北海道大学工学部研究報告
第78号（昭和51年）

Bulletin of the Faculty of Engineering,
Hokkaido University, No. 78 (1976)

有限関係の表現とそのデータ処理への応用

山　口　　忠*

（昭和50年6月30日受理）

A Representation of Finite Relations and its applications to data processing

Tadashi YAMAGUCHI

(Received June 30, 1975)

Abstract

The finite relation is often a useful mathematical tool for describing some relationships. In this paper, its representation with some subsets is discussed. Especially, a space analysis is described for storing finite relations in a computer and a time analysis for searching such relations is also reported.

1. まえがき

有限関係という概念は，数学ばかりではなく他の多くの分野においても基本的でよく用いられている。しかし，有限関係そのものは構造がゆるいというせいもあって，その研究は多くない。それも主に2項関係だけである。Haralick[1]は，2項関係の極大な直積的部分集合（定義1.参照）による表現（Diclique表現）を導入し，その代数的性質について発表している。筆者はHaralickとは独立に先の発表[2]において，一般の有限関係の表現を定義して，2, 3考察を行った。今回導入する表現は，応用を考慮して先の表現を少しゆるめたものである。本報告は，先の発表[2]に続くもので次の2点について述べる。

1) n項関係の直積部分集合による表現とそのrankについて。

2) その表現のデータ表現（構造）への応用とそれに関連する問題の量的考察。

2. 用語と定義

以下で考える集合はすべて有限集合である。n個の集合$S_1, S_2, \cdots, S_n$から作られる直積集合を次のように表す。また$\mathfrak{S}$の部分集合をとくにn項関係という。以下，$n \geqq 2$とする。

$$\mathfrak{S} = \mathsf{X}_{i=1}^{n} S_i = \left\{ (a_1, a_2, \cdots, a_n) \,\middle|\, a_i \in S_i (i=1, 2, \cdots, n) \right\}$$

定義1. n項関係$T \subseteq \mathfrak{S}$に対して，$T_i \subseteq S_i (i=1, 2, \cdots, n)$が存在して，$T = \mathsf{X}_{i=1}^{n} T_i$になるとき，$T$は直積的であるという。このとき，$|T_i| (T_i$の位数$) \geqq 2$なる$i$の個数を$T$の指数という。

定義2. n項関係$R \subseteq \mathfrak{S}$に対して，Rの直積的部分集合（以下，d-集合という）による被覆を

* 工学部電気工学科　系統工学講座

とくに d-被覆と言う。また被覆に寄与している d-集合がたがいに共通部分が空のとき，d-分割という。さらに，いくつかある d-被覆 (d-分割) の中で寄与している d-集合の個数が最小のものを最小 d-被覆 (対応して，最小 d-分割) という。この最小数を rank といい，rank (R) で表す。与えられた R に対して，その最小 d-被覆と最小 d-分割の値は一般に異なるが，どちらの値であるかは文脈からわかる。

3. R の最小 d-被覆と rank について

すべての要素が陽に与えられる形で R が規定されているとき，それから R の最小 d-被覆を求める方法について述べる。また，R に対して rank の上限についての 1 つの結果をあげる。

i) 最小 d-被覆を求める方法

最小 d-被覆は，集合被覆の特別なものであるため，一般の被覆問題[3] へもっていくための前処理を必要とする。以下ではその前処理部分のみについて述べる。

1) 指数 0 の R の d-集合をすべて求める。これは，R の要素 $(a_1, a_2 \cdots, a_n)$ から，d-集合 $\mathsf{X}_{i=1}^{n}\{a_i\}$ を求めることにほかならない。

2) 指数 i の d-集合がすべて求まったとして，それから指数 $(i+1)$ の d-集合を求める (この段階で新らたに得られる d-集合がなかったらここで終える)。

以上で得られた d-集合をもって R の被覆問題を解けばよい。d-分割を求めるには，disjoint の条件を入れて解けばよい。1), 2) の前処理は何の変哲もない単純なものであるが，今のところほかに使えるものがない。

ii) rank について

与えられた R に対して，被覆問題を解かずに rank を評価することは興味あるが，現在得ている次の結果についてだけ述べておく。

結果 1.

1) p_i を S_i の位数とする $(i=1, 2, \cdots, n)$。$R \subseteq \mathfrak{S} = \mathsf{X}_{i=1}^{n} S_i$ に対して次式が成立する。

$$\text{rank}\,(R) \leqq p_1 p_2 \cdots p_n / \max_i \{p_i\}$$

2) どんな $p_1, p_2, \cdots, p_n$ に対しても上式の等号をみたす R が存在する。

証明： 1) については文献 2) を参照。2) について： $p_n = \max\{p_i\}$ とし，各 S_i を次のようにする；$S_i = \{0, 1, \cdots, p_i - 1\}$。

各 $u=(a_1, a_2, \cdots, a_{n-1}) \in \mathsf{X}_{i=1}^{n-1} S_i$ に対して $\sum_{j=1}^{n-1} a_j \pmod{p_n} \in S_n$ を対応づける写像を f とする。この f を用いて次のように R を定義すれば，この R は $\text{rank}\,(R) = |R| = p_1, p_2, \cdots, p_{n-1}$ であることは容易にわかる。

$$R = \left\{ (u, f(u)) \,\middle|\, u \in \mathsf{X}_{i=1}^{n-1} S_i \right\} \subseteq \mathfrak{S} = \mathsf{X}_{i=1}^{n} S_i$$

4. d-被覆のデータ処理への応用

与えられ n 項関係 R を計算機へストアする方法として種々考えれる。そのうち，従来からある辞書式順序での単純なストア方法と，ここで導入する d-被覆を用いてのストア方法を対比させて，ストアに要する領域数とストアされた R に対する探索時間 (項の比較回数) について考える。この両方式の差はある程度容易に見当のつくことであるがあるモデルのもとで量的評価を行った。

i) 方式1： R の各要素に対して n 記憶単位を割り当て，第 n 成分が先に変化するような辞書式順序でストアする（この順序は必要領域数には影響を与えない）。

ii) 方式2： $\{T^1, T^2, \cdots, T^k\}$ が R の d-被覆 (d-分割) として，Fig. 1 のようにストアする。ただし図には $T^i=\mathop{\mathsf{X}}_{j=1}^{n} T_j^i$ の部分のみを示してある。

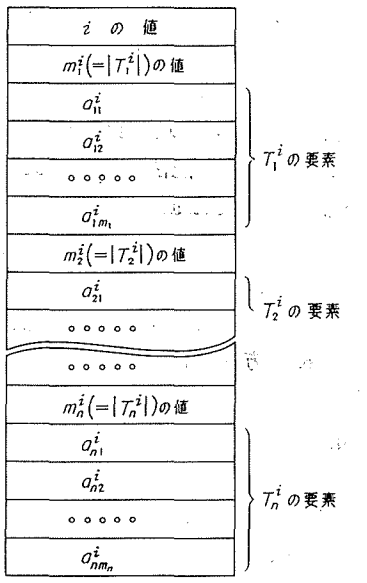

Fig. 1. Storing configuration of T^i

4.1 記憶領域数について

方式1では，$M_1=n\cdot|R|$ 単位の領域が必要になる。一方，方式2においては，T^i をストアするのに $(1+n+\sum_{j=1}^{n}|T_j^i|)$ 単位必要であるから R 全体としては次の M_2 となる。

結果2

1) $M_1=n\cdot|R|$

2) $M_2=(1+n)\,k+\sum_{i=1}^{k}\sum_{j=1}^{n}|T_j^i|$

方式2にとって best なケースを考えてみる。$T^1, T^2, \cdots, T^k$ が disjoint で（すなわち d-分割のとき），すべての i, j について $|T_j^i|=m\geqq 2$ とする。このとき，$|R|=km^n$ となるから，

$$M_1=nkm^n, \qquad M_2=nk\,(m+1)+k$$

となる。一方，$m=1$ のときは，方式2にとっては worst ケースになり，

$$M_1=nk, \qquad M_2=(2n+1)\,k$$

となる。

4.2 探索時間について

以下では，R がすでにストアされているとして，query $u\in R$ に対して，それがストアされている位置を決めるまでに走査する記憶単位の比較回数を考える。探索時間を考える際，他にも関連する量（たとえば，ループのための制御変数の計算など）があるここでの議論ではこれらを無視した。

u に対する比較回数 $c(u)$ は，R のストア方式および使用する探索アルゴリズムによって決まる量なので，一般論は展開できない。そこで次の様なモデルを考えて両方式の対比のもとで $c(u)$ の値および特別なケースの $c(u)$ とその平均値を求める。

次のモデルを考える：

1) $R=\bigcup_{i=1}^{k} T^i$（R の d-分割）；$T^i=\mathop{\mathsf{X}}_{j=1}^{n} T_j^i$，$m_j^i=|T_j^i|$

2) $i\neq i'$ ならば $T_1^i\cap T_1^{i'}=\phi$（この仮定により1)は R の d-分割になる）。

3) 方式1によるストアでは，$T^1, T^2, \cdots$ の順にストアされる。

ただしそのとき，各 T^i ごとに辞書式順序（第 n 成分が先に変化する）でストアされているとする。また，各 T^i に対応する部分を block といい B^i で表わす。さらに，各 block は，その第1成分の値のテーブル T_1^i を持っているとする。

2) と 3) の仮定は，議論を簡単にするためと，方式1の $c_1(u)$ と方式2の $c_2(u)$ の対比のために入れたものであるが，$c_1(u)$ の値の減少化をはかっている。したがって 3) で第1成分以外のテーブルも持たせたらどうなるかという別の問題が出てくるがここでは扱わない。

以下で，$u=(2, 3, \cdots, 10)\in T^j$ とか $u=(l_1, l_2, \cdots, l_n)\in T^j$ のような記法を用いるが，これは u の第1成分が T_1^j の2番目の要素であり，第2成分が T_2^j の3番目，第 n 成分が T_n^j の10番目の成分であることを意味する。方式1でのこの記法から，block の先頭よりの位置を求めるには，Fortran 言語などでよく知られている多次元配列要素の位置ぎめの手法[4]で行う。これは $c_1(u)$ の導出で用られている。

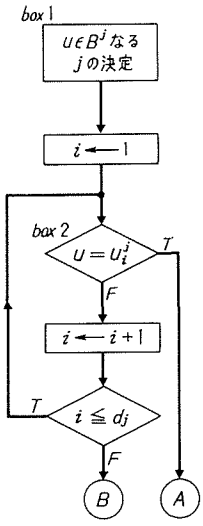

Fig. 2.

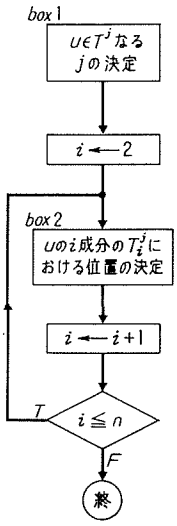

Fig. 3.

4.2.1 方式1における比較回数 $c_1(u)$

R の方式1におけるストアの blocks を $B^1, B^2, \cdots, B^k$ とし，$B_i=\{u_1^i, u_2^i, \cdots, u_{di}^i\}$ とする。$u\in R$ に対して Fig. 2 のアルゴリズムを用いる。ただし box 1 は3)で述べたテールブを用いての探索を行い u の属している B^j を決める。また，box 2 では u と u_i^j の第1成分から順に比較を行い等しくない成分で下の box へ行き，すべての成分が等しいとき Ⓐ へ出るものとす。このアルゴリズムにおいて，$u=(l_1, l_2, \cdots, l_n)\in B^j$ に対して Ⓐ に出るまでの box 1, box 2 で行われる成分ごとの比較回数 $c_1(u)$ は次式で与えられる。

結果3.

$$c_1(u)=\left[\sum_{i=1}^{j-1} m_1^i+l_1\right]+\left[\sum_{i=1}^{n} i(l_i-1)\,D_i^j+n\right] \qquad \text{ただし，} \quad D_i^j=\begin{cases} m_{i+1}^j\cdots m_n^j & (1\leqq i\leqq n-1)\\ 1 & (i=n)\end{cases}$$

第1項，第2項はそれぞれ box 1, box 2 での比較回数である（証明は付録 A を参照）。

4.2.2 方式2における比較回数 $c_2(u)$

$u=(l_1, l_2, \cdots, l_n)\in T^j$ に対して Fig. 3 のアルゴリズムを用いたときの，box 1, box 2 における比較回数 $c_2(u)$ は次式で与えられる。

結果4.

$$c_2(u)=\left[\sum_{i=1}^{j-1} m_1^i+l_1\right]+\left[\sum_{i=2}^{n} l_i\right]$$

第1項，第2項はそれぞれ box 1, box 2 での比較回数である。

このアルゴリズムでは，box 1 の判定の代りに i を1からはじめても同じことになるが，$c_1(u)$ との対応を考えてこうしてある。

4.2.3 特別な場合における c_1, c_2 の値とその平均値

$c_1(u)$, $c_2(u)$ の形からもその違いは読めるが，ここで $m_j^i=m\geqq 2\ (1\leqq i\leqq k,\ 1\leqq j\leqq n)$ という特別な場合について考える。結果5は，結果3, 4の書きかえである。

結果5.

$u=(l_1, l_2, \cdots, l_n)\in T^j(=B^j)$ のとき，c_1, c_2 は次のようになる。

$$c_1(u)=\left[(j-1)\,m+l_1\right]+\left[\sum_{i=1}^{n} i(l_i-1)\,m^{n-i}+n\right]$$

$$c_2(u)=\left[(j-1)\,m+l_1\right]+\left[\sum_{i=2}^{n} l_i\right]$$

さらに各 $u\in R$ が選ばれる確率が一様であるとして，c_1, c_2 の平均値を求めると次の結果を

得る。$E[c_2]$ の導出のみ付録Bにあげておく。

結果 6.

$$E[c_1]=\frac{1}{2}\left\{mk+n+\frac{m(m^n-1)}{m-1}+1\right\}$$

$$E[c_2]=\frac{1}{2}\left\{m(k-1)+n(m+1)\right\}$$

これらの結果から，方式1では m の値が本質的にきいてきていることが分る。$E[c_1]$ を小さくするには，m の減少化を計ればよいが仮定3)より限界がある。一方，方式2では，k と m の兼合いになっていることがわかる。

5. むすび

本報告では，n 項関係についての2通りの記憶方式を対比させ，テータ処理で問題になる領域量と探索時間（比較回数）についてのいくつかの結果をあげた。そこでの k, m の値に対しての R の d-分割（d-被覆）を求めるには前半で述べた被覆問題で解けばよい。また2方式の陽の比較を行うには更に詳細に議論する必要がある。特に方式1で用いた補助テーブル，アルゴリズムは ad hoc である。しかし以上の解析からだけでも，ある程度両方式の差の量的把握が出来たものと思う。

おわりに，日頃ご指導頂いている加地郁夫教授に感謝いたします。

引用文献

1) Haralick, R. M.: J. A. C. M. Vol. 21, No. 3, July, 1974.
2) 山口 忠: 昭和49年電気四学会北海道支部大会論文集.
3) Lawler, E. L.: J. SIAM. Appl. Math. Vol. 14, No. 5, Sept., 1966.
4) 浦 昭二:「データ構造」共立出版 (49年) p. 38.

付録

A. 結果3の証明

$u=(l_1\ l_2,\ \cdots,\ l_n)\in B^j$ なる u に対して，box 1 では，$T_1^i(i=1,\ \cdots,\ j-1)$ のテーブルを探索し，T_1^j の l_1 番目の比較で j が決まる。ゆえに box 1 での比較回数は，$\sum_{i=1}^{j-1}|T_1^i|+l_1$ となり，第1項が出る。

次に，box 2 での比較回数を考える。B^j における $u=(l_1,\ l_2,\ \cdots,\ l_n)$ 以前にストアされている要素で u と第 i 成分ではじめて異なるものの個数は，$(l_i-1)\,D_i^n$ 個あってそれらとは i 回の比較が行われる。このことは $i=1$ から n までについて言える。それとⒶに出る直前で行われる n 回の比較を加えれば，第2項を得る。

B. 結果6. の $E[c_2]$ の証明

$$E[c_2]=\sum_{u\in R}\frac{1}{km^n}\cdot c_2(u)=\frac{1}{km^n}\sum_{u\in R}c_2(u)=\frac{1}{km^n}\sum_{j=1}^{k}\sum_{*}\left[(j-1)\,m+\sum_{i=1}^{n}l_i\right]$$

和 * は，$1\leqq l_i\leqq m$ なるすべての順列 $(l_1,\ l_2,\ \cdots,\ l_n)$ を動く。

$$\sum_{*}[\quad]=m^{n+1}(j-1)+m^{n-1}\cdot\frac{m(m+1)}{2}\cdot n,\qquad \sum_{j=1}^{k}\sum_{*}[\quad]=\frac{km^n}{2}\left[m(k-1)+n(m+1)\right]$$

ゆえに，$E[c_2]=\frac{1}{2}\left[m(k-1)+n(m+1)\right]$。