解　説

アレイ文法†

——その理論と応用——

中村　昭††　会沢邦夫††

1. はじめに

コンピュータ・グラフィックスの画期的発展にともなって，画像処理の基礎理論として図形の生成と認識，空間のディジタル構造などの研究が最近注目されている．特に，**画像の処理と認識 (pictorial pattern recognition)** の研究は情報処理技術の重要な部分を占めるようになってきている（たとえば文献 33), 35), 46) など）．

一般に，形あるものを**ディジタル**計算機で処理するためには離散的な方法によって表現しなくてはならない．この離散的パターンを**ディジタル画像 (digital picture)** という．普通それは整数座標をもつ平面上の点に離散的な値を対応させることによって表される．このようなディジタル画像の処理および認識を統計的決定理論に基づく従来のパターン認識の理論とは異なった**文章論 (syntactic)** 的な立場から論ずる**文章論的パターン認識 (syntactic pattern recognition)** の理論的背景が本稿でとりあげる**アレイ文法 (array grammar)** である．

さて，アレイ文法はある種の**形式文法 (formal grammar)** にほかならない．形式文法としてはチョムスキーによるもの[9]が有名である．チョムスキー型の文法では，**語彙 (alphabet)** に含まれる終端記号の**列 (string)** のなかで文法によって生成されるものは**語 (word)** と呼ばれ，語全体の集合はその文法の**言語 (language)** といわれる．このような形式文法を対象とする理論が**形式言語理論 (formal language theory)** であり，その文法に対応する処理機械に関する理論が**オートマトン理論 (automata theory)** であることはよく知られている．アレイ文法は，このチョムスキー型文法の2次元への拡張と考えてよい．したがって，アレイ文法によって生成された2次元パターンの集合を**2次元言語 (picture language)** という．また，それらを受理・認識する機械が**アレイアクセプタ (array acceptor)** である．

さきに述べた文章論的パターン認識においては，画像処理のある種の問題は生成文法を導入することによって言語の構文解析の議論に還元されるという立場をとっている．そのような立場をとった最初の論文は Kirsch[22] とされている．この論文では図形の認識に形式文法が利用できることが示されている．その他にも，たとえば Fu[13], Miller and Shaw[27], Rosenfeld[36] らがやはり同様の立場から画像の処理および認識を扱った文献である．

本稿では，アレイ文法とその受理機に関する基本的な事柄を整理するとともに，アレイ文法の拡張として並列文法およびグラフ文法を紹介する．また，パターン認識を中心とした応用面での最近のトピックスをとりあげる．

2. アレイ文法と2次元言語

チョムスキー型の文法を2次元的な構造の上に拡張しようとする試みは数多くなされてきた．そのすべてを紹介することはここでは困難である．そこで，本稿では連結な2次元配列を言語として生成する文法を主にとりあげる．この文法は（その生成する言語ゆえに）連結アレイ文法，または（生成規則に付加された制限から）等形 (isometric) アレイ文法と呼ばれるもので，2次元文法としてはもっとも代表的なものである．

本章では，まず2次元配列上での配列の書き換えを紹介し (2.1)，これを利用して等形アレイ文法を定義する (2.2)．最後に，文法の型と生成能力の階層構造を整理する (2.3)．

本章および次章の内容は特に言及しないかぎり

† Array Grammars—Theory and Application by Akira NAKAMURA and Kunio AIZAWA (Department of Applied Mathematics, Faculty of Engineering, Hiroshima University).

†† 広島大学工学部応用数学教室

Rosenfeld によるこの分野での先駆的教科書[39]に従っている.

2.1 配列の書き換え

アレイ文法をチョムスキー型文法の2次元への拡張とするならば，それは連結した有限の大きさをもつ2次元配列を同様な配列で置き換えて，2次元配列の集合を言語として生成するものと考えるのがもっとも自然であろう．ここで2次元配列とは一般に整数の対から記号の有限集合への写像と定義されるものである．整数座標をもつ平面上の点(**格子点**)の位置に記号が書かれていると想像していただいてもよい．このような2次元配列上での連結は隣接関係によって定義される．すなわち，ある格子点はその上下左右の4つの格子点と**隣接**しており，与えられた格子点の集合 S に属する二つの格子点はその二つの格子点を隣接関係で結ぶような一連の S に属する格子点が存在すれば**連結**(**connected**) であると定義される．このような連結関係は同値関係であり，これによって定義される同値類は**連結成分** (**connected component**) と呼ばれる．

さて，配列を配列で置き換えるような生成規則は一般に困った問題を引き起こす可能性を秘めている．なぜならば置き換えられた配列が元の配列と大きさや形が異なっていると，局所での書き換えが配列全体に影響を及ぼす場合があるからである．**図-1** はそのような状況の一例である．置き換えられた配列によって押し退けられた行や列に属している記号は，置き換える以前とは別の格子点上へとずらされることによって他の記号との隣接関係を完全に壊されてしまっている．このような問題を防ぐために，アレイ文法では“生成規則の両辺は幾何学的に同形でなければならない”という**等形性** (**isometric**) の制限が付加されている．等形性の制限の下で配列を生成するために，空白を表す特別な記号 # (**空白記号—blank symbol**) を新たに導入する．空白記号を他の記号に置き換えることによって配列を大きくすることができ，ある記号を空白記号に書き換えることによって配列を削ることができる．この書き換えの例を**図-2** に示す．本節の最初で述べたようにアレイ文法の言語として連結した有限の大きさをもつ2次元配列の集合を考える場合，文法によって生成された配列における非空白記号の連結性が重要となってくる．なぜならば配列のある非空白記号を空白記号に書き換えた場合，配列上の非空白記号全体が非連結になってしまう場合があるからである．連結性を保証する生成規則の条件は次のように与えられる．

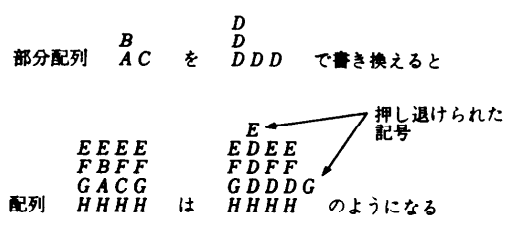


図-1 配列の書き換えにともなう“ずれ”

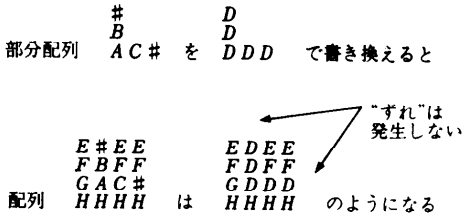


図-2 等積性制限下での配列の書き換え

条件(1)：もし生成規則の左辺にある配列の非空白記号がその配列の周辺に接していないならば，右辺にある配列の非空白記号は連結（かつ空でない）でなければならない

条件(2)：そうでない場合

条件(2a)：生成規則右辺に含まれるすべての非空白記号の連結成分は，左辺のある非空白記号の連結成分と配列の周辺との共通部分を含んでいなくてはならない．

条件(2b)：逆に，そのようなすべての共通部分は右辺のある非空白記号の連結成分に含まれていなければならない．

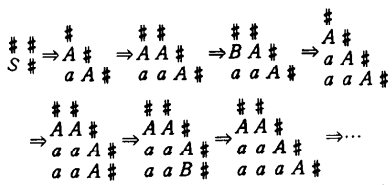
この条件(1)，(2)を満足する書き換え規則が配列に適用された場合，非空白記号を非連結にしたり完全に削除したりすることはない．

2.2 等形アレイ文法

前節で示した配列の書き換えを用いる**等形アレイ文法** (**isometric array grammar—IAG**) は5項 $\langle \Sigma, \Delta, P, S, \# \rangle$ で定義され，$\Sigma - \Delta$ (ただし $\Sigma \supset \Delta$) は変数の有限集合，Δ は終端記号の有限集合，S は $\Sigma - \Delta$ の要素で**公理** (**axiom**)，# は Σ に含まれない記号で空白記号，P は有限の大きさをもつ配列の対 (α, β) の有限集合で，P のおのおのの要素（**書き換え規則—replacement rule**）は次の条件を満足する：

(a) α と β は幾何学的に同形である

(b) α は空白記号のみからなっていてはならない



(空白記号は必要なものだけ表示してある)

図-3 アレイ文法の生成例

(c) β は前節の条件(1)と(2)を満足する.

"等形性"の制限により,アレイ文法における**生成**は配列を配列で置き換えることを除いて1次元の場合と同様に定義できる.すなわち,アレイ文法 G の生成する**言語** $L(G)$ は,空白記号だけからなる無限配列中にただ一つ公理 S が埋め込まれている**初期配列**から出発して,G によって生成される,Δ の記号だけからなる配列の集合と定義される.アレイ文法とその生成の例を示しておく.

[例-1] アレイ文法 $G=\langle\{A,B,S,a\},\{a\},P,S,\#\rangle$ ここで

$$P=\left\{\left(\begin{matrix}\# & A\\ S & \#\end{matrix}\right),\left(\begin{matrix}\# & A\\ A & \#\end{matrix}\right),\left(\begin{matrix}\# & A\\ a & A\end{matrix}\right),(A,B),\right.$$
$$\left.(B\#,aA),\left(\begin{matrix}\# & A\\ B & a\end{matrix}\right),(A,a)\right\}$$

このアレイ文法による生成は**図-3** のようになる.

2.3 アレイ文法のパターン生成能力

前節で示したアレイ文法の生成規則に制限を付け加えることによってチョムスキー型文法と同様にいくつかの型のアレイ文法を定義することができる.それらの制限された文法に対する標準形と,パターン生成能力の階層構造を紹介する.

2.3.1 アレイ文法の型

生成規則になんの制限も加えられない文法は単にアレイ文法と呼ばれる.生成規則の適用によって空白記号が新たに書かれることがない(すなわち,生成規則の右辺に空白記号があるならば左辺の対応する位置にも必ず空白記号が存在する)文法は**単調アレイ文法 (monotonic array grammar—MAG)** と呼ばれる.生成規則の左辺が一つの変数と空白記号(複数でもよい,なくてもよい)だけからなる文法は**文脈自由アレイ文法 (context-free array grammar—CFAG)** と呼ばれる.生成規則が以下に示すような形のものだけからなる文法は**正規アレイ文法 (regular array grammar—RAG)** と呼ばれる.

$$(A\#,aB),(\#A,Ba),\left(\begin{matrix}\# & B\\ A & a\end{matrix}\right),\left(\begin{matrix}A & a\\ \# & B\end{matrix}\right),(A,a)$$

ここで,A, B は変数,a は終端記号である.

2.3.2 標準形とパターン生成能力

たとえば文脈自由文法に対するチョムスキー標準形のように,アレイ文法に対しても標準形定理が示されている.

アレイ文法と単調アレイ文法に対する標準形には**2点標準形 (two-point normal form)**[37] と呼ばれるものがあり,それは生成規則の左右各辺に高々二つの記号が存在するというものである.この結果は,純粋な文脈としての空白記号(生成規則の左辺にある空白記号で他の記号に書き換えられないもの)は生成能力の点からは無意味であることを意味している.文脈自由アレイ文法に対する標準形[6]も以下に示すように,アレイ文法とほぼ同様な形になる.

$$(A\#,BC),(\#A,CB),\left(\begin{matrix}\# & C\\ A & B\end{matrix}\right),\left(\begin{matrix}A & B\\ \# & C\end{matrix}\right),(A,a)$$

ここで,A, B, C は変数,a は終端記号である.

ある型 X に属するアレイ文法の言語全体の集合を $\mathcal{L}(X)$ と記述するならば,アレイ文法のパターン生成能力には次のような階層構造があることが知られている[6].

$$\mathcal{L}(\mathrm{IAG})\supset\mathcal{L}(\mathrm{MAG})\supset\mathcal{L}(\mathrm{CFAG})\supset\mathcal{L}(\mathrm{RAG})$$

具体的なパターン生成能力については,人間が直感的に考えうる離散的パターンの集合の大部分は単調アレイ文法によって生成可能である.驚くべきことに,傾いていない正方形全体の集合を言語として生成する正規アレイ文法!が存在する[53].このような生成は一般に文脈を必要とするので,等形性のために導入された空白記号によってなんらかの文脈がもたらされていると考えられている.

3. アレイアクセプタ

1次元文法と同様にアレイ文法に対してもその言語を受理する機械という見地から議論を展開することができる.ただしアレイ文法の場合,このような受理機械の話題は言語の受理といった伝統的な問題にとどまらず,図形のさまざまな性質(たとえば,連結性とか凹凸性などの幾何学的性質)の認識という側面ももっている.

本章では,まず2次元配列上で逐次形 (sequential) アレイアクセプタの一種である2次元チューリング・アクセプタを定義し,この受理機械と前章で導入したアレイ文法との関連をまとめる (3.1).次に,図形の認識という観点から並列に処理を行う2次元セラー・アクセプタについて解説する (3.2).

3.1 逐次型アレイアクセプタ

3.1.1 2次元チューリング・アクセプタ

おおざっぱに言えば，2次元配列上の**受理機械**(**acceptor**)は有限個の内部状態をもち，配列上の記号を読み書きでき，上下左右の4方向に動くことが可能な，2次元に拡張されたチューリング機械である．

各単位時間ごとにこの機械は以下の動作を実行する．

（a） 現在位置にある配列上の記号を読み取り，それを消去し，新しい記号に書き換える．

（b） 新しい内部状態へ変化する．

（c） 隣接する位置に移動する．まったく移動しなくてもよい．

この(a)～(c)の動作は現在の内部状態，現在位置の記号，直前に移動した方向に依存する．したがって，この受理機械の動作は1次元の場合と同じく，(状態，記号，移動方向)から(状態，記号，移動方向)への写像である**遷移関数**(**transition function**)の形で表現することができる．したがって，2次元配列上の受理機械は5項$\langle Q, V, \delta, q_0, Q_A\rangle$である．ここで，$Q$は内部状態の有限集合，$V$は2次元配列上で使われる記号の有限集合，$\delta: Q\times V\times \Delta\to 2^{Q\times V\times \Delta}$は**遷移関数**（ここで$\Delta=\{R, L, U, D\}$は受理機械の移動方向の集合），$q_0$は受理機械の**初期状態**，$Q_A$は$Q$の部分集合で**受理状態**の集合である．

2次元配列上の受理機械の**計算状況**(**configuration**)は$(q, d, (i, j), \tau)$で表される．ここで，qは現在の状態，dは受理機械が直前に移動した方向，(i, j)は受理機械の現在位置の座標，τは座標(i, j)の記号を表す．

$C_1=(q, d, (i, j), \tau)$とC_2がともに計算状況であるとすると，以下の(1)～(4)の場合，C_2はC_1から**1動作で来た**といわれる．

(1) $(q', \tau', R)\in\delta(q, \tau, d)$で，
かつ $C_2=(q', R, (i+1, j), \tau')$であるとき

(2) $(q', \tau', L)\in\delta(q, \tau, d)$で，
かつ $C_2=(q', L, (i-1, j), \tau')$であるとき

(3) $(q', \tau', U)\in\delta(q, \tau, d)$で，
かつ $C_2=(q', U, (i, j+1), \tau')$であるとき

(4) $(q', \tau', D)\in\delta(q, \tau, d)$で，
かつ $C_2=(q', D, (i, j-1), \tau')$であるとき

2次元配列上の受理機械Tによって受理される配列の集合$L(T)$は，初期計算状況$(q_0, d, (i, j), \tau)$から出発して上記の動作を繰り返すことによってある受理状態qを含む計算状況$(q, d', (i', j'), \tau)$に到達できるような配列の集合である．

このように定義される2次元配列上の受理機械を**2次元チューリング・アクセプタ**(**Turing array acceptor—TAA**)と呼ぶ．

3.1.2 制限された受理機械と階層構造

空白記号を書き込まず，もし空白記号上に移動したならば次の単位時間に空白記号に移動する直前の位置に戻るTAAは**2次元テープ有界アクセプタ**(**tape-bounded array acceptor—TBAA**)と呼ばれる．もし，配列上の記号をまったく書き換えないならば，それは**2次元有限状態アクセプタ**(**finite-state array acceptor—FSAA**)と呼ばれる．

これらの受理機械の受理能力は次の包含関係で表すことができる．

$$\mathcal{F}(\mathrm{TAA})\supset\mathcal{F}(\mathrm{TBAA})\supset\mathcal{F}(\mathrm{FSAA})$$

3.1.3 全面走査

逐次的な処理を行う受理機械の場合，入力配列上にある任意の形をした非空白記号の集合（もちろん連結状態にある）全体をくまなく走査できる（しかも，走査が終了したことが分かる）ことが，その受理機械の図形認識能力に大きな影響を及ぼす．2次元チューリング・アクセプタと2次元テープ有界アクセプタは，この**全面走査**を行うことができる．ところが，2次元有限状態アクセプタはこのような全面走査を行うことができない．確かにFSAAはある点から隣の点へと次々と移動していくことによって，やがてすべての非空白記号を走査することは可能であるが，それがいつ完了したのか，それともまだなのかの判断を行うことができないのである．

このような全面走査の問題は入力テープを端から端まで読めばよい1次元の場合とはまったく異なった状況を生みだしている．すなわち，FSAAに対応する“自然な”アレイ文法の型が存在しないのである．このアレイ文法との関連は次のようにまとめることができる．

3.1.4 アレイ文法との同値性

[定理-1] アレイ文法によって生成される2次元言語は2次元チューリング・アクセプタによって受理される集合と等しい．

[定理-2] 単調アレイ文法によって生成される2次元言語は2次元テープ有界アクセプタによって受理される集合と等しい．

3.2 並列アクセプタ

前節の2次元チューリング・アクセプタは自ら入力配列上を移動しながら受理動作を行うものであった．この受理動作を自らは移動しないアクセプタの2次元配列によって並列的に行うのが**セラー・アクセプタ**(**cellular acceptor**)である．このセラー・アクセプタの一つ一つの構成要素はセルと呼ばれ，上下左右の隣接セルの状態と自身の状態それに自分が見ている入力配列の記号に依存して新しい状態へと遷移する．この動作はすべてのセルで並列に行われる．

このセラー・アクセプタをピラミッド状に積み重ねたのが**ピラミッド・アクセプタ**(**pyramid cellular acceptor**)である．一番底辺にあるセルは $2^n \times 2^n$ の大きさをもつ正方形の配列を構成している．その一つ上の層は $2^{n-1} \times 2^{n-1}$ の大きさである．このようにして，$n+1$ 番目の層はただ一つのセルからなっている(**図-4** 参照)．それぞれのセルは同じ層に4つ，下の層に4つ，上の層に一つの計9つの隣接セルをもつことになる．このピラミッド・アクセプタは一般にセラー・アクセプタと同値であることが分かっている．

これらの並列アクセプタは処理速度の点で逐次形アレイアクセプタに数段勝るという性質があり，古くから研究がなされている(たとえば，文献 5))．また個個のプロセッサのコストの低廉化にともなって，最近では並列アクセプタに関する研究は膨大なものとなっている．これらを網羅することは本稿の主旨から少しはずれるので，次章で図形処理との関連において多少触れるにとどめることにする．

4. アレイ文法の拡張と応用

これまで述べてきたアレイ文法は記号が正方形状の2次元配列に配置されたものを言語として生成した．このような配列構造は図形の生成および認識といった立場からは直感的に理解しやすく便利である．しかし，たとえばファジィ図形では6角形状の2次元配列が便利であるし，グラフのように2次元配列でない構造を扱う文法がより有益となる場合もある．また，文法は一般に一つの生成規則を1カ所に適用するが，これもたとえば生物成長記述の立場からすれば不都合である．このようなさまざまな分野からの要請によってアレイ文法には数多くの拡張された文法が存在している．そのすべてをここで取り扱うことはできないが，いくつかの重要な文法をあげ，それに関する応用の具体例について言及する．

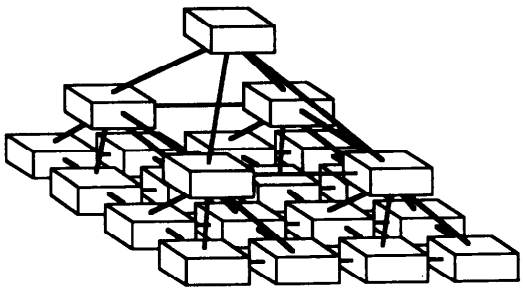


図-4 ピラミッド・アクセプタの概念図

4.1 アレイ文法の拡張

この節ではまず，アレイ文法を正方形以外の配列上で展開したものを紹介し，次にグラフを言語として生成するグラフ文法について述べる．最後に生成規則の並列的な適用を行う文法について解説する．

4.1.1 正方形以外の配列

アレイ文法が定義される配列は通常正方形配列である．しかし，この配列上でのディジタル図形の認識を考えた場合，隣接関係がやや複雑なことに気づく．たとえば，**図-5** に示すように0と1によってラベル付けられている二つのディジタル図形を考える．この両方のディジタル図形に対して上下左右4つの隣接関係しか認めないとすると，ラベル0のディジタル図形は二つの連結成分からなっていることになるが，それを二つに分離しているはずのラベル1のディジタル図形も連結ではないという奇妙な現象が発生する．逆に，双方に対して対角線方向を含む8つの隣接関係を認めると，今度は両方とも連結していることになってしまう．それでも2値のディジタル図形の場合は，片方に4つの隣接関係，他方に8つの隣接関係を定義しておけば問題は解決する．しかし，多値のディジタル図形やファジィ図形[38]に対しては適当な解決策がない．このような問題をうまく解決する手段として，任意の点に対する隣接関係が一様な**6角形配列**(**hexagonal array**)上での図形が考えられている[45]．

またこれとは別に，6角形配列が直方体に対するある意味での2次元表現になっていることからも6角形配列上のアレイ文法が注目されている[48]．

```
# # # # # # # #
# 0 0 0 1 1 # #
# # 0 0 1 1 1 #
# # 0 0 1 1 # #
# 1 1 1 0 0 # #
# 1 1 1 0 0 0 #
# 1 1 # 0 0 # #
# # # # # # # #
```

図-5 2値ディジタル図形の連結性

4.1.2 グラフ構造

グラフ文法 (graph grammar) に関する最初の論文は，Pfaltz and Rosenfeld の**ウェッブ文法** (web grammar)[34] とされている．この論文で彼らは手書き文字の認識という画像処理のための手法・理論としてウェッブ文法を提案した．

グラフ文法は部分グラフ（文法によってはこれが頂点であったり辺であったりもする）をグラフで置き換える規則を生成規則としてもっている．しかし，もっとも特徴的なことはその生成規則によって**置き換えられたグラフ (daughter graph)** を**もとのグラフ (host graph)** へ埋め込むための規則を別個にもっていることである．このような**埋め込み規則 (embedding rule)** は定まった構造のないグラフの生成を扱うためには不可欠のものである．また，グラフ文法の生成能力もこの埋め込み規則の影響を強く受けることもあって，さまざまな埋め込み規則が提案されている[19),21),32)]．

ここでは，グラフ文法の例として比較的分かりやすい生成規則と埋め込み規則をもち，かつ応用面でも注目されている **NLC グラフ文法**[20] **(node label controlled graph grammar)** を紹介しよう．

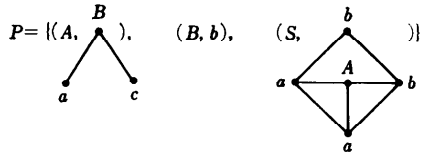
いま，ループも複数辺もない．頂点にラベルのついた無向グラフを考える．すなわち Σ をラベルの有限集合とし，ノードの有限集合 V と，V の要素の対からなる辺の集合 E および写像 $\phi: V \to \Sigma$ からなる4項 $\langle V, N, \Sigma, \phi \rangle$ を Σ 上のグラフという．特に V が空集合のとき，空グラフといい λ で表す．また，Σ 上のグラフのすべての集合を F_Σ で表す．

NLC グラフ文法は5項 $\langle \Sigma, \Delta, P, \mathrm{conn}, S \rangle$ で表される．ここで，$\Sigma - \Delta$ は（ただし $\Sigma \supset \Delta$）変数の集合，Δ は終端記号の集合，S は $\Sigma - \Delta$ の要素で，公理である．P は生成規則の有限集合で，その要素は (α, β) の形であり，α は $\Sigma - \Delta$ の要素，β は F_Σ の要素である．すなわち，NLC グラフ文法はラベル付けされた頂点をグラフで置き換えるような生成規則をもっている．そして conn が埋め込み規則であり，これは Σ から Σ の部分集合全体の集合への写像である．P に属する規則で置き換えられたグラフのそれぞれの頂点 $v \in V$ は，置き換えられる前の頂点が辺で結ばれていた頂点のうち $\mathrm{conn}(\phi(v))$ に含まれるラベルをもつすべての頂点と辺で結ばれる．

NLC グラフ文法 G の生成する言語 $L(G)$ は，S でラベル付けされたただ一つの頂点からなる**初期グラフ**から出発して，P に属す生成規則で頂点を置き換え，それを conn で埋め込むという操作を繰り返して得られる Δ 上のグラフの集合として定義される．

NLC グラフ文法の例を示しておこう．

[例-2] NLC グラフ文法 $G = \langle \{A, B, a, b, c\}, \{a, b, c\}, P, \mathrm{conn}, S \rangle$，ここで



$\mathrm{conn}(a) = \{a\}$

$\mathrm{conn}(b) = \phi$

$\mathrm{conn}(c) = \{a, b\}$

この NLC 文法による生成は**図-6** のようになる．

4.1.3 並列生成

形式文法の世界に並列的な規則の適用を最初に導入したのは Lindenmayer[23] による **L システム (L-system)** である．L システムではすべての記号がなんらかの（同じである必要はない）生成規則によって同時に書き換えられる．また，形式文法では常識的であった変数と終端記号の区別もこのシステムにはない．L システムは元来，生物系の発展成長過程をアルゴリズム的に説明しようとする動機から出発したが，生物学的な立場に留まらずさまざまな観点からの数多くの研究がなされている．

Lindenmayer が最初に提案したモデルは1次元であったが，生物学的な興味もあって，比較的初期のころから高次元への拡張が盛んに行われてきた（たとえば文献 2)）．2次元配列へ拡張されたLシステムは**ア**

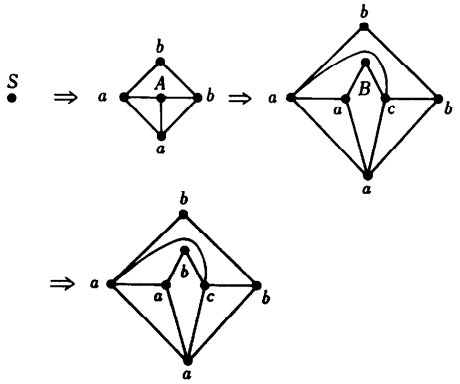


図-6 NLC グラフ文法の生成例

レイ L システム（**array L-system**）（たとえば文献50)）であるが，配列構造上で生成規則を並列に適用すると書き換えられた配列が重なりあう場合があるので，これを防ぐために生成規則のさまざまな制限が試みられている．

グラフ構造上に拡張されたLシステムは**グラフLシステム**（**graph L-system**）と呼ばれ，さかんに研究が行われている[4),10),11)]．実際，配列構造と比較して生成規則の並列的適用がよりすっきりした（制限の少ない）形で行えるという利点がある．

4.2 さまざまな応用分野

本稿で紹介してきたアレイ文法およびその拡張としてのグラフ文法，並列文法は応用面でも大きな広がりをみせている．1. でふれた画像処理の問題もその一つである．

この節では，その応用範囲を Nagl[31)] に従って大きく三つに分類し，それぞれに対して基本的な文献をあげる．また，特に本稿で扱った文法やアクセプタが用いられている最近の文献もいくつかあげることにする．

まず，第1の応用分野では文法は**明確化**（**specification**）のために用いられる．ここでの“明確化”とはあるものを正確に表現するという意味である．ここで対象となるものは非常に複雑なものであり，新たな発見を得るためにはまずそれを明確にしておく必要があるのである．第2の応用分野では文法は**記述**（**description**）のために用いられる．明確化との違いは，この場合すでに明確になっているものが対象として存在していることにある．第3の分野では認識のために文法が用いられる．すなわち，ある性質が存在するか否かの**決定**（**decision**）のために用いられるのである．

明確化の分野ではグラフ文法がよく用いられている[10),11)]．グラフ文法応用の先駆けとなった**2次元プログラミング**[7),9)]もこの分野の一部である．**ソフトウェア開発環境**[30)]，**VLSI エディタ**，**グラフィック・エディタ**，**音楽システム**[52)]などにも応用されている．以上の応用に共通なのは，対象となるものの論理的構造が言語（特にグラフ）によって表現されていることである．明確化という観点のみに関して同じ分類に属するものとしては，**コンパイラの最適化**，**プログラム言語の意味論**，**データベース構造**（**data base scheme**）の明確化があげられる．

記述の分野では，たとえばLシステムによる比較的簡単な植物に対する**成長記述**（**growth description**)[25),47)] をあげることができる．このような分野では，すでに定義（形式的非形式的を問わず）が定まっているものに別の記述を与えるために文法が用いられている．たとえば，すべての平面グラフからなる集合を言語として生成するグラフ文法[51)]を与えたならば，それは平面グラフに対する別の定義を与えたことになるであろう．成長記述の分野では現在，ある種の高次元（ないし3次元）並列文法として**地図生成システム**（または**セル分割システム**—**map generating system or cell division system**)[8),24),26)] が注目されている．この並列文法は国（または細胞）が新たな国境線（細胞壁）で分割されるような生成を行うものであり，Lシステムの拡張形の一種である．

パターンの認識のために文法を使う場合，常に強力な構文解析アルゴリズムが必要とされる．したがって，この分野で用いられる文法の能力はむしろ制限される傾向にある．たとえば，最近では本稿で扱ったNLC グラフ文法が比較的シンプルなグラフ文法として利用されているので簡単に紹介しよう．まず，風景が**図-7** のようなグラフ表現で表されているとする．このグラフは辺に方向とラベルを付与して NLC グラフ文法を拡張したものであり，次のような表の形式で表される．

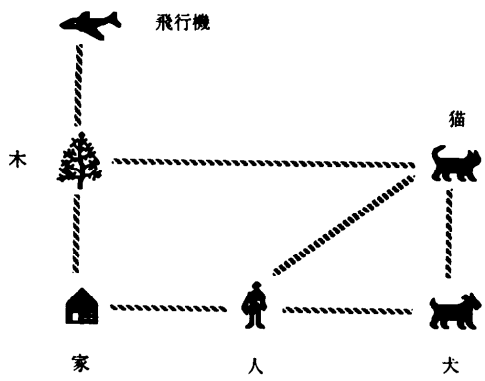
木1	家2	人3	飛4	猫5	犬6
2	2	2	0	1	0
ps	*ps*	*rs*	—	*u*	—
23	45	56	—	6	—

ただし，ここで p は上，u は下，s は右，r は右上，t は右下を表す．このグラフを構文解析することによって風景の理解を行う．構文解析のある時点で，**図-8** の上に示すようなグラフが得られたとする．このグラフは以下のように表される．

木1	家2	動物3	飛4
2	2	0	0
ps	*tp*	—	—
23	34	—	—

この二つの表を比較することによって，図-8 の下に示す生成規則が適用可能であることが分かる．このように，NLC グラフ文法の生成を利用した図形認識アルゴリズムが提案されている[12)]．

また，一般に構文解析は多くの計算量を必要とするので，並列に処理を行う計算モデルを導入することによって高速化への努力がなされている．この目的のために 3. で扱ったピラミッド・アクセプタを利用



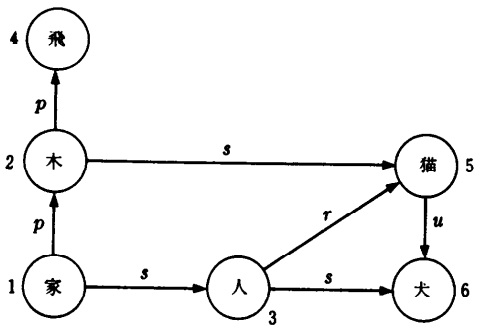
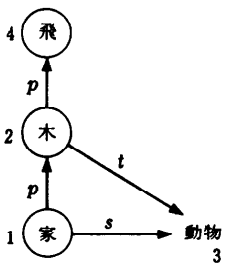


図-7 風景のグラフ表現



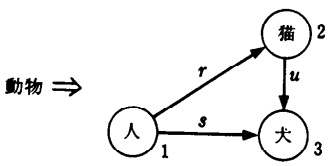


図-8 グラフ文法における生成と風景の認識

した図形解析のための多くの計算モデルが提案されている[1),40),43)]．文章論的パターン認識の利点のひとつはあるパターンが対象に属しているか否かが判明するだけでなく，もし属しているならばそのパターンの記述も同時に得られることである．文章論的パターン認識に関しては Fu[14)~18)] による一連の文献および Nagl[28),29)] による文献集を参照されたい．また，この分野における新しい結果については毎年 Rosenfeld[41),42),44)] によってまとめられている文献集が大変参考になる．

5. おわりに

ディジタル画像処理の基礎理論としてのアレイ文法を中心としたピクチャ理論について解説した．

最初に述べたようにこの理論は構文解析という動機のもとで1次元言語理論の2次元ピクチャへの拡張という形で発展してきた．しかしながら現在，それはすでに2次元ピクチャにとどまらずさまざまな拡張が試みられ広範な応用に支えられて進展している．たとえば，医療用画像処理として3次元ピクチャと3次元ディジタル幾何学の確立もそのひとつであり，生物生成をモデルとしてグラフ文法への拡張も特に並列生成という見地から注目されている．さらにディジタル画像からアナログ画像（たとえばファジィ画像）[38),45)] への展開もそのひとつである．

このような状況のなかで，この分野を推進しようとするいくつかの国際的動向がある．たとえば1978年，1982年，1986年と開催された“グラフ文法とその応用”に関する国際ワークショップにみられる数多くの論文[4),10),11)] は直接アレイ文法に関係している．また Siromoney[49)] はアレイ文法とLシステムに関する理論面の概説をはじめいくつかの関係論文を含んでいる．さらにまた，Wang による IJPRAI の特集“アレイ文法・パターン・認識機械”[54)] はパターン認識・画像処理の問題を AI の立場から論ずる論文集である．

技術の開発にはそれを支える理論が必要であり，アレイ文法およびその拡張された理論が，ディジタル画像処理などの幅広い応用分野における基礎理論として発展しつづけることを期待する．

参考文献

1) Cantoni, V. and Levialdi, S. (eds.): Pyramidal Systems for Computer Vision, Springer-Verlag, Berlin (1986).
2) Carlyle, J. W., Greibach, S. A. and Paz, A.: A Two-Dimensional Generating System Modeling Growth by Binary Cell Division, Proc. 15th SWAT Conf., pp. 1-12 (1974).
3) Chomsky, N.: Three Models for the Description of Language, IEEE Trans. Inf. Theor., Vol. 2, pp. 113-124 (1956).
4) Claus, V., Ehrig, H. and Rozenberg, G. (eds.):

Graph Grammars and their Application to Computer Science and Biology, Lecture Notes in Comput. Sci., 73, Springer-Verlag, Berlin (1979).
5) Codd, E. F.: Cellular Automata, Academic Press, New York (1968).
6) Cook, C. R. and Wang, P. S. P.: A Chomsky Hierarchy of Isotonic Array Grammars and Languages, Comput. Gr. Image Process., Vol. 8, pp. 144-152 (1978).
7) Cristensen, C.: An Example of the Manipulation of Directed Graphs in the AMBIT/G Programming Language, in Klerer and Reinfeld (eds.), Interactive Systems for Applied Mathematics, Academic Press, New York (1968).
8) de Does, M. and Lindenmayer, A.: Algorithms for the Generation and Drawing of Maps Representing Cell Clones, Lecture Notes in Computer Science, 153, pp. 39-57 (1983).
9) Denert, E., Franck, R. and Streng, W.: PLAN2D—Toward a Two-Dimensional Programming Language, Lecture Notes in Computer Science, 26, pp. 202-213 (1975).
10) Ehrig, H., Nagl, M. and Rozenberg, G. (eds.): Graph-Grammars and their Application to Computer Science, Lecture Notes in Comput. Sci., 153, Springer-Verlag, Berlin (1983).
11) Ehrig, H., Nagl, M., Rozenberg, G. and Rosenberg, A. (eds.): Graph-Grammars and their Application to Computer Science, Lecture Notes in Comput. Sci., 291, Springer-Verlag, Berlin (1986).
12) Flasinski, M.: Parsing of edNLC-Graph Grammar for Scene Analysis, Pattern Recogn., Vol. 21, pp. 623-629 (1988).
13) Fu, K. S.: Syntactic Methods in Pattern Recognition, Academic Press, New York (1974).
14) Fu, K. S. (ed.): Applications of Syntactic Pattern Recognition, Springer-Verlag, Berlin (1976).
15) Fu, K. S.: Digital Pattern Recognition, Springer-Verlag, Berlin (1976).
16) Fu, K. S. (ed.): Syntactic Pattern Recognition, Application, Communication and Cybernetics, Lecture Notes in Computer Science, 14, Springer-Verlag, Berlin (1977).
17) Fu, K. S.: Recent Advances in Syntactic Pattern Recognition, Proc. 4th Int. Joint Conf. Pattern Recogn., pp. 13-18 (1978).
18) Fu, K. S.: Syntactic Pattern Recognition and Applications, Chapter 4—High Dimensional Pattern Grammars, Prentice-Hall (1982).
19) Grötsch, E. and Nagl, M.: Explicit versus Implicit Parallel Rewriting on Graphs, Lecture Notes in Computer Science, 73, pp. 237-254 (1979).
20) Janssens, D. and Rozenberg, G.: On the Structure of Node-Label Controlled Graph Languages, Inf. Sci., Vol. 20, pp. 191-216 (1980).
21) Janssens, D. and Rozenberg, G.: Graph Grammars with Neighborhood-Controlled Embedding, Theor. Comput. Sci., Vol. 21, pp. 55-74 (1982).
22) Kirsch, R. A.: Computer Interpretation of English Text and Picture Patterns, IEEE Trans. Comput., Vol. EC-13, pp. 363-376 (1964).
23) Lindenmayer, A.: Mathematical Models for Cellular Interactions in Development, Part I and II, J. Theor. Biology, Vol. 18, pp. 280-315 (1968).
24) Lindenmayer, A.: Models for Plant Tissue Development with Cell Division Orientation Regulated by Preprophase Bands of Microtubules, Differentiation Vol. 26, pp. 1-10 (1984).
25) Lindenmayer, A. and Rozenberg, G. (eds.): Automata, Languages, Development, North-Holland, Amsterdam (1976).
26) Lindenmayer, A. and Rozenberg, G.: Parallel Generation of Maps—Developmental Systems for Cell Layer, Lecture Notes in Computer Science, 73, pp. 301-316 (1979).
27) Miller, W. F. and Shaw, A. C.: Linguistic Methods in Picture Processing—A Survey, Proc. AFIPS Fall Joint Comput. Conf., pp. 279-290 (1968).
28) Nagl, M.: A Tutorial and Bibliographical Survey on Graph Grammars, Lecture Notes in Computer Science, 73, pp. 70-126 (1979).
29) Nagl, M.: Bibliography on Graph-Rewriting Systems (Graph Grammars), Lecture Notes in Computer Science, 153, pp. 415-448 (1983).
30) Nagl, M.: Graph Technology Applied to a Software Project, in Rozenberg, G. and Salomaa, A. (eds.), The Book of L, pp. 303-321, Springer-Verlag, Berlin (1986).
31) Nagl, M.: Set Theoretic Approaches to Graph Grammars, Lecture Notes in Computer Science, 291, pp. 41-54 (1987).
32) Nakamura, A. and Aizawa, K.: Relationships between Coordinate Grammars and Path Controlled Graph Grammars, Int. J. Pattern Recogn. Artif. Intell., Vol. 3, pp. 445-458 (1989).
33) Pavlidis, T.: Algorithms for Graphics and Image Processing, Computer Science Press, Rockville (1982).
34) Pfaltz, J. L. and Rosenfeld, A.: Web Grammars, Proc. Joint Conf. Artif. Intell., pp. 609-619 (1969).

35) Pratt, W. K.: Digital Image Processing, John Wiley Sons, New York (1978).
36) Rosenfeld, A.: Isotonic Grammars, Parallel Grammars and Picture Grammars, in Meltzer, B. and Michie, D. (eds.) Machine Intelligence 6, pp. 281-294, Univ. of Edinburgh Press, Edinburgh (1971).
37) Rosenfeld, A.: Array Grammar Normal Form, Inf. Control, Vol. 23, pp. 172-182 (1973).
38) Rosenfeld, A.: Fuzzy Digital Topology, Inf. Control, Vol. 40, pp. 76-89 (1979).
39) Rosenfeld, A.: Picture Languages, Academic Press, New York (1979).
40) Rosenfeld, A. (ed.): Multiresolution Image Processing and Analysis, Springer-Verlag, Berlin (1984).
41) Rosenfeld, A.: Picture Processing: 1986, Comput. Gr. Image Process., Vol. 38, pp. 147-225 (1987).
42) Rosenfeld, A.: Image Analysis and Computer Vision: 1987, Comput. Gr. Image Process., Vol. 42, pp. 234-293 (1988).
43) Rosenfeld, A.: An Architecture for Picture Parsing, in Narasimhan, R. (ed.), A Perspective in Theoretical Computer Science, World Scientific, Singapore (1989).
44) Rosenfeld, A.: Image Analysis and Computer Vision: 1988, Comput. Gr. Image Process., Vol. 46, pp. 196-264 (1989).
45) Rosenfeld, A.: On Connectivity Properties of Grayscale Pictures, Pattern Recogn., Vol. 16, pp. 47-50 (1989).
46) Rosenfeld, A. and Kak, A. C.: Digital Picture Processing, Academic Press, New York (1976), 長尾他訳：ディジタル画像処理，近代科学社 (1978).
47) Rozenberg, G. and Salomaa, A. (eds.): The Book of L, Springer-Verlag, Berlin (1986).
48) Siromoney, G. and Siromoney, R.: Hexagonal Arrays and Rectangular Blocks, Comput. Gr. Image Process., Vol. 5, pp. 353-381 (1976).
49) Siromoney, R.: Array Languages and Lindenmayer Systems—A Survey, in Rozenberg, G. and Salomaa, A. (eds.), The Book of L, pp. 413-426 (1986).
50) Siromoney, R. and Siromoney, G.: Extended Controlled Table Larrays, Inf. Control, Vol. 35, pp. 119-138 (1977).
51) Uesu, T.: Complete System of Grammars for Plane Graphs, Tsukuba J. Math., Vol. 3, pp. 129-160 (1979).
52) Wankmüller, F.: Application of Graph Grammars in Music Composing Systems, Lecture Notes in Computer Science, 291, pp. 580-592 (1987).
53) 山本，森田，菅田：長方形・正方形を生成する2次元正規アレイ文法 (RAG), 信学技報，AL 81, 109 (1982).
54) 雑誌：International Journal of Pattern Recognition and Artificial Intelligence, Vol. 3, No. 3, Special Issue on Array Grammars, Patterns and Recognizer (1989).

(平成元年9月13日受付)