

位相情報空間型データベースシステム

阪大 工

打浪 清一

手塚 慶一

〔I〕はじめに

意味の包含, 遠近関係と記述する言語モデルとして, 位相情報空間モデルを提案し, その Formal のモデル化とその数学的性質について検討してきた。またこれをデータモデルに拡張し, 意味的の遠近, 包含関係と記述, 処理できる DBMS の構成について検討してきた。学術研究用の DBMS は, 既知知識の管理用と, 生データの整理・管理用の二種類が必要である。筆者らは提案したデータモデルに基づく Private Research Data Management System として Conceptual Schema-free Data Base Management System を提案, その試作を行ってきた。もう一つの Center Data Management System については次のように考へる。7月14日位相情報空間モデルにおける位相情報空間は知識のユニバースを表し, 辞典・辞書として働くとともに, 各 Private Research File 間の

インターフェースとデータの共通軸を提供する。

本稿では、これら二種のデータベースと統合するシステムの構成とその言語についてのべる。

位相情報空間は、意味地図であって、之含さるる概念は之含する概念の部分空間として表現され、意味的に近い概念は、空間中の近い部分空間として表現される。之含関係はある意味では集合関係ともみられる。一般に遠近と之含の二つの性質と同時に記述、処理することは難しく、この二者を十分取り扱うDBMSはまだ見られない。本モデルはこの二つの性質の記述、処理と同時に行うことを目的としている。

位相情報空間モデルにおいては、ある視点を定めると、そのレベルでの認識対象としての個体が定まる。その個体はそのレベルでの位相情報空間内の点あるいは領域を占める。その空間を構成する各座標軸はその個体の属性を表す。それが一段下のレベルの個体であることも多い。そして個体の点と各座標軸に射影したものが各属性の属性値となる。個体の認識レベルは相対的なものであり、P. Chenの個体・関係性モデルにおける関係性に対応するものも、視点をそこに移せば個体と考えられる。視点を置かれたレベルに対し、これら個体間の関係として生じた事象は関係となる。よって関係—個体—属性—属性値—という階層の…し関係は相対的なもの

で、現在視点がおかれているところが、個体レベルとなる。

〔II〕位相情報空間モデル概説

生成意味論の立場でモデル化している。位相情報空間生成文法と基本規則として定義する。この文法により生成された空間と位相情報空間とよび、この空間が意味地図となる。位相情報空間生成文法から統辞生成文法と求める写像が存在し、この写像を用いることにより各概念領域はそれらの表す語、句、または文に変換される。

表層の統辞文法では、生成された文の間に何等直接的な意味関係は陽に表われていないので、ただ単に文の集合と生成すればよいが、空間生成文法では点のみは領域の生成だけでなく、その位相も併せて生成しなければならない。ところが使用時には、位相ではなく特定概念領域を指示すればよいから、結局生成規則には空間全体の知識構造を生成するものと、伝達したい個々の概念を生成するものと2種預める。ユニバース生成規則は位相空間全体（構成要素集合とその上での位相）を生成し、発話・意味解釈時には概念生成規則で特定概念領域を生成する。

位相情報空間の各座標軸は何かも同じ位相としているとは限らない。そこで空間生成文法で生成する位相に関して、

教育漢字約1000字の表現する意味について分析した結果、次のような位相の種類が少くとも必要であることが分った。

(1)数軸 例之は身長、体重のように実数値ある…は整数値
 などで表現されるもので、座標軸は、 R , N , ある…
 は通常の両閉区間、開区間等で定義される。

(2)循環数軸 例之は月の名前：1月, 2月, …, 12月,
 1月, … のように数値で表現され、その値が循環する
 もの。これは(1)の数軸の両端点を同一視することによ
 り定義される。

(3)順序軸 例之は、温度感覚軸：寒い, 涼しい, 快適, 暖
 かい, 暑い… のようにその性質が順序づけられた序列
 として表わされるもの。この軸では順序のみが意味を
 持つ。距離は考へない。これは要素の順序づけ(た列
 挙法)により定義される。

(4)循環順序軸 例之は 四季：春夏秋冬春… のようにそ
 の性質が循環する序列で表わされるもの。これは(3)の
 両端点を同一視することにより定義される。

(5)より一般的位相 例之は 赤, 青, 黄, 緑… のような
 色の関係のように何等かの位相的関係があるが、上
 記(1)~(4)では表現できないもの。これらは何等かの位
 相幾何学的方法により定義される。

(6) 単なる集合 位相関係が無く、単なる寄せ集めのもの。

よって位相情報空間生成文法においては、空間生成時にこれらの属性の上記タイプに合わせて、それに見合った位相の座標軸を生成してゆくように文法を定める。

位相情報空間生成文法は、生成規則の適用に制限をつけた次の3レベルの文法からなる。

[定義1] 空間生成文法 G_1

$$G_1 = \langle V_{N1}, V_{T1}, P_1, S_1 \rangle \quad (1)$$

$$\text{ここで } V_1 = V_{N1} \cup V_{T1}, \quad V_{N1} \cap V_{T1} = \emptyset \quad (2)$$

V_{Ni}, V_{Ti}, P_i, S_i は第 i レベルの中間語彙, 終端語彙, 生成規則, 初期語彙と表す。また他レベルでも(2)に対応する式が成立するとする。

P_1 は次の形の規則からなる。

$$P_1: A \rightarrow \omega \quad A \in V_{N1}, \omega \in L^\Delta(V_1, D) \quad (3)$$

$$L^\Delta(V, D) \stackrel{\text{def}}{=} \{ (\alpha \delta)^* \beta \mid \alpha, \beta \in V, \delta \in D \} \quad (4)$$

$$D = \{ \cdot, \#, | \} \quad (5)$$

G_1 は情報空間を構成する部分空間間の関係と規定するもので、 D は空間構成子で、その種類としては、直積 \cdot , 連結和 $\#$, 直和 $|$ がある。

[定義2] 空間性質規定文法 G_2

$$G_2 = \langle V_{N2}, V_{T2}, P_2, S_2 \rangle \quad (6)$$

P_2 は次の形の規則からなる。

$$P_2: A \rightarrow w \quad A \in V_{N2} \quad a, b \in V_{T2} \quad (7)$$

$$w \in \{ a, \bar{a}, aa, aa^+, aba^+b^+, aba^+ \} \quad (8)$$

G_2 は各部分空間の性質を規定するもので、その性質は w の要素で定まるが、その種類には、

(i) 直線軸 (a) $\mathbb{R}, \mathbb{N}, \mathbb{Z}$ またはその区間

(ii) 円 (a) (i) の直線軸の両端点と同一視したものの。

以下位相幾何学が多様体と表す多角形式の

(iii) 射影平面 (aa)

(iv) 球面 (aa^+)

(v) 一円の輪環面 (aba^+b^+) トーラス

(vi) 一円面 (aba^+)

等の制限型のみである。このレベルで座標軸を組み合わせて部分空間を構成する方法を定める。

[定義3] 位相規定文法 G_3

$$G_3 = \langle V_{N3}, V_{T3}, P_3, S_3 \rangle \quad (9)$$

P_3 は知識全宇宙を生成する P_{3T} と、空間内の領域を生成する

3 P_{3A} とからなる。

$$P_{3T}: A \rightarrow \omega \quad \omega \in \{ \text{各型} \text{の位相空間} \} \quad (10)$$

$$P_{3A}: A \rightarrow \omega \quad \omega \in \{ \text{位相空間の領域} \} \quad (11)$$

G_3 は部分空間の位相を与えるもので、その位相の型として、次のようなものがある。

$T_0, T_1, T_2, R, T_3, CR, T, N, T_4, CN, T_5, FN, PN, T_v, L, C, CH, LC, C_n, U, S, QM, M, B, \dots$

以上の3レベルを統合して、位相情報空間が生成される。

[定義4] 位相情報空間生成文法 G_I

$$G_I = \langle G_1, G_2, G_3 \rangle \quad (12)$$

ここで3レベルの間の関係は次のように定められる。

$$S_1 = S, \quad S_2 = V_{T_1}, \quad S_3 = V_{T_2} \quad (13)$$

各レベルの適用規則に制限をのけることにより、種々の空間が構成される。

[定義5] 位相情報空間 I

$$I = I(G_I) \quad (14)$$

位相情報空間生成文法 G_I から生成された位相空間 I と、位相情報空間という。このとき第3レベルでは、 P_{3T} のル-

ルを用いる。この空間の言語モデルにおいては基礎となる。

データモデルのデータベースへの応用においては、この空間の Conceptual Schema に対応し、位相情報空間と本文法が、この Schema を記述するのに用いられる。

[III] 位相情報空間型データベースシステムでのデータ表現

データベースで取り扱ったデータの種類のほか、宣言的知識、手続的知識がある。本システムではこの二者を次のように扱う。

宣言的知識：概念間の遠近、包含関係をもとに、その関係を反映する形で空間内の領域として配置する。

手続的知識：手続間の遠近、包含関係（定義域、値域の包含関係、計算機能の包含関係）をもとに、その関係を反映する形で空間内の領域として配置する。手続とそのものは、その領域にストラされた。

取り扱ったデータは、別の観点から、内包的知識と外延的知識に分けられる。この取り扱いは次のように表される。

内包的知識：情報空間内に配置されている各概念（点あるいは領域）に対し、各座標軸へ射影し、その

座標値を求めるとき、内己を知るときに相当する。

外延的知識：集合あるものの領域が与えられたとき、それを含む要素概念（点集合）を求めるとき、また逆同様の処理が相当する。

本システムはこのように 宣言的知識、手続的知識、および、内己的知識、外延的知識とカバーするよう考えられているが、どちらかというに宣言的知識を主に考えている。これに対し手続的知識を主として考えると、S-M変換可能な言語とベースとするシステムの方が処理が行いやすい。その積りシステムについては、次の「外延と内己を取り扱える拡張データベースシステム」で議論する。但しまだ第1版なので手続的の遠近と反映する形での格納はまだインプリメントされていない。

データベースに對するデータ群の特性について、2つの構造を考えられる。Inter-entity structure と、Intra-entity structure である。

この2つのデータ構造の取り扱いは次のように行なう。

Inter-entity structure : 個体間構造

データの一対一が構造が異なるとき、一対一のデータと記録するのに一対スキーマが必要となり、効率が悪い。このとき構造と生成する文法を定義し、この文法を用いて構造と生成し、その生成樹 (Generation Tree 又は Derivation Tree) を抄録とする。これと使用生成規則、使用終端語彙で表現し、データベース化したものか。「内容検索可能な画像データベースシステム」であり、そのパイロットシステムとして、植物図鑑データベースシステムと構成し、その有効性を確認した。

この段階では、内部構造の抄録であり、各個体の内部構造の類似性を調べて、個体間の関係、類似性を求めれば、Inter-entity structure は、次の Intra-entity structure に変換できる。

Intra-entity structure : 個体内構造

データ群は同じ構造をしており、内容 (属性) のみが異なるとき、構造をスキーマで記述し、個々のデータは Occurrence として取り扱う。これは従来のセリオの DBMS で実現可能である。

位相情報空間型データベースシステムでは、個体から始まり、集合、線、領域、更にこれらが再帰的に入ったものとして、データタイプが定義される。これは個体内構造であり各要素間の包含、遠近関係などは個体間構造である。

[定義6] 個体

実体を持つか、あるいは概念上のもので、他のものと区別して示さしめるものという。位相情報空間内の一点もしくはある領域を占める。それ故各個体には幾つかの属性(空間の座標軸に対応)が付属し、更にその属性の属性値が対応により定まる。

[定義7] 集合

ある個体(更には Recursive に集合等も入り得るか)の集合が Abstraction により一段上のレベルの一つの個体とみなされたとき、これを集合という。集合においても、属性、属性値が付随するか、属性値が個体等となる場合もある。

[定義8] 線

ある個体(更には Recursive に集合等も入り得るか)の順序づけられた組が抽象化されて一つの個体とみなされたとき、これを線という。最も下位レベルの線は、各座標軸上に並んだ属性値の線である。

[定義9] 領域

位相情報空間内の、(閉)部分空間を領域という。これは一般に個体の Generalization によってできた個体に相当する。一般に概念の階層構造は、個体と領域の関係として表現される。即ち領域の種概念、個体の種概念に対応する。この関係も Recursive に定義されるので、視点が変われば種概念が種概念とわり更に下位レベルの種概念も定義され、又逆に上位方向への視点の移動もあり得る。

ある種概念の内包は、直接上位の種概念の内包に種差と加えたものに等しい。ここで種差とは、ある種概念と同一の種概念に属する他のすべての種概念から区別する特徴、即ち当の種概念の特有の性質という。それ故領域は、その領域を構成する個体集合に対し、領域の内包即ち各属性の属性値の組に、種差の属性を加えると、個体の各属性の属性値の組が得られる。換言すればある軸を無視して出来る部分空間は、Generalization による領域と構成している。

これに対し Aggregation は、属性から空間を構成し個体を形造るプロセスに相当する。換言すれば各属性は個体の部分を表している。

Role は個体の一属性として入ってくるのが普通である。

それが抽象化された個体として取り扱われることもある。

M. L. Brodie の Association is member of relation である。これは個体集合と Role で一つの個体を作ることに相当する。例之は従業員集合と「組合」という Role で職員組合という一線上の個体を作れる。

同一の種概念(領域)に含まれる種概念(部分領域あるものは個体)は互に同位概念となり、選言的概念、反対概念、矛盾概念等も考えられる。これらは Generalization を行って無視した属性軸において、その直交表現や、両端点の表現等に相当し、位相情報空間データベースではこれらの概念の導出も可能である。

一つの概念に別名がある場合、また換言した場合、これらは同一概念と見る。このときその概念領域には複数個の表層の単語または句や節が対応する。例之は「両親」を表す概念領域は、「父母」の表す領域と等しい。

内之は異なり外延は等しい概念は等価概念というが、例之は日本の現首相と鈴木氏というのがこれに当る。これは、前述の Role の処理で記述できない場合が多いと考えられる。

[IV] 位相情報空間型DBMSの言語

位相情報空間型DBMSには、学術研究用のPRDMSと、
 Conceptual Scheme-free DBMSと、CDBMSと
 してのDBMSがある。これら2つの言語はできるだけ共通
 性があるのが望ましい。そこで共通的な骨格部分と現在設計
 中である。以下DDL, DMLについて概略をのべる。

本システムで取り扱うデータの対象としては、事務データ
 のみならず、意味データの取り扱いに大きな重点をおいてい
 る。

(1) 位相情報空間型DBMSのDDL

位相情報空間生成文法によって生成される知識ユニバース
 (P_{3T} を用いて生成された空間) を記述する際には、最低階
 級のものも記述する機能が必要である。

空間と空間の合成法(空間と空間の間関係)

空間の構成法(空間を構成する軸と、その組合せ方)

各軸の詳細

これらを記述する為のDDLとして次の2つのレベルを設
 ける。何れかのレベルで記述してもよい。

(i) Conventional level

正確には全体の Syntax と BNF で記述すべきである。
 ここでは 大切なポイント部分のみを文脈加わりで示す形を示す。

各軸の詳細

PROPERTY <property name>, <property ID #> ; <comment>

[ALIAS(<property alias>)*]

TOPOLOGY TYPE <topology type> ,

TYPE <data type> <data expression length>

[, RANGE <range>]

[, <occurrence>]

[, KEY] [, NONULL]

[, DEFAULT <default value>]

[, ORDERING <sorting key>]

[, UNIT <unit>] :

<topology type> ::= LINE | CYCLINE | ORDER | CYCLORDER |

GTOPOLOGY | SET

<data type> ::= INTEGER | REAL | LOGICAL | CHARACTER | KANJI |

BIT

<data expression length> ::= <integer>

<range> ::= <lower limit> : <upper limit>

<lower limit> ::= [<number> | (<number>

<upper limit> ::= <number>] | <number>)

<number> ::= <integer> | <real> | <TF-value> | <special value>

<default value> ::= <number>

ここで $\langle \text{role} \rangle$ とは、空間中に表現される個体に対し、
extended property の \rightarrow 役割をいう。

空間の構成は Recursive に行われる。低位のレベルでは
空間を構成する要素として属性が入るが、次第に上位のレベ
ルにゆくに従って、個体や部分空間と要素として空間が定義
される。ある空間が2通り以上に分解されるとき、その空間
は、各分解の積で定義される。例之は人間という空間は男、
女という性別の部分空間に分割され、同時に年齢別の部分空
間に分割される。

これは例之げ次のように記述される。

SPACE MAN, E1; NINGEN WO ARAWASU KUUKAN

ALIAS HITO, NINGEN,

SPACE TYPE ENTITY,

UNIQUE,

UNIT NIN,

(NAME, ID, KEY, AIUEO) · (SEX) · (AGE) : 空間構成

空間 E1. 人間

別名 → 人

空間型 → 個体

→ 一価

単位 → 人

PROPERTY NAME, P1 ; NAMAE

ALIAS NAMAE, SEIMEI,

TOPOLOGY TYPE ORDER,

TYPE CHARACTER 20,

属性 P1. 名前

別名 → 姓 名

位相型 → 順序型

型 → 文字型 20 字

UNIQUE,	→ - 西
KEY, NONULL,	キー, 又ル値は許さず.
ORDERING AIUEO :	順序 → アイウエオ順
PROPERTY SEX, P2 ; SEIBETSU	属性 P2. 性
ALIAS SEIBETSU, SEI,	別名 → 性別
TOPOLOGY TYPE ORDER,	位相型 → 順序型
TYPE CHARACTER 1,	型 → 文字型 1 字
UNIQUE,	→ - 西
ORDERING [M,F] :	順序 → M, F の順
PROPERTY AGE, P3 ; NENREI	属性 P3. 年齢
ALIAS NENREI, TOSHI,	別名 → 年齢, 年
TOPOLOGY TYPE LINE,	位相型 → 線
TYPE INTEGER 3,	型 → 整数 3 桁
RANGE [0,150],	値域 → [0, 150]
UNIQUE,	→ - 西
ORDERING ASCENDING,	順序 → 昇順
UNIT SAI:	単位 → 才

Conceptual Scheme の作成時には, この 3 の Syntax の前に, CREATE とついで CREATE SPACE, CREATE PROPERTY のように書く。修正時には, この 3 の Syntax の前に MODIFY とついで MODIFY SPACE, MODIFY PROPERTY のように書く。修正部分のみを記述する。

利用時における External Schema の作成時には、これは a Syntax の前に LET とついで LET SPACE, LET PROPERTY のように書く。

(ii) Generative Rule Level

位相情報空間生成文法と同様に記述してゆく方法である。生成規則を記述し、その生成規則に表われる各語の定義を行う方法で記述する。記述内容は、Conventional Level とほぼ同じである。

```

<space name> → ( <extended property name> [ , role ] )
                [ <space constitutor>
                  ( <extended property name> [ , role ] ) * ] :
WHERE <space name> IS <space ID #> ; <comment>
                [ ALIAS ( <space name> ; ) * ]
                [ SPACE TYPE <space data type> ]
                [ , <occurrence> ]
                [ , UNIT <unit> ] :
                [ <extended property name> IS
                  [ , <key property> ]
                  [ , <sorting order> ] ] :

```

```

<extended property name> ::= <subspace name> [ . <ID name> ] |
<entity name> [ . <ID name> ] | <property name>

```

<space constitutor> ::= # | | | _
 <space data type> ::= ENTITY | SET | LINE | AREA | ...
 <occurrence> ::= UNIQUE | MULT | VARIABLE TIMES
 <key property> ::= KEY | COMPOUND KEY <i> | NON KEY

ここで $\bar{}$ は default 値を表す。

<property name> → <topology type> <topology expression>
 WHERE <property name> IS <property ID # >; <comment>
 [ALIAS (<property alias> ,) *]
 [, <occurrence>]
 [, KEY][, NONULL]:

<topology type> ::= LINE | CYCLINE | ORDER | CYCORDER |
 GTOPOLOGY | SET

<topology expression> ::= <var.> | <var.i> <var.i>
 <var.i> <var.j> <var.i>⁻¹<var.j>⁻¹ | <var.i> <var.i>⁻¹ |
 <var.i> <var.j> <var.i>⁻¹

ここで <var.k> は 軸 k を表す変数である。

<var.k> → <data type> <data expression>
 WHERE <var.k> IS <axis ID # >; <comment>
 [ALIAS (<axis alias> ,) *]
 [, <occurrence>]
 [, KEY][, NONULL]:

```

<data type> ::= <type> <length> <exp.unit>
<type>      ::= INTEGER | REAL | LOGICAL | CHARACTER | KANJI |
              BIT | ...
<exp.unit>  ::= BYTE | BIT | CHARACTER | ...
<data expression> ::= RANGE <range>
                [ , DEFAULT <default value> ]
                [ , ORDERING <sorting order> ]
                [ , UNIT <unit> ]
<range>     ::= <lower limit> : <upper limit>
<lower limit> ::= [<number> | (<number>
<upper limit> ::= <number>)] | <number> )
              :
              :

```

以上の → のついたレベルの生成規則を定義して空間の構成を定め、その詳細は、生成された文および WHERE 句内で定義する。

(2) 位相情報空間型DBMSのDML

位相情報空間生成文法により生成される概念スキーマを、DDLで記述した加、その上での処理を行う際のDMLには、他のモデルで持つ機能に加えて、次の機能が必要であり、またこの機能を持つことにより、他のモデルではシステムがサポートすることのできない意味的な述述、包含関係と取り扱う処理が可能となる。

- ① 意味的に近い概念、遠い概念をアクセスする機能
- ② 反対語、同義語、対語的な概念をアクセスする機能
- ③ 意味的に包含関係にある概念をアクセスする機能
- ④ 全知識が一々にまとまっていたので、知りたいところだけを切り出す Window 的な機能
- ⑤ Window から眺めたデータと、どの視点から見よかという Aspect を指定する機能

これらの機能は他のデータモデルに基づくデータベースでは、十分にサポートすることは、おそらく不可能である。

DMLも次のように 情報代換系レベル, Conventional level, 自然言語風レベル, 空間表示レベル等々のレベルがある。

(i) ファイル情報代数系レベル

Conceptual Scheme-free DBMS を設計した際には、データ統合の為に設定したファイル情報代数系を用いて記述するレベルである。

これは更に2つのレベルに分けられる。一つはファイル情報代数系を用いて欲しいデータを未知数とする方程式と記述し、システムがこれを解いて答えるレベルで、非手続的の高級なレベルである。他の一つはファイル情報代数系の性質を用いながら、処理はこの上で手続的にプログラムしたものとの翻訳、実行するレベルである。このレベルの言語はここでは省略する。

(ii) Conventional level

積極的に External Scheme を記述し、これを用いて処理を行うレベルで、現在普通の DBMS が行っている方式によるものである。

このレベルでは、処理対象を個体とみだし、その属性、属性値の3つ組のうち、何れかを陽に与え、指定した部分と求めた処理が主になる。この処理を行うときには視点が与られて関係性が入ってくる場合がある。そのときは関係と個体とみだし、個体はその属性とみて処理を続けてゆく。

このレベルのデータ検索のコメントの核部分と次に示す。

FIND <var. list> WHERE <conditional clause> ... query st.

<var. list> ::= <var.> | <var.>, <var. list>

<conditional clause> ::= <e.r.f> | <e.r.f> <l.op> <cond. exp.>

<e.r.f> ::= $\left\{ \begin{array}{l} e=<ext. data-item> \\ p=<ext. data-item> \\ v=<ext. data-item> \end{array} \right\}$

$\left\{ \begin{array}{l} e=<ext.data-item> \left(\begin{array}{l} <l.op> <e= ext. data-item> \\ : <p= ext. data-item> \\ : <v= ext. data-item> \end{array} \right)^+ \\ p=<ext.data-item> \\ v=<ext.data-item> \end{array} \right\}$

where *e.r.f* means *entity restricting formulae*

<cond. exp.> ::= <variable><r.op><variable>

<variable> ::= <string> | <functional exp.> | <cond.exp>

<functional exp.> ::= <function>(<expression>)

<function> ::= <set function> | <entity function> | <line function> |
<area function> | ...

<set function> ::= COUNT | MAX | MIN | ...

<line function> ::= LENGTH | RANGE | DOMAIN | ...

<area function> ::= AREA | DIAMETER | ...

<entity function> ::= NO_OF_PROPERTIES | ...

<ext. data-item> ::= [<determiner>]<data-item> [;<string>]

<l.op> ::= AND | OR

<determiner> ::= SOME | EACH | NOT | ALL

<r.op> ::= < | ≤ | > | ≥ | = | ≠ | ⊃ | ⊂ | ≈ | ↔

Ex.

$$\text{FIND } d, \text{AVG}(x) \text{ WHERE } \left\{ \begin{array}{l} e=\text{EACH dept}; d \text{ AND } y \\ p= \text{ emp} \quad : \quad \text{salary} \\ v= \quad y \quad : \quad x \end{array} \right\}$$

$$\text{FIND } x \text{ WHERE } \left\{ \begin{array}{l} e=\text{dept}; z \text{ AND} \quad \text{AND} \quad \text{item}; y \\ p=\text{item}; y \quad : \quad \text{floor} \quad : \quad \text{dept} \\ v= x \quad : \quad 2 \quad : \quad z \end{array} \right\}$$

$$\text{AND COUNT}(z) \geq 2$$

- ・各部門の従業員の平均給料を求めよ。
- ・2階にある少なくとも2つの部門によって売られている品目を求めよ。

このレベルでは検索しては視点を移し、別の観点から検索し、更に視点を移す形で Traverse しながら検索を行ってゆく。これは $\langle e, r, f \rangle$ の中で変数名を共通にするこゝに於ける。 $\langle e, r, f \rangle$ では常にその時点での処理対象を e , r の属化、属性値を p, v として表す。

r, op の \supset , \subset は概念の包含関係を表し, \approx は概念的近似性を表し, \leftrightarrow は反対概念を表す。

関数には, その定義域により, 個体型, 集合型, 線型, 領域型, 9人7がある。例として LENGTH は線の長さを求める関数で, AREA は領域の面積を求める関数である。

(iii) 自然言語風レベル

External Schema は記述せず、質問文の中に適当に Scheme 情報を入れ、曖昧さをなくして処理を行うレベルである。

知識辞書として、辞書や辞典を位相情報空間生成文法に基づいて解析し、データベースを構築し、それと利用する場合を想定する。この場合、毎回 External Scheme を記述するのは現実的でない。そこで自然語に近しい形で質問を出すか、曖昧さの生じような形では、External Scheme に類似したデータを陽に記述し、質問を書くためのレベルである。

DML で最もよく用いる検索に関する部分で、他のシステム、関係モデル型や、転置型 DBMS では記述しにくい、あるいは記述できない部分について重点的にのべる。

```
PRINT <var.list> FIND [ <functional expression>
                       [ <var.list> <restricting clause> ]
```

```
<var.list> ::= <var.> | <var.> , <var.list>
```

```
<var.> ::= [ITS] [ <determiner> ] <e.e.p> [ ; <string> ] { <o.r> <var.> } *
                                                [ <l.op> <var.> ]
```

```
<determiner> ::= ALL | SOME | EACH
```

```
<e.e.p> ::= <extended entity> | <extended property>
```

```
<o.r> ::= OF | RELATED TO
```

```
<l.op> ::= AND | OR
```


<condition> ::= <functional expression> | <var.><r.op><v.n.f.i>

<r.op> ::= > | ≥ | < | ≤ | = | ≠ | ≈ | ≠ | ↔

<viewpoint> ::= <ext.property> | <ext.property><viewpoint>

<substituting state.> ::= ← <query statement>

<judge statement> ::= IS IT TRUE <functional exp.> ? |

IS IT TRUE <var.list><restricting clause> ?

<restricting clause> が何重にも重なり得ることは、後から直前へかき足す制約をつけてゆくように解釈する。

<function> には、位相情報空間モデルの各データタイプに対応したものがある。他のDBMSでは<set function>が無い。

<restricting clause> の意味は、<ext. entity>を対象として、<viewpoint>で指定された観点から、換言すると、<ext. entity>が位置する空間中の<viewpoint>で指定された座標軸に関して<restriction>で与えられた条件で検索することを示している。

<restriction>は位相情報空間を読む命令であって、次のようなものがある。

CONTAINED) は Generalization 関係を読み、<viewpoint>で指定された属性において、<ext. entity>を含む(に含まれる)ものを検索する。

BELONGED) は集合と要素の関係 \in (\ni) にあるもの
の検索である。

CONSTITUTE(D) は Aggregation 関係を読み、 $\langle \text{viewpoint} \rangle$
で指定される属性において $\langle \text{ext. entity} \rangle$ を aggregate
した (\sum) ものを検索する。

COVER (ED) は、領域又は集合における包含関係を検索
する。例として、政界 A, B, C を COVER する人を探
る。

NEAR は空間中 $\langle \text{ext. entity} \rangle$ に近いものを検索する。
($\langle \text{viewpoint} \rangle$ で指定される属性に関して)。

CONTRARY は空間中 $\langle \text{viewpoint} \rangle$ で指定される属性にお
いて対極にある反対概念を検索する。

COORDINATE は $\langle \text{viewpoint} \rangle$ で指定される属性レベル
で $\langle \text{ext. entity} \rangle$ に属する同レベルの概念を全て検索
する。

DISJUNCTIVE は COORDINATE とよく似ているが、
互に素の概念の集合を検索する。

CORRELATIVE は $\langle \text{viewpoint} \rangle$ の観点で対となる相対立
する概念を検索する。

CROSS は $\langle \text{viewpoint} \rangle$ 軸上で、 $\langle \text{ext. entity} \rangle$ と交わる概
念を検索する。

EQUIPOLLENT は $\langle \text{ext. entity} \rangle$ 空間中で $\langle \text{viewpoint} \rangle$ の観点から見て、換言すれば他の属性は無視して、内包は異なり、外延も同じものを検索する。

IDENTICAL は $\langle \text{ext. entity} \rangle$ 空間中で $\langle \text{viewpoint} \rangle$ の観点から内包、外延とも同一のものを検索する。

これらの Restriction は次のように差える。

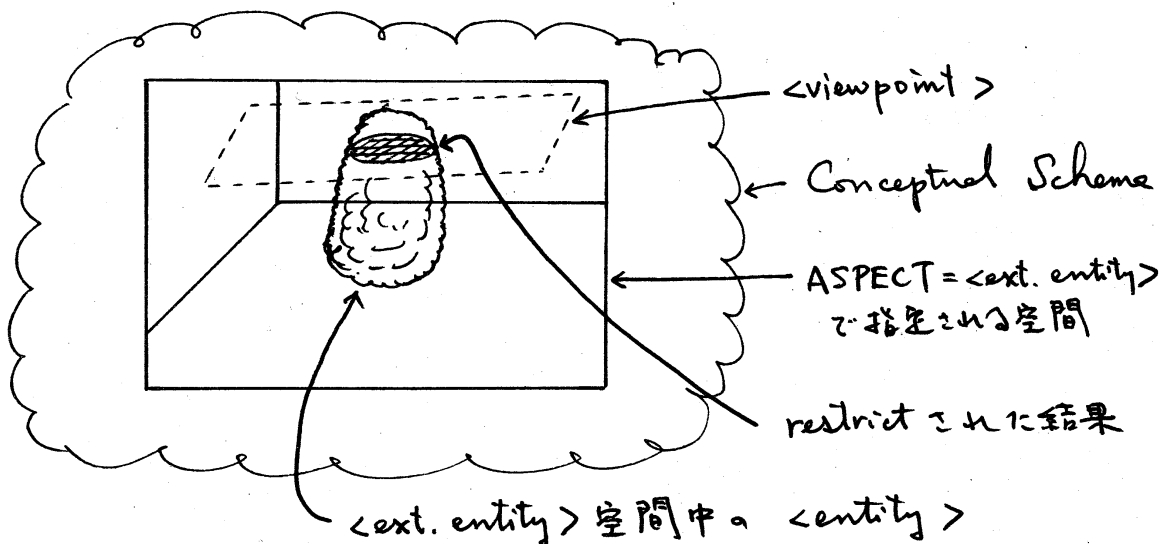


図1 Restriction 処理 (この Restriction は Relational Model の Restriction では無い！)

$\langle \text{range of object of } f_n \rangle$ の形式は OF $\langle \text{ext. entity} \rangle$ の形で使われ、 $\langle \text{ext. entity} \rangle$ の属性を抽出する処理である。これは AMONG $\langle \text{ext. entity} \rangle$ は、個体集合中の個体を抽出

出た処理である。即ち OF では $\langle \text{ext. entity} \rangle$ から \rightarrow レベルの下ったものをとることができるに対し、AMONG では $\langle \text{ext. entity} \rangle$ と同じレベルのものにとることができる。

$\langle \text{judge statement} \rangle$ は、真偽を問合せる命令である。

$\langle \text{ext. entity} \rangle$ は広義の entity であり、個体をとり、一段上の関係が生じ、その関係に視点を移しそれを個体としたものを指す。

$\langle \text{ext. property} \rangle$ は広義の property であり、属性をとり、一段上の関係が生じ、その関係に視点を移しそれを個体としたものをその個体の属性とするもの、換言すればその個体に対応するものを指す。

検索の FIND 以外の INSERT, DELETE, UPDATE なども同様に定義されるが省略する。

Ex.

```
PRINT value FIND AVG( AVG( salary OF employee OF EACH dept.)
                        AMONG dept. )
```

各部門当りの平均給料の、全部門にわたる平均を示す。

```
PRINT name FIND employee WHOSE property COVER FROM STANDPOINT
skill AMONG employee skill = A AND B AND C
```

A, B, C の技能とカバーする従業員をみつけよ。

PRINT name FIND organs CONSTITUTE FROM STANDPOINT parts OF face

顔の構成器官名をみつけよ。

PRINT name, address FIND student WHOSE address NEAR FROM

STANDPOINT distance AMONG students OF department OF

engineering, WHOSE address ≈ Umeda, Osaka

工学部の学生のうち、距離的にいって大阪市梅田に近い所に住んでいる者の名前と住所を求めよ。

(N)空間表示レベル

会話による String 出力だけでなく、空間と指定した条件で 1~3次元空間に射影し、それとフライング管上に表示し、会話型で条件をしばってゆくレベルである。位相情報空間はもともと高次元の地図なので、これと文の表現に直して出力したのでは情報の大切な部分が失われてしまうので、地図と地図のまゝで表示し、人がそれと読む方法である。

これは内容検索可能な画像データベースシステムとして作成したDBMSを modify して動かす予定であるが本稿では省略する。

階層構造は、色分けや、Zooming, Panning 等のより階層と上下の動きにより表示する。

[V] 位相情報空間型DBMS

本DBMSは前述の如く Private Research Data Base Management System と Knowledge Data Base Management System の統合システムとして構成される。

知識管理システムにおいては、データは生成規則に基づき多次元の Facet に分割される。ファイル構成は、転置ファイル索引ととも Clustered Index Multi-key Bucket Reversed File 構成とする。Aggregation 関係は、key 索引とデータを混合させることにより、アクセスの効率化をはかる。

ファイル構成の詳細は別の機会にのべる。

DML の処理は、ファイル情報代数系の上に翻訳して、ファイル情報代数系を用いて行う。

第3レベルにおいては、ユーザはスキーマを知らず、外部スキーマを定義することもある。質問中に適当にスキーマを記述しながら行う。その表DDLの alias を参照しながら、ユーザの意図を個体が配置されてゆく空間をさかしてゆく。構造がユーザの想定したものとは全く同じでなくても、位相情報空間を特定の超平面に射影することによりユーザのスキーマと合うならば、それに合わせて処理を行う。

PRDBMS に関しては Conceptual Schema-free DBMS を用いる。

[VI] むすび

位相情報空間データベースメッセージメントシステムの基礎となる位相情報空間モデルと、そのモデルに基づく位相情報空間型DBMSの構成原理、およびその言語周辺についてのべた。DDL, DMLともまだ完全ではないうが、位相情報空間モデルの持つ表現能力(概念の遠近関係、包含関係等)と十分に生かした言語に可及長に、概念間の複々の関係と検索できるように改善中である。更に他モデルで出たの処理も十分でき、そのSyntaxがあまり複雑にならないように検討しているが、微細な所を表現しようとするればSyntaxが複雑になり、また記憶しなければならぬ予約語が増加する。逆に簡単化すれば能力が低下し、そのtrade-offの大きな問題となっている。

発表関連論文

位相情報空間モデル : 電子通信学会論文誌 J63-D1

1980年1月号 pp 79-86

Schema-free DBS & ファイル情報代数系 : 電子通信学会論

文誌 J64-D3, '81年3月号 pp 190-197

画像データベースシステム : 情報処理学会論文誌

23.2 '82年3月号 pp 116-123