準 LL(2) 文法に対する解析表の構造と解析アルゴリズム†

吉田敬一†† 竹内淑子†††

下降型構文解析法として代表的な LL(k) 文法解析法に関するいままでの研究のほとんどは LL(1) 文法に対するものであった．そして，LR(1) 解析法が実用的になるにつれて，LL(1) 解析法はやや退潮の兆しを示したが，LL 解析法固有の特長のゆえに，根強く利用されている．そしていままた，自然言語処理への LL(1) 解析法の応用という新しい局面が開かれようとしている[8]．しかし，LL 文法は表現能力の点で LR 文法に劣る．LR 文法の場合，LR(k) 文法，$k\geqq 2$ の言語クラスは LR(1) 文法の言語クラスと一致することはよく知られている．しかし，LL 文法では LL(k) 文法と LL($k+1$) 文法の間では後者の文法に対する言語クラスのほうが前者のそれより大きい．そこで，LL 文法のもついくつかの特長を継承しつつ，その言語クラスを高めるには LL(k) 文法，$k\geqq 2$ に対する解析法の研究が待たれる．とくに，実用という点を考えるとき LL(2) 文法に対する解析法の確立に対する期待は少なくない．しかるに，LL(1) 文法はすべて強 LL(1) 文法であるが LL(k)，$k\geqq 2$ に対して強のものと強でないものとが存在し，強でない文法に対しては文脈問題がからむため，解析表作成が複雑になる．$k=2$ の場合についての Aho-Ullman の方法も「解析表が大きい」，「解析表の作成法が複雑」，「解析表作成法が LL(1) 文法の場合と全く異なる」といったいくつかの欠点をもっている．本論文では，LL(k) 文法にわずかに制限を加えた準 LL(k) 文法を提案し，とくに $k=2$ の場合について，解析表の作成が容易な解析表の構造とその解析表による解析法を提案するものである．提案する方法は従来の方法の欠点を解消し，かつ表の部分（解析表と生成規則表）の大きさは Aho-Ullman の方法の約 1/120 程度になった．解析時間は，Aho-Ullman の方法の約 2 倍弱かかるが，これは解析表の工夫により同程度まで速くすることができる．

1. はじめに

LL 構文解析法は LR 構文解析法とともに，解析法の代表的なものの 1 つである．LL 文法は LR 文法にくらべて，表現能力が小さいが「解析表作成が容易」，「属性との相性がよい」，「解析プログラムが小さくてすむ」，「下降型解析は見通しがよい」などといったいくつかの特長をもつ．こうした特長を生かして，LL(1) 文法の解析法を，自然言語処理に応用しているという報告がある[8]．しかし，LL 構文解析法についていくつかの研究がなされ，その成果が報告されているのはいずれもが LL(1) 文法に対するものである[1)~3),5),7),8)]．それにくらべて LL(k)，$k\geqq 2$ に対するものはほとんど見られない．一般に，k が 2 以上になると解析表の作成が LL(1) のそれにくらべて数段の困難さを伴うと言われている．その大きな理由は，LL(1) 文法はすべて強 LL(1) 文法であるが，k が 2 以上になると強 LL のものとそうでないものとが存在し，後者に対する表構造は文脈に依存するためである．いままで，$k\geqq 2$ の場合に対する LL(k) 文法の解析法に関する研究成果はほとんど発表されていない．具体例をあげて示したものは，わずかに文献 1) の方法があるにすぎない．具体例をあげたものとしては他に文献 7) があるが，これは手続き型であるため，全く手法を異にするものであるので，ここでは考察しない．

解析法と表構造は密接な関係がある．そこで，文献 1) による解析表の作成法から表構造を考察する．この作成法はある非終端記号 A に対して，その右側に n 個 ($n\geqq 1$) の異なる文脈があるとき，A に対して新しい非終端記号 $T_1, T_2, \cdots, T_n$ を導入することにより，文脈問題を解決している．つまり，文法を Aho ら固有の強 LL(k) に変換している．この解決法による欠点は，次のとおりである．

(i) A に右接しうる文脈の数だけの T_i を導入するため，膨大な作業量となる．この作業をすべての非終端記号について行い，しかも，同じ非終端記号が右辺に 2 度以上現れてもよいことを考慮するとその作業に費やす時間はかなりの量となる．こうして作り出される T_i の数は，PASCAL−[6] 文法について実験してみたところ，もともと 54 しかなかった非終端記号から作られたその数は $k=2$ としたときでも 400 を超えた．さらに，それによってかきかえられた生成規則の数はもとの PASCAL−文法の生成規則の 98 本に対して 800 本を超えた．

† The Structure of the Parsing-Table and the Parsing Algorithm for Semi-LL(2) Grammars by KEIICHI YOSHIDA (Department of Computer Science, College of Engineering, Shizuoka University) and YOSHIKO TAKEUCHI (Department of Computer Science, Hamamatsu Polytechnic College).

†† 静岡大学工業短期大学部情報工学科
††† 浜松職業訓練短期大学校情報処理科

（ii） 解析表が大きくなる．解析表は最大 $\|\mathcal{G}\cup\Sigma\cup\{\$\}\|\times\|\Sigma^{*k}\|$ になる．ここで，$ はスタックの底を示す特殊な記号，Σ は終端記号の集合，$\mathcal{G}$ は新しくつくられた非終端記号の集合である．また，Σ^{*k} は長さ k 以下の終端記号列の集合で，$\|\mathcal{G}\cup\Sigma\cup\{\$\}\|$, $\|\Sigma^{*k}\|$ はそれぞれ $\mathcal{G}\cup\Sigma\cup\{\$\}$, Σ^{*k} の要素の個数を表す．

LL(k), $k\geqq 2$ の中で実用上重要なのは $k=2$ の場合であるが，いままで述べてきたように LL(2) 文法の解析表作成は LL(1) の場合にくらべて格段に難しくなる．

そこで本論文では準 LL(k) (semi-LL(k) grammar) なる文法を提案し，平易な構造の解析表の実現が容易である $k=2$ の場合に関してその解析法を提案する．準 LL(k) 文法はその表現能力において LL(k) 文法と強 LL(k) 文法の間に位置するものである．本論文で提案する解析法に用いられる解析表の作成法については紙数の都合上，別の論文[4]で述べる．この解析表は基本的には生成規則番号を利用した解析表作成法[3),5)]に基づくもので，生成規則番号と文脈を解析表の中に保存するものである．これにより，従来の方法がもつ上のような問題点（i）および（ii）を解決することができた．従来の方法[1)]と性能を比較した結果，記憶容量はコード部分は 4% ほど筆者らのほうが大きくなったが，表の部分（解析表と生成規則表）は Aho らのものの約 1/120 程度におさまった．解析時間は 45 ステップ前後のプログラムで，Aho らのものより 2 倍弱の時間を必要としたが，6 章の解析速度の項で触れるように解析表の若干の工夫により表は大きくなるが，解析時間をほぼ同程度にまで短縮することができた．

2. 基本的定義と準 LL(k) 文法

2.1 基本的定義

準 LL(k) 文法の記述に必要ないくつかの定義，記法について述べる．

［定義 1］

文脈自由文法 G を

$$G=(N,\Sigma,P,S)$$

とする．ここに，N,Σ はそれぞれ文法 G の非終端記号の集合ならびに終端記号の集合，P は生成規則の集合であり，S は出発記号である．

［記法 1］

N^*,Σ^* 等は，それぞれ N,Σ 上の空列を含むすべての記号列の集合を表す．

［記法 2］

N の要素を $A,B,C,\cdots$，Σ の要素を $a,b,c,\cdots$，$N\cup\Sigma$ の要素を X,Y,Z で表す．また，Σ^* の要素を $s,t,u,\cdots$ で表し，$(N\cup\Sigma)^*$ の要素を $\alpha,\beta,\gamma,\cdots$ で表す．これらの記号は添字をつけて用いることもある．

［定義 2］

集合 $\mathrm{FIRST}_k(\alpha)$ は以下で定義される．

$$\mathrm{FIRST}_k(\alpha)=\{u\mid(\alpha\overset{*}{\Rightarrow}u\beta,\ \|u\|=k)\ \text{または}\ (\alpha\overset{*}{\Rightarrow}u,\ \|u\|<k)\}$$

ただし，$\alpha,\beta\in(N\cup\Sigma)^*$, $u\in\Sigma^*$, $\|u\|$ は u の長さを表す．また，$\overset{*}{\Rightarrow}$ は，生成規則を 0 回以上使用する最左導出である．なお，本論文で扱う導出は，とくに断りがない限り，すべて最左導出である．

［定義 3］

文脈自由文法 $G=(N,\Sigma,P,S)$ において，$S\overset{*}{\Rightarrow}u_iA\xi_i$ のとき集合 $\mathrm{FOLLOW}_k(A)$ は以下で定義される．

$$\mathrm{FOLLOW}_k(A)=\cup_i\mathrm{FIRST}(\xi_i)$$

［定義 4］[1)]

L_1,L_2 を Σ^* の部分集合とするとき，演算子 $\oplus_k$ は以下で定義される．

$$L_1\oplus_kL_2=\{w\mid\text{ある } x\in L_1,\ y\in L_2 \text{ に対して，}\ \|xy\|\leqq k \text{ のとき } w=xy,\ \|xy\|>k \text{ のとき } w=u,\ \text{ただし } xy=uv \text{ で，} \|u\|=k\}$$

2.2 準 LL(k) 文法

準 LL(k) 文法の定義ならびにそれに関連したいくつかの定義，定理を以下に述べる．

［定義 5］

文脈自由文法 $G=(N,\Sigma,P,S)$ において $S\overset{*}{\Rightarrow}u_iAX\xi_i$ のとき，PF (partial-FOLLOW) は以下で定義される．

$$PF_k(A,X)=\cup_i\mathrm{FIRST}_k(X\xi_i)$$

［定義 6］

文脈自由文法 $G=(N,\Sigma,P,S)$ において，$A\to\alpha$, $A\to\beta$ を相異なる生成規則とするとき

$$S\overset{*}{\Rightarrow}uAXv$$

となる任意の $X\in N\cup\Sigma$ に対して

$$(\mathrm{FIRST}_k(\alpha)\oplus_kPF_k(A,X))\cap(\mathrm{FIRST}_k(\beta)\oplus_kPF_k(A,X))=\phi$$

が成り立つとき，G は準 LL(k) 文法であるという．

［定義 7］

文脈自由文法 $G=(N,\Sigma,P,S)$ の以下の 2 つの最左導出

- $S\overset{*}{\Rightarrow}uA\alpha\Rightarrow u\beta\alpha\overset{*}{\Rightarrow}ux$

• $S \overset{*}{\Rightarrow} uA\alpha \Rightarrow u\gamma\alpha \overset{*}{\Rightarrow} uy$

において，ある正整数 k に対して $\mathrm{FIRST}_k(x)=\mathrm{FIRST}_k(y)$ ならば $\beta=\gamma$ であるとき，G は LL(k) 文法であるという．

[定義7] より以下の定理がえられることが知られている．

[定理1][1]

文脈自由文法 $G=(N, \Sigma, P, S)$ が LL(k) であるとき，以下の条件が成り立つ．

$A\to\alpha$, $A\to\beta$ を相異なる生成規則とするとき

$S \overset{*}{\Rightarrow} uA\nu$

に対して

$$(\mathrm{FIRST}_k(\alpha)\oplus_k\mathrm{FIRST}_k(\nu))\cap(\mathrm{FIRST}_k(\beta)\oplus_k\mathrm{FIRST}_k(\nu))=\phi$$

ただし，$u\in\Sigma^*$, $A\in N$, $\nu\in(N\cup\Sigma)^*$

[定義8]

文脈自由文法 $G=(N, \Sigma, P, S)$ において $A\to\alpha$, $A\to\beta$ を相異なる生成規則とするとき

$$(\mathrm{FIRST}_k(\alpha)\oplus_k\mathrm{FOLLOW}_k(A))\cap(\mathrm{FIRST}_k(\beta)\oplus_k\mathrm{FOLLOW}_k(A))=\phi$$

が成り立つとき，G は強 LL(k) 文法であるという．

2.3 LL(k)，強 LL(k) 文法との包含関係

準 LL(k) 文法と，LL(k)，強 LL(k) 文法との包含関係についての定理とその証明を以下に述べる．

[定理2]

LL(k) 文法，準 LL(k) 文法，強 LL(k) 文法のクラスの間で，以下の包含関係が成り立つ．

LL(k)⊃ 準 LL(k)⊃ 強 LL(k)

とくに，$k\geqq 2$ の場合には

LL(k)⊋ 準 LL(k)⊋ 強 LL(k)

が成り立つ．

[証明]

LL(k)⊋ 準 LL(k)

$A\to\alpha$, $A\to\beta$ を相異なる生成規則とするとき

$S \overset{*}{\Rightarrow} uAX\xi$

なる導出に対して，準 LL(k) 文法の条件は PF の定義より

$$\begin{aligned}&(\mathrm{FIRST}_k(\alpha)\oplus_k PF_k(A, X))\cap(\mathrm{FIRST}_k(\beta)\oplus_k PF_k(A, X))\\&=(\mathrm{FIRST}_k(\alpha)\oplus_k(\cup_i\mathrm{FIRST}_k(X\xi_i)))\cap(\mathrm{FIRST}_k(\beta)\oplus_k(\cup_j\mathrm{FIRST}_k(X\xi_j)))\\&=\cup_{i,j}((\mathrm{FIRST}_k(\alpha)\oplus_k\mathrm{FIRST}_k(X\xi_i))\cap(\mathrm{FIRST}_k(\beta)\oplus_k\mathrm{FIRST}_k(X\xi_j)))=\phi\end{aligned}$$

となる．ゆえに，すべての i, j について

$$(\mathrm{FIRST}_k(\alpha)\oplus_k\mathrm{FIRST}_k(X\xi_i))\cap(\mathrm{FIRST}_k(\beta)\oplus_k\mathrm{FIRST}_k(X\xi_j))=\phi$$

が成り立つ．とくに，$i=j$, $X\xi_i=X\xi_j=\nu$ とおくと，上式は LL(k) 文法の成立条件 [定理1] を満足する．

∴ LL(k)⊋ 準 LL(k)

しかるに，$k\geqq 2$ の場合は [例1] に示すように，準 LL(k) でない LL(k) 文法が存在する．ゆえに，$k\geqq 2$ の場合には

LL(k)⊋ 準 LL(k)

である．

準 LL(k)⊋ 強 LL(k)

$S \overset{*}{\Rightarrow} u_iA\nu_i$ のとき FOLLOW の定義より

$\mathrm{FOLLOW}_k(A)=\cup_i\mathrm{FIRST}_k(\nu_i)$

であるから，強 LL(k) 文法の条件は

$$\begin{aligned}&(\mathrm{FIRST}_k(\alpha)\oplus_k\mathrm{FOLLOW}_k(A))\cap(\mathrm{FIRST}_k(\beta)\oplus_k\mathrm{FOLLOW}_k(A))\\&=(\mathrm{FIRST}_k(\alpha)\oplus_k(\cup_i\mathrm{FIRST}_k(\nu_i)))\cap(\mathrm{FIRST}_k(\beta)\oplus_k(\cup_j\mathrm{FIRST}_k(\nu_j)))\\&=\cup_{i,j}((\mathrm{FIRST}_k(\alpha)\oplus_k\mathrm{FIRST}_k(\nu_i))\cap(\mathrm{FIRST}_k(\beta)\oplus_k\mathrm{FIRST}_k(\nu_j)))=\phi\end{aligned}$$

となる．ゆえに，すべての i, j に対して

$$(\mathrm{FIRST}_k(\alpha)\oplus_k\mathrm{FIRST}_k(\nu_i))\cap(\mathrm{FIRST}_k(\beta)\oplus_k\mathrm{FIRST}_k(\nu_j))=\phi$$

でなければならない．とくに，$\nu_i=X\xi_i$, $\nu_j=X\xi_j$ の場合を考えると

$$(\mathrm{FIRST}_k(\alpha)\oplus_k(\cup_i\mathrm{FIRST}_k(X\xi_i)))\cap(\mathrm{FIRST}_k(\beta)\oplus_k(\cup_j\mathrm{FIRST}_k(X\xi_j)))=\phi$$

よって，可能な i, j について和集合をとっても

$$\cup_{i,j}(\mathrm{FIRST}_k(\alpha)\oplus_k\mathrm{FIRST}_k(X\xi_i))\cap(\mathrm{FIRST}_k(\beta)\oplus_k\mathrm{FIRST}_k(X\xi_j))=\phi$$

となる．この式は準 LL(k) 文法の成立条件である．

∴ 準 LL(k)⊋強 LL(k)

しかるに，$k\geqq 2$ の場合は [例2] に示すように強 LL(k) でない準 LL(k) 文法が存在する．ゆえに，$k\geqq 2$ の場合には

準 LL(k)⊋強 LL(k)

である． [証明終]

[例1] 次の文法 G_{1m} ($m\geqq 2$) は，$k=m$ なる k に対して，LL(k) ではあるが準 LL(k) ではない．

G_{1m}:
1. $S\to aY_1Xa^m$
2. $S\to bY_2Xb^{m-1}a$
3. $Y_1\to aA$
4. $Y_2\to bA$
5. $A\to b^{m-1}$
6. $A\to\varepsilon$
7. $X\to\varepsilon$

[証明]

G_{1m} の A を含む最左文型は $aaAXa^m$ と $bbAXb^{m-1}a$ のみである．この2つの文型に対して，$k=m$ の場合に，LL(k) と準 LL(k) の成立条件を適用してみる．

（i） $S\overset{*}{\Rightarrow}aaAXa^m$, $A\to b^{m-1}$, $A\to\varepsilon$ に対して，$\alpha=b^{m-1}$, $\beta=\varepsilon$, $\nu=Xa^m$ とすると，$k=m$ のとき

$$(\mathrm{FIRST}_k(\alpha)\oplus_k\mathrm{FIRST}_k(\nu))\cap(\mathrm{FIRST}_k(\beta)\oplus_k\mathrm{FIRST}_k(\nu))$$
$$=(\{b^{k-1}\}\oplus_k\{a^k\})\cap(\{\varepsilon\}\oplus_k\{a^k\})$$
$$=\{b^{k-1}a\}\cap\{a^k\}$$
$$=\phi$$

$S\overset{*}{\Rightarrow}bbAXb^{m-1}a$, $A\to b^{m-1}$, $A\to\varepsilon$ に対して同様の対応づけをすると

$$(\{b^{k-1}\}\oplus_k\{b^{k-1}a\})\cap(\{\varepsilon\}\oplus_k\{b^{k-1}a\})$$
$$=\{b^k\}\cap\{b^{k-1}a\}=\phi$$

ゆえに，G_{1m} は $k=m$ なる k に対して，LL(k) である．

（ii） $S\overset{*}{\Rightarrow}aaAXa^m$, $S\overset{*}{\Rightarrow}bbAXb^{m-1}a$ より

$$PF_k(A,X)=\{a^k,b^{k-1}a\}$$

したがって，$A\to b^{m-1}$, $A\to\varepsilon$ に対して，$\alpha=b^{m-1}$, $\beta=\varepsilon$ とすると，$k=m$ のとき

$$(\mathrm{FIRST}_k(\alpha)\oplus_kPF_k(A,X))\cap(\mathrm{FIRST}_k(\beta)\oplus_kPF_k(A,X))$$
$$=(\{b^{k-1}\}\oplus_k\{a^k,b^{k-1}a\})\cap(\{\varepsilon\}\oplus_k\{a^k,b^{k-1}a\})$$
$$=\{b^{k-1}a,b^k\}\cap\{a^k,b^{k-1}a\}\neq\phi$$

ゆえに，G_{1m} は準 LL(k) ではない．したがって，G_{1m} は $k=m$ なる k に対して，LL(k) ではあるが準 LL(k) ではない．

[例2] 次の文法 G_{2m} ($m\geqq2$) は，$k=m$ なる k に対して，準 LL(k) ではあるが強 LL(k) ではない．

G_{2m}: 1. $S\to aAXa^m$　　4. $A\to\varepsilon$
2. $S\to bAb^{m-1}a$　　5. $X\to\varepsilon$
3. $A\to b^{m-1}$

[証明]

A を含む最左文型は $aAXa^m$ と $bAb^{m-1}a$ のみである．この2つの文型に準 LL(k) と強 LL(k) の成立条件を適用してみる．

（i） 準 LL(k) の成立条件を調べると，$S\overset{*}{\Rightarrow}aAXa^m$ より

$$PF_k(A,X)=\{a^k\}$$

したがって，$S\overset{*}{\Rightarrow}aAXa^m$, $A\to b^{m-1}$, $A\to\varepsilon$ に対して

$$(\{b^{k-1}\}\oplus_k\{a^k\})\cap(\{\varepsilon\}\oplus_k\{a^k\})$$
$$=\{b^{k-1}a\}\cap\{a^k\}$$
$$=\phi$$

また，$S\overset{*}{\Rightarrow}bAb^{m-1}a$ より

$$PF_k(A,b)=\{b^{k-1}a\}$$

ゆえに，$S\overset{*}{\Rightarrow}bAb^{m-1}a$, $A\to b^{m-1}$, $A\to\varepsilon$ に対して

$$(\{b^{k-1}\}\oplus_k\{b^{k-1}a\})\cap(\{\varepsilon\}\oplus_k\{b^{k-1}a\})$$
$$=\{b^k\}\cap\{b^{k-1}a\}=\phi$$

ゆえに，G_{2m} は $k=m$ なる k に対して，準 LL(k) である．

（ii） 次に，強 LL(k) の成立条件を調べると，$S\overset{*}{\Rightarrow}aAXa^m$, $S\overset{*}{\Rightarrow}bAb^{m-1}a$ より

$$\mathrm{FOLLOW}_k(A)=\{a^k,b^{k-1}a\}$$

したがって，$A\to b^{m-1}$, $A\to\varepsilon$ に対して

$$(\mathrm{FIRST}_k(\alpha)\oplus_k\mathrm{FOLLOW}_k(A))\cap(\mathrm{FIRST}_k(\beta)\oplus_k\mathrm{FOLLOW}_k(A))$$
$$=(\{b^{k-1}\}\oplus_k\{a^k,b^{k-1}a\})\cap(\{\varepsilon\}\oplus_k\{a^k,b^{k-1}a\})$$
$$=\{b^{k-1}a,b^k\}\cap\{a^k,b^{k-1}a\}\neq\phi$$

ゆえに，G_{2m} は強 LL(k) ではない．したがって，G_{2m} は $k=m$ なる k に対して準 LL(k) ではあるが，強 LL(k) ではない．

[系]

文脈自由文法 $G=(N,\Sigma,P,S)$ に対して，以下の関係が成り立つ．

LL(1)↔準 LL(1)↔強 LL(1)

[証明]

[定理2] より，LL(k)，準 LL(k)，強 LL(k) の各クラスの間で，$k\geqq1$ に対して

$$\text{LL}(k)\supset\text{準 LL}(k)\supset\text{強 LL}(k)\qquad(2.1)$$

が成り立つ．しかるに，$k=1$ の場合は

$$\text{LL}(1)\leftrightarrow\text{強 LL}(1)\qquad(2.2)$$

である[1]．そこで(2.1)，(2.2)より

LL(1)↔準 LL(1)↔強 LL(1)

が成り立つ．　[証明終]

3. 解 析 表

3.1 基本的な定義

前章では，準 LL(k) 文法を提案した．ここでは準 LL(2) 文法に対する解析表 T の構造とその性質について述べるのに必要ないくつかの定義を与える．

[定義9]

文脈自由文法 G を $G=(N,\Sigma,P,S)$ とするとき，G を用いて次の拡大文法 G' を定義する．

$$G'=(N',\Sigma',P',S'),\quad N'=N\cup\{S'\},$$
$$\Sigma'=\Sigma\cup\{\$\},\quad P'=P\cup\{S'\to S\$\$\}$$

以下では，この拡大文法 G' を用いて議論を展開する

が，便宜上，N'，Σ'，P' をそれぞれ N，Σ，P と読みかえるものとする．

［記法 3］

生成規則ならびに導出

$A\underset{p}{\rightarrow}\alpha$，$uA\alpha\underset{q}{\Rightarrow}u\beta\alpha$

における p，q は，それぞれ生成規則 $A\rightarrow\alpha$ につけられた固有の番号，ならびに最左導出 $S'\fallingdotseq uA\alpha\Rightarrow u\beta\alpha$ で用いられた生成規則 $A\rightarrow\beta$ につけられた固有の番号を示す．

［記法 4］

記号列 $\gamma\in(N\cup\Sigma)^+$ の左から i 番目の記号を，$(i)\gamma$ で表す．

［定義 10］

生成規則の番号を p で表すとき［ ］p または［X］p を τ 型生成規則番号という．ここで X は，最左導出 $S'\fallingdotseq uA\alpha\underset{p}{\Rightarrow}u\beta\alpha$ における α の最左端一記号を表す．このとき［ ］p はそれまでの解析（つまり，文脈）に関係なく A に対して p が適用可能であることを表す．一方，［X］p は最左文型上で現在導出の対象となっている非終端記号 A のすぐ右に隣接している α の先頭が X のときに限り，A に対して番号 p をもつ生成規則が適用可能であることを表す．つまり，［X］p のように［ ］内が空でないときは文脈に関連して生成規則 p が用いられることを示す．

［定義 11］

解析表 T は行，列がそれぞれ $(N-\{S'\})\cup\Sigma$，Σ の要素によって名付けられたマトリクスとする．解析表 T の A 行，a 列の要素を $T(A,a)$，また a 行，b 列の要素を $T(a,b)$ で表し，これら各要素には τ 型生成規則番号の集合が記入される．

3.2 解析表の性質

準 LL(2) の定義より，解析表 T は以下の性質をもっていなければならない．このような性質をもつ解析表の作成方法はすでに述べたように，紙数の都合上，別論文[4]で述べる．ここで，$T(A,a)$，$T(a,b)$ はそれぞれ［定義 11］に従うものとする．

［性質 1］

$S'\fallingdotseq uA\alpha\underset{p}{\Rightarrow}u\beta\alpha\fallingdotseq uab\gamma\alpha$

$\longleftrightarrow$［ ］$p\in T(A,a)$ かつ［ ］$p\in T(a,b)$

［説明］

この性質は，A から長さ 2 以上の終端記号列を導出できるので，A の PF の助けを必要としない場合を表す．つまり，$A\underset{p}{\Rightarrow}\beta\fallingdotseq ab\gamma$ のとき，表の $T(A,a)$ および $T(a,b)$ のエントリに対して，［ ］p が記入されることを意味する．逆に，［ ］$p\in T(A,a)$ かつ［ ］$p\in T(a,b)$ であれば，導出 $A\underset{p}{\Rightarrow}\beta\fallingdotseq ab\gamma$ が存在する．

［性質 2］

$S'\fallingdotseq uAX\alpha\underset{p}{\Rightarrow}u\beta X\alpha\fallingdotseq uaX\alpha\fallingdotseq uab\delta$

$\longleftrightarrow$［ ］$p\in T(A,a)$ かつ［X］$p\in T(a,b)$

［説明］

この性質は，A から長さ 1 の終端記号しか導出されず，A の PF_1 の助けを必要とする場合で，このときは文脈を問題としなければならない．いま，A を含む文型 $uAX\alpha$ があって，$AX\alpha\fallingdotseq aX\alpha\fallingdotseq ab\delta$ であるとする．このとき，解析表の中で

［ ］$p\in T(A,a)$ かつ［X］$p\in T(a,b)$

としていることは，以下のことを意味する．つまり，［ ］$p\in T(A,a)$ は A から a を導出するのに最初に用いる生成規則が p であることを示し，［X］$p\in T(a,b)$ は a につづいて b が導出できるのは A のすぐ右となりに X がきている場合であることを示す．

［性質 3］

$S'\fallingdotseq uAX\alpha\underset{p}{\Rightarrow}u\beta X\alpha\fallingdotseq uX\alpha\fallingdotseq uab\gamma$

$\longleftrightarrow$［X］$p\in T(A,a)$ かつ［X］$p\in T(a,b)$

［説明］

この性質は，A が $A\fallingdotseq\varepsilon$ となって終端記号を 1 つも導出しないため，A の PF_2 の助けにより 2 つの終端記号を求める場合である．

［性質 4］

解析表の要素への記入は，前記 3 つの場合以外にはない．

［説明］

この性質は，その他の組合せ，すなわち $T(A,a)$ もしくは $T(a,b)$ のいずれか一方のみの記入，および［X］$p\in T(A,a)$ かつ［ ］$p\in T(a,b)$ となる記入は生じないことを意味する．

4. 解析アルゴリズム

4.1 予備的事項

アルゴリズムの説明に必要な二，三の事柄について述べる．

［定義 12］

任意の $A\in N$，$a,b\in\Sigma$，解析表要素 $T(X,Y)$ に対して，集合演算子 $\sqcap$ を以下のように定義する．

$T(A,a)\sqcap T(a,b)=U$

とするとき，

① (［ ］$p\in T(A,a)$)$\wedge$(［ ］$p\in T(a,b)$) なら，

［ ］$p\in U$

② $([\ ]p\in T(A,a))\wedge([X]p\in T(a,b))$ なら，
$[X]p\in U$

③ $([X]p\in T(A,a))\wedge([X]p\in T(a,b))$ なら，
$[X]p\in U$

とする．

④ ①，②，③の場合がなければ，U は空集合とする．

[定理3]

演算 $\sqcap$ により得られる集合 U が2つ以上の要素を含むとき，文法が準 LL(2) であるとすれば要素のいずれの2つをとり出しても以下の組合せのものは起こりえない．

(i) $[\ ]p,[\ ]q,\ p\neq q$

(ii) $[\ ]p,[X]q,\ p\neq q$

(iii) $[X]p,[X]q,\ p\neq q$

[証明]

$U=T(A,a)\sqcap T(a,b)$

とする．

(i)の証明

(i)の場合が起こりうるとすれば [定義12] に関して①-①の組合せが成り立たねばならない．つまり，以下の組合せのときに限られる．

(イ) $[\ ]p\in T(A,a)$, $[\ ]p\in T(a,b)$ かつ
$[\ ]q\in T(A,a)$, $[\ ]q\in T(a,b)$

(イ)：①-①の組合せ．

この場合は [性質1] より

(1) $S'\doteqdot u_1A\gamma_1\underset{p}{\Rightarrow}u_1\alpha\gamma_1\doteqdot u_1ab\zeta\gamma_1$ かつ

(2) $S'\doteqdot u_2A\gamma_2\underset{q}{\Rightarrow}u_2\beta\gamma_2\doteqdot u_2ab\xi\gamma_2$

が成り立つ．そこで，$S'\doteqdot u_1A\gamma_1$, $A\to\alpha$, $A\to\beta$ に対して

$$\begin{aligned}&(\mathrm{FIRST}_2(\alpha)\oplus_2PF_2(A,X))\cap\\&\qquad(\mathrm{FIRST}_2(\beta)\oplus_2PF_2(A,X))\\&=(\{ab,\cdots\}\oplus_2\{\cdots\})\cap(\{ab,\cdots\}\oplus_2\{\cdots\})\\&=\{ab,\cdots\}\cap\{ab,\cdots\}\\&\neq\phi\qquad\text{ただし，}X=^{(1)}\gamma_1\end{aligned}$$

である．また，$S'\doteqdot u_2A\gamma_2$, $A\to\alpha$, $A\to\beta$ に対しても同様の結果をうる．ところが $p\neq q$ であるから，これは準 LL(2) の定義に反する．ゆえに，(イ)つまり①-①の組合せは生じないので，(i)は起こりえない．

(ii)の証明

(ii)の場合が起こりうるとすれば [定義12] に関して①-②または①-③の組合せが成り立たねばならない．そこで，以下の組合せが得られる．

(イ) $[\ ]p\in T(A,a)$, $[\ ]p\in T(a,b)$ かつ
$[\ ]q\in T(A,a)$, $[X]q\in T(a,b)$

(ロ) $[\ ]p\in T(A,a)$, $[\ ]p\in T(a,b)$ かつ
$[X]q\in T(A,a)$, $[X]q\in T(a,b)$

(イ)：①-②の組合せ．

この場合は [性質1]，[性質2] より

(1) $S'\doteqdot u_1A\gamma_1\underset{p}{\Rightarrow}u_1\alpha\gamma_1\doteqdot u_1ab\zeta\gamma_1$ かつ

(2) $S'\doteqdot u_2A\gamma_2\underset{q}{\Rightarrow}u_2\beta\gamma_2\doteqdot u_2a\gamma_2\doteqdot u_2ab\gamma_2'$

が成り立つ．導出(2)より $bc\in PF_2(A,X)$ である．ただし，$bc\in\mathrm{FIRST}_2(\gamma_2)$, $X=^{(1)}\gamma_2$. ゆえに，$S'\doteqdot u_2A\gamma_2$, $A\to\alpha$, $A\to\beta$ に対して

$$\begin{aligned}&(\mathrm{FIRST}_2(\alpha)\oplus_2PF_2(A,X))\cap\\&\qquad(\mathrm{FIRST}_2(\beta)\oplus_2PF_2(A,X))\\&=(\{ab,\cdots\}\oplus_2\{bc,\cdots\})\cap\\&\qquad(\{a,\cdots\}\oplus_2\{bc,\cdots\})\\&=\{ab,\cdots\}\cap\{ab,\cdots\}\\&\neq\phi\end{aligned}$$

である．ところが $p\neq q$ であるから，これは準 LL(2) 文法の定義に反する．ゆえに，(イ)つまり①-②の組合せは生じない．

(ロ)：①-③の組合せ．

この場合も同様に

(1) $S'\doteqdot u_1A\gamma_1\underset{p}{\Rightarrow}u_1\alpha\gamma_1\doteqdot u_1ab\zeta\gamma_1$ かつ

(2) $S'\doteqdot u_2A\gamma_2\underset{q}{\Rightarrow}u_2\beta\gamma_2\doteqdot u_2\gamma_2\doteqdot u_2ab\gamma_2'$

が成り立つ．導出(2)より $ab\in PF_2(A,X)$ である．ただし，$ab\in\mathrm{FIRST}_2(\gamma_2)$, $X=^{(1)}\gamma_2$. ゆえに，$S'\doteqdot u_2A\gamma_2$, $A\to\alpha$, $A\to\beta$ に対して

$$\begin{aligned}&(\mathrm{FIRST}_2(\alpha)\oplus_2PF_2(A,X))\cap\\&\qquad(\mathrm{FIRST}_2(\beta)\oplus_2PF_2(A,X))\\&=(\{ab,\cdots\}\oplus_2\{ab,\cdots\})\cap\\&\qquad(\{\varepsilon,\cdots\}\oplus_2\{ab,\cdots\})\\&=\{ab,\cdots\}\cap\{ab,\cdots\}\\&\neq\phi\end{aligned}$$

である．ところが $p\neq q$ であるから，これは準 LL(2) 文法の定義に反する．ゆえに，(ロ)つまり①-③の組合せは生じない．したがって，(ii)が生じるための組合せはすべて生じないので，(ii)は起こりえない．

(iii)の証明

(iii)の場合が起こりうるとすれば [定義12] に関して②-③または③-③の組合せが成り立たねばならない．このとき，以下の組合せが得られる．

(イ) $[\ ]p\in T(A,a)$, $[X]p\in T(a,b)$ かつ
$[X]q\in T(A,a)$, $[X]q\in T(a,b)$

(ロ) $[X]p\in T(A,a)$, $[X]p\in T(a,b)$ かつ

$[X]q \in T(A, a)$, $[X]q \in T(a, b)$

(イ)：②-③の組合せ．

この場合は［性質2］，［性質3］より

(1) $S' \fallingdotseq u_1 A \gamma_1 \underset{p}{\Rightarrow} u_1 \alpha \gamma_1 \fallingdotseq u_1 a \gamma_1 \fallingdotseq u_1 ab \gamma_1'$ かつ

(2) $S' \fallingdotseq u_2 A \gamma_2 \underset{q}{\Rightarrow} u_2 \beta \gamma_2 \fallingdotseq u_2 \gamma_2 \fallingdotseq u_2 ab \gamma_2'$

が成り立つ．導出(1)，(2)より

$bc \in PF_2(A, X)$ かつ $ab \in PF_2(A, X)$

ただし，$c \in \mathrm{FIRST}_1(\gamma_1')$，$X = {}^{(1)}\gamma_1 = {}^{(1)}\gamma_2$

であるから，$S' \fallingdotseq u_1 A \gamma_1$（または $S' \fallingdotseq u_2 A \gamma_2$），$A \to \alpha$，$A \to \beta$ に対して

$$
\begin{aligned}
&(\mathrm{FIRST}_2(\alpha) \oplus_2 PF_2(A, X)) \cap \\
&\qquad (\mathrm{FIRST}_2(\beta) \oplus_2 PF_2(A, X)) \\
&= (\{a, \cdots\} \oplus_2 \{ab, bc, \cdots\}) \cap \\
&\qquad (\{\varepsilon, \cdots\} \oplus_2 \{ab, bc, \cdots\}) \\
&= \{aa, ab, \cdots\} \cap \{ab, bc, \cdots\} \\
&\neq \phi
\end{aligned}
$$

となって，文法は準 LL(2) の定義を満足しない．ゆえに，(イ)つまり②-③の組合せは生じない．

(ロ)：③-③の組合せ．

この場合は，［性質3］より $A \fallingdotseq \varepsilon$ となる A-生成規則 p, q $(p \neq q)$ が同時に存在することになる．これは準 LL 文法の定義を満足しないことは明らかである．ゆえに，(ロ)つまり③-③の組合せは生じない．したがって，(ⅲ)が生じるための組合せはすべて生じないので，(ⅲ)は起こりえない． ［証明終］

4.2 解析アルゴリズム

本論文で提案する解析アルゴリズムの基本的な考えを述べる前に，この解析アルゴリズムで用いるスタックならびに領域について述べる．

(1) 解析スタック R：解析の経過を記憶するためのスタックで，初期値は $S\$\$$ である．

(2) 入力テキスト記憶用領域 M：プログラム・テキストを記憶するための領域．プログラム・テキストの終りには $\$\$$ が付加される．

(3) $\mathrm{CURRENT}_1$, $\mathrm{CURRENT}_2$：入力テキスト M の目下解析の対象となっている要素，ならびにその右隣りの要素を格納するための領域．

(4) TOP, NEXT：スタック R の一番上の要素，ならびに2番目の要素を表す．

(5) i：入力テキストの i 番目の要素 $M(i)$ を表すための整数型変数．

(6) Y：生成規則番号 p を保存するための整数型変数．

また，解析上必要とする固有の命令には以下のものがある．

(i) POP：解析スタック R より，TOP を1回に1個取り出す．したがって，NEXT が先頭になる．

(ii) PUSH(p)：解析スタック R に生成規則番号 p の右辺全体を，右辺の左端がスタック R の TOP になるように押し込む．

解析の手続きとその意味づけは以下のとおりである．

(1) $\mathrm{TOP} = \mathrm{CURRENT}_1$，かつ $\mathrm{NEXT} = \mathrm{CURRENT}_2$ ならば，POP を2度行い，M のポインタを右へ2つずらす．

(2) $\mathrm{TOP} = \mathrm{CURRENT}_1$ で $\mathrm{NEXT} \neq \mathrm{CURRENT}_2$ ならば POP し，M のポイントを右へ1つずらす．

(3) $\mathrm{TOP} \neq \mathrm{CURRENT}_1$ で TOP が終端記号ならばテキスト・エラー．

(4) TOP が非終端記号ならば集合 U を求める．

$$U = T(\mathrm{TOP}, \mathrm{CURRENT}_1) \cap T(\mathrm{CURRENT}_1, \mathrm{CURRENT}_2)$$

(イ) $|U| = 0$ のとき，つまり $U = \phi$ のときは TOP の非終端記号に適合する生成規則がないことを意味しているからテキスト・エラー．

(ロ) $|U| = 1$ のとき，もし $U = \{[\ \ ]p\}$ であれば NEXT の記号の如何によらず p を適用すればよいことを意味するから生成規則番号 p を選択し，POP し，PUSH(p) を行う．また，$U = \{[X]p\}$ であれば NEXT の記号の次第により p を選択するかどうかをきめることを意味するから，NEXT と X を比較し $\mathrm{NEXT} = X$ であれば POP し，PUSH(p) を行う．$\mathrm{NEXT} \neq X$ であればテキスト・エラー．

(ハ) $|U| \geqq 2$ のとき

① $[\ \ ]p \in U$ のとき．この場合は［定理3］の①および②より $U = \{[\ \ ]p, [X_1]p, [X_2]p, \cdots, [X_n]p\}$ $(n \geqq 1)$ に限られる．この場合は，次の3つのいずれかの場合を意味する．

$S' \fallingdotseq u_1 A \gamma_1 \underset{p}{\Rightarrow} u_1 \alpha \gamma_1 \fallingdotseq u_1 ab\zeta\gamma_1$

$S' \fallingdotseq u_2 A \gamma_2 \underset{p}{\Rightarrow} u_2 \alpha \gamma_2 \fallingdotseq u_2 a \gamma_2 \Rightarrow u_2 ab \gamma_2'$

$S' \fallingdotseq u_3 A \gamma_3 \underset{p}{\Rightarrow} u_3 \alpha \gamma_3 \fallingdotseq u_3 \gamma_3 \fallingdotseq u_3 ab \gamma_3'$

しかし，いずれにしても $A \underset{p}{\to} \alpha$ の適用から

導出は始められるので，生成規則番号 p を選択し，POP し，PUSH(p) を行う．

② [ ]$p \in U$ のとき．この場合は [定理3] の③より，$U=\{[X_1]p_1, [X_2]p_2, \cdots, [X_n]p_n\}$ ($n \geqq 2$, $X_i \neq X_j$, $1 \leqq i, j \leqq n$) である．この場合はもし NEXT$=X_i$ であれば TOP の非終端記号に対して生成規則 p_i が排他的に適用可能であることを意味するから，NEXT$=X_i$ を満足する p_i を選択し，POPし，PUSH(p_i) を行う．

③ ①，②のいずれをも満足しないときは明らかにテキスト・エラー ([定理3])．

さらに，[性質1] から [性質4] のことを配慮すると，上記の手続きを実現する解析アルゴリズムは以下のようになる．

Parsing Algorithm

```
begin
/* M(i)はテキストMの i 番目の要素を示す */
  R←'S$$'; /* スタックの初期化 */
  M←text'$$';/* Mの初期化 */
  i←1;
  CURRENT1←M(i);
  CURRENT2←M(i+1);
  repeat
    if TOP=CURRENT1 and NEXT
                          =CURRENT2 then
      begin
        POP; POP;
        i←i+2;
        CURRENT1←M(i);
        CURRENT2←M(i+1);
      end
    else
      if TOP=CURRENT1 then
        begin
          POP;
          i←i+1;
          CURRENT1←CURRENT2;
          CURRENT2←M(i+1);
        end
      else
        if TOP∈Σ then text error
        else
          begin
            find U such that U=T(TOP,
                CURRENT1)∩T(CURRENT1,
                              CURRENT2);
            case |U|=0: text error;
                 |U|=1: if there exists [ ]p in
                           U then select p and
                                      Y←'p'
                        else
                          if there exists
                          [NEXT]p in U then
                            select p and Y←'p'
                          else text error;
                 |U|≧2: if there exists [ ]p in
                           U then select p and
                                      Y←'p'
                        else
                          if there exists
                          [NEXT]p1 in U then
                           select p1 and Y←'p1'
                          else text error;
            end of case;
            POP; PUSH(Y);
          end
    until TOP='$' and CURRENT1='$'
end.
```

[End of Algorithm]

5. 適 用 例

準 LL(2) 文法により生成される言語に対して，上記の解析アルゴリズムを適用する．いま，解析の状態を

(入力列，スタックの状態，解析列)

で表し，初期状態を

(入力列 \$\$, S\$\$, ε)

で表す．解析列とは，解析に用いられた生成規則番号の並びで，その並びの順序は解析に使用された順に従う．解析がある状態 (K_i, Π_i, τ_i) から，次の状態 (K_{i+1}, Π_{i+1}, τ_{i+1}) に推移したとき，これを記号$\vdash$を用いて

$$(K_i, \Pi_i, \tau_i) \vdash (K_{i+1}, \Pi_{i+1}, \tau_{i+1})$$

で表す．また，解析終了時は

(\$\$, \$\$, τ)

である．なお，以下の例題中の $A, B, \cdots, a, b, \cdots$ 等は，4章までの議論の展開に使用したそれらと異なり，それぞれの例題に固有の文法記号を意味する．

[例 3][1] 強である準 LL(2) 文法

$G_3=\{\{S, A\}, \{a, b\}, P, S\}$

1. $S\rightarrow\varepsilon$
2. $S\rightarrow abA$
3. $A\rightarrow Saa$
4. $A\rightarrow b$

文法 G_3 に対する解析表は，3.2 節の性質に従って**図 1** で与えられる．文法 G_3 が生成する言語 $L(G_3)$ は

$$L(G_3)=\{\varepsilon, (ab)^n(aa)^n, (ab)^nb(aa)^{n-1}\} \quad (n\geqq 1)$$

である．この中から，二，三を選んで前出の解析アルゴリズムに従って解析する．

[例 3a] 正しいテキスト *abaa* に対する解析

まず，入力のはじめの頭 2 つ *ab* を考えると，図 1 の解析表 T より

$T(S, a)=\{[\ \ ]2, [a]1\}$

$T(a, b)=\{[\ \ ]2, [\ \ ]3\}$

$\therefore\ T(S, a)\sqcap T(a, b)=\{[\ \ ]2\}$

であるから POP を行い，生成規則番号 2 を選択し PUSH(2) を行う．つまり，スタック R の S を生成規則番号 2 の右辺でおきかえる．この手順を前述の記法にならうと以下のように表される．

$(abaa\$\$, S\$\$, \varepsilon)\vdash(abaa\$\$, abA\$\$, 2)$

次に，入力列とスタックを調べると，TOP=CURRENT$_1$=$\{a\}$ かつ NEXT=CURRENT$_2$=$\{b\}$ であるから，POP を 2 度行うことにより a と b をスタック R より降ろし，M のポインタを右へ 2 つずらす．以下，同様にして上の記法に従ってはじめから書くと下記のようになる．

$(abaa\$\$, S\$\$, \varepsilon)\vdash(abaa\$\$, abA\$\$, 2)$

$\vdash(aa\$\$, A\$\$, 2)$

$\vdash(aa\$\$, Saa\$\$, 23)$

$\vdash(aa\$\$, aa\$\$, 231)$

$\vdash(\$\$, \$\$, 231)$

[例 3b] 正しいテキスト ε に対する解析

この場合，テキストが ε であるから，M の初期値は \$\$ となる．したがって，

$T(S, \$)=\{[\$]1\}$

$T(\$, \$)=\{[\$]1\}$

$\therefore\ T(S, \$)\sqcap T(\$, \$)=\{[\$]1\}$

$(\varepsilon\$\$, S\$\$, \varepsilon)\vdash(\$\$, \$\$, 1)$

[例 3c] 誤ったテキスト *ab* に対する解析

$(ab\$\$, S\$\$, \varepsilon)\vdash(ab\$\$, abA\$\$, 2)$

$\vdash(\$\$, A\$\$, 2)$

$T(A, \$)=\phi$ なのでテキスト・エラー．

[例 4][1] 強でない準 LL(2) 文法

$G_4=\{\{S, A\}, \{a, b\}, P, S\}$

1. $S\rightarrow aAaa$
2. $S\rightarrow bAba$
3. $A\rightarrow b$
4. $A\rightarrow\varepsilon$

文法 G_4 に対する解析表は，3.2 節の性質に従って**図 2** で与えられる．文法 G_4 が生成する言語 $L(G_4)$ は

$L(G_4)=\{aaa, abaa, bba, bbba\}$

である．この中から，二，三を選んで前出の解析アルゴリズムに従って解析する．

[例 4a] 正しいテキスト *abaa* に対する解析

$(abaa\$\$, S\$\$, \varepsilon)\vdash(abaa\$\$, aAaa\$\$, 1)$

$\vdash(baa\$\$, Aaa\$\$, 1)$

ここで $T(A, b)=\{[\ \ ]3, [b]4\}$，$T(b, a)=\{[a]3, [b]4\}$ であるから，$T(A, b)\sqcap T(b, a)=\{[a]3, [b]4\}$ となって選択すべき生成規則が一意に定まらないが，これは文法 G_4 が強 LL(2) でないことによる．そこで，NEXT を調べると NEXT=$\{a\}$ であるから，$[a]3$ を選択する．ゆえに解析は以下のようにつづく．

$\vdash(baa\$\$, baa\$\$, 13)$

$\vdash(a\$\$, a\$\$, 13)$

$\vdash(\$\$, \$\$, 13)$

[例 4b] 誤ったテキスト *aab* に対する解析

$(aab\$\$, S\$\$, \varepsilon)\vdash(aab\$\$, aAaa\$\$, 1)$

$\vdash(ab\$\$, Aaa\$\$, 1)$

	a	b	$
S	[ ]2 [a]1		[$]1
A	[ ]3	[ ]4	
a	[ ]3 [a]1	[ ]2 [ ]3	
b	[a]4		[$]4
$			[$]1

図 1 文法 G_3 に対する解析テーブル
Fig. 1 A parsing-table for grammar G_3.

	a	b	$
S	[ ]1	[ ]2	
A	[a]4	[ ]3 [b]4	
a	[ ]1 [a]4	[ ]1	
b	[a]3 [b]4	[ ]2 [b]3	
$			

図 2 文法 G_4 に対する解析テーブル
Fig. 2 A parsing-table for grammar G_4.

ここで，$T(A,a)=\{[a]4\}$，$T(a,b)=\{[\ \ ]1\}$ であるから

$\therefore\ T(A,a)\cap T(a,b)=\phi$

となるのでテキスト・エラー

6. 性能評価

筆者らの提案する解析アルゴリズムの性能を評価するため，Aho らの解析法と，以下の条件の下で比較検討した．

- 使用機種：SUN-3 モデル 60 M
- 使用 O/S および言語：UNIX/PASCAL
- 対象言語：PASCAL−[6] (pascal minus)

プログラムは，個人差をできるだけ少なくするため，すべて同一人が作成した．

記憶容量

筆者らと Aho らの解析表の構造を**図 3**（a），（b）に示す．ただし，計算機の記憶容量の関係で図 3（b）の斜線部については表から除外して，解析ルーチンの中で処理するようにした．

PASCAL− に対する解析表等の記憶容量の実測値を，**表 1** に示す．表中のⅡの部分は，解析表と生成規則表の大きさのみであるのに対し，Ⅰのほうはこの 2 つのほかに，解析時に用いられるいくつかのこまごまとした表（入力テキスト用，解析スタック，解析木用など）の領域がふくまれている．両者の差がそうした表のための領域である．表 1 から分かるように，コード部分は筆者らのほうが 4% 程度大きくなるが，プログラム全体に占める割合は半分以下であるので，あまり問題とならない．Ⅱの部分（解析表と生成規則表）は Aho らの約 1/120 程度になる．筆者らと Aho らの間にこれだけ大きな差が出たのは，主に次の 2 つに起因する．

（i）解析表の大きさは，筆者らが $\|N\cup\Sigma\|\times\|\Sigma\|$ ですむのに対し，Aho らは $\|\mathcal{T}\cup\Sigma\|\times\|(\Sigma-\{\$\})^2\cup\Sigma\|$ を要するためである．ここで，$\mathcal{T}$ は新しく生成された非終端記号の集合であ

表 1　PASCAL− に対する解析プログラムの記憶容量（実測値）（単位：バイト）
Table 1　Actual memory capacity occupied in the parsing program for PASCAL−. (memory unit: byte)

		筆者ら X	Aho ら Y	$X:Y$
コード部分　(A)		57424	55264	1 : 0.96
表	Ⅰ．全　体*(B)	80896**	7657064	
	Ⅱ．解析表＋生成規則表	60228	7439496	1 : 124
合　計　(A)＋(B)		138320	7712328	1 : 56

* 全体には，解析表，生成規則表のほかに，解析時に必要となる入力テキスト用の領域，解析スタック用の領域，解析木用の領域などが含まれる．

** 筆者らの解析表では，1 つのエントリに複数個の τ 型生成規則番号が入る場合がある．そうした場合に，プログラム上では各生成規則番号をポインタで結ぶことにより実現している．この値には，こうした領域も含まれている．

* The values consist of memories for the parsing-table and the production rules, in addition, for the input text, a parsing-stack, and parsing-trees.

** In authors' parsing-table, it is possible to enter multiple τ type production indices into a single entry of tables. In such a case, the linked-list technique is applied in authors' program. The value involves one for these linked-lists.

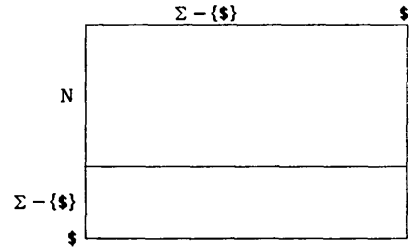

図 3（a）　筆者らの解析表の構造
Fig. 3(a)　The structure of authors' parsing-table.

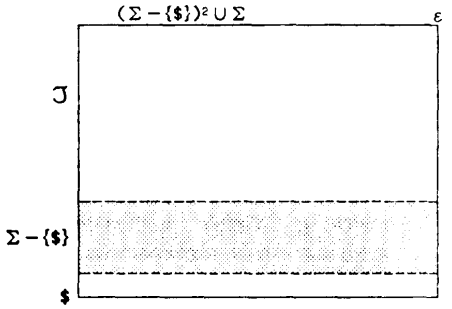

図 3（b）　Aho らの解析表の構造
計算機の記憶容量の関係で斜線部（生成規則による置きかえ作業のないところ）については表から除外して，解析ルーチンの中で処理するようにしてある．ただし，$(\Sigma-\{\$\})^2$ は長さ 2 の終端記号列の集合を表す．

Fig. 3(b)　The structure of Aho and Ullman's parsing-table.
We didn't construct the oblique portion of the Fig. 3(b) as an array because authors' computer does not have enough memory size, but we can code this part of tables in the parsing steps. $(\Sigma-\{\$\})^2$ denotes a set of terminal string of length 2.

り，$(\Sigma-\{\$\})^2$ は長さ2の終端記号列の集合を表す．ただし，記憶容量の関係で実験では図3(b)の斜線部は表からは除外したので，Aho らは $(\|\mathcal{I}\|+1)\times\|(\Sigma-\{\$\})^2\cup\Sigma\|$ の領域となっている．したがって，斜線部を考慮すると筆者らとの差はさらに拡大する．

(ii) 生成規則表の大きさは，対象となる文法の生成規則数を n とするとき，筆者らは n 個の生成規則に対する記憶領域ですむ．これに対して，Aho らは書きかえにより生成規則の数が増えるのでその分余計に記憶領域を必要とする．

PASCAL−文法を Aho ら用に書きかえたところ**表2**を得た．新しい生成規則の数がもとの数の8倍になっているが，この増える割合は，もとになる文法の生成規則の数が多くなるほど大きくなると思われる．

解析速度

次に，両者の解析速度の比較を，以下の2つのサンプル・プログラムを用いて行った．

- 8王妃 (eight queen) 問題プログラム (46 ステップ)
- クイック・ソートプログラム (42 ステップ)

実験は，各プログラムを5回ずつ解析しその平均所要時間を求めた (**表 3**)．

この表によれば，筆者らのほうが，解析時間が約2倍弱多く要している．その原因は [定義12] で与えられる集合 U の計算にある．試みに U の演算所要時間を測定してみたところ，8王妃で U を求める演算が855回行われており，時間にして 0.045 秒，クイック・ソートのほうは 689 回で，0.038 秒であった．この時間を表3の筆者らの実験値より差し引くと，ほぼ Aho らの所要時間と一致する．したがって，次に述べるように，この時間を節約できるように解析表を工夫すれば，解析速度はほぼ同程度に改良されると推測される．

U の計算を解析時点から除くには，解析表作成時にその計算をすませてしまう方法が考えられるが，このことについて少し考察する．図3(a)の解析表をもとにして，U を求めながら**図4**のような解析表を作ることが考えられる．こうすることによって，解析時にその都度 U を求める必要がなくなるので，U の計算時間の分だけ解析時間は短くなる．しかも，PASCAL−での実験結果では8王妃問題，クイック・ソートいずれも $|U|$ の値は最大が2で，ほとんどが1であった．このことを考慮に入れると，U の要素の中からさらに所定の生成規則を選択するにはせいぜい2回の判定ですむので，このことによる遅れはほとんどないものと思われる．U の計算後の表を実験的に試作し，解析速度を測定した結果，予想したように Aho らのものとほとんど一致した．

したがって，解析時間については Aho らと同程度まで改良できると推定される．ただし，このような表は $\|N\|=n, \|\Sigma\|=m$ とするとき，表の大きさはほぼ m^2n で表されるから，図3(a)のものに比べると，かなり大きくなる．しかし，Aho らの図3(b)に比べると，$\|N\|\ll\|\mathcal{I}\|$ であるからなお比較にならないほど，小さい．さらに，Aho らの場合，書きかえられた生成規則の数がもとの数の少なくとも8倍になることを考えると，記憶容量の点でもかなり小さくてすむと思われる．

表 2 書きかえによる生成規則数，非終端記号数等の変化 (PASCAL−の場合) (実測値)
Table 2 Actual numbers of production rules, terminals, and non-terminals resulted from rewriting Wirth's production rules for applying Aho's algorithm (in the case of PASCAL−).

	準 LL(2)	アルゴリズム θ	Aho ら	増えた割合 準LL(2) : Aho ら
生成規則数	98	98	804	1 : 8
非終端記号数	54	54	427	1 : 8
終端記号数	46	46	46	

表 3 解析速度 (実測値) (単位：秒)
Table 3 Actual times taken for parsing. (time unit : second)

	筆者ら X	Aho ら Y	$X : Y$
8王妃	0.11	0.06	1.8 : 1
クイック・ソート	0.10	0.06	1.7 : 1

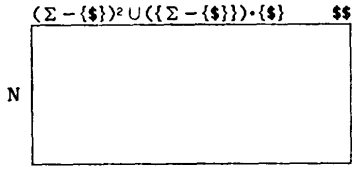

図 4 U を計算したあとの解析表
・は記号の連接を示す演算子．
Fig. 4 The structure of parsing-table after calculating U.
The symbol · indicates an operator for concatenation between terminal symbols.

誤りの検出位置

最後に，誤りの検出位置について考察する．スタックの先頭が終端記号の場合，Aho らも筆者らもその解析表の構成上，エラーが発見されるのはそのエラー記号が入力された時点であり，この発見位置については両者とも同じである．一方，スタックの先頭が非終端記号のときは，Aho らも筆者らもその非終端記号から対象となる入力2記号が導出できるか否かを調べているので，この場合も，誤りの検出については両者とも差がない．

7. むすび

本論文では，準 LL(2) 文法に対する解析表の構造とそれにもとづいた解析法を提案した．従来，LL(k), $k\geqq 2$ の場合解析表の作成法も含めて解析法の研究はほとんどなされていなかった．本論文で提案する解析表には文脈の保存が可能な τ 型生成規則番号が記入される．この解析表を用いて，集合 U を求めながら解析していくという本方法は，解析時間は Aho らのものより2倍弱かかるが，解析表の構造の工夫により，Aho らのものと同程度に改良が可能であると思われる．

一方，解析時に要するいくつかの表の中で，大きいものは解析表と生成規則表であるが，この2つで比較すると Aho らのものの 1/120 程度ですむ（表1）．表構造から分かるように，こうした傾向は他のプログラミング言語についても一般的に言える．

計算機の所有が，ますます個人化する傾向のなかで，記憶容量の節約はやはり大きな魅力である．今後の課題としては，試行した表構造の工夫をより一段とすすめ，解析速度をあげてきいたい．

謝辞　本研究をすすめるにあたり，長年にわたりご指導いただいている東京理科大学・井上謙蔵教授に心から感謝の意を表したい．また，本論文の査読者の有益なるご助言に謝意を表する．

参考文献

1) Aho, A. V. and Ullman, J. D.: *The Theory of Parsing, Translation, and Compiling*, Vol. 1, pp. 348-356, Prentice-Hall (1972).
2) Lewis, P. M. and Stearns, R. E.: *Compiler Design Theory*, pp. 262-276, Addison-Wesley (1976).
3) 吉田，竹内：生成規則番号を用いた LL(1) 解析テーブル作成のアルゴリズム，情報処理学会論文誌，Vol. 27, No. 11, pp. 1095-1105 (1986).
4) 吉田，竹内：準 LL(2) 文法に対する構文解析表の作成アルゴリズム，情報処理学会論文誌，Vol. 31, No. 6, pp. 916-929 (1990).
5) Yoshida, K. and Takeuchi, Y.: Some Properties of an Algorithm for Constructing LL(1) Parsing-Tables Using Production Indices, *J. Inf. Process.*, Vol. 11, No. 4, pp. 258-262 (1988).
6) Hansen, P. B.: *Brinch Hansen on Pascal Compilers*, pp. 15-16, Prentice-Hall (1985).
7) Backhouse, R. C.: *Syntax of Programming Languages* (*Theory and Practice*), pp. 92-152, Prentice-Hall (1979).
8) Grune, D. and Jacobs, C. J. H.: A Programmer-friendly LL(1) Parser Generator, *Software*, Vol. 18, No. 1, pp. 29-38 (1988).

(平成2年1月23日受付)
(平成2年6月 4 日採録)

吉田　敬一（正会員）

昭和13年生．昭和37年法政大学工学部電気工学科卒業．昭和43年法政大学大学院工学研究科修士課程中退．日本電気(株)を経て，現在，静岡大学工業短期大学部・情報工学科助教授．工学博士．コンパイラの自動生成，情報処理教育のあり方，人工知能に関心を持っている．著書「コンパイラの理論と応用」（共訳，学研），「コンピュータ・サイエンスのための言語理論入門」（共訳，共立出版）など．日本ソフトウェア科学会，ACM, IEEE, ACS（オーストラリア）各会員．

竹内　淑子（正会員）

昭和29年生．昭和55年静岡大学工業短期大学部・情報工学科卒業．(株)東京システム技研（コンパイラ開発担当）を経て，現在，浜松職業訓練短期大学校情報処理科教官．静岡大学工業短期大学部非常勤講師．コンパイラの自動生成に関心を持っている．著書「BASIC プログラミングのすべて」（共訳，一橋出版）．日本ソフトウェア科学会，IEEE 各会員．