# 非正準 LR 構文解析と拡張 LR 構文解析の提案†

張 又普†† 中田育男†† 佐々政孝††

LR($k$) 構文解析は後戻りのない決定性構文解析として，事実上もっとも広い範囲の言語に適用できるものであるが，場合によっては，LR($k$) の適用範囲がまだ不十分と思われる[4),10)]．正準上昇型構文解析は最右導出を逆順にたどることに対応するが，必ずしも最右導出でない導出（非正準導出）方式に対する構文解析法の一つとして，本論文では LR 構文解析法を拡張した非正準 LR($k$)（LNR($k$) と呼ぶことにする）構文解析法とその一般的なアルゴリズムを提案し，さらに，この方法の正当性を証明する．また，LNR($k$) 構文解析と LR($k$) 構文解析を組み合わせた拡張 LR($k$)（EXLR($k$) と呼ぶことにする）構文解析法を提案する．LR($k$) 構文解析法で解析できないいくつかの例について，LNR($k$) 構文解析法でそれらを解析する方法も述べる．

## 1. はじめに

正準ボトムアップ (canonical bottom-up) 構文解析は最右導出を逆順にたどることに対応するが，必ずしも最右導出を逆順にたどる必要がない方式を非正準ボトムアップ構文解析と呼ぶことにする．近年，正準構文解析技法を非正準方式まで一般化しようとする試みがいろいろなされている．文献 3) は単純順位構文解析を非正準方式まで拡張し，全順位構文解析法を提案した．文献 11) は BRC (Bounded Right Context Grammars) を非正準方式まで拡張し，BCP (Bounded Context Parsable Grammar) を提案した．文献 9) は非正準導出の一般理論とモデルを提案した．文献 10) は SLR（1）構文解析を非正準方式まで拡張し，非正準 SLR（1）構文解析のアルゴリズムを提案した．本論文では正準 LR($k$) 構文解析を非正準方式 (LNR($k$) と呼ぶことにする）まで拡張し，その一般的なアルゴリズムを提案し，さらに，この方法の正当性を証明する．ここで L は Left to right scanning of the input の，NR は producing a Noncanonical Right parser の略で，$k$ は先読み記号の数である．また，LNR($k$) と LR($k$) 構文解析を組み合わせた拡張 LR($k$) 構文解析 (EXLR($k$) と呼ぶことにする）を提案する．

LR($k$) 構文解析は後戻りのない決定性構文解析として，現在事実上もっとも広い範囲の言語に適用できるものであるが，場合によっては，LR($k$) の適用範囲がまだ不十分と思われる[4),10)]．たとえば，以下のような問題がある．

† A Proposal of Noncanonical LR Parsers and Extended LR Parsers by YOUPU ZHANG, IKUO NAKATA and MASATAKA SASSA (Institute of Information Sciences and Electronics, University of Tsukuba).

†† 筑波大学電子・情報工学系

問題 1：次の $G1$ は Pascal の変数宣言の一部である[10)]．

$G1$：変数宣言→var 名称並び：型表記
　名称並び→名称，名称並び|名称
　型表記→integer|real　　名称→id

$G1$ では id を変数の名称に還元するときに，変数の型が分からない．変数の型が何になるかは構文規則だけでは明記されていない．それを明記するために $G1$ を次の $G2$ のように変えたいが，$G2$ は LR($k$) 文法ではない．

$G2$：変数宣言→var 整数名称並び：整数型表記 |
　　var 実数名称並び：実数型表記
　整数名称並び→整数名称，整数名称並び | 整数名称
　実数名称並び→実数名称，実数名称並び | 実数名称
　整数型表記→integer　　実数型表記→real
　整数名称→id　　実数名称→id

この場合，「整数名称並び」に還元することはその右側にある「整数型表記」への還元が先に行われていれば，可能になる．そのためには，LR 構文解析の順序を変更するような拡張が必要である．このように右側にある情報（型表記など）を左側で利用するのは構文解析後の意味解析時に行われるのが通常であるが，上記のようにすれば，それが構文解析の枠内で行えるようになる．

問題 2：コード生成に構文解析の技法を使う場合，中間語の仕様が前置記法での生成規則として表現される[6)]．その場合，$S\to A|AA$；$A\to a|b|ab$（条件 $C$）；のような条件付き生成規則がよく現れる．すなわち，$ab$ が入力であるときに，条件 $C$ が満たされれば，$A\to ab$ で還元するが，条件 $C$ が満たされなければ，$A\to a$ と $A\to b$ で還元する．LR($k$) 文法で解析しようとすれば，シフト還元競合が起こる．シフト優先で解

決しようとすれば，ブロック状態（シフト，還元のどちらもできない状態を意味する）が起こる[4]．この場合にもLR構文解析の順序を変えて $A\to a$ による還元を（やめるのでなく）一時遅らせたい．

問題3：問題2と同じように，中間語の仕様が前置記法での生成規則として表現される場合，次のような生成規則がよく現れる．$G3=(N, T, P, E)$, $P=\{E\to+EA|+AE|A;\ A\to a\}$．ここで，$E$ は rEgister（レジスタ）で，$A$ はアドレスモードである．$+AA$（注：$A$ が二つあるので，区別するために以下に $+A_1A_2$ のように書くことにする）のような文形式になったときに，$A_1$, $A_2$ のどちらをレジスタにロードするのがよいか問題になる．複数個のレジスタを有効に使う目的譜を作り出すためには，複雑なオペランドをレジスタにロードするほうが効率がよい[6]．一般に，二項演算子の一番目のオペランド $A_1$ が解析された時点では，$A_1$ をレジスタにロードする命令を作るべきか判断できない．二つのオペランドをともに解析しておいてから，どちらをレジスタにロードするかを判断するのが望ましい．しかし，これは LR($k$) 解析ではうまく実現できない[5]．この場合にも $A_1$ からの還元を遅らせたい．

以上のような問題を解決するために，LR($k$) 文法よりもっと広い範囲の文法に適用でき，効率が同じである構文解析法があるとよい．そのようなものとして，LNR($k$) 構文解析と EXLR($k$) 構文解析を提案する．以下，2章で術語と LNR 構文解析の考え方を述べる．3章で LNR($k$) 構文解析の理論を示す．4章で LNR($k$) 文法の特徴と EXLR($k$) 構文解析のアルゴリズムを述べる．5章で前にあげた LR($k$) 構文解析で解決できない例を LNR（1）で解決する方法を述べ，LNR($k$) の手法が適用できる分野の一端を示す．付録で基本定理を証明する．

## 2. 術語の定義と LNR 構文解析の考え方

$G=(N, T, P, S)$ は文脈自由文法 (CFG)．$N$ は非終端記号の有限集合，$T$ は終端記号の有限集合，$P$ は生成規則の有限集合，$S\in N$ は出発記号である．$G'=(N', T', P', S')$ は $G$ に「$S'\to S\$$」を加えて得られる拡大文法とする．とくに断わりのないかぎり，以下のような記号を使うことにする．$V=N\cup T$, $A, B, C, D, E, F\in N$, $X, Y, Z\in V$, $a, b, c\in T$, $u, v, w\in T^*$, $x, y, z\subseteq N^*T^*$, $\alpha, \beta, \gamma, \cdots\in V^*$, $\varepsilon$ は空文字列，$\phi$ は空集合とする．

定義 2.1：文形式 (sentential form) の定義．

文法を $G=(N, T, P, S)$ とするとき，$S\Rightarrow\alpha$ のような導出があったら，$\alpha$ は文形式である． ■

定義 2.2：関数 $\mathrm{Head}_k$ の定義

$\forall\beta\in V^*$ に対し $\beta=X_1\cdots X_k\gamma$ なら，$\mathrm{Head}_k(\beta)=\{X_1\cdots X_k\}$，$\beta=X_1\cdots X_i$，$(i<k)$ なら，$\mathrm{Head}_k(\beta)=\{X_1\cdots X_i\}$．この場合，$\mathrm{Head}_k(\beta)$ は要素が一つしかない文字列の集合である．

$\forall s\subseteq V^*$ に対し，$\mathrm{Head}_k(s)=\cup_{\beta\in s}\mathrm{Head}_k(\beta)$．とくに $k=1$ なら，$k$ を省略できる． ■

定義 2.3：関数 $\mathrm{First}_k$ の定義

$$\forall\beta\in V^*\text{ に対し，}\mathrm{First}_k(\beta)=\{a_1\cdots a_k\,|\,\beta\Rightarrow a_1\cdots a_k\gamma\}\cup\{a_1\cdots a_i\,|\,\beta\Rightarrow a_1\cdots a_i,\ i<k\}$$

$\forall s\subseteq V^*$ に対し，$\mathrm{First}_k(s)=\cup_{\beta\in s}\mathrm{First}_k(\beta)$．すなわち，$\mathrm{First}_k(\beta)$ は $\beta$ が導出できる終端記号列の長さ $k$ の語頭になりうるものの集合である．とくに，$k=1$ なら，$k$ を省略できる． ■

本文で扱う文法はすべて次の条件を満たすとする．

1) $\forall A\in N$ に対し，$A\Rightarrow A$ のような導出は存在しない；

2) $\forall A\in N$ に対し，$\mathrm{First}(A)\neq\phi$．

次に簡単な例を挙げて，直感的に LNR 構文解析の考え方を述べる．

例1：$G4=(N, T, P, S)$ は CFG で，$N=\{S', S, A, B, C, D\}$，$T=\{a, b, c, \$\}$，$P=\{p_0: S'\to S\$;\ p_1: S\to AC;\ p_2: S\to BD;\ p_3: A\to a;\ p_4: B\to a;\ p_5: C\to ab;\ p_6: D\to ac\}$ とする．$p_j$ は生成規則の番号である．

$G4$ は LR（2）文法で，LR（1）文法ではない．その LR（2）構文解析表の一部を示す．ここで Goto 関数は普通の LR 構文解析の Goto 関数で，Goto $(I, X)$ は LR 状態 $I$ で $X$ を読んだとき遷移する LR 状態を示す（定義 3.6 と 4.5 参照）．

（1） $I_0=\{[S'\to .\ S\$,], [S\to .\ AC, \$], [S\to .\ BD, \$], [A\to .\ a, ab], [B\to .\ a, ac]\}$

（2） $I_1=\mathrm{Goto}(I_0, a)=\{[A\to a.,\ ab], [B\to a.,\ ac]\}$

（2）で生成規則 $A\to a$ と $B\to a$ の二つの還元を区別できるためには，先読み記号が2つなければならない．しかしここで，（2）の先読み $ab$ と $ac$ を生成規則 $C\to ab$ と $D\to ac$ の左辺の $C$ と $D$ で置き換えれば，（2）は次の（3）に変わる．

（3） $\{[A\to a., C], [B\to a., D]\}$

ここで，$C, D$ を先読み記号と考えれば，競合(conflict)がなくなり，先読みの長さが1になる．（4），（5）は（1），（2）に対応する LNR（1）状態である．すなわ

ち，$[A\to a.,C]$ のように非終端記号を先読み記号とすることは $A\to a$ による還元より先に $C$ への還元を行うことを意味する．これは LNR 構文解析の基本的な考え方である．場合によって LNR 構文解析に対応する導出順序は LR 構文解析の最右導出と異なるが，最左導出ではない．LR と LNR 構文解析法に対応する文字列 $aab$ の導出順序（還元の逆順）を示すと，それぞれ(6)，(7)のようになる．

(4) $I_0=\{[S'\to .\ S\$,],\ [S\to .\ AC,\$],\ [S\to .\ BD,\$],\ [A\to .\ a,\ C\$],\ [B\to .\ a,\ D\$]\}$

(5) $I_1=\mathrm{Goto}(I_0,a)=\{[A\to a.,\ C\$],\ [B\to a.,\ D\$],\ [C\to .\ ab,\$],\ [D\to .\ ac,\$]\}$

(6) $S'\Rightarrow S\$\Rightarrow AC\$\Rightarrow Aab\$\Rightarrow aab\$$（最右導出である）

(7) $S'\Rightarrow S\$\Rightarrow AC\$\Rightarrow aC\$\Rightarrow aab\$$（最右導出でない）

非正準導出は必ずしも最右（または最左）導出ではないが，ランダムな導出ではない．一定の基準で導出順序を決めているのである．その決め方についてのいくつかの発想を示そう．上の例の $S\to AC$ で $C$ への還元を先にするのは $AC$ のように非終端記号が並んでいるときは左側を先に導出したほうがよい場合があることを示している．そこで，全体的には最右導出とし，非終端記号が並んだ部分については最左導出することが考えられる．そこで

方法 1：一般に，文形式 $\alpha$ は次のような形である．$\alpha=\beta aA_1\cdots A_iw$，$\beta\in V^*$，$a\in T$，$A_1\cdots A_i\in N^*$，$w\in T^*$ である．$A_1\cdots A_i$ は一番右側の非終端記号列である．その左側の記号 $A_1$ を先に展開する．

ところが，方法 1 のように導出順序を定義すると，後の定理 3.3 と定理 4.2 で述べるようにいろいろな問題が生じる．そこで，導出順序をすべての非終端記号について一律に決めるのでなく，最右導出をするものとしないものに分けることを考える．すなわち，非終端記号集合 $N$ を $L$，$R$ 二つの集合に分割する．$N=L\cup R$，$L\cap R=\phi$ という条件を満たせば，分割は自由である．$R$ に属する非終端記号の導出順序は最右導出で，$L$ に属する非終端記号の導出順序は最右導出ではない．いうまでもなく，$R=N$，$L=\phi$ のように $N$ を分割すれば，LNR 構文解析は LR 構文解析になる．方法 1 は次のように修正される．

方法 2：一般に，文形式 $\alpha$ は次のような形である．$\alpha=\beta XA_1\cdots A_iw$，$\beta\in V^*$，$X\in T\cup R$，$A_1\cdots A_i\in L^*$，$w\in T^*$ である．$A_1\cdots A_i$ は $\alpha$ の中で $L$ の要素からなる一番右側の非終端記号列である．その左側の文字 $A_1$ を先に展開する．

定義 2.4：方法 2 の導出順序を非正準最右導出（noncanonical rightmost derivation または縮めて nr 導出と呼ぶことにする）と名付ける．$\Rightarrow^{nr}$ で表現する．一般に，文形式 $\alpha$ は次のような形である．$\alpha=\beta XA_1\cdots A_iw$，$\beta\in V^*$，$X\in T\cup R$，$A_1\cdots A_i\in L^*$，$w\in T^*$．この文形式 $\alpha$ に対し，先に展開する非終端記号は $A_1$ である．$i=0$ なら，$\beta X$ の最右非終端記号を展開する．言い換えれば，任意の文形式 $\alpha X\beta$ に対し，$\alpha,\beta\in V^*$，$X\in T\cup R$ なら，$\beta$ を終端記号列まで展開してから，$X\in R$ なら $X$ が展開でき，$X\in T$ なら $X$ の左側の非終端記号が展開できる．任意の文形式 $\beta XA_1\cdots A_iw$ に対し，$X\in T\cup R$，$A_1\cdots A_i\in L^*$ なら，左から右へ $A_1\cdots A_i$ を展開する． ■

任意の文形式 $\beta A_1\cdots A_ia_1\cdots a_kw$ $(A_1\cdots A_i\in L^+)$ に対し，LR($k$) 構文解析では，$A_i$ に還元される時点で $a_1\cdots a_k$ の情報が必要なので，$A_i$ が展開される時点で，生成規則「$A_i\to\delta_i$」$\in P$ なら，$a_1\cdots a_k$ が項 $[A_i\to .\ \delta_i]$ の先読みでなければならない．非正準最右導出の場合には，$A_1$ から $A_i$ へ展開する．$A_1$ に還元される時点では $\mathrm{Head}_k(A_2\cdots A_ia_1\cdots a_kw)$ の情報しか必要がないが，以下の理由で，そこでの先読み記号は $A_2\cdots A_ia_1\cdots a_k$，すなわち，$A_i$ につづく長さ $k$ の終端記号まで含めたものとする．LR($k$) の場合と同じように，LNR($k$) 構文解析表は導出の順序にしたがって作成され，LNR($k$) 項 $[A_1\to .\ \delta_1]$ の先読みから項 $[A_i\to .\ \delta_i]$ の先読みが計算される．その途中で，新たな先読み終端記号列が付け加えられることはないので，前者に終端記号列 $a_1\cdots a_k$ が入っていないと，それが後者の先読み記号列として求められない．ゆえに次の $\mathrm{Left}_k$ を定義する．

定義 2.5：$\forall\alpha\in V^*$ に対し，$\mathrm{Left}_k(\alpha)=\{A_1\cdots A_iu\mid\alpha=A_1\cdots A_iX\beta,\ A_1\cdots A_i\in L^*,\ X\in T\cup R,\ u\in\mathrm{First}_k(X\beta)\}$

すなわち，$\mathrm{Left}_k(\alpha)\subseteq\cup_{j\le k}L^*T^j$ は $\alpha$ の先頭の $L$ の文字列の後に先読み終端記号列を付けたものの集合である． ■

定義 2.6：$\forall s\subseteq V^*$，$\mathrm{Left}_k(s)=\cup_{\beta\in s}\mathrm{Left}_k(\beta)$ ■

定義 2.7：次のような記法を定義する．

$\forall\alpha\in V^*$，$y\subseteq V^*$ に対し，$\alpha y=\{\alpha\beta\mid\beta\in y\}$．ここで $\alpha\beta$ は「文字列 $\beta$ を文字列 $\alpha$ の後ろに付ける」を意味する． ■

定理 2.1：$\forall\alpha,\beta\in V^*$ に対し，$y=\mathrm{Left}_k(\beta)$ なら，$\mathrm{Left}_k(\alpha\beta)=\mathrm{Left}_k(\alpha y)$.

定理 2.2：「$S' \doteqdot^{nr} \alpha A\eta \Rightarrow^{nr} \alpha\beta\eta$」が nr 導出なら，$\mathrm{Head}_k(\eta) = \mathrm{Head}_k(\mathrm{Left}_k(\eta))$

証明：定理 2.1，2.2 はほぼ自明であるので，証明を省略する． ■

## 3. LNR($k$) 構文解析理論

LNR($k$) 文法の定義

定義 3.1：$G=(N, T, P, S)$ は CFG で，$G'=(N', T', P', S')$ は $G$ に「$S' \rightarrow S\$$」を加えて得られる拡大文法とする．次の条件（1），（2），（3）から，必ず（4）が導かれるとき，$G$ は LNR($k$) 文法である．

（1） $S' \doteqdot^{nr} \alpha A\eta \Rightarrow^{nr} \alpha\beta\eta$

（2） $S' \doteqdot^{nr} \gamma B\lambda \Rightarrow^{nr} \gamma\delta\lambda = \alpha\beta\mu$

（3） $\mathrm{Head}_k(\eta) \cap \mathrm{Head}_k(\mathrm{Left}_k(\mu)) \neq \phi$

（4） $\alpha=\gamma$, $A=B$, and $\beta=\delta$. ■

定義 3.1 の（1），（2）の「$\doteqdot^{nr}$」と「$\Rightarrow^{nr}$」を「$\doteqdot^{rm}$」と「$\Rightarrow^{rm}$」，（3）を $\mathrm{First}_k(\eta) = \mathrm{First}_k(\mu)$ と書けば，定義 3.1 は LR($k$) の定義になる[1]．ここで，「$\Rightarrow^{rm}$」は最右導出を意味する．

定義 3.2：LNR($k$) 項 (item)

$G=(N, T, P, S)$ は CFG で，$A \rightarrow \beta_1\beta_2 \in P$，$x \subseteq L^*T^*$ のとき，$[A \rightarrow \beta_1.\ \beta_2,\ x]$ を LNR($k$) 項と定義する．または単に項という．「$A \rightarrow \beta_1.\ \beta_2$」部分を項の核と定義し，$x$ 部分を項の先読みと定義する．一般の LR($k$) 項と同じように，同一の核「$A \rightarrow \beta_1.\ \beta_2$」を持つ LNR($k$) 項 $[A \rightarrow \beta_1.\ \beta_2,\ x_i]$ の集合を $[A \rightarrow \beta_1.\ \beta_2,\ x]$ で示し（$x = \bigcup_i x_i$），やはり LNR($k$) 項と呼ぶことにする． ■

定義 3.3：成長可能な語頭

nr 導出 $S' \doteqdot^{nr} \alpha A\gamma \Rightarrow^{nr} \alpha\beta_1\beta_2\gamma$ が存在するとき，$\alpha\beta_1$ を成長可能な語頭と呼び，さらに $x = \mathrm{Left}_k(\gamma)$ のとき，LNR($k$) 項 $[A \rightarrow \beta_1.\ \beta_2,\ x]$ は成長可能な語頭 $\alpha\beta_1$ に対して正当であると定義する．語頭を明示する必要がない場合に，単に正当な項という． ■

次の定理 3.1 は LNR($k$) 構文解析理論の基礎である．

定理 3.1：$G=(N, T, P, S)$ が LNR($k$) 文法である必要十分条件は「項 $[A \rightarrow \beta.,\ x]$ が成長可能な語頭 $\alpha\beta$ に対して正当なら，同じ語頭 $\alpha_1\beta_1 = \alpha\beta$ に対して，$\mathrm{Head}_k(x) \cap \mathrm{Head}_k(\mathrm{Left}_k(\beta_2 y)) \neq \phi$ であるような別の正当な項 $[A_1 \rightarrow \beta_1.\ \beta_2,\ y]$ が存在しない」ことである．ここで，$\beta_2$ は $\varepsilon$ かもしれない．

証明：付録で証明する． ■

定義 3.4：LNR($k$) 状態

$G=(N, T, P, S)$ は CFG で，$\gamma \in V^*$ とする．$V_k(\gamma)$ を語頭 $\gamma$ に対して正当なすべての項の集合と定義する．これを語頭 $\gamma$ に対して正当な LNR($k$) 状態と呼ぶことにする．語頭を明示する必要がない場合に，単に状態という． ■

$S' \doteqdot^{nr} \beta A\alpha$ があるとすると，次の関数 $\mathrm{Rightof}_k(A, \alpha)$ は非終端記号 $A$ の先読みを計算する関数である．$A \in R$ なら，$A$ の右側の非終端記号は $A$ より先に展開するから，$A$ の先読みは終端記号列である．一方，$A \in L$ なら，$A$ の先読みは非終端記号を含むかもしれないから，関数 $\mathrm{Left}_k$ を用いて計算しなければならない．

定義 3.5：$\forall A \in N$, $\forall \alpha \in V^*$ に対し，次のように $\mathrm{Rightof}_k(A, \alpha)$ を定義する．

$A \in R$ なら，$\mathrm{Rightof}_k(A, \alpha) = \mathrm{First}_k(\alpha)$

$A \in L$ なら，$\mathrm{Rightof}_k(A, \alpha) = \mathrm{Left}_k(\alpha)$

さらに，$y \subseteq V^*$ に対して，$\mathrm{Rightof}_k(A, y) = \bigcup_{\alpha \in y} \mathrm{Rightof}_k(A, \alpha)$ と定義する． ■

次に $V_k(\gamma)$ を計算するアルゴリズムを提案する．

アルゴリズム 3.1：$V_k(\gamma)$ を作る

入力：$G=(N, T, P, S)$, $\gamma \in V^*$, 整数 $k$, $N$ の分割 $L$, $R$

出力：$V_k(\gamma)$

方法：$\gamma = X_1X_2\cdots X_n$ とすると，$V_k(\varepsilon)$ から $V_k(X_1)$, $V_k(X_1X_2)$, …, $V_k(X_1\cdots X_n) = V_k(\gamma)$ を作る

（1） 次のように $V_k(\varepsilon)$ を作る

（1a） $V_k(\varepsilon) := \{[S' \rightarrow .\ S\$,\ ]\}$

（1b） $[A \rightarrow .\ B\beta, x] \in V_k(\varepsilon)$ なら，$\{[B \rightarrow .\ \delta, y] \mid B \rightarrow \delta \in P,\ y = \mathrm{Rightof}_k(B, \beta x)\}$ の要素を $V_k(\varepsilon)$ に加える．

（1c） $[A \rightarrow \alpha.,\ Bx] \in V_k(\varepsilon)$ なら，$\{[B \rightarrow .\ \delta, x] \mid B \rightarrow \delta \in P\}$ の要素を $V_k(\varepsilon)$ に加える．

（1d） 新たな項が $V_k(\varepsilon)$ に付け加えられなくなるまで（1b），（1c）を繰り返す．

（2） $V_k(X_1\cdots X_{i-1})$ が作られたと仮定する．次のように $V_k(X_1\cdots X_i)$ を作る．

（2a） $V_k(X_1\cdots X_i) := \{[A \rightarrow \alpha X_i .\ \beta, x] \mid [A \rightarrow \alpha .\ X_i\beta, x] \in V_k(X_1\cdots X_{i-1})\}$

（2b） $[A \rightarrow \alpha .\ B\beta, x] \in V_k(X_1\cdots X_i)$，（$B \in R$ or $X_i \in T \cup R$）なら，$\{[B \rightarrow .\ \delta, y] \mid B \rightarrow \delta \in P,\ y = \mathrm{Rightof}_k(B, \beta x)\}$ の要素を $V_k(X_1\cdots X_i)$ に加える．

（2c） $[A \rightarrow \alpha.,\ Bx] \in V_k(X_1\cdots X_i)$，$X_i \in T \cup R$ なら，$\{[B \rightarrow .\ \delta, x] \mid B \rightarrow \delta \in P\}$ の要素を $V_k(X_1$

$\cdots X_i$) に加える.

(2d) 新たな項が $V_k(X_1\cdots X_i)$ に付け加えられなくなるまで (2b), (2c) を繰り返す. ■

アルゴリズム 3.1 の (2b) と (2c) の説明: (2b) で $B\in L$ かつ $X_i\in L$ の場合に, 項 $[B\rightarrow .\,\delta, y]$ を $V_k(X_1\cdots X_i)$ に加えるということは次の (3.1) のような導出があることに対応する.

(3.1) $S'\overset{*}{\Rightarrow}X_1\cdots X_iB\cdots\Rightarrow X_1\cdots X_i\delta\cdots$

一方, nr 導出の定義 2.4 により, $X_i$ が $B$ より先に展開されなければならない. ゆえに, (3.1) は nr 導出ではなく, 項 $[B\rightarrow .\,\delta, y]$ が成長可能な語頭 $X_1\cdots X_i$ に対して正当ではない. (2c) の条件「$X_i\in T\cup R$」の理由も同じである.

定義 3.6: Goto $(V_k(X_1\cdots X_{i-1}), X_i)=V_k(X_1\cdots X_i)$ ■

定理 3.2: 項 $[A\rightarrow\beta_1.\,\beta_2, x]$ がアルゴリズム 3.1 の計算によって $V_k(\gamma)$ に加えられる必要十分条件は「$[A\rightarrow\beta_1.\,\beta_2, x]$ が成長可能な語頭 $\gamma$ に対して正当である」ことである.

証明: 付録で証明する. ■

例2: 例1の $G4$ の LNR(1) 状態の一部を示す. ここで, $L=N$, $R=\phi$ とする.

$V(\varepsilon)=\{[S'\rightarrow .\,S\$, ], [S\rightarrow .\,AC, \{\$\}], [S\rightarrow .\,BD, \{\$\}], [A\rightarrow .\,a, \{C\$\}], [B\rightarrow .\,a, \{D\$\}]\}$

$V(A)=\{[S\rightarrow A.\,C, \{\$\}]\}$

$V(a)=\{[A\rightarrow a.,\{C\$\}], [B\rightarrow a.,\{D\$\}], [C\rightarrow .\,ab, \{\$\}], [D\rightarrow .\,ac, \{\$\}]\}$

$V(aa)=\{[C\rightarrow a.\,b, \{\$\}], [D\rightarrow a.\,c, \{\$\}]\}$

ここで, $A, C\in L$ であるから, 状態 $V(A)=\{[S\rightarrow A.\,C, \{\$\}]\}$ に項 $[C\rightarrow .\,ab, \{\$\}]$ を加えられない.

アルゴリズム 3.1 は一つの語頭 $\gamma$ に対して正当な状態 $V_k(\gamma)$ を計算する方法であるが, われわれはすべての語頭 $\gamma$ に対して正当な状態 $V_k(\gamma)$ を求めたい.

定理 3.3: $G=(N, T, P, S)$ は CFG で, $L=N, R=\phi, p=|P|, m=\max(|\alpha|)$ for $A\rightarrow\alpha\in P$ とする. 正当な LNR($k$) 項 $[B\rightarrow .\,\beta, x]$ の先読み $x$ の要素の最大長が「$m*(p+1)+k$」より長くなる必要十分条件は「$^{\exists}A\in N, A\overset{+}{\Rightarrow}\alpha A\mu, \mu\in N^+$」である.

証明: 紙幅の関係で省略する. ■

定理 3.3 の系:

$A\overset{+}{\Rightarrow}\alpha A\mu, \alpha\in V^*, \mu\in N^+$ になる $A$ の集合を $R$, $L:=N-R$ として非終端記号の集合 $N$ を $L, R$ に分割すれば, すべての正当な項の先読みの長さは有限になり, 正当な項の数は有限になる. したがって, アルゴリズム 3.1 の (1d) と (2d) は停止し, 正当な状態の数は有限になる. ■

定義 3.7: 二項関係 $\pi$ の定義.

$A\rightarrow\alpha B\mu\in P$, $A, B\in N$, $\alpha\in V^*$, $\mu\in N^+$ のとき, $A\pi B$ と定義する. ■

アルゴリズム 3.2 は定理 3.3 の系に基づいて非終端記号集合 $N$ を $L, R$ に分割するアルゴリズムである.

アルゴリズム 3.2: 非終端記号集合 $N$ を $L, R$ に分割する.

入力: $G=(N, T, P, S)$

出力: $L\cup R=N$, $L\cap R=\phi$ を満たす非終端記号集合 $L$ と $R$

方法: 初期処理: $L:=\phi$, $R:=\phi$

(1) $G$ に対し, 二項関係 $\pi$ と $\pi$ の推移的閉包 $\pi^+$ を計算する.

(2) $^{\forall}A\in N$ に対し, $A\pi^+A$ なら, $A$ を $R$ に加える.

(3) $L:=N-R$ ■

定義 3.8: $G$ の LNR($k$) 状態の集合

$G=(N, T, P, S)$ を CFG とする. $Q_k(G)$ で $G$ のすべての正当な状態の集合を表す. すなわち, $k$ を先読みの長さとして, $Q_k(G)=\{V_k(\gamma)|\gamma$ は $G$ の成長可能な語頭$\}$ と定義する. これを $G$ の LNR($k$) 状態の集合と呼ぶことにする. ■

次に, $L\cup R=N$, $L\cap R=\phi$ になる $N$ の分割 $L, R$ が与えられたとき, $Q_k(G)$ を計算するアルゴリズムを提案する.

アルゴリズム 3.3: $G$ の LNR($k$) 状態の集合 $Q_k(G)$ を作る.

入力: $G=(N, T, P, S)$, 整数 $k$ と $L, R$ ($N=L\cup R$, $L\cap R=\phi$)

出力: $Q_k(G)=\{V_k(\gamma)|\gamma$ は $G$ の成長可能な語頭$\}$

方法: (1) $Q_k(G):=\{V_k(\varepsilon)\}$

(2) $^{\forall}V_k(\gamma)\in Q_k(G)$ と $X\in V$ に対し, Goto $(V_k(\gamma), X)$ が空集合でなく, $Q_k(G)$ に入っていなかったら, それを $Q_k(G)$ に加える.

(3) (2)を新たな集合が $Q_k(G)$ につけ加えられなくなるまで繰り返す. ■

定理 3.3 の系により, アルゴリズム 3.3 は停止する.

定理 3.4: $G=(N, T, P, S)$ の状態 $V_k(\gamma)$ がアルゴリズム 3.3 によって $Q_k(G)$ に加えられる必要十分条件は「$\gamma$ が $G$ の成長可能な語頭である」ことである.

証明: ほぼ自明であるので, 証明を省略する. ■

例 3：アルゴリズム 3.2 によって $G4$ の $N$ を分割すると，$L=N$, $R=\phi$ になる．そのように $N$ を分割した場合のすべての LNR (1) 状態を示す．各状態の左側の数字は状態の番号である．

$I_0=V(\varepsilon)=\{[S'\rightarrow .\ S\$,\ ],[S\rightarrow .\ AC,\ \{\$\}],$
$[S\rightarrow .\ BD,\ \{\$\}],[A\rightarrow .\ a,\ \{C\$\}],[B\rightarrow .\ a,\ \{D\$\}]\}$
$I_1=V(S)=\mathrm{Goto}(I_0,S)=\{[S'\rightarrow S.\ \$,\ ]\}$
$I_2=V(A)=\mathrm{Goto}(I_0,A)=\{[S\rightarrow A.\ C,\ \{\$\}]\}$
$I_3=V(B)=\mathrm{Goto}(I_0,B)=\{[S\rightarrow B.\ D,\ \{\$\}]\}$
$I_4=V(AC)=\mathrm{Goto}(I_2,C)=\{[S\rightarrow AC.,\{\$\}]\}$
$I_5=V(BD)=\mathrm{Goto}(I_3,D)=\{[S\rightarrow BD.,\{\$\}]\}$
$I_6=V(a)=\mathrm{Goto}(I_0,a)=\{[A\rightarrow a.,\{C\$\}],[B\rightarrow a.,$
$\{D\$\}],[C\rightarrow .\ ab,\ \{\$\}],[D\rightarrow .\ ac,\ \{\$\}]\}$
$I_7=V(aa)=\mathrm{Goto}(I_6,a)=\{[C\rightarrow a.\ b,\{\$\}],$
$[D\rightarrow a.\ c,\{\$\}]\}$
$I_8=V(aab)=\mathrm{Goto}(I_7,b)=\{[C\rightarrow ab.,\{\$\}]\}$
$I_9=V(aac)=\mathrm{Goto}(I_7,c)=\{[D\rightarrow ac.,\{\$\}]\}$

以上，LNR($k$) 構文解析のアルゴリズムを提案し，その正当性を証明した．LNR 構文解析表は LR 構文解析表と同じように作成することができる．LNR 構文解析の状況とは最初の成分がパーサスタックであり，2番目の成分が入力列スタックである次のような対のことである．

$(s_0X_1s_1X_2\cdots X_ms_m,\ Y_iY_{i+1}\cdots Y_n\$)$

ここで，$X$, $Y\in V$, $s$ は状態記号である．LNR($k$) 構文解析の次の動作は現在の入力列スタックの最上段の文字列 $Y_i\cdots Y_{i+k-1}$ とパーサスタックの最上段の状態 $s_m$ とを読み，LNR 構文解析表の記入欄を調べることによって決まる．

例 4：$G4$ の文 $aab$ を解析するシミュレーション．

$(I_0, aab\$)\vdash(I_0aI_6, ab\$)\vdash(I_0aI_6aI_7, b\$)\vdash$
$\vdash(I_0aI_6aI_7bI_8, \$)\vdash(I_0aI_6, C\$)\vdash(I_0, AC\$)\vdash$
$\vdash(I_0AI_2, C\$)\vdash(I_0AI_2CI_4, \$)\vdash(I_0, S\$)\vdash(I_0SI_1, \$)$

## 4. LNR($k$) 文法の特徴と EXLR($k$) 構文解析

定理 4.1：$G=(N, T, P, S)$ が LNR($k$) 文法なら，$G$ は曖昧性のない文法である．

証明：自明であるので証明を省略する． ■

定義 4.1：$\forall Y\in T$ に対して，$\mathrm{First}'(Y)=\{Y\}$
$\forall Y\in N$ に対して，$\mathrm{First}'(Y)=\{X\mid Y\Rightarrow X\alpha, X\in V\}$
$\forall s\subseteq V$ に対して，$\mathrm{First}'(s)=\cup_{Y\in s}\mathrm{First}'(Y)$ ■

次に LNR($k$) 文法の適用範囲が LR($k$) 文法より大きいことを証明する．LNR($k$) の適用範囲は $N$ を $L$, $R$ に分割する方法に左右される．$N$ を $L$, $R$ へ分割する方法の一つをアルゴリズム 3.2 に示したが，実はこの分割では，LR($k$)⊆LNR($k$) が成り立たない．その原因は次のとおりである．$G$ に対し，$S'\overset{nr}{\Rightarrow}\alpha AB\beta$; $S'\overset{nr}{\Rightarrow}\alpha AY\gamma$; $A, B\in L$; $Y\in T\cup R$; $Y\in\mathrm{First}'(B)$ なら，$G$ は LNR($k$) 文法ではないが，LR($k$) 文法かもしれない．そこで，LR($k$)⊆LNR($k$) を満たす分割法を次の定義 4.2 とアルゴリズム 4.1 に示す．この方法は LR($k$)⊆LNR($k$) を成り立たせなくするような非終端記号を探し，それを全部 $R$ に入れるものである．そのように $N$ を $L$, $R$ に分割すれば LR($k$)⊆LNR($k$) が成り立つ．

定義 4.2：二項関係 $\tau$ と $\rho$ を定義する．
$A\rightarrow\alpha BX\beta\in P$, $A, B\in N$, $X\in V$ なら，$B\tau X$ と定義する．
$A\rightarrow\alpha B\in P$, $A, B\in N$ なら，$A\rho B$ と定義する． ■

アルゴリズム 4.1：$G=(N, T, P, S)$ の非終端記号集合 $N$ を $L$, $R$ に分割する．

初期処理：$R:=\phi$

1) $\forall A\in N$ に対し，$A\Rightarrow\alpha A\beta$, $\alpha\in V^*$, $\beta\in N^+$ なら，$R:=R\cup\{A\}$；

2) $\forall A\in N$ に対し，$A\Rightarrow\varepsilon$ なら，$R:=R\cup\{A\}$；

3) $L:=N-R$；

4) $\forall A\in L$ に対し，$\exists B\in L$, $\exists Y\in T\cup R$, $A\rho^*\tau B$, $A\rho^*\tau Y$, $\mathrm{First}'(B)\cap\mathrm{First}'(Y)\neq\phi$ なら，$R:=R\cup\{A\}$；$L:=L-\{A\}$

5) 4) を新たな非終端記号が $R$ に付け加えられなくなるまで繰り返す． ■

定理 4.2：$G=(N, T, P, S)$ は CFG で，アルゴリズム 4.1 のように $N$ を $L$, $R$ に分割する．$G$ が LR($k$) 文法なら，$G$ は LNR($k$) 文法である．

証明：紙幅の関係で省略する． ■

定理 4.2 と例 1 により，LR($k$) 文法は LNR($k$) 文法の真部分集合である．

次に，2スタックパーサ[1] が LNR($k$) 構文解析表を駆動する時間の効率を1スタックパーサが LR($k$) 構文解析表を駆動する時間効率と比べる．$G=(N, T, P, S)$ が曖昧性のない文脈自由文法なら，$\forall w\in L(G)$ に対し，$S\Rightarrow w$ の導出木がユニークに決められる．以下，$w$ の導出木の内部節点の数を $m$，葉節点の数を $n$ とする．$G$ が LR($k$) 文法で，$G$ の LR($k$) 構文解析表と $G$ の LNR($k$) 構文解析表 ($L=N$ とする) が作られたとする．次に $\forall w\in L(G)$ に対し，1スタックパーサと2スタックパーサそれぞれが LR($k$) と

LNR($k$) 構文解析表を駆動し，$w$ を解析する時間を比べる．

1スタックパーサが LR($k$) 構文解析表を駆動し，$w$ を解析する時間：1) $n$ 回シフトする；2) $m$ 回還元する；3) $m$ 回 goto する；4) $n+m$ 回 Action 表を調べる；5) $m$ 回 goto 表を調べる．

2スタックパーサが LNR($k$) 構文解析表を駆動し，$w$ を解析する時間：1) $n+m$ 回シフトする；2) $m$ 回還元する；3) $n+2m$ 回 Action 表を調べる．

この二者を比較すると，goto 動作はシフト動作と同じ程度の手間だから，LNR($k$) の 1) は LR($k$) の 1) と 3) の和と同じである．表を調べる時間を同一視すれば，LNR($k$) の 3) は LR($k$) の 4) と 5) の和と同じである．したがって，2スタックパーサが LNR($k$) 構文解析表を駆動する時間の効率は1スタックパーサが LR($k$) 構文解析表を駆動する時間効率と同じである．

次に，LR($k$) と LNR($k$) 構文解析を組み合わせることを提案する．これを拡張 LR($k$) 構文解析と呼び，対応する文法を拡張 LR($k$) 文法（または縮めて EXLR($k$) 文法）と呼ぶことにする．簡単に言えば，$G=(N, T, P, S)$ に対し，まず $G$ の LR($k$) 構文解析表ができるかどうか調べる．LR($k$) で解析できない部分だけに対し，導出順序を変えて LNR($k$) で解析するような解析器を作るのである．次に示すアルゴリズム 4.2 と 4.3 は以上の要求に合うようにしたアルゴリズムである．アルゴリズム 4.2 の（a），（b），（d）は $G$ の LR($k$) 構文解析器を作るアルゴリズムであるが，とりあえず LNR($k$) 構文解析のように先読みを計算する．（c）は LR($k$) で解析できない部分だけに対し，導出順序を変えて，LNR($k$) で解析するように修正する所である．（e）は LR($k$) で解析できる部分に対し，LNR($k$) の先読みを LR($k$) の先読みに変えるアルゴリズムである．

定義 4.3：LR($k$) 項または LNR($k$) 項のことを EXLR($k$) 項と呼ぶことにする． ■

定義 4.4：EXLR($k$) 項の集合 $I$ に対して，その EXLR 閉包 (Closure($I$)) とは，次のアルゴリズム 4.2 によって得られるものであるとする． ■

アルゴリズム 4.2：EXLR 閉包を求める

入力：EXLR($k$) 項の集合 $I$

出力：Closure($I$)

方法：（a） Closure($I$) := $I$

（b） $[A\rightarrow .\, B\beta, x]\in$Closure($I$) なら，$\{[B\rightarrow .\, \delta, y] \mid B\rightarrow\delta\in P, y=\mathrm{Rightof}_k(B, \beta x)\}$ の要素を Closure($I$) に加える．

（c） $\forall[A\rightarrow\alpha .,\ Bx]\in$Closure($I$) に対して，$\exists[D\rightarrow\beta_1 .\ \beta_2, y]\in$Closure($I$), $\mathrm{First}_k(\beta_2 y)\cap\mathrm{First}_k(Bx)\neq\phi$ なら，$\{[B\rightarrow .\, \delta, x] \mid B\rightarrow\delta\in P\}$ の要素を Closure($I$) に加える．

（d） 新たな項が Closure($I$) に付け加えられなくなるまで（b），（c）を繰り返す．

（e） $\forall[A\rightarrow\alpha .,\ Bx]\in$Closure($I$) に対して，$\forall[D\rightarrow\beta_1 .\ \beta_2, y]\in$Closure($I$) に対して，$\mathrm{First}_k(\beta_2 y)\cap\mathrm{First}_k(Bx)=\phi$ なら $[A\rightarrow\alpha., \mathrm{First}_k(Bx)]$ で $[A\rightarrow\alpha., Bx]$ を置き換える ■

定義 4.5：EXLR($k$) 項の集合 $I$ に対して，Goto($I$, $X$)=Closure($\{[A\rightarrow\alpha X .\ \beta, y] \mid [A\rightarrow\alpha .\ X\beta, y]\in I\}$) と定義する． ■

アルゴリズム 4.3：LR($k$) と LNR($k$) の組合せ

入力：$G=(N, T, P, S)$, $\gamma\in V^*$, 整数 $k$, $N$ の分割 $L, R$

出力：EXLR($k$) の構文解析の状態 $V_k(\gamma)$

方法：$\gamma=X_1X_2\cdots X_n$ とすると，$V_k(\varepsilon)$ から $V_k(X_1)$, $V_k(X_1X_2)$, $\cdots$, $V_k(X_1\cdots X_n)=V_k(\gamma)$ を作る

（1） $V_k(\varepsilon)$ := Closure($\{[S'\rightarrow .\ S\$, ]\}$)

（2） $V_k(X_1\cdots X_{i-1})$ が作られたと仮定する．$V_k(X_1\cdots X_i)$ := Goto($V_k(X_1\cdots X_{i-1}), X_i$) として $V_k(X_1\cdots X_i)$ を作る． ■

定理 4.3：$G=(N, T, P, S)$ は CFG で，アルゴリズム 3.2 のように $N$ を $L, R$ に分割する．$G$ が LR($k$) 文法なら，$G$ は EXLR($k$) 文法である．

証明：EXLR($k$) 構文解析の定義からすぐ分かる． ■

## 5. LNR($k$) または EXLR($k$) 文法で解決できる実用的な例

次に1章であげた LR($k$) 文法で解決できない実用的な例の解決の概略を述べる．

問題 1：$G2$ は明らかに LR($k$) 文法ではないが，アルゴリズム 4.1 によって，$L=N, R=\phi$ のように $N$ を分割すれば，$G2$ が LNR(1) 文法になる．ゆえに，id を変数の名称に還元するときに，変数の型が一意に決められる．

問題 2：$L=N, R=\phi$ として，LNR (1) 文法で解析すれば，シフト還元競合を還元還元競合に変えることができ，還元のときに効率のよいコードを選ぶこと

ができるし，ブロック状態が起こらない．LNR（1）構文解析表の一部は次のようである．

$I_0$= {[$S'\rightarrow$ $S\$$, ], [$S\rightarrow$. $A$, {\$} ], [$S\rightarrow$. $AA$, {\$} ],
[$A\rightarrow$. $a$, {\$, $A$ \$} ], [$A\rightarrow$. $b$, {\$, $A$ \$} ],
[$A\rightarrow$. $ab$, {\$, $A$ \$} ]}

$I_1$=Goto($I_0$, $a$)= {[$A\rightarrow a.$, {\$, $A$ \$} ],
[$A\rightarrow a.$ $b$, {\$, $A$ \$} ],
[$A\rightarrow$. $a$, {\$} ], [$A\rightarrow$. $b$, {\$} ], [$A\rightarrow$. $ab$, {\$} ]}

$I_2$=Goto($I_1$, $b$)= {[$A\rightarrow ab.$, {\$, $A$ \$} ], [$A\rightarrow b.$, {\$} ]}

状態 $I_2$ で還元還元競合があるが，入力が $ab$ であるときに，条件 $C$ が満たされれば，状態 $I_2$ の [$A\rightarrow ab.$, {\$, $A$ \$} ] で還元する．条件 $C$ が満たされなければ，まず状態 $I_2$ の [$A\rightarrow b.$, {\$} ] で還元し，その後状態 $I_1$ の [$A\rightarrow a.$, {\$, $A$ \$} ] で還元する．

問題 3：EXLR（1）文法で解決する方針は次のようである．まず $E$ と $A$ に合成属性 $t$[8] を付随させる．$t$ は $E$ や $A$ が使っているレジスタの数を示すとする．$L=N$, $R=\phi$ とすると，EXLR（1）構文解析表の一部は次のようになる．

$I_0$= {[$S'\rightarrow$. $E\$$, ], [$E\rightarrow$. $+EA$, {\$} ], [$E\rightarrow$. $+AE$, {\$} ],
[$E\rightarrow$. $A$, {\$} ], [$A\rightarrow$. $a$, {\$} ]}

$I_1$=Goto($I_0$, $+$)= {[$E\rightarrow +$. $EA$, {\$} ],
[$E\rightarrow +$. $AE$, {\$} ], [$E\rightarrow$. $+EA$, {$A$\$} ],
[$E\rightarrow$. $+AE$, {$A$\$} ], [$E\rightarrow$. $A$, {$A$\$} ],
[$A\rightarrow$. $a$, {$E$\$, $A$\$} ]}

$I_2$=Goto($I_1$, $A$)= {[$E\rightarrow +A.$ $E$, {\$} ], [$E\rightarrow A.$, {$A$\$} ],
[$E\rightarrow$. $+EA$, {\$} ], [$E\rightarrow$. $+AE$, {\$} ],
[$E\rightarrow$. $A$, {\$} ], [$A\rightarrow$. $a$, {\$} ]}

$I_3$=Goto($I_1$, $E$)= {[$E\rightarrow +E.$ $A$, {\$} ], [$A\rightarrow$. $a$, {\$} ]}

$I_4$=Goto($I_2$, $A$)= {[$E\rightarrow A.$, {\$} ]}

$I_5$=Goto($I_2$, $E$)= {[$E\rightarrow +AE.$, {\$} ]}

$I_6$=Goto($I_3$, $A$)= {[$E\rightarrow +EA.$, {\$} ]}

状態 $I_2$ で [$E\rightarrow A.$, {$A$\$} ] と [$E\rightarrow$. $A$, {\$} ] のシフト還元競合があるが，パーサスタックと入力スタックの先頭にある非終端記号の合成属性 $t$ の値によってシフト優先か還元優先かが決められる．$+aa$ を入力したときに，$+aa$ を $+A_1A_2$（注：添字は二つの非終端記号 $A$ を区別するためのものである）に還元するときに $A_1$ と $A_2$ の合成属性 $t$ が決められたとする．その後の解析は次のようである．

| パーサスタック | 入力列スタック |
|---|---|
| $A_1.t > A_2.t$ の場合 | |
| $I_0+I_1A_1I_2$（還元優先） | $A_2\$$ |
| $I_0+I_1$ | $EA_2\$$ |
| $I_0+I_1EI_3A_2I_6$ | \$ |
| $I_0$ | $E\$$ |

| パーサスタック | 入力列スタック |
|---|---|
| $A_1.t \leqq A_2.t$ の場合 | |
| $I_0+I_1A_1I_2$（シフト優先） | $A_2\$$ |
| $I_0+I_1A_1I_2A_2I_4$ | \$ |
| $I_0+I_1A_1I_2$ | $E\$$ |
| $I_0+I_1A_1I_2EI_5$ | \$ |
| $I_0$ | $E\$$ |

## 6. おわりに

非正準 LR ($k$) 構文解析の一般的なアルゴリズムとして LNR($k$) 構文解析および，それと LR($k$) 構文解析を組み合わせた EXLR($k$) 構文解析を提案し，この方法の正当性を証明した．BNF で文法を与えれば，それが LNR($k$) か EXLR($k$) 文法である限り，構文解析のプログラムを自動生成できる．

この方法の適用範囲は LR ($k$) より広く，第1章に上げたような従来 LR ($k$) 構文解析ではうまくできない問題も解決できる．我々は現在，LNR($k$), EXLR($k$) 文法を属性文法[8]に拡張し，EXLR（1）属性文法の処理系の作成について研究中であるが，以上のことを考えると，プログラミング言語の意味解析やコード自動生成に応用できる有望な方法であると期待される．

## 参考文献


1) Aho, A. V.: *The Theory of Parsing, Translation, and Compiling*, Volume I: Parsing, Prentice-Hall, Englewood Cliffs, N. J. (1972).
2) Aho, A. V., Sethi, R. and Ullman, J. D.: *Compilers Principles, Techniques, and Tools*, Addison-Wesley, Reading, Massachusetts (1986).
3) Colmerauer, A.: Total Precedence Relation, *J. ACM*, Vol. 17, No. 1, pp. 14-30 (1970).
4) Ganapathi, M. and Fischer, C. N.: Affix Grammar Driven Code Generation, *ACM Trans. Prog. Lang. Syst.*, Vol. 7, No. 4, pp. 560-599 (1985).

5) Graham, S. L.: Table-Driven Code Generation, *IEEE Computer*, Vol. 13, No. 8, pp. 25-34 (1980).
6) Nakata, I.: On Compiling Algorithms for Arithmetic Expressions, *Comm. ACM*, Vol. 10, No. 8, pp. 492-494 (1967).
7) 中田育男：コンパイラ，産業図書，東京 (1982).
8) 佐々政孝：属性文法，コンピュータソフトウェア，Vol. 3, No. 4, pp. 73-91 (1986).
9) Szymanski, T. G. and Williams, J. H.: Noncanonical Extensions of Bottom-up Parsing Techniques, *SIAM J. Comput.*, Vol. 5, No. 2, pp. 231-250 (1976).
10) Tai, K. C.: Noncanonical SLR(1) Grammars, *ACM Trans. Prog. Lang. Syst.*, Vol. 1, No. 2, pp. 295-320 (1979).
11) Williams, J. H.: Bounded Context Parsable Grammars, Tech. Rep. 72-127, Dept. of Computer Sci., Cornell Univ., Ithaca, N. Y. (1972).

## 付録 重要定理の証明

Lemma 3.1: $G=(N, T, P, S)$ が LNR($k$) 文法でなければ，次のような導出がある．

(1) $S' \doteqdot^{nr} \alpha A\eta \Rightarrow^{nr} \alpha\beta\eta$

(2) $S' \doteqdot^{nr} \gamma_1 B_1 \lambda_1 \Rightarrow^{nr} \gamma_1\delta_1\lambda_1 = \alpha\beta\mu_1$

(3) $\mathrm{Head}_k(\eta) \cap \mathrm{Head}_k(\mathrm{Left}_k(\mu_1)) \neq \phi$

(4) $\alpha \neq \gamma_1$ or $A \neq B_1$ or $\beta \neq \delta_1$

(5) $|\gamma_1\delta_1| \geqq |\alpha\beta|$

(6) $|\gamma_1| \leqq |\alpha\beta|$

証明：紙幅の関係で省略する． ■

定理 3.1: $G=(N, T, P, S)$ が LNR($k$) 文法である必要十分条件は「項 $[A \rightarrow \beta., x]$ が成長可能な語頭 $\alpha\beta$ に対して正当なら，同じ語頭 $\alpha_1\beta_1=\alpha\beta$ に対して，$\mathrm{Head}_k(x) \cap \mathrm{Head}_k(\mathrm{Left}_k(\beta_2 y)) \neq \phi$ であるような別の正当な項 $[A_1 \rightarrow \beta_1. \beta_2, y]$ が存在しない」ことである．ここで，$\beta_2$ は $\varepsilon$ かもしれない．

証明：必要条件であること：

項 $[A \rightarrow \beta., x]$ と $[A_1 \rightarrow \beta_1. \beta_2, y]$ は異なる項で，ともに成長可能な語頭 $\alpha\beta$ に対して正当で，$\mathrm{Head}_k(\mathrm{Left}_k(\beta_2 y)) \cap \mathrm{Head}_k(x) \neq \phi$ とする．項の定義 3.2 により，次の四つが成り立つ．

(7) $S' \doteqdot^{nr} \alpha A\gamma \Rightarrow^{nr} \alpha\beta\gamma \qquad x = \mathrm{Left}_k(\gamma)$

(8) $S' \doteqdot^{nr} \alpha_1 A_1 \gamma_1 \Rightarrow^{nr} \alpha_1\beta_1\beta_2\gamma_1 = \alpha\beta\beta_2\gamma_1$
$y = \mathrm{Left}_k(\gamma_1)$

(9) $\mathrm{Head}_k(x) \cap \mathrm{Head}_k(\mathrm{Left}_k(\beta_2 y)) \neq \phi$

(10) $A \neq A_1$ or $\beta \neq \beta_1$ or $\beta_2 \neq \varepsilon$

$\alpha_1\beta_1=\alpha\beta$ から，$\alpha=\alpha_1, A=A_1$ かつ $\beta=\beta_1\beta_2$ なら，$\beta_1=\beta$, $\beta_2=\varepsilon$ になる．これは(10)と矛盾する．ゆえに，$\alpha \neq \alpha_1$ or $A \neq A_1$ or $\beta \neq \beta_1\beta_2$ が成り立つ．定理 2.1，定理 2.2 と(9)により，$\mathrm{Head}_k(\gamma) \cap \mathrm{Head}_k(\mathrm{Left}_k(\beta_2\gamma_1)) \neq \phi$．ゆえに，$G$ は LNR($k$) 文法ではない．

十分条件であること：$G$ は LNR($k$) 文法でないとする．Lemma 3.1 により，次のような導出がある．

(11) $S' \doteqdot^{nr} \alpha A\eta \Rightarrow^{nr} \alpha\beta\eta$

(12) $S' \doteqdot^{nr} \gamma B\lambda \Rightarrow^{nr} \gamma\beta_1\beta_2\lambda = \alpha\beta\beta_2\lambda$

(13) $\mathrm{Head}_k(\eta) \cap \mathrm{Head}_k(\mathrm{Left}_k(\beta_2\lambda)) \neq \phi$

(14) $\alpha \neq \gamma$ or $A \neq B$ or $\beta \neq \beta_1\beta_2$

ゆえに，項 $[B \rightarrow \beta_1. \beta_2, y]$ と $[A \rightarrow \beta., x]$ は成長可能な語頭 $\alpha\beta$ に対して正当で，$x=\mathrm{Left}_k(\eta)$, $y=\mathrm{Left}_k(\lambda)$ である．定理 2.1，定理 2.2 と(13)より，$\mathrm{Head}_k(x) \cap \mathrm{Head}_k(\mathrm{Left}_k(\beta_2 y)) \neq \phi$ である．

$[B \rightarrow \beta_1. \beta_2, y]$ と $[A \rightarrow \beta., x]$ が同じ項なら，$B=A$, $\beta=\beta_1$, $\beta_2=\varepsilon$ になる．一方，$\gamma\beta_1=\alpha\beta$ から，$\alpha=\gamma$ になる．これは(14)と矛盾する．ゆえに，$[B \rightarrow \beta_1. \beta_2, y]$ と $[A \rightarrow \beta., x]$ は異なる項である． ■

定理 3.2: 項 $[A \rightarrow \beta_1. \beta_2, x]$ がアルゴリズム 3.1 の計算によって $V_k(\gamma)$ に加えられる必要十分条件は「$[A \rightarrow \beta_1. \beta_2, x]$ が成長可能な語頭 $\gamma$ に対して正当である」ことである．

証明：基底段階：$V_k(\varepsilon)$ の計算は明らかに正しい．言い換えれば，$V_k(\varepsilon)$ の中の項は全部語頭 $\varepsilon$ に対して正当である．逆に，項 $[A \rightarrow \beta_1. \beta_2, x]$ が語頭 $\varepsilon$ に対して正当なら，アルゴリズム 3.1 の計算によって，$[A \rightarrow \beta_1. \beta_2, x]$ は $V_k(\varepsilon)$ に加えられる．

帰納段階：十分条件であること：

成長可能な語頭 $X_1 \cdots X_{i-1}$ に対して正当な項はすべてアルゴリズム 3.1 によって，$V_k(X_1 \cdots X_{i-1})$ に加えられることが証明されたと仮定する．項 $[A \rightarrow \beta_1. \beta_2, x]$ が語頭 $X_1 \cdots X_i$ に対して正当なら，アルゴリズム 3.1 の計算により $[A \rightarrow \beta_1. \beta_2, x]$ が $V_k(X_1 \cdots X_i)$ に加えられることを証明する．

$[A \rightarrow \beta_1. \beta_2, x]$ が語頭 $X_1 \cdots X_i$ に対して正当なら，項の定義により，次のような導出がある．

(1) $S' \doteqdot^{nr} \alpha A\delta \Rightarrow^{nr} \alpha\beta_1\beta_2\delta = X_1 X_2 \cdots X_i\beta_2\delta$

ここで $x=\mathrm{Left}_k(\delta)$. 二つの場合に分けて証明する．

Case 1: $\beta_1=\beta_1' X_i$

この場合 $[A \rightarrow \beta_1'. X_i\beta_2, x]$ は語頭 $X_1 \cdots X_{i-1}$ に対して正当である．したがって帰納仮定により，$[A \rightarrow \beta_1'. X_i\beta_2, x]$ は $V_k(X_1 \cdots X_{i-1})$ に加えられるので，アルゴリズム 3.1 の (2a) により $[A \rightarrow \beta_1' X_i. \beta_2 x]$ は $V_k(X_1 \cdots X_i)$ に加えられる．

Case 2: $\beta_1=\varepsilon$

この場合，$\alpha=X_1\cdots X_i$. $X_i\in T\cup R$ or $A\in R$ である．

(1)の $A\delta$ を終端記号列まで導出してから，$\alpha$ の中の非終端記号は導出できる．だから，$X_i$ が導出される時点で，$X_i$ の左側の文字列が $X_1X_2\cdots X_{i-1}$ でなければ，(1)は nr 導出になれない．最後に $X_i$ を導出する生成規則は $B\to\delta_1X_i\delta_2$ とすると，(1)が次の(2)に変わる．

(2) $S\overset{*}{\Rightarrow}{}^{nr}\ \zeta_1B\zeta_2\Rightarrow{}^{nr}\ \zeta_1\delta_1X_i\delta_2\zeta_2=X_1\cdots X_i\delta_2\zeta_2$
$\overset{*}{\Rightarrow}{}^{nr}\ X_1X_2\cdots X_iA\delta\Rightarrow{}^{nr}\ X_1X_2\cdots X_i\beta_2\delta$

ゆえに，$[B\to\delta_1.\ X_i\delta_2,\ y]$ $(y=\mathrm{Left}_k(\zeta_2))$ は語頭 $X_1\cdots X_{i-1}$ に対して正当である．帰納仮定により $[B\to\delta_1.\ X_i\delta_2,\ y]$ は $V_k(X_1\cdots X_{i-1})$ に加えられるから，アルゴリズム 3.1 の (2 a) により，$[B\to\delta_1X_i.\ \delta_2,\ y]$ が $V_k(X_1\cdots X_i)$ に加えられる．$\delta_2\zeta_2\overset{*}{\Rightarrow}{}^{nr}A\delta$ から，次のような非終端記号列 $D_1,D_2,\cdots,D_m\in N$ と記号列 $\theta_2,\cdots,\theta_m\in V^*$ を探すことができる．$\mathrm{Head}(\delta_2\zeta_2)=D_1$, $A=D_m$, 生成規則 $D_i\to D_{i+1}\theta_{i+1}\in P$ for $1\leqq i\leqq m-1$. アルゴリズム 3.1 の (2 c) ($\delta_2=\varepsilon$ なら) と (2 b) を繰り返して実行すれば，$[A\to.\ \beta_2,\ x]$ は $V_k(X_1\cdots X_i)$ に加えられる．ただし，$x=\mathrm{Rightof}_k(A,\theta_m\cdots\theta_2y)$.

必要条件であること：アルゴリズム 3.1 によって，$V_k(X_1\cdots X_{i-1})$ に加えられた項は成長可能な語頭 $X_1\cdots X_{i-1}$ に対して正当であることが証明されたと仮定する．項 $[A\to\beta_1.\ \beta_2,\ x]$ がアルゴリズム 3.1 の計算によって $V_k(X_1\cdots X_i)$ に加えられたとすれば，$[A\to\beta_1.\ \beta_2,\ x]$ は成長可能な語頭 $X_1\cdots X_i$ に対して正当であることを証明する．$V_k(X_1\cdots X_i)$ に加えられる項の順序についてもう一度数学帰納法で証明する．

基底段階：$[A\to\beta_1.\ \beta_2,\ x]$ が $V_k(X_1\cdots X_i)$ に最初に加えられた項であるとする．この場合，$[A\to\beta_1.\ \beta_2,\ x]$ はアルゴリズム 3.1 の (2 a) で $V_k(X_1\cdots X_i)$ に加えられた項である．ゆえに，$\beta_1=\beta'X_i$ で $[A\to\beta'.\ X_i\beta_2,\ x]$ は $V_k(X_1\cdots X_{i-1})$ にあって，語頭 $X_1\cdots X_{i-1}$ に対して正当なので，次のような導出がある．

$S'\overset{*}{\Rightarrow}{}^{nr}\ \alpha A\delta\Rightarrow{}^{nr}\ \alpha\beta'X_i\beta_2\delta=X_1X_2\cdots X_{i-1}X_i\beta_2\delta$,
$x=\mathrm{Left}_k(\delta)$.

ゆえに，$[A\to\beta_1.\ \beta_2,\ x]$ は語頭 $X_1\cdots X_i$ に対して正当である．

帰納段階：Case 1：$[A\to\beta_1.\ \beta_2,\ x]$ はアルゴリズム 3.1 の (2 a) で $V_k(X_1\cdots X_i)$ に加えられた項であるとする．この場合の証明は基底段階と同じである．

Case 2：$[A\to\beta_1.\ \beta_2,\ x]$ は (2 b) で $V_k(X_1\cdots X_i)$ に加えられた項であるとする．この場合，$\beta_1=\varepsilon$ で，$[B\to\zeta.\ A\eta,\ y]\in V_k(X_1\cdots X_i)$, $X_i\in T\cup R$ or $A\in R$ なるものが存在する．帰納仮定により，$[B\to\zeta.\ A\eta,\ y]$ は語頭 $X_1\cdots X_i$ に対して正当である．したがって，次のような導出がある．

$S'\overset{*}{\Rightarrow}{}^{nr}\ \alpha B\delta\Rightarrow{}^{nr}\ \alpha\zeta A\eta\delta=X_1\cdots X_iA\eta\delta$
$\overset{*}{\Rightarrow}{}^{nr}\ X_1\cdots X_iA\lambda\Rightarrow{}^{nr}\ X_1\cdots X_i\beta_2\lambda$

ここで，$y=\mathrm{Left}_k(\delta)$, $x=\mathrm{Rightof}_k(A,\eta y)$.

ゆえに，$[A\to.\ \beta_2,\mathrm{Left}_k(\lambda)]$ は語頭 $X_1\cdots X_i$ に対して正当である．次に $x=\mathrm{Left}_k(\lambda)$ を証明する．$A\in R$ なら，nr 導出の定義により，$\lambda\in T^*$ であるから，

$\mathrm{Left}_k(\lambda)=\mathrm{First}_k(\lambda)=\mathrm{First}_k(\eta\delta)=\mathrm{First}_k(\eta y)$
$=\mathrm{Rightof}_k(A,\eta y)=x$.

$A\in L$ なら，nr 導出の定義により，$\mathrm{Left}_k(\lambda)=\mathrm{Left}_k(\eta\delta)$ であるから，$\mathrm{Left}_k(\lambda)=\mathrm{Left}_k(\eta\delta)=\mathrm{Left}_k(\eta y)=\mathrm{Rightof}_k(A,\eta y)=x$.

Case 3：$[B\to\beta_1.\ \beta_2,\ x]$ は (2 c) で $V_k(X_1\cdots X_i)$ に加えられた項であるとする．この場合，$\beta_1=\varepsilon$ で，$[A\to\zeta.,Bx]\in V_k(X_1\cdots X_i)$, $X_i\in T\cup R$ なるものが存在する．帰納仮定により，$[A\to\zeta.,Bx]$ は語頭 $X_1\cdots X_i$ に対して正当である．したがって，次のような導出がある．

$S'\overset{*}{\Rightarrow}{}^{nr}\ \alpha AB\delta\Rightarrow{}^{nr}\ \alpha\zeta B\delta=X_1\cdots X_iB\delta$
$\Rightarrow{}^{nr}\ X_1X_2\cdots X_i\beta_2\delta$

ここで，$Bx=\mathrm{Left}_k(B\delta)$. ゆえに，$B\in L$, $x=\mathrm{Left}_k(\delta)$ で，$[B\to.\ \beta_2,\ x]$ は語頭 $X_1\cdots X_i$ に対して正当である． ■

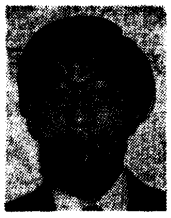
(昭和 62 年 5 月 7 日受付)
(昭和 63 年 1 月 19 日採録)

**張　又普**（正会員）

1953 年生．1982 年 2 月中国西北大学電子計算机系卒業．1985 年電気通信大学大学院修士課程修了．工学修士．現在筑波大学大学院博士課程工学研究科に在学中．プログラミング言語，属性文法，コンパイラの生成系の研究に興味を持っている．ソフトウェア科学会会員．

**中田 育男**（正会員）

1935年生．1958年東京大学理学部数学科卒業．1960年同大学院修士課程修了．1960～1979年(株)日立製作所中央研究所．同システム開発研究所勤務．1979年4月より筑波大学電子・情報工学系教授．理学博士．プログラム言語，言語処理系，ソフトウェア工学などに興味を持っている．著書「コンパイラ」(産業図書)．日本ソフトウェア科学会，電子情報通信学会，ACM，IEEE 各会員．



**佐々 政孝**（正会員）

1948年生．1970年東京大学理学部物理学科卒業．1974年同理学系研究科博士課程中退，東京工業大学理学部情報科学科助手となる．1981年筑波大学電子情報工学系講師，1986年より同助教授．理学博士．プログラミング言語，属性文法，コンパイラ，コンパイラ生成系，プログラミング支援系に興味を持っている．1981年本学会論文賞受賞．ソフトウェア基礎論研究会幹事．ソフトウェア科学会，ACM，IEEE 各会員．