Vol. 5 No. 1 '64 Jan. 1964

Information Algebra について*

刀根 薫**

はじめに

技術計算と事務計算の違いをごく大雑把にのべると，技術計算はデータの数は少ないが解法のアルゴリズム（算法）が複雑であり，それにたいして，事務計算では計算の全体的な流れは簡単であるが，データの数が多く，その出し入れと計算機内でのグループとしての計算，分類などに多くの技術を要するという点であろう．したがって，技術計算では，問題を数式に表現して，流れ図にまとめれば，容易に計算機のプログラム言語によってとくことができる．つまり四則算，論理算が強力な意味をもっているといえる．

また計算機のハードウェアの面でもこの方向の進歩はめざましいものがある．一方，事務計算——もっと一般にデータ処理——の方は流れ図は簡単にかけるが，数学的な問題を数式で定義して計算機に乗せる場合のようにコンパクトな形で問題を定義して，それで終るわけではない．問題はむしろ，その先にある．このことは，問題の性質に由来するのであって，大量のデータ，たとえばファイル群を処理するような算術演算，論理演算がまだ計算機のハード・ウェアの面で十分発達していないということにもよるわけである．

そこで，種々のシステム言語を作って，これら計算機のハード・ウェアにとって手にあまる種類の問題を解決しようとする企てがあり，実用に供されている．たとえば，COBOL, REPORT WRITER, SORT GENERATOR などである．

ところで事務計算に登場するデータなるものの素性をもっと分析していけば，データを処理するメカニズムの解析がもっとうまくできるのではあるまいか．従来のシステム言語では計算機の側からの制約が何といっても一番比重をもっているためにそういう点の配慮があまりみられないのである．

データ，ファイルなどの構造をしらべること，それらの上に一つの理論を作りあげることは，理論上はもとより，十分実用上の興味もよぶものと思う．システム・アナリストやプログラマが暗々裡に感得していたことを，計算機の立場を一応離れて検討してみることは有益なことであろう．この方向の論文として COD ASYL のグループによる Information Algebra[1] や L. Lombardi による ADSYL[2] があるが，ここでは前者について解説を試みることにする．後者まで含めた総合紹介としては，たとえば 3) を参照してもらいたい．

1. 属性空間 (Property Space)

データ処理の問題を“代数学”として取り扱うためには，まず，データとは何かということが問題になるが，それをここではある“個体”(entity)のある“属性”(property) の“値”(value) と考える．つまりデータは普通，ある記号（数値，アルファベットなど）で表現されるわけであるが，それはある個体のある属性の値を表わしていることが判別できねばならない．

この三つの要素は情報代数学では最も基本的な素子となるものである．これは普通の帳簿の構造を抽象的にいい表わしたにすぎないし，データというものが明確に表わされているときには必ずこういう構造をもっているわけである．

普通の代数学では，基本的な構成要素は定義されるものであるが，ここでは問題の特殊性からこれら三つの概念については未定義のまま話を集めていく．しかし，直観的な概念はもちろん，存在するわけであり，特にそれぞれの間の関係を仮定して代数学は作られていく．

（1） 三つの未定義の概念；

属性（q）
値 （v）
個体（e）

（2） 仮定：各属性は一つかつただ一つの値の集合をもっている．このような属性のもつ値の集合を属性値集合（V）という．

（3） 仮定：すべての属性値集合は少なくとも二つの値を含んでいる．すなわち，

Ω（未定義，無関係）

* On Information Algebra, by Kaoru Tone (Tokyo University of Agriculture and Technology, Faculty of Engineering)

** 東京農工大学工学部

θ（不明，関係あり，しかし未知）

この仮定は，論理上の首尾一貫性のためと，未定義または不明の値をゼロと区別するための手段として取り入れられたものである．

（4） 仮定：各個体は各属性値集合に一つ，かつただ一つの値をもっている．

（5） 定義：“座標集合”

異なる有限個の属性の集合を座標集合（Q）と呼ぶ

（6） 定義：“Null 座標集合”

属性を全く含まない座標集合を null 座標集合という．

（7） 定義：二つの座標集合は，同一の属性をもつとき，かつそのときに限り“同等”であるという．

（8） 定義：“属性空間”

ある座標集合（Q）の“属性空間”（P）とは次のデカルト積のことをいう．

$$P=V_1\times V_2\times V_3\times\cdots\cdots\times V_n$$

ここに V_i は Q の i 番目の属性につけられた属性値集合である．

属性空間の各点（p）は $p=(a_1, a_2, \cdots\cdots a_n)$ で表示される．ここに a_i は V_i に含まれる，ある値である．

（9） 定義：属性空間（P）におけるある個体（e）の“データ点”（d）を次のように定義する．

$$d=(a_1, a_2, \cdots\cdots, a_n)$$

ただし，a_i は P の i 番目の属性値集合から e につけられた値である．（d）を（P）における（e）の“表現”と呼ぶ．

（10） 定理：“すべての個体は与えられた属性空間においてちょうど一個のデータ点をもつ”

（11） 定義：ある個体集合に対する“判別属性空間”とは各データ点が一個体以上を表現しないような属性空間をいう．それに対応する座標集合を“判別座標集合”という．

（12） 定義：もし

（a） Q および Q' が座標集合で，

（b） Q が判別的で

（c） Q に含まれるすべての Q' に対して，Q' は判別的でない

ならば，Q を“基底判別座標集合”と呼ぶ．

〔**解説 1**〕属性というのは普通の帳簿の項目（item）にあたるものである．したがって属性空間の座標軸としては対象とするすべての項目が含まれねばならない．第9節（29 頁）の例題では給与計算を取り扱うが，その場合 q_1 から q_9 までの9個の属性が考えられる．（**第1表**参照）すなわち q_1＝ファイルの区分，q_2＝man 番号，q_3＝名前，q_4＝時間給率，q_5＝勤務時間数，q_6＝週日数，q_7＝全支給額，q_8＝支給期（週），q_9＝支給額である．この問題はマスター・ファイルを週単位で更新し，また新らしい従業員があればマスター・ファイルに追加していくのである．したがってファイル（これも厳密には後に定義する）は3個ありそれを区別するためには q_1 が必要である．さらに属性軸上の値の範囲は**第1表**（29 頁参照）の値集合の欄のとおりである．

2. 線および線の函数

これからの話では，ただ一つの属性空間 P を仮定する．データ点を他の点と区分するためのこれ以上の展開は行なわない．したがってまた，議論のなかでも区別しない．一般の点について成立することは，データ点についても成立することに注意してもらいたい．前節では，属性空間を定義する方法と個体，属性，値の間の関係についてしらべた．一たびある座標系がえらばれると，この属性空間の中で演算を行なえばよい．この代数学をデータ処理問題に応用するために，それに必要なすべての情報―入力，中間結果，出力―を含むような属性空間を定義しなければならない．

（1） 定義：P からえらばれたある順序のついた点集合を“線”と呼ぶ．線を構成する点の数をその線の“長さ”という．

（2） 記号：長さ n の線 L を

$$L=(p_1, p_2, \cdots\cdots, p_n)$$

とかく．

（3） 定義：P の各線に対して一つかつただ一つの値を付ける写像を“線の函数”（FOL＝function of line）と呼ぶ．

ある FOL で付けられた異なる値の集合をその FOL の値集合と呼ぶ．

（4） 記号：FOL を函数 $f(X)$ で表わす．ここに f は FOL，X は線を示す．

（5） 定義：二つの FOL は，P に属するすべての線に対して，同一線に同一値を付けるとき，かつそのときに限り“等しい”という．

（6） 定義：ある FOL が線上の点の順序に無関係に，同一点を含むすべての線に同一値を付けるとき，その FOL は“順序に無関係である”という．

（7） 定義：値集合として Ω のみを含むものを

“Null FOL”と呼ぶ.

（8） 定義：ある FOL が null でなく，かつ長さが n でないすべての線に対して Ω を付けるとき，その FOL は長さ n であるという.

（9） 定義：ある FOL があって，その値集合が，非反射的*，非対称**，推移的***なある関係子Ⓡが存在する定義集合のなかに含まれているとき，その FOL を“順序 FOL”（OFOL＝Ordinal FOL）と呼ぶ.

さらに，

（a） 次に示す関係子をもつ定義集合には特別な名前をつける.

定義集合	名　前
$\Omega<\theta<-\infty<$実数の集合$<+\infty$	実数（Ⓡは“より小”）
$\Omega<\theta<$負の整数の集合$<0<$正の整数の集合，	整数（Ⓡは“より小”）
ΩⓇθⓇ False (F)Ⓡ True (T)	Boolean
False(F)Ⓡ True(T)	Selection

（b） 順序 FOL はその定義集合と同じ“タイプ”であるといわれる.

10） 定義：順序 FOL でない FOL を“非順序”FOL という.

〔解説 2〕 前節では対象のすべての情報を含むような空間を考えた．この空間の点を連結したり，あるいは特定の値の集計をしたりしてデータを処理するわけであるが，そのための道具を揃えるために先ず線を導入する．給与計算の例でいえばある一人の従業員の1週間分の勤務記録は一つの線をなしている.

次に $F(X)$ という線の函数 FOL を導入する．これはその線にたいして一義的に定まる値である．その値集合の特殊なもの——というよりはよく出て来るもの——に名前をつけて呼ぶことにする．なお，$F(X)$ の具体的な作り方や，線の例は次節以降に出て来る.

3. 領域および領域の函数

これまで，あるデータ処理システムのなかで表現できる全情報を，全体として，考察してきた．しかし，実際問題としては，情報代数学は，この全情報の一部分，すなわち部分集合と関係をもたねばならない．これらの部分を領域と名づける．ファイルおよびファイルの一部に当たるものである.

（1） 定義：“領域（area）”とは P の任意の部分集合のことをいう.

（2） 定義：各領域に一つ，かつただ一つの値を付ける写像を“領域の函数”（FOA＝function of area）と呼ぶ．ある FOA により写像される異なる値の集合を FOA の値集合と呼ぶ.

（3） 記号：FOA を $f(X)$ とかく.

ここに f は FOA があり，X は領域である.

（4） 定理：“いま

（a） B が P の領域で，

（b） M が B に属する点をすべて含む1本の線でかつそれ以外の点を含まないとする.

そのとき，各 FOA f に対して，FOL f' が存在して，f' は順序に関係なく，（b）をみたすすべての線 M に対して

$f(B)=f'(M)$ である.”

（証明） 明らか.

〔解説 3〕 **第1表**の例でいえばファイルとしては Pay File (PF), Daily Work File (DW), New Employee File (NE) の3個がありそれらは属性空間の三つの部分集合，すなわち領域である．PF は q_1, q_2, q_3, q_4, q_7, q_8, q_9 と関係をもち，DW は週間の勤務記録であるから q_1, q_2, q_5, q_6 を含むだけで十分である．（表の X）．Ω は無関係であることを示す.

4. バンドルとバンドルの函数

データ処理の応用には，関係のあるデータを種々の源から集めて来る必要がある．そしてこのデータに種々の形の函数を適用し，新らしい情報を作る必要がある．たとえば，ファイルをマージして新らしいファイルを作る．ある属性空間に属する種々の部分集合から来る情報の機能を結合したり，評価したりする方法をこの節で定義する.

（1） 定義：位数 n の“領域集合” Z とは順序づけられた n 個の領域 $(A_1, A_2, \cdots\cdots, A_n)=Z$ をいう.

（2） 定義：ある selection OFOL b に対する領域集合 Z の“バンドル” $B=B(b, Z)$ とは次のような性質をもつすべての線 L の集合をいう．もし

（a） $Z=(A_1, A_2, \cdots\cdots, A_n)$ で

（b） $L=(p_1, p_2, \cdots\cdots, p_n)$（ここに $p_i(i=1, 2, \cdots\cdots, n)$ は A_i の一点である.）

ならば $b(L)=$True（真）.

この函数 b は B に対するバンドリング函数（bun-

* 非反射的とは aⓇa が成立しないこと
** 非対称とは aⓇb, bⓇa が同時に成立しないこと
*** 推移的とは aⓇb, bⓇc ならば aⓇc が成立することをいう．これだけの関係が成立すれば Ⓡ という関係子を仲介にして順序がつけられるのである.

dling function）といわれる．バンドルの位数は領域集合の位数と同一とする．

（3） 定義：ある領域をあるバンドルに次のように対応させる写像を“バンドルの函数”FOB(＝function of bundle）という．

（a） バンドルに属する線と領域の点の間に多対一（many to one）の対応がある．

（b） その領域に属する各点の各属性の値はバンドルのなかの対応する線の FOL により定義される．

（c） そのような FOL の値の集合は対応する属性値集合の部分集合でなければならない．

（4） 定義：null FOB とはすべてのバンドルを唯一の点（$\Omega, \Omega, \cdots\cdots, \Omega$）を含む領域に対応させる写像をいう．

（5） 記号：FOB を次の三とおりの等価な方法で表現することができる．

（a） $F\equiv(f_1, f_2, \cdots\cdots, f_k, \cdots\cdots, f_m)$

（b） $F\equiv(q_1'=f_1, \cdots\cdots, q_k'=f_k, \cdots\cdots, q_m'=f_m)$

（c）$F\equiv$
$$\begin{cases} q_1'=f_1 \\ q_2'=f_2 \\ \vdots \\ q_k'=f_k \\ \vdots \\ q_m=f_m \end{cases}$$

ここに F は FOB であり，q_k' はその FOB により対応づけられた領域の点の k 番目の属性である．m は属性の個数，f_k は q_{ij}(i=1, 2, ……, n, j=1, 2, ……, m) の函数である．ただし q_{ij} は領域集合の i 番目の領域の j 番目の属性である．n はバンドルを構成する線の長さである．

（a） の形ではすべての q_k' に対して函数形 f を明示しなければならない．

(b) と (c) では $q_k'=f_k$ という表現がないこともあり得る．その場合には $q_k'=q_{nk}$ とみなされる．ここに q_{nk} は n 番目の領域，すなわち各線の最後の点を含む領域の点に対する属性 q_k の値を示す．

もし f のいずれもが明示されてなければ，すなわち $F\equiv(\ )$ ならば $q_k'=q_{nk}$, (k=1, 2, ……, m) である．

（6） 記号：バンドル B に対する FOB F により対応づけられた領域 A を次のように表わす．

$$A=F(B)$$
$$A=F(b, Z)$$
$$A=F(b, A_1, A_2, \cdots\cdots, A_n)$$

（7） 記号：$A(s)$ は

$$A(s)=F(s, P) \text{ ここに } F\equiv(\ ).$$

により決定される領域を示す．

s は A に対する“選択函数”と呼ばれる．

〔**解説 4**〕 二つのファイルを match-merge して一つのファイルを作ることは事務処理には必ず出て来ることである．この節のバンドル，バンドルの函数によりそれが可能になる．バンドリング函数はその key にあたるもので，たとえば，二つのファイルのなかから“man 番号の等しいデータ”を一緒にすることにより，一つのバンドルができるわけである．バンドルはもともと束の意味であり，データを束ねることを考えるのである．

次にその束にしたものを処理して，たとえば勤務時間の和をとったりすることをバンドルの函数 FOB により行なう．第9節の例題で（1）式に

$$F_1[q_{12}=q_{22}, H(q_2, DW), OP]$$

という表現があるが，これは $H(q_2, DW)$ という次節で説明する領域と OP という領域から $q_{12}=q_{22}$ という selection OFOL によりバンドルを作る．ここで q_{ij} の i の 1, 2 は第1の領域 $H(q_2, DW)$ か第2の領域 OP かを指定するものであり，j はその領域のどの属性かを示している．この例では領域 $H(q_2, DW)$ と OP の q_2（man 番号）を比較してそれが等しいもの（$q_{12}=q_{22}$ が真なるもの）をバンドルとしたわけである．次にそうしてできたバンドルに演算 F_1 を施してまた一つの領域を作るが，その領域の各属性の値は（1）式の

$$F_1\equiv\begin{cases} q_7'=q_{27}+q_{15}*q_{24} \\ q_8'=q_{28}+1 \\ q_9'=q_{15}*q_{24} \end{cases}$$

によって計算する．たとえば F_1 の q_7（全支給額）は q_{27}（Old Pay File の全支給額）と今週の支給額 $q_{15}*q_{24}$ の和である．これが新らしい全支給額になる．q_1', ……, q_6' までは明示されていないが，規則により，それは OP の q_{21}, ……, q_{26} と等しいことを意味している．

5. グランプおよびグランプの函数

属性空間の数種の部分集合（ファイル）を含むデータ処理操作の議論の基礎を確立したので，この節では，P またはその部分集合（ファイル）の中での演算の議論の基礎を固める．あるファイルの総和をとるといった処理がこの節で展開される概念により可能となる．

（1） 定義：A を領域，g を A に含まれる長さ1

の線の上で定義された FOL とするとき FOL g に対する領域 A のグランプ $G=G(g, A)$ とは次のような g による A の分割 (partition) をいう.

この分割の要素は g に対して同一値をもつ A のすべての点から成る. 函数 g を G に対するグランピング函数と呼ぶ.

グランプの概念は要素中の点の順序づけ, またはグランプ内の要素の順序づけを意味しない.

(2) 記号: g に対する値が c であるような点で定義された A の部分集合はグランプの一要素であるが, それを $G(g, A)|_{g=c}$ で表わす.

(3) 定義: "グランプの函数" (FOG=function of glump) とは, グランプに対し, 領域を次のように対応させる写像のことである.

(a) グランプの要素と, 領域の点の間に多対一の対応がある.

(b) 対応された領域における点の各属性の値はグランプの対応する要素の FOA により決められる.

(c) その FOA の値集合は対応する属性値集合の部分集合である.

(4) 記号: FOG を次の三つの等価な方法で表現することができる.

(a) $H\equiv(f_1, f_2, f_3, \cdots\cdots, f_k)$

(b) $H\equiv(q_1'=f_1, q_2'=f_2, \cdots\cdots, q_k'=f_k)$

(c) $$H\equiv\begin{cases} q_1'=f_1 \\ q_2'=f_2 \\ \vdots \\ q_k=f_k \end{cases}$$

ここは H は FOG,
f_i は FOA,
q_i' はその領域の点の i 番目の属性,
k は属性の数,

とする.

(a) の形では, すべての q' に対応する f は明示されねばならない. (b) と (c) では $q'_i=f_i$ という指定がないこともあり得る. この場合は $q_i'=\Omega$ とする.

(5) 記号: FOG H によりグランプ $H(g, A_1)$ に対応づけられた領域は

$A=H(G)$ または
$A=H(g, A_1)$

と書き表わされる.

[解説 5] バンドルでは二つ以上の領域の結びつけを行なったが, ここでは一つの領域内での sorting を考える. 第9節の例題では (1) 式の中に $H(q_2, DW)$ という項があるが, 全従業員の1週間分の勤務記録からなる DW というファイルを q_2 (man 番号) によって分類する. その分類した結果をグランプといい, それを処理するのがグランプの函数である. 処理された結果はまた一つの領域をなしている.

$$H\equiv\begin{cases} q_2'=q_2 \\ q_5'=\Sigma[q_5\leftarrow(q_5<8)\rightarrow 1.5*q_3-4]+f_1 \\ f_2=\Sigma[q_5\leftarrow(q_5<8)\rightarrow 8] \\ f_1=0\leftarrow(f_2<40)\rightarrow 0.5*f_2-20 \end{cases}$$

この式の意味は, $H(q_2, DW)$ により新らしい領域を定義し, その属性の値は, 上式の右辺とする (式の個々の意味は第7節で明らかとなる).

6. 領域の順序づけ

領域を導入したことはファイルの概念と似たものを与え, バンドリングはマッチングと同じ機能をもち, グランピングは一つのファイルの中での総和をとるといった操作を可能とした. これら三つの概念では "順序" といった考えはかなり無視されて来た. というのは必要でないからである. しかしながら実際の問題では, それを導入する必要がある.

(1) 定義: 長さ1の OFOL f による領域 A の "順序づけ" $\bigcirc(f, A)$ とは A からえらばれた点列 $(p_1, p_2, \cdots\cdots, p_n)$ で, その点列が A を尽し, かつ $f(p_i)<f(p_{i+1})$ または $f(p_i)=f(p_{i+1})$ が成立するもののことをいう.

(2) 定義: ある順序づけは $f(p_i)=f(p_{i+1})$ をみたす点が存在しないときかつそのときに限り, その OFOL に対して "単純" であるという.

また, $f(p_i)=f(p_{i+1})$ をみたす点の対 (p_i, p_{i+1}) が少なくとも一つ存在する場合, かつそのときに限りその順序づけは OFOL に対して "部分的" であるという.

(3) 定理: "f を領域 A の上で定義された長さ1の OFOL, $G(f, A)=(G_1, G_2, \cdots\cdots, G_m)$ を f による A のグランプとし, n_i を G_i に含まれる点の数とすれば ($i=1, 2, \cdots\cdots, m$), f による A の順序づけはちょうど $n_1!\ n_2!\ \cdots\cdots\ n_m!$ とおりだけある."

(4) 系: "OFOL f による領域 A の単純な順序づけは高々一個に限る."

7. FOL の演算

前節までの例では, いろいろなタイプの FOL を用いた. この節では FOL の演算を取り扱う. FOL は一つの属性として定義できるので, FOL に対する演

算は属性に対する演算である．しかしながら FOL の定義は再帰的である．という意味は，FOL は値の上で演算することもできるし，また，値の函数の上で演算することもできるし，値の函数の函数の上で演算することもできるのである．

この節では一つの例を用いて話を進める．個体は従業員で，名前と住所で一義的に定まるものとする．従業員は月曜から金曜まで働き，労働時間と生産個数に基いて，給料の支給をうけるものとする．

この個体に対して設定される空間の座標集合は（名前，住所，週日，労働時間，生産個数）である．

この属性空間から撰択された長さ5の線を考える．すなわち従業員の週の各日における労働記録を示す5個の点から1本の線はなる．説明用に次のような FOL を定義する．

f_1＝名前
f_2＝住所
f_3＝週間労働時間
f_4＝1日8時間以内での週間労働時間
f_5＝週間生産個数
f_6＝ボーナス時間（.20/時間　で支払われる）
　＝f_3-f_4
f_7＝ボーナス指数＝($f_5/f_3<13$)
f_8＝生産性奨励率＝(.20 ← f_7 → .25)
f_9＝超勤支給指数＝($f_3=f_4$)

（1） 定義：“連結”(concatenation)(⊂) とは次のような性質をもつ binary* 演算子をいう．

（a） $f_1 \subset f_2 = f_3$ ならば，任意の L に対して $f_3(L)$ はカップル $[f_1(L), f_2(L)]$ である．

（b） $f_1 \subset (f_2 \subset f_3) = f_4$ ならば，
$f_4(L)$ はトリプル $[f_1(L), f_2(L), f_3(L)]$ である．

（c） $f_1 \subset (f_2 \subset f_3) = (f_1 \subset f_2) \subset f_3 - f_1 \subset f_2 \subset f_3$

（d） 一般に $f_1 \subset f_2 \neq f_2 \subset f_1$

連結は associative であるが commutative ではない．連結函数は異なる属性の値を結合させるために用いられる．この例では $f_1 \subset f_2$ により“名前と住所”はリンクされたことになる．

（2） 定義

“和”(+) とは f_1+f_2 が実数順序 FOL で，その値が次の表により決る binary 演算子をいう．

f_1 \ f_2	Ω	θ	R_2	A_2
Ω	Ω	Ω	Ω	Ω
θ	Ω	θ	θ	Ω
R_1	Ω	θ	R_1+R_2	Ω
A_1	Ω	Ω	Ω	Ω

ここに R_1 および R_2 は任意の実数，
A_1 および A_2 はそれ以外の値とする．

（3） 定義

“積”(*) とは f_1*f_2 が実数順序 FOL でその値が次の表により決まるような binary 演算子をいう．

f_1 \ f_2	Ω	θ	R_2	A_2
Ω	Ω	Ω	Ω	Ω
θ	Ω	θ	θ	Ω
R_1	Ω	θ	R_1*R_2	Ω
A_1	Ω	Ω	Ω	Ω

上の例では
$f_9*f_5+.2*f_6$＝週給である．

（4） 定義：“商”(／) は f_1/f_2 が実数順序 FOL でその値が次の表に従って決定される binary 演算子である．

f_1 \ f_2	Ω	θ	O	R_2	A_2
Ω	Ω	Ω	Ω	Ω	Ω
θ	Ω	θ	Ω	θ	Ω
O	Ω	θ	Ω	O	Ω
R_1	Ω	θ	Ω	(R_1/R_2)	Ω
A	Ω	Ω	Ω	Ω	Ω

ここで，R_1 と R_2 は0でない実数とする．

上の例では f_5/f_3 は生産性指数を与え，ボーナス指数を決定するために用いられる．

（5） 定義：“負号”(−) とは $-f$ が実数順序 FOL でその値が次の表により決定される unary** 演算子のことをいう．

f	Ω	θ	O	R	A
$-f$	Ω	θ	O	$-R$	Ω

（6） 定義：OR(⊕) とは $f_1 \oplus f_2$ が boolean FOL でその値が次の表に従って決るような binary 演算子のことである．

f_1 \ f_2	Ω	F	θ	T	A_2
Ω	Ω	Ω	Ω	Ω	Ω
F	Ω	F	θ	T	Ω
θ	Ω	θ	θ	T	Ω
T	Ω	T	T	T	Ω
A_1	Ω	Ω	Ω	Ω	Ω

* ここでいう binary は2進法のことではない．演算子の対象として二つの被演算子があることを意味している．
** unary も上の binary と同じような使い方である．次頁の ternary も同様．

上の例では $f_7=F$, $f_9=T$ の場合には $f_7 \oplus f_9 = T$ である.

(7) 定義: AND @ とは f_1 @ f_2 が boolean FOL でその値が次の表により決定される binary 演算子のことである.

f_1 \ f_2	Ω	F	θ	T	A_2
Ω	Ω	Ω	Ω	Ω	Ω
F	Ω	F	F	F	Ω
θ	Ω	F	θ	θ	Ω
T	Ω	F	θ	T	Ω
A_1	Ω	Ω	Ω	Ω	Ω

(8) 定義: "否定" (¬) とは $\neg f$ が boolean FOL でその値が次の表により決定される unary 演算子をいう.

f	Ω	F	θ	T	A
$\neg f$	Ω	T	θ	F	Ω

(9) 定義: "等号" (=) は $f_1=f_2$ が selection FOL でその値が次の表から決定される binary 演算子をいう.

	$f_1=f_2$
$v_1=v_2$	T
$v_1 \neq v_2$	F

ここに v_1, v_2 はそれぞれ f_1, f_2 の値である.

上の例では f_9 を定義するために用いる.

(10) 定義: "より小" (<) とは $f_1<f_2$ が selection FOL で, その値が次の表により決定される binary 演算子をいう.

	$f_1<f_2$
$v_1<v_2$	T
その他の場合	F

上の例では f_7 を定義するために用いた.

(11) 定義: "If—otherwise" (←, →) とは $f_1 \leftarrow f_2 \rightarrow f_3$ が FOL でその値が次の表から決定される ternary 演算子をいう.

f_2	$f_1 \leftarrow f_2 \rightarrow f_3$
T	v_1
F	v_3
θ	θ
その他の場合	Ω

上の例では f_8 は, $.20 \leftarrow f_7 \rightarrow .25$ で定義されている.

8. 情報代数学の拡張

この節では情報代数学の基本概念のいくつかの使用法を説明し, それらをより身近な概念に結びつける. とくに, 以下で行なうファイルについての議論により, 属性空間におけるファイル情報の表現はユニークなプロセスでないことが明らかになる. このことから, この代数学のある種の拡張が示唆され, その若干を以下で取り扱う. バンドル, グランプまたはその両方の基本的な定義から抽出された表現法の導入をバンドル, グランプまたはバンドルーグランプの "コンストラクト" と呼ぶ.

(1) 定義: "バンドル・アーギュメント" A_i とは領域集合に含まれる領域の一つで, その上でバンドル B

$$B=B(b;\ Z)$$

が定義されているものをいう.

(2) 定義: F をバンドルの函数, L をそのバンドルのある線とするとき

$$p_F(L)$$

は F により L につけられた点を表わす.

(3) 定義: "バンドル B とバンドル・アーギュメント A_i との共通部分 $I_i(B;\ A_i)=B_nA_i$" とは, バンドル B のどれかの線 L の上にある A_i のすべての点の集合をいう.

(4) 定理: "ある領域集合 Z の上のバンドル B とそのバンドル・アーギュメント A_i に対し

$$I_i(B;\ A_i)=F_i(B)$$

なる FOB F_i が存在する."

(証明)

$F_i \equiv \{q_j'=q_{ij},\ j=1,\ 2,\ \cdots\cdots,\ m\}$ とすれば, F_i はバンドル B のすべての線 L に対し, L と A_i の共通集合である点 $p_F(L)$ を対応させる.

(5) 定理: "任意の領域集合 Z とその集合の任意の領域 A_i に対して, Z 上のバンドル B と FOB F_i が存在して,

$$A_i=F_i(B)$$

である."

(証明)

すべての可能な線に対して True という値をもつようなバンドリング函数を b とする.

各 A_i のすべての点はこのバンドルのある線に含まれる. 函数 F_i を

$$F_i \equiv \{q_j'=q_{ij},\ j=1,\ 2,\ \cdots\cdots,\ m\}$$

で定義する. そのとき

$$I_i(B;\ A_i)=A_i=F_i(B)$$

である.

（6） 定義：バンドルの各線に含まれている点の全体の集合を“そのバンドルの領域”（AOB=area of bundle）と呼び，$A(B)$ とかく．

この定義から $A(B)$ は新らしい primitive になると思われる．実際，$A(B)$ は B と A_i の共通部分 I_i の結びである．各 I_i は FOB $F_i(B)$ により定義される．したがってバンドルの領域（AOB）はバンドル・コンストラクトである．すなわち

$$A(B)=F_1(B)\cup F_2(B)\cup\cdots\cdots\cup F_n(B)$$

（7） 定理：“任意の領域集合 $Z=(A_1, \cdots\cdots, A_n)$ に対して

$$A(B)=A_1\cup A_2\cup\cdots\cdots\cup A_n$$

なるバンドル B が存在する．”（証明略）

（8） 記号：A_i の点で I_i に含まれないものは A_i に関する I_i の補集合である．それを $I_i'(A_i)$ とかく．また，全属性空間に対する I_i の補集合を I_i' で表わす．

$I_i'(A_i)=A_i\cap(I_i')$ である．

8.1 ファイル更新へのハンドルの応用

ファイル更新を最も簡単な場合について考察しよう．すなわちマスター・ファイルに，あるレコードを加えたり，引いたりあるいは他のファイルのレコードとマッチングして得られた情報を用いてマスター・ファイルのレコードを更新すること．その際，マスター・ファイルの残りの部分は完全に手をつけずに保つ．

A_n をマスター・ファイルの記録を表示する点からなる領域とする．A_1 を新レコード，$A_2, A_3, \cdots\cdots, A_{n-1}$ をトランザクションレコードのファイルを示す領域とする．また，A_n^{new} を A_n を“更新して”得られた領域，すなわち更新マスター・ファイルとしよう．バンドル・コンストラクト U_p に対して次のような表現が得られるであろう．

$$A_n^{new}=U_p(b;\ F_1;\ A_1, A_2, \cdots\cdots, A_n)$$

各，A_i についてはすでに説明したとおり，b と F_1 については以下に説明する．

U_p を導く前に，まず，b は $A_2, A_3, \cdots\cdots, A_{n-1}$ のレコードを A_n のレコードとマッチさせるために定義されたバンドリング函数であることに注意しよう．このことからバンドル $B(b, A_2, A_3, \cdots\cdots, A_n)$ が得られる．FOB F_1 により $A_2, \cdots\cdots, A_{n-1}$ のトランザクションと A_n のマッチしたマスター・ファイルの処理が行なわれる．その結果，更新されたレコードが得られ，領域 $M=F_1(B)$ ができる．FOB F_1 は各更新作業に応じて異なるものである．それは，マスター・ファイル（A_n）のレコードがトランザクション・ファイル（$A_2, A_3, \cdots\cdots, A_{n-1}$）のレコードにより更新される規則を述べるものである．たとえば，給与計算ファイルの場合では，普通の更新としては，給与率の変更，アドレス，免税の変化 year-to-date の総支給額の変更が行なわれる．各々の場合に，FOB はある線に対しては null 点を対応させ，あるレコードを“削除”する．

b により更新のために選出される A_n のレコードは A_n^{new} のなかに合体してしまってはならない．それらのレコードはバンドル B と A_n との共通部分 $I_n(B; A_n)$ を構成する．レコード $I_n(B; A_n)$ を除去し，A_n の更新されないレコードを合体化することは，A_n に関する $I_n(B; A_n)$ の補集合，領域 C を作ることにより達成される．すなわち $C=I_n'(A_n)$

したがって領域 A_n^{new} は C，A_1 および M の和集合である．

$$A_n^{new}=A_1\cup M\cup C=U_p(b;\ F_1;\ A_1, A_2, \cdots\cdots, A_n)=A_1\cup F_1(B)\cup I_n'(A_n)$$

8.2 グランプをファイルストラクチャーに適用すること

磁気テープ上での通常の情報構成を考えよう．

一組のレコード（個体を表現する）R_1 からなる簡単なファイル・File-1 を先ず考察する．各レコードはそれぞれのフィールド（属性）$f_1, \cdots\cdots, f_{nR}$ をもつものとする．

最初のフィールド f_1 は sort-key であり，レコード R はテープ上にこの sort-key f_1 の順序でレコードされているものとする．情報代数学の概念では $f_1, \cdots\cdots, f_{nR}$ は属性空間 P の属性座標としてえらばれており，File-1 のすべてのレコード集合は P のある領域を示すであろう．

File-1 は一種類の個体に関する情報しか含まないという意味で単純なものである．その個体としてたとえば，衝突保険証書をとり，レコードされた属性としては証書番号，エンジン番号，その他車種，有効期間等々をとってもよい．

さて File-1 より複雑な File-2 を考察してみよう．今度は header R と trailer S をもち次のような形式としよう．

R

$f_1(R)$

$f_2(R)$

⋮

$f_{nR}(R)$

S　　$f_1(R)$

$f_1(S)$

$f_2(S)$

⋮

$f_{nS}(S)$

このファイルでは，R は属性 $f_1(R), f_2(R)\cdots\cdots f_{nR}(R)$ をもつ一つの個体型であると考えられる．Rは衝突保険証書，$f_1(R)$ は証書番号，$f_2(R)$ は証書の有効期間，等々とする．各 R はまた，別の型のエントリイを表現する任意の数のグループ S を含むことができる．その属性 $f_1(R)$ は，それが属する R を指定し，属性 $f_1(S), f_2(S), \cdots\cdots, f_{nS}(S)$ は S を記述するフィールドである．S はその衝突証書で保証された車であり，$f_1(S)$ は車のエンジン番号，等々である．こうすれば一つの証書の下で，任意の台数の車が保証されたことになる．したがって各 R には多くの S が個体として属することになる．

このファイルを情報代数学で取り扱うには，次のような属性をもつ属性空間が選ばれる．

$f_1(R), f_2(R), \cdots\cdots, f_{nR}(R), f_1(S), f_2(S), \cdots\cdots, f_{nS}(S)$

R-header としては属性 $f_1(R), f_2(R), \cdots\cdots, f_{nR}(R)$ をもち，属性 $f_1(S)\cdots\cdots f_{nS}(S)$ が Ω であるようなデータ点によって表現された個体を考える．同様に S グループとしては $f_2(R), \cdots\cdots, f_{nR}(R)$ が Ω で $f_1(R), f_1(S), \cdots\cdots, f_{nS}(S)$ が定義されているデータ点によって表現された個体を考える．前者のデータ点集合は領域 A_R をなし，後者は A_S を作る．ある与えられた R に属する S を表示するすべてのデータ点は常数 $f_1(R)$ による A_S のグランプの要素を構成する．両方のタイプの個体を表現するすべてのデータ点の集合は，ある領域 A_{RS} をなし，それは A_R と A_S の和集合である．

9. 例題（給与計算）

問題は与えられた Old Pay File と Daily Work File と New Employee File から New Pay File を作ることである．Daily Work File は各従業員の毎日の労働時間を一週間分ファイルしたもの，Old Pay File と New Pay File はその従業員の稼いだ給料の total を含むその他の情報を含んでいる．New Employee File は新らしい従業員に関する支給率情報をもち，それが New Pay File に組込まれる．

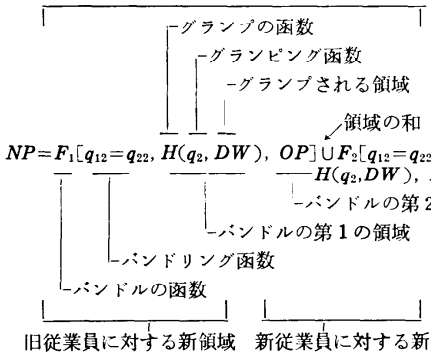
第1表　Payroll Problem-System Information

属　性	値集合	領域			
		Old Pay File (OP)	Daily Work File (DW)	New Employee File (NE)	New Pay File (NP)
q_1=File ID	PF, DW, NE	X (always PF)	X (always DW)	X (always NE)	X (always PF)
q_2=Man ID	00000…99999	X	X	X	X
q_3=Name	20 alphabetic characters	X	Ω	X	X
q_4=Rate	00.00…99.99	X	Ω	X	X
q_5=Hours	00…24	Ω	X	Ω	Ω
q_6=Day	0…7	Ω	X	Ω	Ω
q_7=total salary	00000.00…99999.99	X	Ω	Ω	X
q_8=pay period	00…52	X	Ω	X	X
q_9=Salary	000.00…999.00	X	Ω	Ω	X

この Payroll 問題に対する解は次の関係によって完全に表現される．New Pay File は二つの領域の和として表わされる．一つは旧従業員の領域，一つは新従業員の領域である．これらの領域の各々は逆に二つの領域のバンドルの函数である．各バンドルの最初の領域は等しく，Daily Work File のグランプの函数である．一方のバンドルの第二の領域は Old Pay File であり，他方のバンドルの第二の領域は New Employee File である．

（1）式の各項は次図と等価である．

New Pay File に対する表現



$$NP=F_1[q_{12}=q_{22},\ H(q_2,\ DW),\ OP]$$
$$\cup F_2[q_{12}=q_{22},\ H(q_2,\ DW),\ NE]\quad(1)$$

$$H\equiv\begin{cases} q_2'=q_2 \\ q_5'=\Sigma[q_5\leftarrow(q_5<8)\rightarrow 1.5*q_5-4]+f_1 \\ f_2=\Sigma[q_5\leftarrow(q_5<8)\rightarrow 8] \\ f_1=0\leftarrow(f_2<40)\rightarrow 0.5*f_2-20 \end{cases}$$

$$F_1 \equiv \begin{cases} q_7' = q_{27} + q_{15} * q_{24} \\ q_8' = q_{28} + 1 \\ q_9' = q_{15} * q_{24} \end{cases}$$

$$F_2 \equiv \begin{cases} q_1' = PF \\ q_7' = q_{15} * q_{24} \\ q_8' = q_{28} + 1 \\ q_9' = q_{15} * q_{24} \end{cases}$$

〔解説 6〕

(1) 式の $H(q_2, WD)$ については〔解説 5〕でふれたが，要するに Daily Work File (WD) を，q_2(man 番号) をグランピング函数 (key) としてまとめて，そのグランプから

$$H \equiv \begin{cases} q_2' = q_2 \\ q_5' = \Sigma[q_5 \leftarrow (q_5 < 8) \rightarrow 1.5 * q_5 - 4] + f_1 \\ f_2 = \Sigma[q_5 \leftarrow (q_5 < 8) \rightarrow 8] \\ f_1 = 0 \leftarrow (f_2 < 40) \rightarrow 0.5 * f_2 - 20 \end{cases}$$

によって新らしい領域を作ったわけである．f_1, f_2 は領域の函数であり，$q_5 \geqq 8$ (週 5 日とも) ならば $f_2 = 40$, $q_5 < 8$ (週 5 日とも) ならば $f_2 = \Sigma q_5$ である．また $f_2 < 40$ ならば $f_1 = 0$, そうでなければ 40 時間を越えた分の半分が f_1 である．q_5' の over-time 支払い規則は次のとおり．毎日の 8 時間を越える勤務時間にたいしては通常時間の 1.5 倍とみなす．そして，週 40 時間を越える分にはさらにその半分を加算する．(週 5 日以上の勤務)．$H(q_2, WD)$ の q_2, q_5 以外の属性は未定義である．

次に $F_1[q_{12} = q_{22}, H(q_2, DW), OP]$ については〔解説 4〕でふれたとおりである．この函数では q_7', q_8', q_9' 以外の属性は定義されていないが，OP のものがそのまま生きるので

$q_1' = PF$, $q_2' = \text{man}\, ID$, $q_3' = \text{Name}$, $q_4' = \text{Rate}$, $q_5' - \Omega$ $q_6' = \Omega$

である．

同様に $F_2[q_{12} = q_{22}, H(q_2, DW), NE]$ が定義されている．これは New Employee File の処理である．F_1 と F_2 を集合として加えることにより New Pay File が得られるわけである．

参考文献

1) An Information Algebra Phase I Report-Language Structure Group of the CODASYL Development Committee, Communications of the ACM Vol. 5, No. 4, April. 1962 pp. 190～204
2) Lionello Lombardi: Mathematical Structure of Nonarithmetic Data Processing Procedure, Journal of the ACM, Vol. 9, No. 1, January 1962 pp. 136～159
3) 関根，刀根：データ処理理論の展望，第 4 回プログラミング，シンポジュウム報告集，January 1963, pp. Al～A 40.

(昭和 38 年 11 月 8 日受付)