言語の構造図の一応用について

鹿児島大学理学部紀要（数学・物理学・化学），
No. 8, p. 1-16, 1975.

言語の構造図の一応用について

藤野精一・富樫昭

（1975年9月30日　受理）

On an application of structural diagrams of languages

By

Seiiti Huzino and Akira Togasi

Abstract

In this paper we shall develop a method for constructing a finite automaton recognizing a given regular language by making use of the structural diagram of the language, refining Brzozowski's results about regular expression more directly, and also show its applicability to other languages by examples.

1 序

本論文の目的は，言語の導写像（derivative mapping）がオートマトンの構造の解明に密接に関連していることを示すことである。言語の導写像という概念は，それぞれすこしづつ重点のおきかた，記法等の相違はあるけれども，オートマトンの研究者の間には，早くから知られていた概念であった（[1]，[2]）。

しかし，正規事象に関して導写像を導入して，有限オートマトンとの間の関連を示したのはBrzozowski が最初である（[3]）。本論文では，これを発展させて，一般の形式言語への応用を試み，さきに [4] で導入した言語の構造図の概念を使用して，より一層明確な形で導写像の応用を述べようと思う。

2 言語の導写像と構造図

X_0 をアルファベット，その要素を記号とよぶ。X_0^* を X_0 の上の記号列の全体とする。X_0^* の任意の記号列の長さとは，その記号列を構成する記号の個数をいい，長さ0の記号列をとくに λ と表わす。いま X_0^* の任意の部分集合 L を言語（language）とよぶ。言語は，このように，記号列の集合を指すので，集合に関する演算はここでも適用される。和集合の記号∪を，記号 $+$ で表わすことにしよう。言語に関する演算 $+$, $\cdot$, $*$, c を次のように定義する（u, v, w, $\cdots$ で記号列を示すことにする）。

$$L_1 + L_2 \equiv \{u \mid u \in L_1 \text{ または } u \in L_2\}$$

$$L_1 \cdot L_2 \equiv \{uv \mid u \in L_1 \text{ かつ } v \in L_2\}$$

Seiiti Huzino (Department of Mathematics, Kyushu, University.)
Akira Togasi (Department of Mathematics, Kagoshima University.)

ここで uv は u と v との連接 (concatenation) とよばれ，記号列 u の後に記号列 v をひきつづいて書いた記号列をさす。

$$L_1^* \equiv \sum_{n=0}^{\infty} L_1^n$$
$$= L_1^0 + L_1^1 + L_1^2 + \cdots + L_1^n + \cdots$$

ただし

$$L_1^0 = \{\lambda\}$$
$$L_1^n = L_1 \cdot L_1^{n-1} \quad (n = 1,\ 2,\ \cdots)$$

である。

また，

$$L_1^c \equiv \{u \mid u \in X_0^* \text{ かつ } u \notin L_1\}$$

演算 $+$, $\cdot$, $*$, c をそれぞれ，和，積，kleene 積，補言語演算とよぶ。

ここで新しく言語の導写像という概念を導入しよう。

定義 2.1 X_0^* の中の言語 L，X_0^* の中の記号列 u に対して $\partial_u L$ を次のように定義し，L の u に関する導言語 (derivative language) とよぶ。

$$\partial_u L \equiv \{\, w \mid uw \in L \,\}$$

あきらかに，$\partial_\lambda L = L$ である。また，$u,\ v \in X_0^*$ に対して，

$$\partial_{uv} L = \partial_v\, (\partial_u L)$$

がなりたつ。

例 2.1 $X_0 = \{a,\ b\}$，$L = \{(ab)^n \mid n = 1,\ 2,\ \cdots\} = \{ab,\ abab,\ ababab, \cdots\}$ とすると，

$$\partial_a L = \{b,\ bab,\ babab,\ \cdots\}$$
$$= b\,\{\lambda,\ ab,\ abab,\ \cdots\}$$
$$= b\,(\lambda + L)$$
$$\partial_{ab} L = \{\lambda,\ ab,\ abab,\ \cdots\}$$
$$= \lambda + L$$

ここで，$\{\lambda\}$，$\{b\}$ のように単一要素集合に関しては，単に λ，b のように表わすことにした。このようにしても混同の恐れはないであろう。

定義 2.2 L を X_0 の上の言語 ($\subset X_0^*$) とする。このとき，写像 δ を

$$\delta\,(L) = \begin{cases} \lambda & (\lambda \in L \text{ のとき}) \\ \phi & (\lambda \notin L \text{ のとき}) \end{cases}$$

で定義する。この写像 δ を (言語に関する) λ-判定 (λ-test) とよぶ。

導写像は次の性質をもっている。

定理 2.1 ([3])

X_0 の上の任意の言語 L_1，L_2，任意の $x \in X_0$ に対して，次の関係がなりたつ。

$$\partial_x\,(L_1 + L_2) = \partial_x L_1 + \partial_x L_2$$
$$\partial_x\,(L_1 \cdot L_2) = (\partial_x L_1) \cdot L_2 + \delta\,(L_1) \cdot \partial_x L_2$$
$$\partial_x (L_1^*) = (\partial_x L_1) \cdot L_1^*$$

定理 2.2 L を X_0 の上の正則言語 (regular language)，u を X_o^* の任意の記号列とするとき，導言語 $\partial_u L$ は正則言語である。

証明 $u \equiv \lambda$ のときは $\partial_\lambda L = L$ であるから，自明である。一般の場合は $u = x_1 x_2 \cdots x_k$ ($x_i \in X_0$; $i = 1,\ 2,\ \cdots,\ k$) とするとき，関係式

$$\partial_u L = \partial_{x_k}(\partial_{x_{k-1}}(\cdots(\partial_{x_1} L)\cdots))$$

がなりたつので $u \equiv x$ $(x \in X_0)$ のときに $\partial_x L$ が正則であることを証明すればよい。仮定により，L が正則であるから，$L = T(\mathcal{A})$ $(\equiv \{w \mid w \in X_0^*$ かつ $f^*(s_0, w) \in D\})$ となる有限オートマトン $\mathcal{A} \equiv <S, f, s_0, D>$ が存在する ([5])。状態 s_x $(\in S)$ を $s_x = f(s_0, x)$ で定められる状態とする。有限オートマトン $\mathcal{A}_x$ を

$$\mathcal{A}_x \equiv <S, f, s_x, D>$$

ととる。S, f, D は $\mathcal{A}$ の S, f, D と同じものである。あきらかに $\mathcal{A}_x$ は有限オートマトンである。次の同値関係が成立する。

$$\begin{aligned} w \in \partial_x L &\Leftrightarrow xw \in L && (\partial_x \text{ の定義}) \\ &\Leftrightarrow f^*(s_0, xw) \in D && (L = T(\mathcal{A}) \text{ より}) \\ &\Leftrightarrow f^*(f(s_0, x), w) \in D && (f^* \text{ の定義}) \\ &\Leftrightarrow f^*(s_x, w) && (s_x = f(s_0, x)) \\ &\Leftrightarrow w \in T(\mathcal{A}_x) \end{aligned}$$

よって $\partial_x L = T(\mathcal{A}_x)$ となり，$\partial_x L$ が正則なることが証明された。 (証終)

定義 2.3 L を X_0 の上の言語とする。記号列 u, $v \in X_0^*$ に対して，(集合として) $\partial_u L = \partial_v L$ となるとき，$u \underset{L}{\equiv} v$ と書く。関係 $\underset{L}{\equiv}$ は同値関係 (equivalence relation) である。したがって商集合 $X_0^*/\underset{L}{\equiv}$ が考えられる。

定理 2.3 L が X_0 の上の正則言語であるための必要かつ十分条件は商集合 $X_0^*/\underset{L}{\equiv}$ が有限集合であることである。(これを $\underset{L}{\equiv}$ が有限指数 (finite index) をもつ同値関係であるという)

証明 1° 必要なること： L が正則であるから，$L = T(\mathcal{A})$ となる有限オートマトン $\mathcal{A} = <S, f, s_0, D>$ が存在する。$S = \{s_0, s_1, \cdots s_p\}$ とする。任意の $u \in X_0^*$ に対して，$s_u \equiv f^*(s_0, u)$ とし，有限オートマトン $\mathcal{A}_u \equiv <S, f, s_u, D>$ を構成すると，前定理の証明と同様にして，

$$\partial_u L = T(\mathcal{A}_u)$$

を得る。いま，$f^*(s_0, u) = f^*(s_0, v)$ なる記号列 u, $v \in X_0^*$ に対しては $T(\mathcal{A}_u) = T(\mathcal{A}_v)$ であるから，(集合として) $\partial_u L = \partial_v L$ である。また，

$$W_i \equiv \{w \mid w \in X_0^* \text{ かつ } f^*(s_0, w) = s_i\} \quad (i = 0, 1, 2, \cdots p)$$

とすると，あきらかに

$$X_0^* = W_0 + W_1 + \cdots + W_p$$

であり，任意の u, $v \in W_i$ に対して $\partial_u L = \partial_v L$ である。したがって，任意の u, $v \in W_i$ に対して $u \underset{L}{\equiv} v$ である。このことは各 W_i は $\underset{L}{\equiv}$ による X_0^* の同値類のいずれか1つにすっかり含まれることを示す。すなわち，W_i $(i=0, 1, \cdots, p)$ は $\underset{L}{\equiv}$ の細分となっている。また W_i $(i=0, 1, \cdots, p)$ は有限個であるから，$\underset{L}{\equiv}$ は有限指数である。

2° 十分なること： 逆に $X_0^*/\underset{L}{\equiv}$ は有限集合であるとする。関係 $\underset{L}{\equiv}$ は右不変 (right invariant) である。すなわち，$u \underset{L}{\equiv} v$ ならば，任意の $w \in X_0^*$ に対して $uw \underset{L}{\equiv} vw$ である。何となれば，$u \underset{L}{\equiv} v$ ならば $\partial_u L = \partial_v L$，したがって

$$\partial_{uw} L = \partial_w(\partial_u L) = \partial_w(\partial_v L) = \partial_{vw} L$$

すなわち，$uw \underset{L}{\equiv} vw$ である。

$[u]$ を，$u \in X_0^*$ を含む $X_0^*/\underset{L}{\equiv}$ の元（同値類）とする。写像 $\tilde{f}: X_0^*/\underset{L}{\equiv} \times X_0 \to X_0^*/\underset{L}{\equiv}$ を

$$\tilde{f}([u],\ x) = [u\,x] \quad (u \in X_0^*,\ x \in X_0)$$

で定義する。この定義は，$\underset{L}{\equiv}$ が右不変であるので，整合（consistent）である。$\tilde{S} \equiv X_0^*/\underset{L}{\equiv}$，$\tilde{s}_0 \equiv [X]$，$\tilde{D} \equiv \{[u] \mid u \in L\}$ とする有限オートマトン

$$\tilde{\alpha} = <\tilde{S},\ \tilde{f},\ \tilde{s}_0,\ \tilde{D}>$$

を作ると，$L = T(\tilde{\alpha})$ である。何となれば，次の同値関係が成立するからである。

$$\begin{aligned} w \in L &\Leftrightarrow [w] = [\lambda\, w] \in \tilde{D} \\ &\Leftrightarrow \tilde{f}^*([\lambda],\ w) \in \tilde{D} \\ &\Leftrightarrow w \in T(\tilde{\alpha}) \end{aligned}$$

よって L は正則である。（証終）

系 2.4 L が X_0 の上の正則言語であるための必要かつ十分なる条件は集合族 $\{\partial_u L \mid u \in X_0^*\}$ の中で，集合として相ことなるものは有限個であることである。

この結果は L が正則であるかどうかの判定に使用することができる。

例 2.2 $X_0 = \{a,\ b\}$ の上の言語 $L = \{a^n b^n \mid n = 1,\ 2,\ \cdots\} = \{ab,\ aabb,\ aaabbb,\ \cdots\}$ は正則ではない。何となれば

$$\begin{aligned} \partial_{a^m} L &= \{b^m,\ ab^{m+1},\ a^2 b^{m+2},\ \cdots\} \\ &= \{\lambda,\ ab,\ a^2 b^2,\ \cdots\}\ b^m \\ &= (\lambda + L)\ b^m \quad (m = 1,\ 2,\ \cdots) \end{aligned}$$

となり，$\partial_{a^m} L$ $(m=1,\ 2,\ \cdots)$ は集合として，すべて相ことなる。

系 2.4 を適用して次の定理を得る。

定理 2.5 L_1，L_2 を X_0 の上の正則言語とするとき，L_1+L_2，$L_1 \cdot L_2$，L_0^*，L_1^c は正則である。すなわち，正則言語の族は $+$，$\cdot$，$*$，c に関して閉じている。

証明 任意の $u \in X_0^*$ に対して

$$\begin{aligned} \partial_u (L_1 + L_2) &= \partial_u L_1 + \partial_u L_2 \\ \partial_u (L_1^c) &= (\partial_u L_1)^c \end{aligned}$$

であるから，集合族 $\{\partial_u (L_1 + L_2) \mid u \in X_0^*\}$，$\{\partial_u (L_1^c) \mid u \in X_0^*\}$ の中で，集合として相ことなるものは有限個である。よって，L_1+L_2，L_1^c は正則である。また，帰納法により，任意の $u \in X_0^*$ に対して，

$$\partial_u (L_1 \cdot L_2) = (\partial_u \cdot L_1) \cdot L_2 + \sum_{\substack{v,\ w \\ vw = u}} \delta(\partial_v L_1)\ \partial_w L_2$$

がなりたつことが証明される。よって，集合族 $\{\partial_u (L_1 \cdot L_2) \mid u \in X_0^*\}$ の中で集合としてことなるものは有限個である。また同様にして，$\partial_u (L_1^*)$ は，集合 $(\partial_v L_1) \cdot L_1^*$ $(v \in X_0^*)$ の形の有限和で表現されるから，L_1 の仮定より，正則であることが示される。（証終）

3 正則言語の構造図と有限オートマトン

前節の系 2.4 は正則言語を受理する有限オートマトンの作成に利用される。最初に言語の構造図を定義しよう。

定義 3.1 X_0 の上の言語 L の構造図とは，任意の $u \in X_0^*$ に対して，$\partial_u L$ を作成し，集合族 $\{\partial_u L \mid u \in X_0^*\}$ の中で集合として相ことなる言語を $L_0\,(\equiv L)$，L_1，L_2，$\cdots$（一般に 無限

個）として，構成要素の各言語 L_i の間を ∂_x $(x \in X_0)$ をラベルにもつ矢線 → で結んだ図形をいう。この際，たとえば，$\partial_x L_i = L_j$ のとき，図 3.1 のように，L_i から L_j へ矢線 ∂_x をひく。

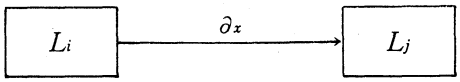

図 3.1

実際には L $(\equiv L_0)$ を出発点にして，∂_x $(x \in X_0)$ を順々にくりかえして，導言語を作成し，集合として以前にでてきた導言語とことなるものがあれば，∂_x $(x \in X_0)$ をその言語にさらにくりかえして進んでいけばよい。

例 3.1

$$X_0 = \{a,\ b\}$$
$$L_0 = \{(ab)^n | n = 1,\ 2,\ \cdots\}$$

この言語の構造図は図 3.2 のようになる。

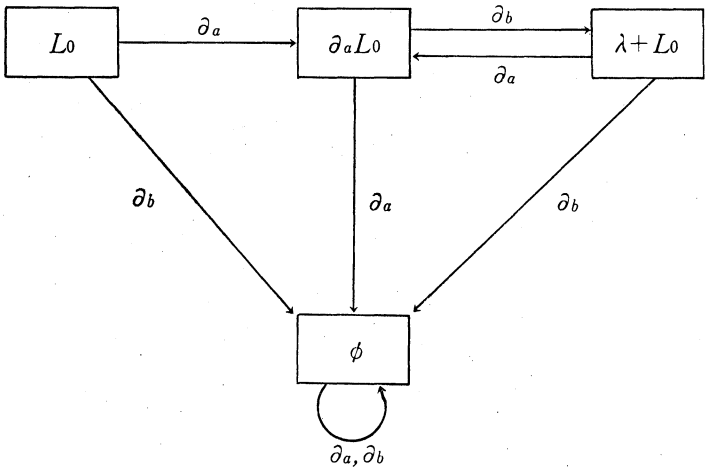

図 3.2

図 3.2 は L_0 から出発して導言語を次のように計算して図表示したものである。

$$\begin{aligned}
L_0 &= \{(ab)^n | n = 1,\ 2,\ \cdots\} \\
&= \{ab,\ abab,\ \cdots\ \ \ \ \ \ \ \ \} \\
\partial_a L_0 &= \{b,\ bab,\ babab,\ \cdots\} \\
\partial_b L_0 &= \phi \\
\partial_a (\partial_a L_0) &= \phi \\
\partial_b (\partial_a L_0) &= \{\lambda,\ ab,\ abab,\ \cdots\} = \lambda + L_0 \\
\partial_a (\partial_b (\partial_a L_0)) &= \partial_a (\lambda + L_0) = \partial_a L_0 \\
\partial_b (\partial_b (\partial_a L_0)) &= \phi
\end{aligned}$$

例 3.2

$$X_0 = \{a,\ b\}$$
$$L_0 = \{a^n b^n | n = 1,\ 2,\ \cdots\}$$

この言語の構造図は図 3.3 のようになる。
これは次のように計算される。

$$\partial_a L_0 = \{b,\ ab^2,\ \cdots,\ a^{n-1}b^n,\ \cdots\}$$

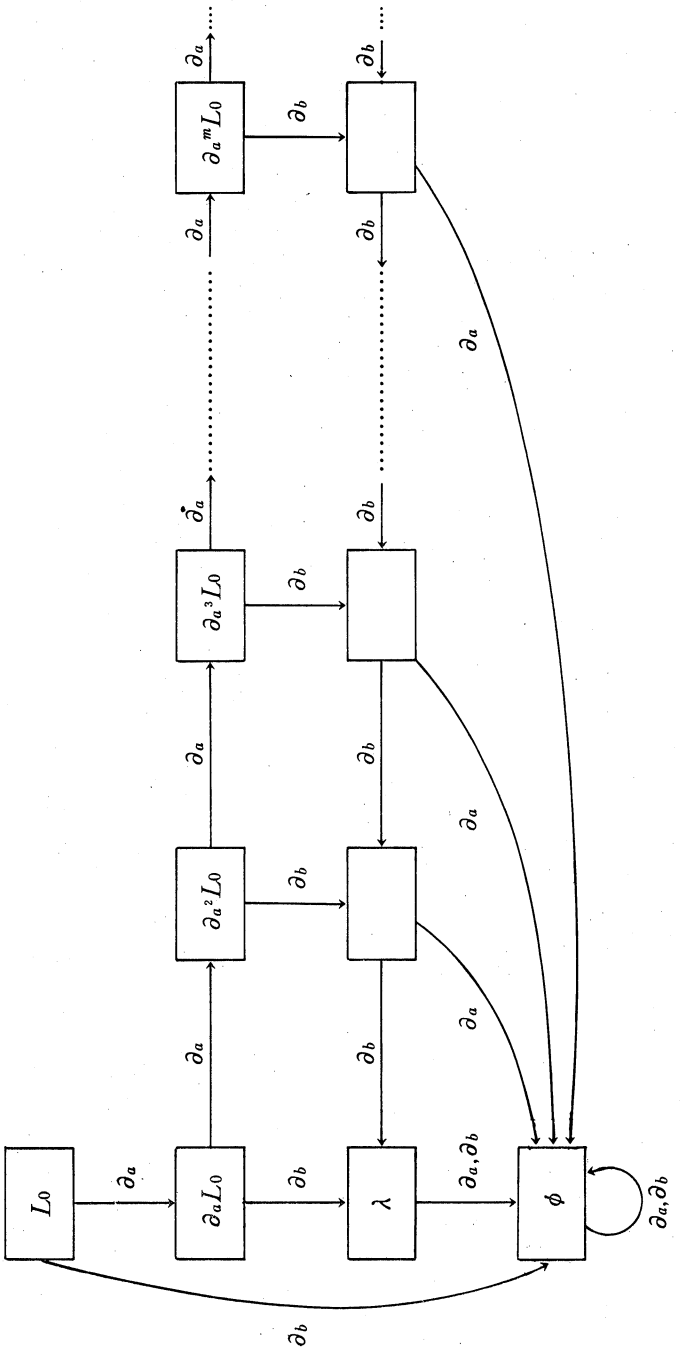
L_0
$\partial_a L_0$
$\partial_a^2 L_0$
$\partial_a^3 L_0$
$\partial_a^m L_0$
λ
ϕ
∂_a
∂_b
∂_a, ∂_b

図 3.3

$$= (\lambda + L_0)\, b$$

$$\partial_b L_0 = \phi$$

$$\partial_{a^m} L_0 = \{b^m,\ ab^{m+1},\ \cdots\} = (\lambda + L_0)\, b^m \quad (m = 1,\ 2, \cdots)$$

$$\partial_b (\partial_{a^m} L_0) = b^{m-1} \quad (m = 1,\ 2,\ \cdots;\ b^0 = \lambda)$$

定義 3.2　X_0 の上の言語 L の構造図を構成する導言語の族を $L_0\ (\equiv L)$, L_1, $L_2, \cdots$ とする。このとき，次の手順によって構成される図を（一般には無限個の状態をもつ）状態図という。

手順 1°　$\boxed{L_i}$ を (s_i) に変更する。

start

2°　(s_0) を初期状態 (s_0) とする。

3°　$\lambda \in L_i$ なる L_i に対応する (s_i) を 2 重丸指定 $((s_i))$ として，s_i を指定状態とする。

4°　矢線の上の ∂_x $(x \in X_0)$ を x $(x \in X_0)$ に書きかえる。

この手続き 1°～4° をアルゴリズム $\mathcal{A}$ という。

例 3.3　(i) 例 3.1 の言語 $L_0 = \{(ab)^n \mid n = 1,\ 2,\ \cdots\}$ にアルゴリズム $\mathcal{A}$ を適用すると，図 3.4 の状態図が得られる。

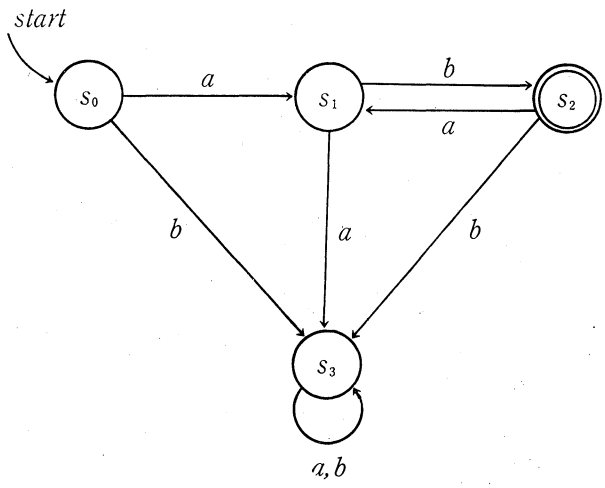

図 3.4

(ii) 例 3.2 の言語 $L_0 = \{a^n b^n \mid n = 1,\ 2,\ \cdots\}$ に対しては，図 3.5 の（無限）状態図が得られる。

状態の添数のつけ方を除いて，状態図は一意に定まる。このようにして得られる言語 L の状態図は，無限オートマトン $\mathfrak{a} = <S,\ f,\ s_0,\ D>$ を構成する。

ここに

$S = \{s_0,\ s_1, \cdots, s_n, \cdots\}$，すなわち言語 L の構造図を構成する 各言語に対応する状態の集合

$f : S \times X_0 \to S$ は

$f(s_i,\ x) =$（$\partial_x L_i$ に対応する状態）で定義する。

s_0 : L_0 に対応する状態

D : λ をふくむ導言語に対応する状態の集合

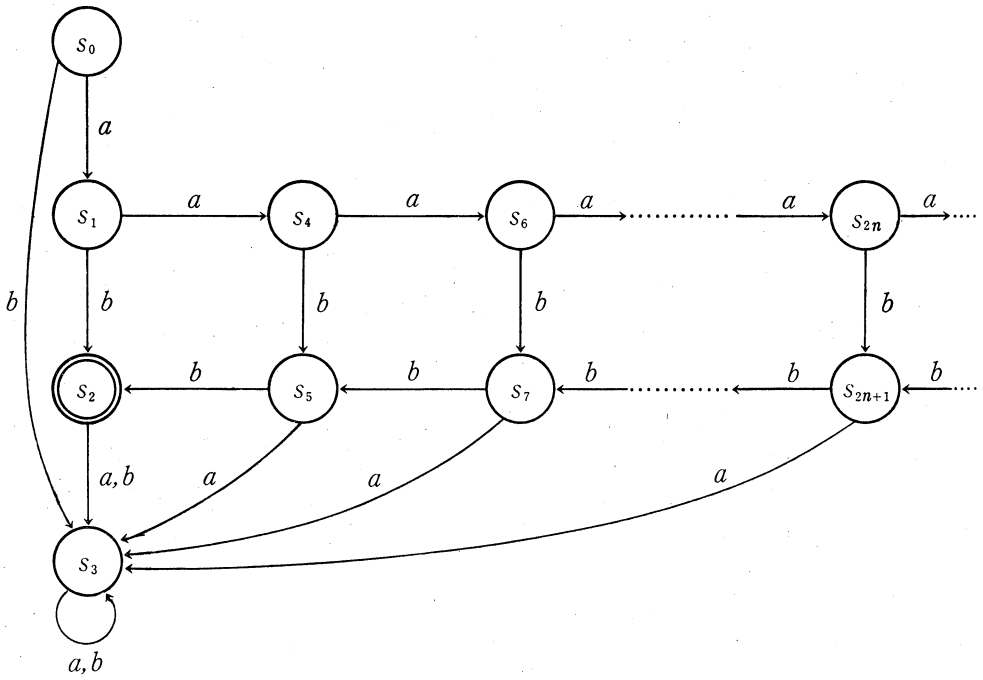

図 3.5

定理 3.1 X_0 の上の任意の言語 L に対して，L の構造図にアルゴリズム $\mathscr{A}$ を施して得られる状態図をオートマトンの機構とするオートマトン $\mathfrak{a}=<S, f, s_0, D>$（上述）を作る。このとき，$L$ は $\mathfrak{a}$ で確認される。すなわち，

$$L=T(\mathfrak{a})=\{w \mid w\in X_0^*,\ f^*(s_0,\ w)\in D\}$$

（ここに f^* の定義は $\mathfrak{a}$ が有限オートマトンの場合と同様に定義する）

証明 次の同値関係が成立する。

$$\begin{aligned} w\in L &\Leftrightarrow \lambda\in\partial_w L \\ &\Leftrightarrow \partial_w L \text{ に対応する状態が } D \text{ の要素である。} \\ &\Leftrightarrow f^*(s_0,\ w)\in D \quad (\mathscr{A} \text{ の定義より}) \\ &\Leftrightarrow w\in T(\mathfrak{a}) \end{aligned}$$

系 3.2 L が正則言語であれば，定理3.1で作成されるオートマトン $\mathfrak{a}=<S, f, s_0, D>$ は $L=T(\mathfrak{a})$ となる有限オートマトンである。

この系により，正則言語を確認する有限オートマトンは，言語の構造図を作成して，手続き $\mathscr{A}$ を適用して自動的に得られることがわかる。

例 3.4 $X_0=\{0,\ 1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 7,\ 8,\ 9\}$

$L_0=\{w \mid w\in X_0^*,$ かつ w は5で整除される$\}$

この言語 L_0 が正則であることを，L_0 を確認する有限オートマトンを系3.2によって作成することにより示そう。

$$\begin{aligned} L_0 &= \{u\,0 \mid u\in X_0^*\}+\{u\,5 \mid u\in X_0^*\} \\ &= X_0^*\,(0+5) \\ \partial_i L_0 &= \partial_i\,(X_0^*\,(0+5)) \\ &= (\partial_i X_0^*)\,(0+5)+\delta\,(X_0^*)\ \partial_i\,(0+5) \\ \partial_i X_0^* &= (\partial_i X_0)\ X_0^*=X_0^* \\ \delta\,(X_0^*) &= \lambda \end{aligned}$$

$$\partial_i\ (0+5) = \begin{cases} \lambda & (i=0,\ 5\ \text{のとき}) \\ \phi & (i \neq 0,\ 5\ \text{のとき}) \end{cases}$$

これより

$$\partial_i L_0 = \begin{cases} X_0^*\ (0+5) + \lambda \\ \quad (= L_0 + \lambda) & (i=0,\ 5\ \text{のとき}) \\ X_0^*\ (0+5)\ \ (= L_0) & (i \neq 0,\ 5\ \text{のとき}) \end{cases}$$

また

$$\partial_i\ (L_0 + \lambda) = \partial_i L_0$$

よって，L_0 の構造図は次のようになる。(図 3.6)

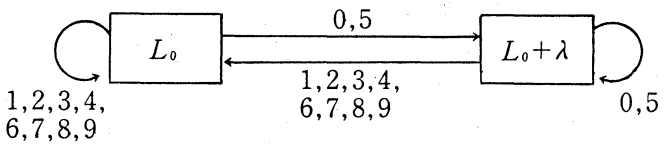

図 3.6

したがって，これにアルゴリズム $\mathcal{A}$ を適用して L_0 を確認する有限オートマトンが得られる。(図 3.7)

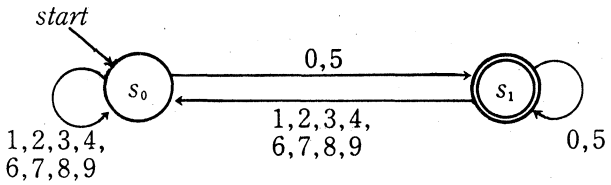

図 3.7

系 3.3 言語 L が正則であるための必要十分条件は，L の構造図が，有限節点をもつ有向グラフとなっていることである。

定理 3.4 正則言語 L に対して定理 3.1 で得られる有限オートマトンは，L を確認する有限オートマトンのなかで，最小の状態をもつ有限オートマトンである。

証明 L の構造図にアルゴリズム $\mathcal{A}$ をほどこして得られる有限オートマトンを $\alpha = <S, f, s_0, D>$ とする。$L=T(\alpha)$。いま L を確認する任意の有限オートマトンを $\tilde{\alpha} = <\tilde{S}, \tilde{f}, \tilde{s}_0, \tilde{D}>$ とする。$L=T(\tilde{\alpha})$。また，$\tilde{S} = \{\tilde{s}_0, \tilde{s}_1, \cdots, \tilde{s}_n\}$ とする。$\tilde{f}^*(\tilde{s}_0, u) = \tilde{f}^*(\tilde{s}_0, v)$ なる $u, v \in X_0^*$ をとると，いつでも $u \underset{L}{\equiv} v$ である。すなわち，$\partial_u L = \partial_v L$ である。いま，$W_i \equiv \{w \mid \tilde{f}^*(\tilde{s}_0, w) = \tilde{s}_i\}$ $(i=0,\ 1,\ 2, \cdots n)$ とすると，任意の $w \in W_i$ は $[w]_{\underset{L}{\equiv}}$ の要素である。よって

$$\begin{aligned} \#\ (\tilde{S}) &\geqq \#\ (X_0^*/\underset{L}{\equiv}) \\ &= \#\ (S) \end{aligned}$$

4. 自由言語の構造図とプッシュダウン・オートマトン

前節で言語の構造図を応用して，正則言語を確認する有限オートマトンを構成できることを

知った。さて，言語が自由 (context free) であるとき，この言語は，一般に非決定性プッシュダウンオートマトンによって確認されることを知っている ([6])。この確認プッシュダウンオートマトンを，前節の有限オートマトンの場合のように構成できないものであろうか？ この問題は，かなり興味ある問題である。いままでのところ部分的には，A. A. Letichevskii ([7]) が研究し，これをわかりやすい形で A. Salomaa ([8]) が解説している。しかし，これは完全な解決とはいえない。その理由は，確認プッシュダウンオートマトンが，無限個の状態を許した形でとらえられているところにある。彼等はそのようなオートマトンに，抽象的プッシュダウンオートマトン (abstract pushdown automaton) となづけた。これを，普通の意味のプッシュダウンオートマトンを構成するように変形することができれば，問題は解決されるが，それはかなり困難なようである。そこで，彼等の方法はとらないで，言語の構造図を直接に利用する方法はないだろうか，と考えた。いまのところ，まだ完全な解決にはいたっていないが，本節でその方法の成功した例をあげ，このような方法が一般に適用され得るのではないかということを，今後の問題として提示しよう。

はじめに，$X_0=\{a, b\}$ の上の言語 $L_0=\{a^n b^n \mid n=1, 2, \cdots\}$ を代表的にとってみよう。この言語を確認する無限オートマトンは，その構造図より図 4.1 のように得られた (前節例 3.3)。

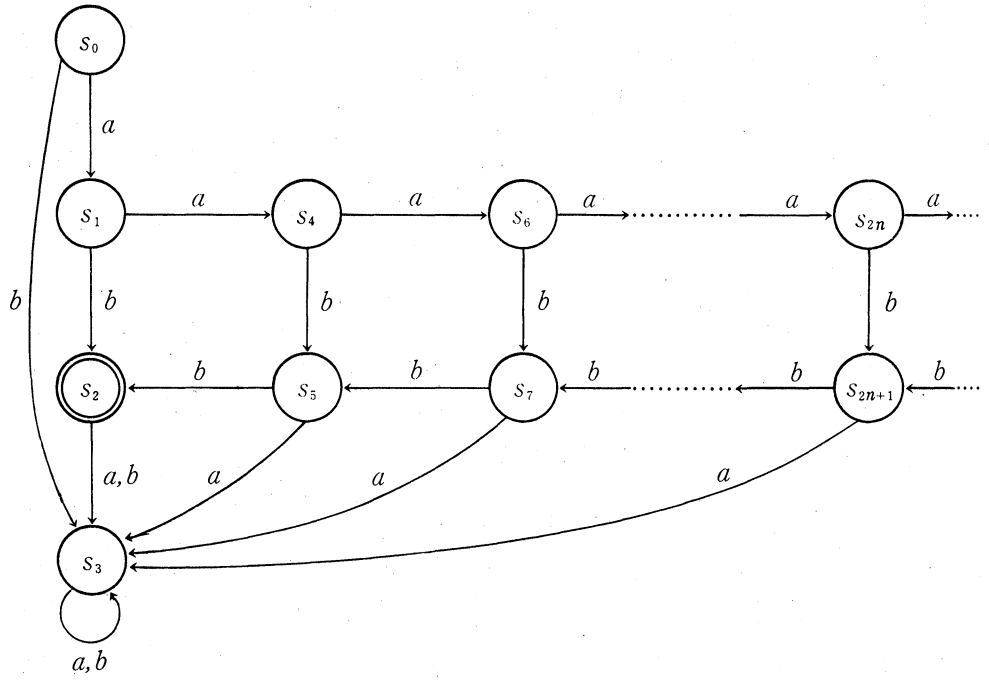

図 4.1

この図は図 4.2 のような (1 矢線が外へ飛びだした) 有限オートマトンに，図 4.3 の形の (外に飛びだす矢線を 3 本もった) 有限オートマトンが，無限個一列にくっついたものと考えられる (図 4.4)。ということは，状態 s_1 で a を読んだとき，a がつづくかぎり，つぎつぎに図 4.3 の形の有限オートマトンを，もとのオートマトン (図 4.2) の上にプッシュダウンし，そのプッシュダウンされたオートマトンを，現時点で使用することによって，確認作業を続行するものと解釈されよう。したがって，2 回目以後 a がでるたびに，図 4.3 の形で使用する有限

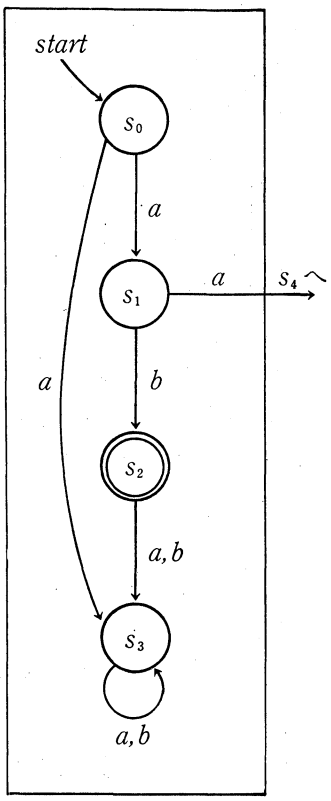

図 4.2

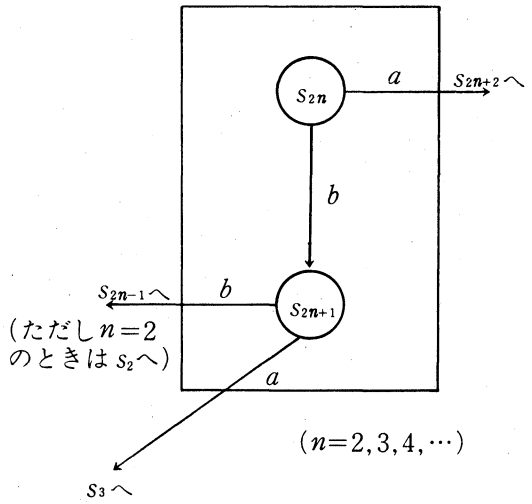

図 4.3

オートマトンを1つ増加し，2回目の b 以後は，この形の有限オートマトンを1つずつ減じていくようにしておけばよい。ところで，このことは，回数を表示する何等かの指標をくっつけておけば，図4.3の形のオートマトンを無限個使用する必要はなく，ただ1個のオートマトンで代用することができる。その指標は，プッシュダウンスタックを利用することによって達成

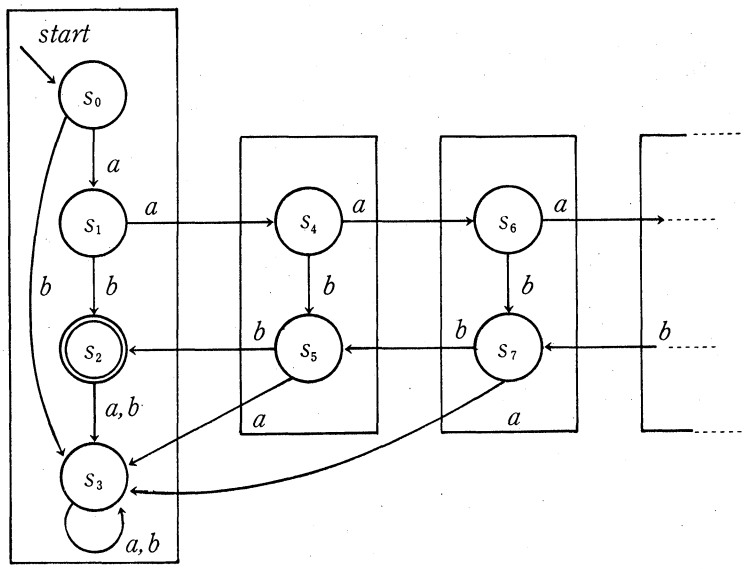

図 4.4

される。それを次のように構成すればよい。

まず図4.5をみてみよう。この図の最左端のオートマトンが，外へでるラベル a の矢線の行先を s_6 にしないで，s_4 へ逆もどりさせる。そのかわり，s_4，s_5 の組を s_6，s_7 の組のかわりに今かりに使用するということを表現するために，プッシュダウンスタックを1つ設置したオー

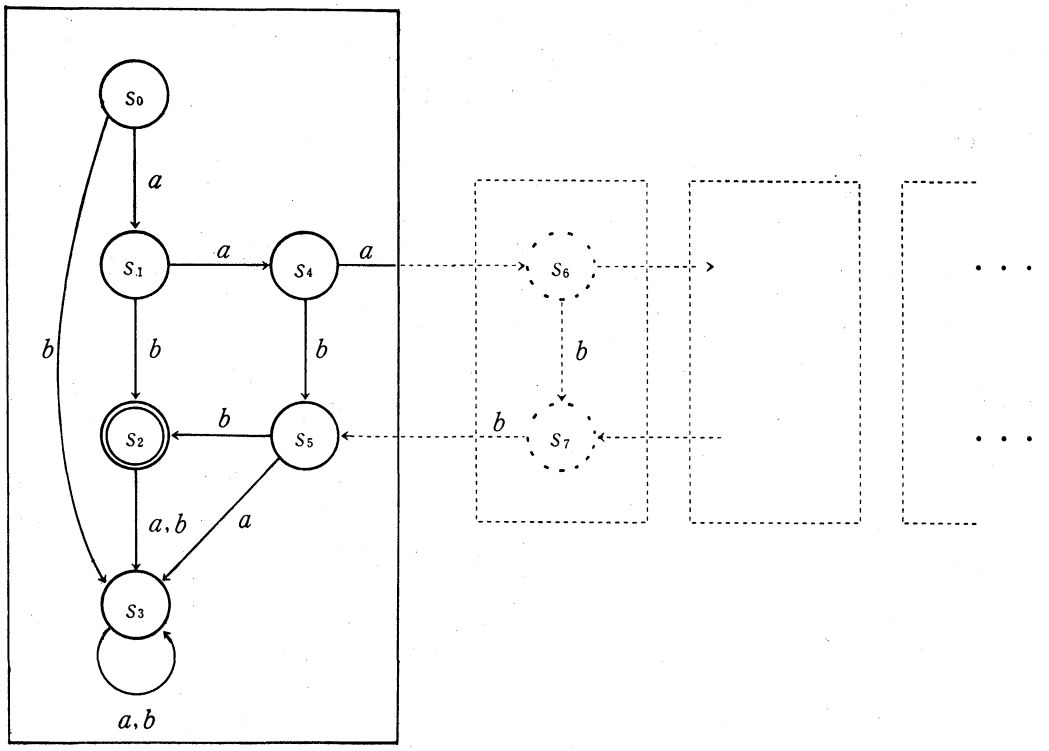

図 4.5

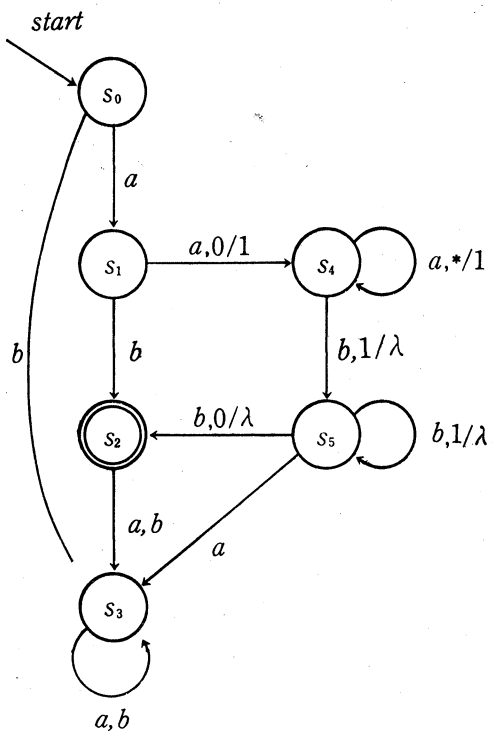

図 4.6

トマトンを考えることにし，プッシュダウンスタックに1を入れて，このことを表示することにする。スタックにはあらかじめ0が1個入っているものとする。

プッシュダウンスタックに入っている1の個数で，現在使用しているオートマトンが，どの組 (s_{2n}, s_{2n+1}) であるかがわかるようになる。b がきたとき，スタックの最上位の1をとりさり，使用の組を1つ下げる。このことを表わすため，矢線の上の a のかわりに a, $*/1$ という記号を同じ矢線の上に書いて，この意味を 'a を読んだとき，プッシュダウンスタックに現在何が入っていようともその上に記号1を入れる' とする。また b のかわりに b, $1/\lambda$ と書き，その意味を 'b を読んだとき，プッシュダウンスタックの上の記号が1のとき，これを1字消す' とする。$*$ は任意の記号を表わすものとする。これを s_4 へ入る矢線 a と，s_5 からでる矢線 b にも適用して，結局次の図4.6を構成する。

あきらかに，これは $L_0=\{a^nb^n \mid n=1, 2, \cdots\}$ を確認するプッシュダウンオートマトンである。

このように言語の構造図で，同じ形が無限個くりかえされるところを見出せばよい。プッシュ

ダウンスタックを利用して，これを1個で代用することができる。この方法は次の場合にもうまく適用される。

例 4.1 $X_0 = \{a,\ b,\ c\}$

$L_0 = \{w\,c\,w^R | w \in \{a,\ b\}^*\}$

ただし w^R は w の鏡像，すなわち

$w = \sigma_1 \sigma_2 \cdots \sigma_k \in \{a,\ b\}^*$ のとき，

$w^R = \sigma_k \sigma_{k-1} \cdots \sigma_1$

L_0 の構造図は関係式

$$\partial_u L_0 = L_0 u^R \quad (u \in \{a,\ b\}^*)$$
$$\partial_c L_0 = \lambda$$

より次のようになる（図4.7）。

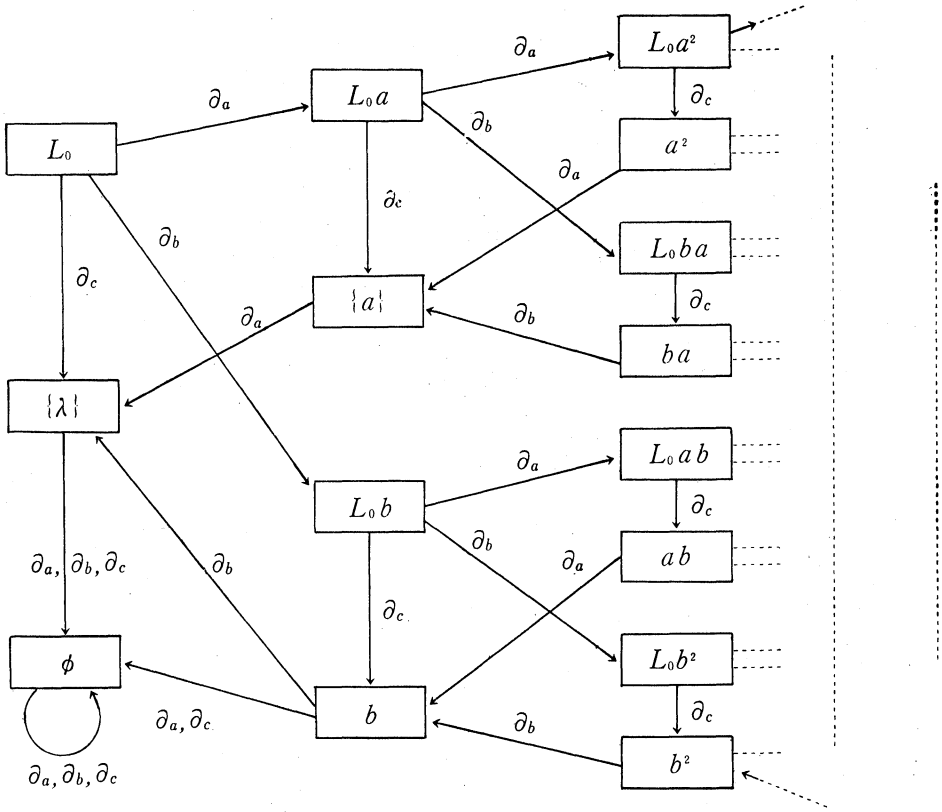

図 4.7

したがって，この L_0 を確認するオートマトンとして，次の状態図をもつ無限オートマトンが得られる（図4.8）。

最初の例と同様に考えると，今度の場合は2つの有限オートマトン（図4.9，図4.10）の無限のくりかえしになっている。

したがって，プッシュダウンスタックを1個設置した有限オートマトンを，非決定的に使用することによって，図4.11のような非決定性プッシュダウンオートマトンが得られる。

ここで $a,\ A/\lambda$ は入力記号が a で，プッシュダウンスタックの最上位の記号が A のとき実行されて，プッシュダウンスタックの最上位の記号を消す（λ）。$a,\ */A$ は入力記号が a で，

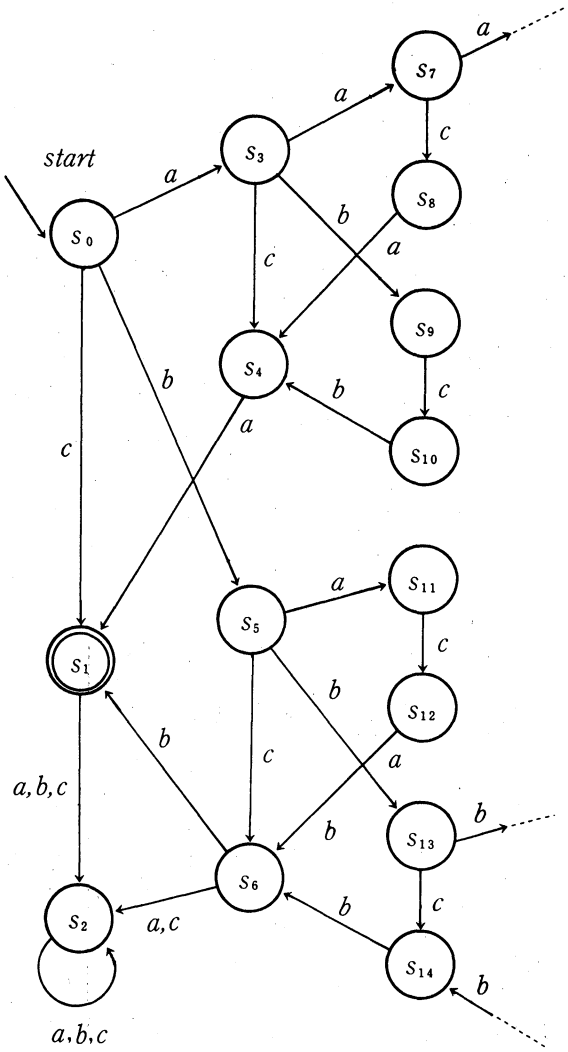

図 4.8
(s_2 へ達する矢印は，はじめの部分以外は省略した)

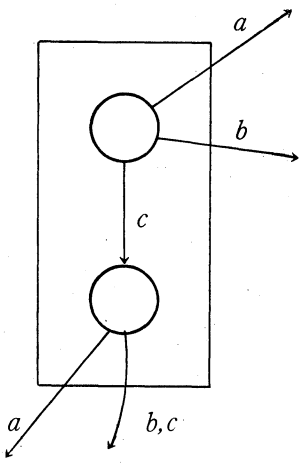

図 4.9

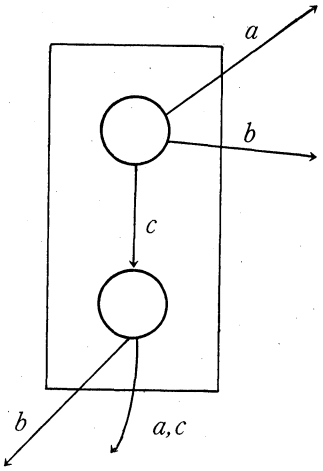

図 4.10

プッシュダウンスタックに何が入っていようとも，(*)，A をプッシュダウンするという意味である。

このオートマトンが，L_0 を確認することは，プッシュダウンスタックの使用の仕方を追跡してみればあきらかであろう。

この方法は今のところ，個々の場合に，構造図の関連の状態をみて，そのプッシュダウンのあり方を決定している。一般の場合に，どのような形で規則が表われるかは，わかっていない。今後に残された問題である。

5 その他の言語の構造図

最後に文脈関連言語の構造図の例として，$X_0=\{a, b, c\}$ のときの言語 $L_0=\{a^nb^nc^n \mid n=1, 2, \cdots\}$ をとりあげよう。この言語の構造図は図5.1のようになる。

仮に有限オートマトンに，プッシュダウンスタックを1個使用しただけでは，無限個使用さ

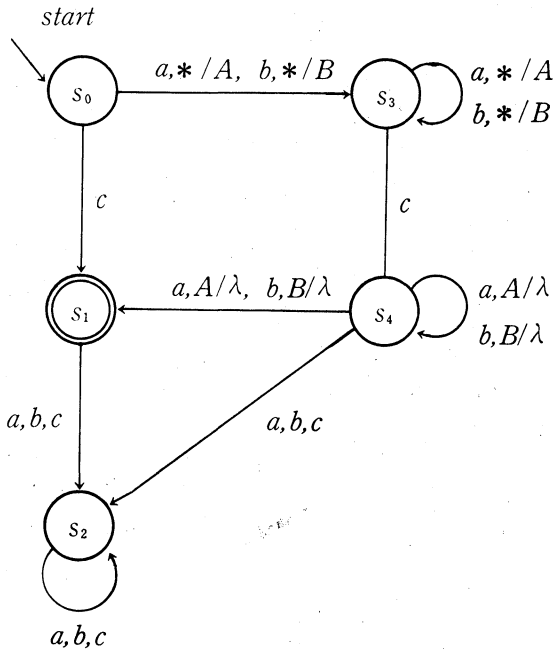
start
S_0
$a,*/A,\ b,*/B$
S_3
$a,*/A$
$b,*/B$
c
c
S_1
$a,A/\lambda,\ b,B/\lambda$
S_4
$a,A/\lambda$
$b,B/\lambda$
a,b,c
a,b,c
S_2
a,b,c

図 4.11

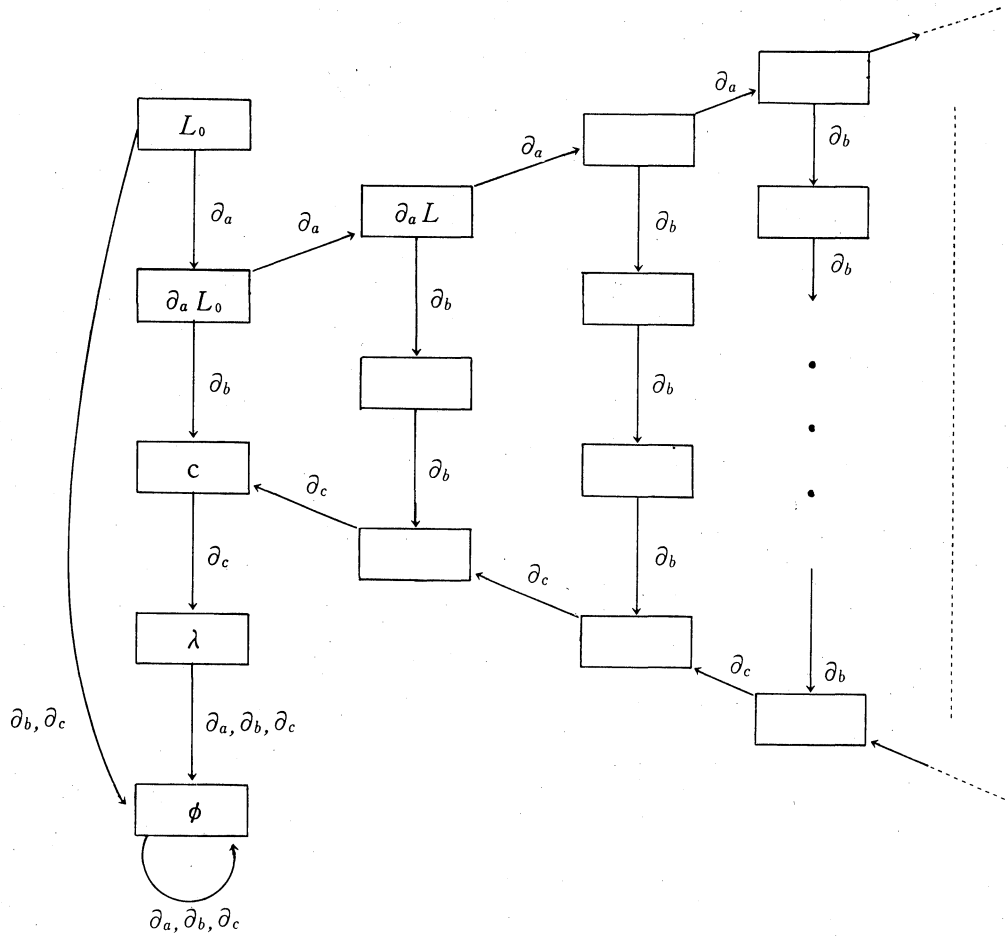
L_0
∂_a
$\partial_a L_0$
∂_b
c
∂_c
λ
$\partial_a, \partial_b, \partial_c$
ϕ
$\partial_a, \partial_b, \partial_c$
∂_b, ∂_c
∂_a
$\partial_a L$
∂_b
∂_c
∂_a
∂_b
∂_c
∂_a
∂_b
∂_b
∂_b
∂_b
∂_b
∂_b
∂_c
∂_c
∂_b

図 5.1

れる各有限オートマトン（たての列）の，使用順を記憶することができても（たての列）の，各オートマトンの状態の個数の増加を記憶することができない。プッシュダウンオートマトンでは，この言語の確認はできないことを，この言語の構造図が示しているとみることができよう。このことは，言語の構造図が，確認オートマトンの構造と非常に密接に関連していることを示すものといえよう。

参考文献

[1] C.C. Elgot: Decision problems of finite automata design and related arithmetic, Trans. Amer. Math. Soc. 98 (1961), 21–51.
[2] S. Ginsburg and E.H. Spanier: Quotients of context free languages, J. ACM. 10 (1963), 487–492.
[3] J.A. Brzozowski: Derivatives of regular expressions, J. ACM. 11 (1964), 481–494.
[4] S. Huzino: On some properties of derivative mappings, structural diagrams and structural equations, Part I. Mem. Fac. Sci. Kyushu Univ. Ser. A. 20 (1966), 179–265.
[5] M.O. Rabin and D. Scott: Finite automata and their decision problems, IBM J. Res. Devel 3 (1959), 114–125.
[6] J.E. Hopcroft and J.D. Ullman: Formal languages and their relation to automata, (1969), Addison Wesley, Reading, Mass.
[7] A.A. Letichevskii: The representation of context free languages in automata with a pushdown type store, Kibernetika, 1 (1965), 80–84. (Russian)
[8] A. Salomaa: Theory of automata, Pargamon Press (1969), N.Y.