Object Identity as a Query Language Primitive. In *Proc. SIGMOD Conference*, pp. 159-173 (1989).
3) Ait-Kaci, H.: An Algebraic Semantics Approach to the Effective Resolution on Type Equations, *Theor. Comput. Sci.*, Vol. 45, pp. 293-351 (1986).
4) Bancilhon, F. and Khoshafian, S.: A Calculus for Complex Objects. In *Proc. Symp. on Principles of Database Systems* (1986).
5) Buneman, P., Jung, A. and Ohori, A.: Using Powerdomains to Generalize Relational Databases, *Theor. Comput. Sci.* (To Appear). Available as a technical report from University of Pennsylvania (1989).
6) Cardelli, L.: A Semantics of Multiple Inheritance, *Inf. Comput.*, Vol. 76, pp. 138-164 (1988). (Special issue devoted to Symp. on Semantics of Data Types, 1984).
7) Cardelli, L. and Wegner, P.: On Understanding Types, Data Abstraction, and Polymorphism, *Comput. Surv.*, Vol. 17, pp. 471-522 (1985).
8) Courcelle, B.: Fundamental Properties of Infinite Trees, *Theor. Comput. Sci.*, Vol. 25, pp. 95-169 (1983).
9) Felleisen, M. and Friedman, D. P.: A Calculus for Assignment in Higher-Order Languages. In *Proc. Symp. on Principles on Programming Languages*, pp. 314-325 (1987).
10) Huet, G.: *Résolution d'équations dans les langages d'ordre 1, 2, …ω*, PhD thesis, University Paris (1976).
11) Hull, R. and Yap, C. K.: The Format Model: A Theory of Database Organization, *J. ACM* Vol. 31, No. 3, pp. 518-537 (1984).
12) Jategaonkar, L. A. and Mitchell, J. C.: ML with Extended Pattern Matching and Subtypes. In *Proc. Conference on LISP and Functional Programming*, pp. 198-211 (1988).
13) Kifer, M. and Lausen, G.: F-Logic: A Higher Order Language for Reasoning about Objects, Inheritance and Schema. In *Proc. SIGMOD Conference*, pp. 134-146 (1989).
14) Maier, D.: A Logic for Objects. In *Proc. Workshop on Deductive Databases and Logic Programming* (1986).
15) Milner, R.: A Theory of Type Polymorphism in Programming. *J. Comput. Syst. Sci.*, Vol. 17, pp. 348-375 (1978).
16) Fuh, Y-C. and Mishra, P.: Type Inference with Subtypes. In *Proc. European Symp. on Object-oriented Programming*, pp. 94-114 (1988).
17) Mitchell, J. C.: Type Systems for Programming Languages, A chapter in *Handbook of Theoretical Computer Science*, van Leeuwen. et al. eds.,

North-Holland (1990).
18) Moggi, E.: Computational Lambda-Calculus and Monads. In *Proc. Symp. on Logic in Computer Science* (1989).
19) Ohori, A.: Representing Object Identity in a Pure Functional Language. In *Proc. International Conference on Database Theory*, pp. 41-55 (1990).
20) Ohori, A.: Semantics of Types for Database Objects, *Theor. Comput. Sci.*, Vol. 76, pp. 53-91 (1990). (Special issue devoted to International Conference on Database Theory, 1988.)
21) Ohori, A. and Buneman, P.: Type Inference in a Database Programming Language. In *Proc. Conference on LISP and Functional Programming*, pp. 174-183 (1988).
22) Rémy, D.: Typechecking Records and Variants in a Natural Extension of ML. In *Proc. Symp. on Principles of Programming Languages* (1989).
23) Rounds, W.: Complex Objects and Morphisms I. A Set-Theoretic Semantics, Technical report, University of Michigan (1989).
24) Stansifer, R.: Type Inference with Subtypes. In *Proc. Symp. on Principles of Programming Languages*, pp. 88-97 (1988).
25) Tanaka, K. and Yoshikawa, M.: Towards Abstracting Complex Database Objects: Generalization, Reduction and Unification of Set-Type Objects. In *Proc. International Conference on Database Theory*, pp. 252-266 (1988).
26) 横田：オブジェクトの形式化とスキーマ，電子情報通信学会データ工学研究会，*DE* 88-25，京都，11月 (1988).
27) 横田，西尾：演繹・オブジェクト指向データベース，情報処理，Vol. 31, No. 2, pp. 234-243 (1990).
28) Wand, M.: Complete Type Inference for Simple Objects. In *Proc. Symp. on Logic in Computer Science*, pp. 37-44 (1987).
29) Wand, M.: Corrigendum: Complete Type Inference for Simple Objects. In *Proc. Symp. on Logic in Computer Science* (1988).
30) Wand, M.: Type Inference for Records Concatenation and Simple Objects. In *Proc. Symp. on Logic in Computer Science*, pp. 92-97 (1989).

(平成3年1月7日受付)

大堀 淳 (正会員)

1957年生．1981年東京大学文学部哲学科卒業．同年沖電気工業(株)入社．1989年米国ペンシルバニア大学計算機・情報科学科博士課程修了．Ph. D. 1989年10月より1年間，英国王立協会よりリサーチフェローシップを受け，グラスゴー大学にて客員研究．1990年10月より，沖電気工業(株)関西総合研究所勤務．プログラム言語の型理論，データベース・プログラム言語，データモデル理論，オブジェクト指向データベースなどに興味をもつ．