認識プログラムFERTのリスト論的形式体系における表現

鈴木昇一

A Representation of Recognition Program FERT from the Point of View of a List-Theoretic Formalism

by Shoichi SUZUKI

A structural-fertilization pattern-recognition program of fixed-point type, FERT, proposed in a mathematical theory of recognizing patterns is represented in terms of a list-theoretic formalism, that is, by the use of five list-processing functions, i. e. head, tail, cons, null and if-then-else-fi.

1．まえがき

知識工学（knowledge engineering）上のIS-A（概念の階層関係），PART-OF（全体一部分関係），NUMBER-OF（集合内の要素数）から眺めた立場で，パターン認識（pattern recognition）を考えてみよう。

パターン（pattern）はあるカテゴリ（category，そのパターンが表現・帰属している範ちゅう）の具体例であり，パターンはカテゴリの下位概念である。パターンとカテゴリとの間のIS-A関係をパターン（全体）とこのパターンを構成する部分（パターン要素）との間のPART-OF関係から推断するのがパターン認識の働きである。

問題は最上位概念なるカテゴリの具体例としてのパターンの数（NUMBER-OF）があまりにも多いということで，知識工学上のいわゆる図式論理（schematic logic）が役に立たないことである。

パターン φ の持つ意味はPART-OF関係から定まるその構造（structure）に表わされていると考えられ，パターン φ においてはその構造に意味（主として，パターン φ の帰属するカテゴリ）が対応する。パターン φ に関し意味の処理を行なうということは，パターン φ の構造からその意味を取り出すことである。

原パターン φ は一般に，ISP（ill-structured pattern，不良構造パターン）であることが多く，これをWSP（well-structured pattern，良構造パターン）に変換していく多段の過程を介して，パターン φ の意味処理が行なわれ，この多段過程の最終結果はBSP（best-structured pattern，最良構造パターン）すなわち，WFP（well-formed pattern）としての，パターン φ の帰属するカテゴリの代表パターン ω であると考えてみよう。

$$\mathrm{ISP}\ (=\varphi)\rightarrow \mathrm{WSP}\ (=\varphi^{0})\rightarrow \mathrm{WSP}\ (=\varphi^{1})\rightarrow \cdots \rightarrow \mathrm{WSP}\ (=\varphi^{m-1})\rightarrow \mathrm{WFP}\ (=\omega)\ (=\mathrm{BSP}=\varphi^{m})$$

という多段過程を生成する構造受精法をパターン認識の数学的理論[1]が採用したのは，ISP（$=\varphi$）からその意味を取り出すよりも各WSP（$=\varphi^{t}$＝第t認識段階で得られるパターン）からその意味を取り出す方が容易で，更に，BSP即ちWFP（$=\omega$）からその意味を取り出す方が一層容易であるからである。

本論文では，パターン認識の数学的理論が提唱した不動点形構造受精認識プログラム（structural-fertilization pattern-recognition program of fixed-point type）FERT をリスト論的形式体系（list theoretic formalism）内で表現する。具体的には，5個のリスト処理関数（list-processing function）head, tail, cons, null, if-then-else-fi で表現することになる。

まず，第2，第3章で，FERT がある不動点方程式を解くことによって，あるいは短期記憶（short term memory）としてのある対リストを書き換えることによって認識の働きを発現していることを説明し，第4章，5章では FERT が人間のもつ認識の働きと似た性質をもっていることなどの特色を説明し，また，FERT に関し既に明らかになっている性質を説明する。第6章で，第7，8，9の3章で用いられるプログラム言語としての λ 言語を簡単に説明し，第7章でリストに対する操作を説明する。第8，第9章が本論文の要となるところであって，認識プログラム FERT の λ 言語による表現が与えられる。

2．認識プログラム FERT と不動点方程式

収縮写像Tと，その時点で判明しているカテゴリ候補番号のリスト γ を助変数とする構造受精作用素 $A(\gamma)$ とを導入し，パターン $\eta \in \Phi$ を変数とする関数

$$g(\eta)=TA(\gamma)T\eta-\eta$$

を考えよう。非線形方程式（不動点方程式）

$$g(\eta)=0$$

を有限回の演算で解くことを主目標としよう。

初期値として，その帰属するカテゴリを一意的に決定したい入力パターン $\varphi \in \Phi$ のモデル $\varphi^0=T\varphi$ を選び，この初期値から始め，反復法で真の解

$\eta=T\omega$（あるカテゴリの代表パターン ω のモデル）

に収束する近似解の例 $\{\varphi^t\}_{t=0,1,2,\cdots}$ を生成すればよいだろう。

その際，真の解に速く収束する列を可能な限り少ない手間で作りだすことが必要である。本論文で考察の対象とする認識プログラム FERT が提供する構造受精的反復法が果たしてこの意味で効率的といえるかどうかという疑問は残っているが，認識プログラム FERT は上の不動点方程式を解くことで認識の働きを発現するものである。

3．認識プログラム FERT と書き換え規則

その帰属するカテゴリを決定したい処理対象パターンを $\varphi \in \Phi$ とする。この φ から次の対 $[\varphi^t, \gamma^t]$ の列を考えよう：

$$[\varphi^0, \gamma^0], [\varphi^1, \gamma^1], \cdots, [\varphi^t, \gamma^t], \cdots, [\varphi^u, \gamma^u]. \qquad (3.1)$$

$\varphi^t \in \Phi$ は認識の第 t 段階で受精されて得られるパターンであり，γ^t はパターン φ^t の帰属する可能性のあるカテゴリ候補 $\mathfrak{C}_j$ のカテゴリ番号 j のリスト（カテゴリ候補番号リスト）であり†，

φ^t が γ^t 内のいずれか一つのカテゴリ番号をもつカテゴリに帰属すること

を，対リスト $[\varphi^t, \gamma^t]$ は表わしていると考えられる。

γ^0 はすべてのカテゴリ番号から成るリストであり，対リスト $[\varphi^0, \gamma^0]$ は公理と考えられ，（3．1）で示される対リストの列は公理 $[\varphi^0, \gamma^0]$ の下で成立する命題の列

$[\varphi^1, \gamma^1], [\varphi^2, \gamma^2], \cdots$

を $[\varphi^u, \gamma^u]$ が結論できるまで生成していくこと††

† 第 $j \in J$ 番目のカテゴリ $\mathfrak{C}_j$ にパターン φ^t が帰属する可能性の程度は類似度関数 SM を用いて，$SM(\varphi^t, \omega_j)$ と表わされる。ここに，$\omega_j \in \Omega \subset \Phi$ は $\mathfrak{C}_j$ の代表パターンである。

†† 入力パターン $\varphi \in \Phi$ から得られる最終のパターン $\varphi^u \in \Phi$ は ψ の意味モデルといわれ，

$$\varphi^u=0 \quad \vee \quad \varphi^u=TA(\gamma^u)T\varphi^u$$

を満たしていなければならない。

を表現している。

φ^0, φ^1, …, φ^{t-1}, φ^tから成るリストを

$\Psi^t=〔\varphi^t, \varphi^{t-1}, \cdots, \varphi^1, \varphi^0〕$

と表わし，γ^0, γ^1, …, γ^{t-1}, γ^tから成るリストを

$\Gamma^t=〔\gamma^t, \gamma^{t-1}, \cdots, \gamma^1, \gamma^0〕$

と表わし，対リスト

$〔\Psi^t, \Gamma^t〕$

を考えよう。本論文で考察の対象とする認識プログラム FERT の働きは上述の事態を反映し，この対リスト$〔\Psi^t, \Gamma^t〕$を用いて次のように表現される。

認識プログラム FERT に対リスト$〔\Psi^0, \Gamma^0〕$を入力して得られる対リストの系列（多段階認識過程）

$〔\Psi^0, \Gamma^0〕, 〔\Psi^1, \Gamma^1〕, \cdots, 〔\Psi^t, \Gamma^t〕, \cdots$

は入力パターン $\varphi \in \Phi$ に対する認識計算列（recognitive computation sequence）を提供している。見方を変えれば，認識プログラム FERT は短期記憶としての対リスト

$〔\Psi^{t-1}, \Gamma^{t-1}〕$

を一つだけ段階が進んだ対リスト

$〔\Psi^t, \Gamma^t〕$

に書き換えていく書き換え規則（rewriting rule）を内蔵しているシステムと考えられよう。

4．本認識プログラムの特色

知識の表現法，利用法（推論法），獲得・管理法，ユーザ・インタフェースなどを中心課題とする知識工学によれば，知識には三種類があり，パターン認識の数学的理論では，この知識の分類を次のように考えている。

（i）対象とする世界に関する事実

パターン φ とその集まり Φ，収縮写像T，類似度関数 SM，大分類関数 SGN，構造受精作用素A（γ）

（ii）新しい事実（解こうとする問題の状態）を作り出すのに必要な規則（operator あるいは可能な準備）

問題の状態$〔\Psi^t, \Gamma^t〕$を今一つ進んだ状態$〔\Psi^{t+1}, \Gamma^{t+1}〕$へと変換する規則

（iii）規則適用の戦略

不動点形構造受精認識プログラム FERT

□

不動点を探索する形式としての構造受精認識法のもつ特色を簡単に指摘してみよう。

(a)途中の認識過程を人間が理解可能な形式で示している。

$\varphi^t=〔\varphi^t, \varphi^{t-1}, \cdots, \varphi^1, \varphi^0〕$

$\Gamma^t=〔\gamma^t, \gamma^{t-1}, \cdots, \gamma^1, \gamma^0〕$

として，第 t 段階の認識過程は$〔\Psi^t, \Gamma^t〕$という問題状態で表現されているが，第 t 段階で新たに得られたパターン φ^tが帰属する可能性のあるカテゴリ候補はリスト γ^t内の各カテゴリ番号をもつカテゴリである。

しかも，

$\gamma^t=〔j_1, j_2, j_3, \cdots〕$

とすると，φ^tがカテゴリ C_{j_k}に帰属している確率は SM (φ^t, ω_{jk}) で与えられ，不等式

SM $(\varphi^t, \omega_{j1}) \leqq$ SM (φ^t, ω_{j2})

$\leqq$ SM $(\varphi^t, \omega_{j3}) \leqq \cdots$

が成立している。

(b)経験を利用できる形式である。

一度問題解決（認識対象としての入力パターンの帰属するカテゴリを決定すること）に成功したならば，そのとき得られた認識過程

$〔\Psi^0, \Gamma^0〕, 〔\Psi^1, \Gamma^1〕, \cdots, 〔\Psi^t, \Gamma^t〕, \cdots, 〔\Psi^u, \Gamma^u〕$

を一定の形式で蓄え記憶しておいて，途中で

$\Psi_2{}^{t2}=\Psi_1{}^{t1}$, $\Gamma_2{}^{t2}=\Gamma_1{}^{t1}$

が成立して合流する別な問題

$〔\Psi_2{}^0, \Gamma_2{}^0〕, 〔\Psi_2{}^1, \Gamma_2{}^1〕, 〔\Psi_2{}^2, \Gamma_2{}^2〕, \cdots$

の解決に利用できる。

(c)自己組織化の効果を取り入れ可能な形式である。

誤認識した場合，帰属可能性のないカテゴリを除外する役目をもつ大分類関数 SGN 内の隠されたパラメータ（自己組織化可能な要素）を変更できる余地を残している。

(d)入力パターン φ を変形できなくなる迄

変形してしまい，しかもこの変形の過程

$$\langle \Psi^t, \Gamma^t \rangle \quad \rightarrow \quad \langle \Psi^{t+1}, \Gamma^{t+1} \rangle$$

は非可逆である。変形は不動点方程式

$$\varphi^n = 0 \quad \vee \quad \varphi^u = \mathrm{TA}(\gamma^u)\, \mathrm{T}\varphi^u$$

が成立する迄続いて，認識の過程

$$\langle \Psi^0, \Gamma^0 \rangle, \langle \Psi^1, \Gamma^1 \rangle, \cdots, \langle \Psi^t, \Gamma^t \rangle, \cdots, \langle \Psi^u, \Gamma^u \rangle$$

が得られ，一種の reduction（還元）を行なっている。しかも不等式

γ^t内の要素の総数$\geqq\gamma^{t-1}$内の要素の総数

が成立しているから，認識の段階が進むにつれて，入力パターン φ の帰属するカテゴリの候補は増加していない。

このように，不動点探索形構造受精認識法は，ある意味では，人間の認識的思考に近いレベルで認識活動を実現するとみることができよう。

5．これ迄に解決している諸点

パターン認識の数学的理論[1]は唯単に特定の問題を解決するのではなく，特定のパターン認識問題に関する知識が与えられたときに，それを用いて，この与えられた特定の問題を解決できるための枠組であり，

「機械的に実行可能な認識の働き」

を数学的に明確にしようとしている。

これ迄，処理対象とする入力パターン $\varphi \in \Phi$ に対し，

(1)認識プログラム FERT の実行とは φ から得られる対リストに不動点形構造受精変換を繰り返すことであり，

(2)認識プログラム FERT の停止とは，実行の結果，大部分の場合 φ の既約形に一致する意味パターン φ_{sem} を得ることである，

という考え方に関連する研究がなされている。

まず，プログラム FERT による認識（recognition according to program FERT）における停止問題（halting problem）が議論された。FERT は当然ながら，停止する認識（terminating recognition）を提供すること，いいかえれば，認識プログラム FERT は多段認識過程を提供するが，カテゴリ総数内の認識段階で停止すること（認識停止定理）が証明された。この有限停止性は次の3性質 i，ii，iiiから保証されていることがその証明からわかる。

（i）カテゴリ総数が有限

（ii）入力パターンの帰属するカテゴリ候補が唯一つになる迄認識の段階が進むことがあり，一つ進む度に，帰属カテゴリ候補が一つずつ少なくなっている。

(iii)帰属カテゴリ候補が唯一つになる認識の最終段階では必ず不動点方程式

$$\mathrm{TA}(\gamma)\, \mathrm{T}\varphi^u = \varphi^u$$

ここに，γ は唯一つの要素から成るカテゴリ番号リスト

が成立する。

よって，認識プログラム FERT を改良するには各々，ii，iiiに関連し，

(イ)認識の段階が進む度に除外するカテゴリの決定方法

(ロ)帰属カテゴリ候補が唯一つになったときにどんな関係が成立するか見つけ，それを停止条件とする方法

を考案すればよい。

(イ)の方法を探究するに際して，類似度関数 SM の値がある値以下のカテゴリ候補はすべて次の認識の段階で除くようにすると，一つの入力パターンを認識するのに要する段階の総数はカテゴリ総数より少なくなることに注意すべきである。また，(ロ)に関連して，類似度関数 SM の値があるカテゴリに関し最大値の1に近くなると，不動点方程式が近似的に成立することが期待されるようにすることなどにも留意すべきである。

これ迄に次の4事実 I ～IVが明らかにされている。

（I）処理対象パターン φ の変換先（既約形）としての意味モデル φ_{sem} が $\varphi_{sem} \neq 0$

なる限り，あるカテゴリ$\mathfrak{C}_j$の代表パターンω_jのモデル$T\omega_j$になること……意味モデル定理，認識既約形定理

(II)処理パターンから既約形を得る過程が有限の記憶と有限の計算（書き換え）でなされること……認識停止定理

(III) ある程度以下の範囲で相違している，つまりある程度以上似ている二つのパターンは共に同じ既約形をもつこと……認識同値・既約形定理

(IV)ある程度を超えて異なる二つのパターンは各々別の既約形をもつこと……認識同値・既約形定理

6．λ言語

本論文では，認識プログラムFERTなどを関数型プログラム言語 (functional programming language) としてのλ言語 (λ language)で表現する。本章では，その簡単な記法を説明しておく。

変数xを引数 (argument) とする関数Qを

$G=\lambda x.\ Q$

と表わし，関数抽象(functional abstraction)という。更に，Qで自由な[†]変数xの出現をすべてRで置き換えて得られるλ式

$E=((\lambda x.\ Q)\ R)$

は，関数Gを値Rに適用することを抽象的に表わしたものであり，関数適用 (function application) と呼ばれる。

$\lambda x.\lambda y.M$すなわち

$(\lambda x.(\lambda y.M))$

を $\lambda xy.M$ と書くことがある。また，$\lambda x.AB$は$((\lambda x.A)\ B)$ではなくして，$(\lambda x.(AB))$のことである。

7．リストとその操作

リスト (list) とは有限個の要素に順序をつけて一列に並べたものであって，n個の要素$z_1, z_2, \cdots, z_n$から成るリストを，左括弧記号〔，右括弧記号〕を用いて

〔$z_1, z_2, \cdots, z_n$〕

で表わす。各要素z_jはアルファベット (alphabet) から選ばれた有限個の要素から成る列（有限系列）であれば，この要素記号はアトム (atomic symbol) と呼ばれる。アトムはこれ以上分解しては意味を持たないもので，少なくとも左右括弧〔，〕は含まれない。リストの要素がまたリストから成っているような

$L=〔z_1, 〔y_1, y_2, \cdots, y_m〕, z_3, z_4, \cdots, z_n〕$

なるリストLも定義できる。ここに，$y_1, y_2, \cdots, y_m, z_1, z_2, \cdots, z_n$はアトムである。この場合，Lの頂上レベルの要素とは$z_1, 〔y_1, y_2, \cdots, y_m〕, z_3, z_4, \cdots, z_n$なるn個の各々である。

リスト論的形式体系(list theoretic formalism)におけるリストLに対する操作として，次の5個なるリスト処理関数 (list processing function) i～vを考えよう。

(i) head (L) …リストLの最初の要素を与えるリスト頭部関数

headはheader part（頭部）の略であり，$L=〔x, y, z〕$ の場合 head (L)＝x．また，$L=〔〔a, b〕, x, 〔d, e〕, z〕$の場合head (L)＝〔a, b〕.

(ii) tail (L) …リストLから最初の要素(＝head (L))を取り除いて得られる要素から成るリストを与えるリスト尾部関数。

tailはtail part（尾部）の略であり，$L=〔〔a, b〕, x, 〔d, e〕, z〕$の場合，tail (L)＝〔x, 〔d, e〕, z〕.

なお，Lが一要素zのみから成るリスト〔z〕の場合（L＝〔z〕），tail (L)＝noth-

† Qの変数xによるλ抽象（λ－abstraction）とも呼ばれる（λx. Q）のxを束縛変数（bound variable）といい，束縛されていない変数を自由変数（free variable）という。

ing, つまり

(ii$_2$) tail ([z])=nothing

と約束し,

(ii$_3$) 空リスト nothing (=[ ]) は一つのアトム

と約束する。

(iii) cons (L_1, L_2)…要素L_1をリストL_2の頭に挿入して得られるリストを与えるリスト合成関数。

cons は constructor (構成子) の略であり, L_1=[a, b], L_2=[x, [d, e], z] の場合

cons (L_1, L_2)=[[a, b], x, [d, e], z].

また, L_1=a, L_2=[x, [d, e], z] の場合

cons (L_1, L_2)=[a, x, [d, e], z].

(iv) null (L) …リストLがnothingの場合 trueなる値をとり, そうでない場合 falseなる値をとる述語関数 (predicate function)

(v) cond (b, x, y)…述語 (predicate) bがtrueの場合xをとり, falseの場合yをとる条件関数 (conditional function)。

しかしながら, 以後, cond (b, x, y) を便宜的に

if b then x else y fi

で表わす。以後

(vi) head (nothing)=nothing

tail (nothing)=nothing

と約束する。

さて, head, tail, null を用いて定義される関数

(vii) last is

λL. if null (tail (L)) then head (L)

else last (tail (L))

fi

は, 引数としてのリストLの頂上レベルの最後の要素を与える関数である。last は last part (最尾部) の略であり,

L=[x, y, [a, b]]

の場合

last (L)=[a, b]

(viii) length is

λL. if null (L) then 0

else add 1 (length (tail (L)))

fi

, where add 1 (x)=x+1 (xは数値) は, リストLの頂上レベルの要素の総数 (リストLの長さ) を与える関数である。

たとえばL=[x, y, [a, b], z] の場合 length (L)=4

(ix) tail1 is

λL. if null (tail (L)) then L

else tail (L)

fi

は, リストLが空リスト nothing または唯1個の要素zから成るリスト[z]ならばLを与え, そうでない場合には tail (L) に等しい関数である。

(x) eq (x, y) … xがアトムかつyがアトムかつx=yならば true を与え, その他の場合は false を与える述語関数。

ii$_3$の約束の下で, eg を使えば, ivの null は

(iv$_2$) null is λL. eg (L, nothing)

と書ける。

(xi$_1$) equal (x, y) …x=yならば true を与え, その他の場合 false を与える述語関数

(xii) atom (L)…Lがアトムならば true を与え, その他の場合は false を与える述語関数

二つの基本的述語 atom, eq と二つの基本的関数 head, tail を用いれば, xi$_1$, の equal は次のように書ける。

(xi$_2$) equal is

λx. λy. if atom (x) then

if atom (y) then eq (x, y)

```
        else false
        if
  else
    if equal (head (x), head (y))
    then equal (tail (x), tail (y))
    else false
    fi
  fi
```

(xiii) member is

```
λx. λL.
if equal (L, nothing) then false
else
  if equal (x, head (L))        then
                                true
  else member (x, tail (L))
  fi
fi
```

は，xがリストLの頂上レベルの要素としてあればtrue，そうでなければfalseを与える述語である。たとえば，

member (E, 〔A, B, C, D〕) は true,

member (E, 〔A, B, 〔E, D〕〕) は false.

(xiv) less (x, y) …… 2つの数値x，yは関し，x＜yであればtrueを与え，x≧yであればfalseを与える述語関数

(xv) greater (x, y) …… 2つの数値x，yに関し，x＞yであればtrueを与え，x≦yであればfalseを与える述語関数

(xvi) assoc is

```
λx. λL.
if null (L) then nothing
else
  if equal (x, head (head (L)))
       then head (L)
  else assoc (x, tail (L))
  fi
fi
```

は，対リスト〔x_j, y_j〕（j≦1～n）を要素とするリストLにおいて，頭部がxなる対リストが存在すればそのような最初の対リストを与え，存在しなければnothingを与える関数である。

たとえば，

assoc (x_2, 〔〔x_3,y_3〕, 〔x_1,y_1〕, 〔x_2,y_2〕, 〔x_4, y_4〕〕)

は〔x_2,y_2〕.

(xv) sorta is

```
λL.
if null (tail (L)) then L
else cons (max (L), sorta (remax (L)))
fi
, where
max is
λL.
if null (tail (L)) then head (L)
else
  if greater (head (L), max (tail (L))) then
  head (L)
  else max (tail (L))
  fi
fi
and
remax is
λL.
if null (tail (L)) then nothing
else
  if greater (head (L), max (tail (L))) then
  tail (L)
  else
       cons (head (L), remax (tail (L)))
  fi
fi
```

は，リストL＝〔x_1, x_2, …, x_n〕の各要素が数値であるとき，これを大きい順に並べかえたリストM＝〔y_1, y_2, …, y_n〕を与える関数である。ただし，n＞0。なお，max (L) はリストLの中の最大要素を与える関数であり，また，remax (L) はリストLから最大要素を一つ取り除いて得られるリストを与える関数である。

(xvi) sortb is
λL.
if null (L) then L
else
cons (least (L), sortb (delete (least (L), L)))
fi
, where
least is
λL.
if null (tail (L)) then head (L)
else
if less (head (L), least (tail (L))) then head (L)
else least (tail (L))
fi
fi and
delete is
λx. λL.
if null (L) then nothing
else
if equal (x, head (L)) then tail (L)
else
cons (head (L), delete (x, tail (L)))
fi
fi

は，Lが空リスト nothing ならば，nothing を与え，そうでない場合，数値を要素とするリストLについて各要素を小さい順に並べかえて得られるリストを与える関数である。なお，least (L) はリストLの最小要素を与える関数であり，また，delete (x, L) はリストLの頂上レベルの要素で初めてアトムまたはリストなるxが生じる（があればそれ）を除いて得られるリストを与える関数である。たとえば，

delete (b, 〔a, 〔b〕 ,b, e, b〕)= 〔a, 〔b〕 , e, b〕 .

8．認識プログラム FERT の記述

その帰属するカテゴリを決定したい処理対象パターンを $\varphi \in \Phi$ としよう。

$\varphi^0 = T\varphi$

γ^0 = SORT (〔1, 2, ⋯, #(J)〕 , φ^0)

として，初期値としての対リスト

〔Ψ^0, Γ^0〕

をつくる。ここに，

nothing を空リスト，

#（J）を最大カテゴリ番号

として，

Ψ^0 = cons (φ^0, nothing)

Γ^0 = cons (γ^0, nothing)

である。SORT (γ, η) の説明は次の通りである。

認識抽象(recognition-abstraction)は次の六つの概念 i ～viで確保されている。

（i）パターン抽象　φ：M→Zの集まり $\Phi \subset$ NS（あるノルム空間）

ここに，M，Zは各々ある集合，複素数の集合（複素数体）で，集合Aから集合Bへの写像gを　g：A→B　と表わしている。

（ii）収縮写像抽象T：$\mathfrak{D}$(T) → Φ

ここに，$\mathfrak{D}$(T) はTの定義域であり，

$\Phi \subseteqq \mathfrak{D}$(T)

（iii）モデル抽象 $T\varphi \in \Phi$

ここに，写像Tを $\varphi \in \Phi$ に作用して得られる像を　$T\varphi$, T(φ), T・φ などで表わしている。

（iv）類似度関数抽象 SM: $\Phi \times \Omega \rightarrow \{s \mid 0 \leqq s \leqq 1\}$

ここに,Ω は第 $j \in J$ 番目のカテゴリ $\mathfrak{C}_j$ の代表パターンを　$\omega_j \in \Phi$　として，

$\Omega = \{\omega_j \mid j \in J\}$

と定義されており，集合A，Bの直積を　A×B　と表現している。

（v）大分類関数抽象 SGN：　$\Phi \times J \rightarrow \{0, 1\}$

ここに，Jはカテゴリ番号の全集合

(vi) カテゴリ候補番号リスト γ を助変数にもつ構造受精作用素抽象 $A(\gamma) : \Phi \to \Phi$.

さて，$SM(\varphi, \omega)$ は二つのパターン $\varphi \in \Phi$, $\omega_j \in \Omega$ の間の類似度(similarity)を与えることに注意して，$SORT(\gamma, \eta)$ はカテゴリ番号を要素とするリスト（カテゴリ番号リスト）γ とパターン $\eta \in \Phi$ とを変数にもち，一つのカテゴリ番号リストをその値とする関数であり，

すべての $j_k \in \gamma$ について，$SM(\eta, \omega_{jk}) \leqq SM(\eta, \omega_{jk+1})$, $k = 1, 2, 3, \cdots$ であれば，

$SORT(\gamma, \eta) = [j_1, j_2, j_3, \cdots]$. □

認識プログラム FERT に初期対リスト $[\Psi^0, \Gamma^0]$ を入力して得られる対リストの系列を

$[\Psi^0, \Gamma^0], [\Psi^1, \Gamma^1], \cdots, [\Psi^t, \Gamma^t], \cdots$

とする。ここに，

$\varphi^t \equiv TA(\gamma^{t-1})\, T\varphi^{t-1}$

$\gamma^t \equiv tail1\ (SORT(\gamma^{t-1}, \varphi^t))$

と定義されており，対リスト $[\Psi^t, \Gamma^t]$ 内の二つのリスト Ψ^t, Γ^t は

$\Psi^t = cons(\varphi^t, \Psi^{t-1})$

$\Gamma^t = cons(\gamma^t, \Gamma^{t-1})$

と表わされる。その実，Ψ^t, Γ^t は

$\Psi^t = [\varphi^t, \varphi^{t-1}, \cdots, \varphi^1, \varphi^0]$

$\Gamma^t = [\gamma^t, \gamma^{t-1}, \cdots, \gamma^1, \gamma^0]$

である。認識プログラム FERT は，等式

$FERT([\Psi^u, \Gamma^u]) = [\Psi^u, \Gamma^u]$

を満たす第 u 段階で停止する。ここに，#(J) は集合 J に含まれる要素の総数（カテゴリ総数）であり，

$0 \leqq u \leqq \#(J)$

が証明されている。

このとき，

$\varphi^u = head(head(FERT([\Psi^0, \Gamma^0]))))$

$\gamma^u = head(head(tail(FERT([\Psi^0, \Gamma^0])))))$

とおけば，

$\varphi^u = 0 \quad \vee \quad \varphi^u = TA(\gamma^u)\, T\varphi^u$

が成立している。

$\varphi_{sem} = \varphi^u \in \Phi$

$j_{sem} = last(\gamma^u) \in J$

を求め，認識器（recognizer）は，

I. 入力パターン φ をあたかもその意味モデル（semantic model）φ_{sem} のごとく理解しており，

II. 入力パターン φ の帰属する意味カテゴリ番号（semantic category number）を j_{sem} と解釈し，

A pattern φ belongs to category Cj_{sem} as though φ were a semantic model φ_{sem}

と認識推断する。

上で説明された働きをもつ認識プログラム FERT を関数言語としての λ 言語で表わすと次のようになる。FERT は不動点形構造受精認識プログラム（structural-fertilization patter-recognition program of fixed-point type)と称せられる。φ^{t-1}, γ^{t-1} からパターン φ^t を構造的につくりだしていることに注意しておこう。

```
FERT
=λ[Ψ, Γ].
  if head(Ψ)=0  ∨  head(Ψ)=ψ then
   [Ψ, Γ]
  else
     FERT (
          [cons (ψ, Ψ),
            cons (tail1  (SORT (head (Γ),
  ψ)                      )
                  ,Γ)]
               )
  fi,

  where
     ψ=TA (head (Γ))T head (Ψ).
```

9．SORTの，関数プログラムによる表現

関数プログラム（functional program）とは，プログラムとしての関数のデータ(引数)に対する適用（application）のみによって出力値を求めるプログラムのことである。認識プログラムFERTは関数プログラムの形で前章で記述されたが，このFERTの中に，SORTが関数プログラムの形で使われている。

プログラムSORT（γ, η）については

γ=〔j，k，l，…〕，j，k，l，… $\in$ J

$\eta \in \Phi$

として，

(a) γとSORT(γ, η)とは集合としては同一

(b) $\forall j_k \in \gamma$, SM (η, ω_{jk}) $\leqq$ SM (η, ω_{jk+1}), k＝1，2，3，…であれば，

SORT（γ, η)＝〔j_1，j_2，j_3，…〕

が成り立つ。このSORTの関数プログラムを本章で示そう。

まず，リスト

MP (η)＝〔〔1，SM (η, ω_1)〕，〔2，SM (η, ω_2)〕，…〕，ここに

J＝〔1，2，…〕

を導入する。カテゴリ番号j $\in$ Jと，パターンηがカテゴリ$\mathfrak{C}_j$に帰属している確率（帰属確率，membership probability) SM (η, ω_j)とから成る対リスト〔j, SM (η, ω_j)〕を要素とするリストがMP（η）である。

まず，関数プログラム

（i）SORT 1 (γ, MP (η))

＝〔〔k_1, SM (η, ω_{k1})〕，〔k_2, SM (η, ω_{k2})〕，…〕，ここに

γ＝〔k_1，k_2，…〕

を作る。これはMP (η)からj $\in$ γなる対リスト〔j, SM (η, ω_j)〕を集めて得られるリストである。次に

（ii）SORT 2 (SORT 1 (γ, MP（η))

＝〔〔η_1, SM (η, ω_{n1})〕，，〔η_2, SM (η, ω_{n2})〕，…〕，ここに，

n_1，n_2，… $\in$ γで，

SM (η, ω_{nk}) $\leqq$ SM (η, ω_{nk+1}),

k＝1，2，…

なる関数プログラムは，リストSORT 1 (γ, MP（η)）の頂上レベルの要素である〔j, SM (η, ω_j)〕をSM (η, ω_j）の小さい順に並べて得られるリストである。最後に，リストSORT 2 (SORT 1 (γ, MP (η)) の各要素である対リスト〔n_k, SM (η, ω_{nk})〕から，すべてのn_kを集めて得られるリストがSORT（γ, η）であるから，

（iii）SORT（γ, η）

＝SORT 3 (SORT 2 (SORT 1 (γ, MP (η)))) とおける関数プログラムSORT 3が存在する。

SORT 1，SORT 2，SORT 3，は再帰的関数プログラムとして，次のI，II，IIIに示される。

（I）K＝MP（η）とおくと，

```
SORT 1
=λγ. λK.
if null (γ) then nothing
else
   cons (assoc (head (γ), K),
         SORT 1 (tail (γ), K)
        )
fi.
```

（II）K＝SORT 1 (γ, MP (η)）とおく．

関数プログラムminはリストKの要素

〔j, SM (η, ωj)〕，j $\in$ γ

の中の最小のSM（η, ωj）をもつ対リスト〔j, SM (η, ωj)〕を値にするもので，

```
min
=λK.
if null (tail (K)) then head (K)
else
  if less (head (tail (head (K))),
```

head (tail (min (tail (K)))))
then head (K)
else min (tail (K))
fi
fi

と与えられる。また，reminは，リストKの要素〔j, SM (η, ω_j)〕, j $\in$ γ の中の最小の SM (η,ω_j) をもつ対リスト〔j, SM (η, ω_j)〕を一つ取り除いて得られるリストを値にするものであり，

remin
=λK.
if null (tail (K)) then nothing
else
if less (head (tail (head (K))), head (tail
(min (tail (K)))))
then tail (K)
else
cons (head (K), remin (tail (K)))
fi
fi

と与えられる。そうすれば，SORT 2 は

SORT 2
=λK.
if null (tail (K)) then K
else
cons (min (K), SORT 2 (remin (K)))
fi

と与えられる。

(III) M=SORT 2 (SORT 1 (γ, MP (η)))

とおくと，SORT 3 は

SORT 3
=λM.
if null (M) then nothing
else
cons (head (head (M)), SORT 3 (tail
(M)))
fi

と与えられる。

10. むすび

不動点形構造受精認識プログラム FERT を関数形言語としてのλ言語を用いて，リスト論的形式体系における表現に直したが，これを更に具体的に実際のリスト処理言語 LISP に書き直すことは容易である。現在その全容を発表しつつあるパターン認識の数学的理論においては，対リスト〔Ψ, Γ〕† を今一つの対リスト〔Ψ', Γ'〕に書き換えるのが認識プログラムの主要な機能ということになっているが，このような各種の認識プログラムをリスト論的形式体系内で表現可能なことは本論文の説述内容から容易に推察できよう。

（文教大学情報学部教授）

文　献

(1)鈴木昇一：パターン認識の数学的理論，
第I部（考え方，PRL84－6，pp. 1－10，1984－05）
第II部（認識抽象と公理系・定理系，PRL84－30，pp. 65－74，1984－07）
第III部（認識抽象と不動点諸定理，PRL84－38，pp. 65－73，1984－09）
第IV部（パターンの素領域，PRL85－27，pp. 1－10，1985－09）
第V部（認識停止と認識同値，PRU86－8，pp. 65－74，1986－05）
第VI部（類似度関数の三構成法，PRU86－35，pp. 51－60，1986－07）
第VII部（類似度関数の実現と解析，PRU86－69，pp. 1－8，1986－12）
第VIII部（大分類関数の自己組織化，PRU87－1，pp. 1－8，1987－05）

†対リスト〔Ψ, Γ〕内の Ψ は認識の途中におけるそれ迄に生成されたパターンのリストであり，また，Γ は Ψ 内の対応する要素としてのパターンが帰属する可能性のあるカテゴリの番号のリストを要素とするリストである。たとえば，head（ψ）なるパターンが帰属する可能性のあるカテゴリの番号のリストがhead（Γ）である。

第IX部（帰属関係あいまい度と認識情報量，PRU87－28，pp. 1－10，1987－07）
電子情報通信学会技術研究報告〔パターン認識と学習，パターン認識・理解〕
(2)鈴木昇一：収縮映像に関する一考察，情報研究（文教大学情報学部），Vol. 6，pp. 19－30（1985－12）
(3)鈴木昇一：認識工学（上），柏書房，1975－02

（認識プログラムFERTのリスト論的形式体系における表現・了）

情報研究第7号に掲載された「連想形記憶器MEMOTRONと日本語母音系列の再生に関する計算機シミュレーション」における誤りを下記正誤表によって訂正いたします。

正　誤　表

場　所	誤	正
17頁左上から6行目	$X_{l1}(\varphi)$	$X_l(\varphi)$
21頁左下から14行目	周期P	周期p
25頁左上から12行目	システム階数Nは	システム階数（＝3）は
25頁左上から14行目	約18周期で	約18（＝6N）周期で
28頁右下から15行目	$\forall\ \varphi\varepsilon\ \mathfrak{D}$	$\forall\ \varphi\in\mathfrak{D}$
28頁右下から10行目	$\varepsilon\ \mathfrak{D}$	$\in\mathfrak{D}$
29頁左上から8行目	$Uj\in J$	$\cup_{j\in J}$
29頁右上から4,5行目	$\parallel\theta_l(H)\langle\varphi\parallel\varphi-\xi\parallel\xi\parallel^{-1}\parallel^2$ (A4.7)	$\parallel\theta_l(H)\langle\varphi\parallel\varphi\parallel^{-1}-\xi\parallel\xi\parallel^{-1}\rangle\parallel^2$ (A4.7)
29頁右上から9行目	$\vec{e}=\{e_1\mid l\in L\}$	$\vec{e}=\{e_l\mid l\in L\}$
29頁右上から12行目	$\mathfrak{F}_1(\phi)$	$\mathfrak{F}_1(\varphi)$