

CODASYL DBTG提案, Relationalモデルと情報空間モデル

(第)日本エニフック総合研究所 小林功哉

1974年10月5日より10月10日の6日間 Stockholm 郊外の Mässan で IFIP 174 が開催された。400名を越える出席者を迎え、併行に進められた4つの会場で10 sessionはどれもほぼ満員の盛況であった。筆者はデータ・ベース関係を中心に、いくつかの興味を持つ session を聴き回った。データ・ベース技術は中心議題のひとつであり、会期中を通じてほぼ何卒の関連 session があった。二つ

- session 701 : Data Representation
 - session 712 : File Structure and Accessing
 - session 216 : Files and Data Management
 - session 416 : Data Base Management Systems
 - session 615 : Data Base Design
 - session 404 : Logic and Data Bases
- の直接データ・ベース、マニピュレーション技術に因るものが多くあり、その他
- session 106 : Theoretical Aspects of Information Systems for Management
 - session 115 : Design and Implementation of Management Information Systems
 - session 715 : Business Applications of Information Systems
 - session 816 : Information Systems for Governmental Administration
 - session 906 : Control Aspects of Management and Information Systems
- のまじりに情報システムの設計を論ずるものもあった。

筆者は主としてデータ・ベース、マニピュレーション技術に直接関係する session に出席した。過去の二つ年の学会の報告に多かった特定のソフトウェア開発についての報告は割合と減り、より本格的、原理的な議論が増えているように見受けられ、データ・ベース、マニピュレーション技術が「オペレーティング・システムやプログラムの言語と並んで、計算機の基本技術としての地位を獲得したように思われた。

特に目につくのは CODASYL DBTG と Relational モデルとの対立であった。session 416 が CODASYL DBTG の立場から Relational モデルを批判すれば、615 が Relational モデルの利点を強調し、1974年5月の ACM SIGFIDET の席上で行われた1021、ディスクウェアが IFIP にも持ち越されたことがあった。しかし、双方の論点が噛み合っていないと見えず、互に別の土俵で勝負しているように見受けられる。この点では T.W. Otle [1], E.F. Codd [2] の両者の論文を対比してみるとよい。両者は SIGFIDET で発表されたものがある。session 中の質問もほとんどこの両者の対比についてであったが、明解な解答を得られず

"too much fruitless discussion concerning DBTG versus Relational Data Base Model"

と嘆息するを得た。このことに関しは、情報処理学会データ・ベース研究会データ・ベース・モデル・ワーキンググループでのかなりの討議を行ったものがあり、筆者自身の見解もあるが、以下紹介した。

1. 対象, 属性, 情報空間

解題の便のためにまず CODASYL 65G の情報代数[4] を基源とした筆者自身の情報空間モデルを概説する。このモデルは CODASYL DBTG と Relational モデルの双方を記述するのには便利である。まず世の中に存在する対象で構成されたと考えられる。対象には種々の属性が付帯し、ひとつの対象は属性についてそれぞれ一意の値を持つ。属性は対象集合からその属性について値の集合への関数と考えられる。対象の集合を E とし、 i 番目の属性 A_i 、その値集合を V_i とすると $f_i: E \rightarrow V_i$ となる。いくつもの属性の組をとると対象集合の E のひとつの元に対して i と j の属性値の組がつけられる組 (tuple) が与えられる。これを対象集合の元に対応する記録としよう。

属性値を記録に收容する n 属性の組を座標系としよう、各属性を座標軸とする n 次元の空間、つまり属性値集合の直積を情報空間としようことにしよう。計算機が与えるのは対象集合自身ではなく、 n 次元で構成された情報空間である。属性値集合に2個の特殊元素 (定義不能) \emptyset (不明) を与えて、あらゆる対象を投射する情報空間を考へ、形式的議論を容易にするのが可能である。

情報空間の構成にまつ座標系の設定が問題になる。伝統的なデータ処理ではシステムの出力が選択すべき属性を決定しなくては、システムの出力を予感すべき情報システムでは本来情報の持つ価値ポテンシャルと蓄積コストを天秤にかけざるべからざる。文献検索に於けるキーワード選択、これに伴うシーソー作り、パターン認識における特徴抽出は情報空間の座標軸選択の問題に他ならない。アプリケーションにまつ情報空間に特定のマトリックスが導入される。このマトリックスを判別しと判別分析やクラスフィニングが文献検索やパターン認識の目的となる[4,5]。

データベース R は情報空間 $V = V_1 \times V_2 \times \dots \times V_n$ の部分集合である。表現すべき対象に対応する情報空間の点の集合と考えよう。さらにこれをある基準によつて互に素な部分集合、つまりファイルに分割する。

情報空間モデルではすべての対象を表現できるユニバーサルな空間が \mathcal{P} 。つまりに与えられると考へる。CODASYL DBTG [6] では情報空間の意識はないが、記録は \mathcal{P} 。つまりに与えられるものであり、よく情報空間モデルに対応する。一方、Relational モデルでは情報はすべての項の関数であると考えられる。すべての情報は言語で表現され、言語は述語で構成される。述語とは n 個の個体の間の関係を表す $P(a_1, a_2, \dots, a_n)$ であり、これを n -tuple で表示するものである。述語の単純に応じて異なる形の n -tuple ができる。述語に対応する n -tuple は形の上では対象に対応する n -tuple と区別がないが、二に至る過程が異なるといわれる。

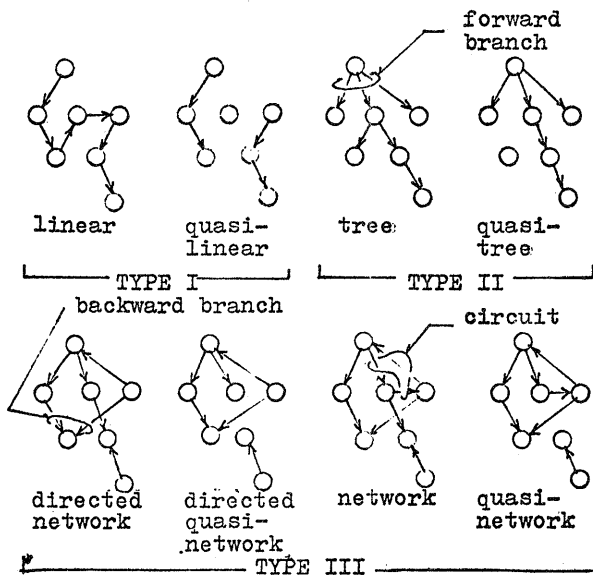
2. 記録間の関係と構造

情報代数はそれなりに奇麗なモデルではあるが、情報空間内の各点はバラバラである。つまり情報空間内の各点の単純な集合が考えられている。一般には対象と対象、情報空間の中で点と点の間に諸種の関係がある。一般の関係は n 項関係であるがこれを2項関係に分割することができる。同種の2項関係を羅列したもの構造としようことにする。構造とは2項関係の集合であると定義するのである。

記録をノード、記録間の二項関係をアークとすると、これらの構造はひとつの
 有向グラフと与える。多層の構造はハイパー・グラフとなる[7]。構造をその中
 のアークの接続具合とグラフとしての連結性を使っていくつかのクラスに分類
 することが出来る。構造中に前方向の枝分れも後方向の枝分れもなしでサーキットも
 存在しないときはこれを骨工型と呼ぶ。後向き枝分れとサーキットのない場合を骨
 直型と呼び、前向き枝分れとサーキットのないとき逆骨工型と言う。サーキット
 がないという条件だけを残すと方向づけられただけの骨工型、全部の条件を撤去すると
 骨直型が得られる。このほかにもサーキットだけあつた枝分れがないものなどの
 クラスが考へられるが、実用上これらは骨工型と考へてよい。骨工型、骨直型、
 逆骨工型、骨直型構造がグラフとして連結しているとき、それぞれ線型、木型、
 逆木型、ネットワーク型と呼び、連結してないときは各々に群を冠すること
 にしよう(附図)。

線型構造は通常順序構造と呼ばれるが、順序構造とは物理的な順序配置のなご
 りを意味するといふ意味があるのびニニは厳密に区別し、前者を線型構造、後者を
 順序配置表現ということにした。オセリスト構造やリング構造等の用語も上記の
 意味の構造の概念と異なるため使用を避け、リスト表現やリング表現ということに
 した。線型構造は構造の定義として点集合に全順序が導入可能であることに
 関がならない。データ処理上重要な役割を持つものである。D.L. Childsの集合論
 的データ構造はとくにこの点ととの表現について詳細に議論している[8]。

上記の構造のクラスがはに階層構造の概念は重要である。階層構造は層々木
 構造と混同して用いられるが、前者はこれに明確な定義を与えた。一般にひとつ



附図 構造の分類

ファイルA_iをひとつの点、B_{ij}をA_iとA_jに結合アークと考へると{A_i}の上
 で{B_{ij}}がひとつの構造と考へられる。これを骨格構造と呼ぶ。準階層構造Bの
 骨格構造が骨工型のときBを階層構造と呼ぶ(第2図)。とくにBが骨工型でかつ
 階層的である階層骨工型構造は通常のデータ処理に重要である。実際多くの可変
 長記録はいくつかの固定長記録ファイルの上で定義された階層骨工型構造と理解

のファイルの上に複数個の構
 造が定義されるし、並にこれ
 つつ構造がいくつかのファイ
 ルにわたつて定義される。
 この構造Bがいくつかのファ
 イルA₁, A₂, ..., A_nの上で定
 義されたとすると

$$B_{ij} = \{ \alpha | \alpha \in B, \alpha \in A_i, \alpha \in A_j \}$$

ただしαはアークの始点
 α(α)は終点と定義したときB
 は

$$B = \bigcup_{i=1}^m \bigcup_{j=1}^m B_{ij}$$

の形に一意的に分解される。
 B_{ij}はBの互に素部分集合
 である。すべてのB_{ij}=φの
 ときBを準階層構造と言う。

である。

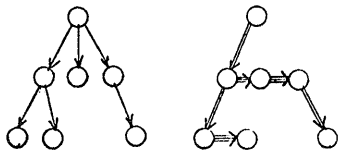
3. ファイルと構造の同値関係

情報空間内の点の集合であるファイルと点の間の2項関係の集合である構造は情報の論理的モデルであるが、この論理的モデルは一意的に与えられるものではない。実際全く同じ情報と異なる論理モデルで捉えることができる。既存のデータ・ベース・マネジメント・システムは、これを「自身の持つ表現機能に都合のよい形」の論理的モデルの敷設を強制しているが、これがデータ・ベース・マネジメント・システム機能の評價の基準となつていゝ。議論を明確にするために、この直接の結果としていくつかの論理的モデルの同値関係に就いておこす。

まず、ファイル群 A_1, A_2, \dots, A_m とこの上で定義された階層構造 B は、 A_1, A_2, \dots, A_m に B を分解して得られる2層の階層構造群 B_{ij} と同値である。ここで言う同値とは片側からもう一方が導かれ、その逆も可能なことである。 B が階層第II型であれば、この B_{ij} は2層の階層第II型となる。GIS[9], TDMS[10], MARK N[11] などのシステムは多層の階層第II型構造が表現可能であるが、これらのファイルに対して上層の多層の階層第II型構造の定義が許されていない。これに対し IDS[12] やその延長の CODASYL DBTG はこれらのファイルに対して任意個の2層分型構造の定義を許すのが特色である。上記の事実を IDS 系列のシステムが GIS, TDMS, MARK N などのデータ定義機能をカバーしていることを示す。

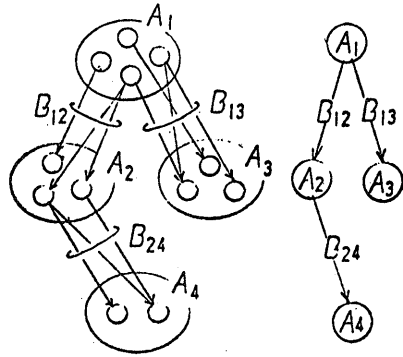
つぎに重要な事実は、ファイル A_1, A_2, \dots, A_m 上で定義された2層の階層第II型構造 B は、 A_1, A_2, \dots, A_m と B の骨格構造を示す $A' = \{A_k\}$, $B' = \{B_{ij}\}$ および A_1, A_2, \dots, A_m の上で定義された1層の階層第II型構造 B'' の組と同値であるということである。 B が B', B'' を導く、あるいは B', B'' が B を導くアルゴリズムは容易である。階層構造の表現は第II型構造の表現より遙かに容易である。この事実は GIS, TDMS, MARK N などのインテグレーションに活用されている。IDS 系のシステムで2層の階層構造を (owner-coupled) set と取っているが、その表現はこれと第II型構造に変換したものが B' の表現にまつている。

第三に、ファイル A とこの上で定義された2層の階層第II型構造 B と同値な A と第II型構造の対 B', B'' が存在することが示される(第3図)。



→ arc in type II structure B
 ⇒ arc in type I structure B'

第3図 第II型構造に同値な第I型構造の対。



第4図 階層構造とその骨格構造

導くことは不可能である。指し示す構造がホインで表示にわづら、第II型構造を導くことは困難である。ホインで表示できないため、第II型についてはこれが成立しないことが示されるが、これは上記の事実による。

さらにファイル A と任意の構造 B は、ファイル A および B (構造をファイルと見做す) と A, B 間に定義される2層の階層第II型構造の対 B', B'' と同値であることが示される(第4図)。IDS 系のシステムに対する第II型構造

報告書に見られる用語や概念が適切であると考えない[16,17]。しかし Coddの批判 [2] は的を得ていると思えない。これは下記の理由による。

第一、データベースの計算機上の表現の技術は結局単位データのどれどれへのアドレス割付けの方法と、単位データ間の連合の方法に集約される。後者は多層に重収される。アドレス割付けはそれ単独に考えられることは稀で、何層かの連合の表現と関連して考慮されるのが普通である。二重という連合とはひとつの連合を構成する要素の1個乃至複数個が与えられるとき、他の要素を検索し得る、つまり連想を可能にするメカニズムのことである。現在の計算機ハードウェアのちとで可能な連合の方法は僅かに3種しかない。その第一は連合の要素を記憶装置上に隣接配置する二重による二重連合で、基本的には2要素間の連合である。第二は連合要素の一方のアドレスを単位データとして他の要素に第一の連合方式で付加するポインタ方式の連合で、これも基本的に2要素間連合である。第三の方法は連合の各要素が同質の場合に行なわれる方法で、これらの各要素に共通のプログラムアドレス割付けを行なうものである。通常要素としては $A = n(x)$ と b で決められるアドレス a が与えられる。これは要素の長さ、またはある定数、 $n(x)$ はある定数 n_0 に対して $0 \leq n(x) \leq n_0$ となるような2項関数である。 $n(x)$ の定め方として直接アドレス法、順次アドレス法があり、それぞれ異なる種の連想を可能にする。第三の方法は任意個の要素の連合であるが、第3項 $P(x)$ なる第項の述語を表現する手段、つまり一項の関係表現と考える方がよい。

第一の方法は容易にネストして用いられ、 n 項の関係表現に使われるが、第二の方法の n 項表現への利用は必ずしも容易でない。第二の方法を用いた n 項表現が一意的に定まらないことが議論を複雑にするのである。通常属性値を集めて記録を作るのは可能なが、第一の方法を用い、記録を集めてファイルを作るには第三の方法が用いられる。これ以外は第二の方法を利用するのである。

記録内関係と記録間関係を区別する理由の1つは、この方が直観的に見出しのよい場合があることだろう。実際言語の「ページ」に現れる木構造や、ネットワーク解析の対象となるネットワーク構造などがその場合である。著者はこの点について Codd と意見を異にする。もうひとつの大きな理由は記録内関係が通常第一の連合方法で表現されるに対し、記録間関係の表現に第二の方法を用いることが多いことと思われる。前述の理由で記録内関係は2項関係に分解され、その集合として構造が定義される。

構造つまり記録間の2項関係の表現も3種類ある。第一は記録の隣接配置による方法で第一型構造の表現に限って適用可能である。第二はポインタによる方法つまりリストやリンク表示で、構造型に依って様々な変化が有り得る。第三は2項関係自身を要素と考えて少くとも始点および終点の2個の関係属性を持つ記録(論理的リスト)のファイルを作るものである。第二と第三はどんな構造型にも適用できるが、第二の場合には種々の変化があるわけである。

連合の構成はいわゆるアクセス・パスの準備と古い換えることができる。Coddの批判はアクセス・パスを「ユーザ」に意識させるべきでないと言っているが、カニエール、ユーザを主たる対象とする Relational モデルはとて、プログラマ、ユーザ向きの IDS, CODASYL DBTG では構造概念を注入の方が自然であろう。もしもばアクセス・パスをユーザの考慮対象から除外するとしても、その考慮のないシステム、インプリメンテーションは悲惨なパフォーマンスを齎らすに違いない。DBTG提案での構造概念の導入は理論的に奇麗な形であると云い難い

し、これをベースとしたDDLの設計もラール・デザインと優められたものではないが、構造を全く重視すべからずというCoddの批判は多を得ていないように思われる。

なお、情報空間モデルでは記録内2項関係と記録間2項関係も考えたが、記録内関係も2項関係に分解する方向もある。LEAP [18] と TRAMP [19] がそれで、すべての情報はA (Attribute), O (Object), V (Value) の連想三つ組に分解される。連想三つ組はA(O, V)の2項の連想の表現である。A, O, Vの任意の1個乃至2個を与えて他を探索可能にするメカニズムも考えているが、これは連想記憶装置のソフトウェア・シミュレーションであると述べている。M.E. SenkoのDIAMモデルもこの方向で作られている [20]。

6. データ処理のモデル

CODASYL DBTG 例からのRelationalモデルの批判は主としてRelationalモデルの理解不足に起因するようである [1]。Relationalモデルの優れた点のひとつはRelational Calculus とその周辺としてAlpha言語の設計 [21, 22] にある。これはRelational データ・ベース・モデルの上のデータ処理モデルの議論である。データ処理のモデルはデータ構造のモデル群の題目を兼ねていない。

ホスト言語システムではデータの検索、更新以外の処理はホスト言語に任せている。ホスト言語そのものが長い歴史とある程度の理論的検討を経たものであるが、データの検索、更新については同程度の理論的検討がなされたとは考え得ない。さらに独立言語システムやシステム言語に至っては便宜的に考慮されたものとして考えられる。

筆者は情報空間モデルの上のデータ処理モデルとして10種の集合演算を考えた。データ処理は情報空間のいくつかの点集合(入力ファイル)からいくつかの点集合(出力ファイル)への変換であり、したがって集合演算である。任意のデータ処理がいくつかの基本的な集合演算群の組として記述できると考える。Chudisは17種の演算を挙げた。筆者の10種はこれにかわるものである。CoddのRelational Algebra はこの集合演算に対応しない。CoddはAlpha言語に対応する操作を入れたが、便宜的に過ぎる嫌いがある。10種の集合演算を規定するパラメータとして記録乃至記録群の上で定義される関数があり、その特殊な形のRelational Calculus となる。Relational Calculus がよりエレメンタリな操作群への分解がデータ・ベースの探索問題として重要である [23, 24]。筆者のモデルでは10種の集合演算を中間レベルの言語とし、コンピュータ・ユーザに使用に供することを提案する。なおこれらの操作をワークスペースの利用を考慮した操作に分解したもので、Coddのいわゆる piped mode をホスト言語に連繋するデータ・ベース・サブルーチンに対応させた。さらに単位処理を完全に非平順的に記述できる上位レベルの言語を考へ、パラメータ・ユーザの使用に供するが、このレベルでは限られた数の単立操作でデータ処理すべてを記述することは不可能であることがわかる。これらの議論は平順的、非平順的の概念の明確化に際し、また構造的プログラミングに似るプログラム作成の道も提起する。

情報空間モデルについては [25] に、その物理的表現については [26] に詳述した。またこのモデルを土台として開発した実験システム FORIMS [27] も参照されたい。

REFERENCES

1. T.W.Olle. Current and Future Trends in Data Base Management Systems. Proc. IFIP '74 Preprint, pp. 998 - 1006.
2. E.F.Codd. The Relational and Network Approaches: Comparison of the Application Programming Interfaces. IBM Research, RJ 1401. June, 1974.
3. CODASYL Development Committee. An Information Algebra: Phase 1 Report. Comm. ACM, Vol. 5, No. 4. pp. 190 - 204. April, 1962.
4. G. Salton. Automatic Information Organization and Retrieval. McGraw-Hill, New York, 1968.
5. 柳田健一郎. クラスティフィケーション. The Soken Kiyo, Vol. 4, No. 1. 1974.
6. CODASYL Data Base Task Group. '71 Report. April, 1971.
7. C. Berge. Graphs and Hypergraphs. North-Holland, Amsterdam, 1973.
8. D.L.Childs. Feasibility of a Set-theoretic Data Structure. Proc. IFIP '68, pp. 420 - 430.
9. International Business Machine Corp. GIS Application Description Manual. H20-0574.
10. System Development Corp. TS/DMS User's Guide. TM-4132. 1968.
11. Informatics Corp. MARK IV User's Guide.
12. General Electric Co. Integrated Data Store: A new Concept in Data Management. GE Information System Dept. GPB-483.
13. E.F.Codd. A Relational Model for Large Shared Data Banks. Comm. ACM, Vol. 13, No. 6. pp. 377 - 387. June, 1970.
14. E.F.Codd. Normalized Data Base Structure: A brief Tutorial. IBM Research, RJ 935. November, 1971.
15. E.F.Codd. Further Normalization of Data Base Relational Model. IBM Research RJ 909. August, 1971.
16. CODASYL Systems Committee. A Survey of Generalized Data Base Management Systems. May, 1969. 柳田健一郎編著. データベース管理 基礎編 - 1971.
17. CODASYL Systems Committee. Feature Analysis of Generalized Data Base Management Systems. June, 1971. 柳田健一郎編著. データベース管理 応用編 bit 柳田健一郎 December, 1973.
18. J.A.Feldman and P.D.Rovner. An ALGOL-based Associative Language. Comm.ACM, Vol. 12, No. 8. pp. 439 - 449. August, 1969.
19. W.L.Ash and E.H.Sibley. TRAMP: An Interactive Associative Processor with Deductive Capabilities. Proc. ACM National Conference '68. pp. 143 - 156.
20. M.E.Senko et al. Data Structure and Accessing in Data Base Systems. IBM Sys. Jour. Vol. 12, No. 1. 1973.
21. E.F.Codd. A Data Base Sublanguage founded on the Relational Calculus. IBM Research, RJ 893. July, 1971.
22. E.F.Codd. Relational Completeness of Data Base Sublanguage. IBM Research, RJ 987. March, 1972.
23. F.P.Palermo. A Data Base Search Problem. IBM Research, RJ 1072. July, 1972.
24. Y. Chiba. A Data Base Search Algorithm based on Complicated retrieval Conditions. The Soken Kiyo, Vol. 5, No. 1. 1974.
25. I. Kobayashi. A Formalism of Information and Information Processing Structure: Revised Report. The Soken Kiyo, Vol. 5, No. 1. 1974
26. I. Kobayashi. Physical Representation of Information Structure. The Soken Kiyo, Vol. 5, No. 1. 1974.
27. K. Kohri and Y. Chiba. FORIMS Phase 2 Design Specification: A FORTRAN Oriented Information Management System. The Soken Kiyo, Vol. 5, No. 1. 1974.