3H－9

ＬＬ(２)文法から強ＬＬ(２)文法への書き換えアルゴリズム

広瀬卓也† 竹内淑子‡ 吉田敬一†
静岡大学大学院理工学研究科†
浜松職業能力開発短期大学校‡

1 はじめに

文法の大小と言語の大小は異なる。つまり文法クラスが異なるものでも同じ言語を生成する場合がありうるということはよく知られている。同じ言語に対しては、文法クラスの小さな文法の方が解析表の大きさや、解析速度、実装の容易さから有利である。

そこで本研究ではＬＬ(２)文法から強ＬＬ(２)文法への書き換えアルゴリズムを提案する。

2 基本定義と記法

［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の要素をA、B、C...、Σの要素をa、b、c...、$N \cup \Sigma$の要素をX、Y、Zで表す。また、Σ^*の要素をｓ、ｔ、ｕ...で表し、$(N \cup \Sigma)^*$の要素をα、β、γ...で表す。特にε、ϕはそれぞれ空列、空集合を表す。ε、ϕ以外のこれらの記号は添字をつけて用いることもある。またp、qは生成規則番号である。

［定義２］ 集合$FIRST_k(\alpha)$は以下で定義される。

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

ただし、α、$\beta \in (N \cup \Sigma)^*$、$u \in \Sigma^*$、$|u|$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。

［定義３］ 集合$FOLLOW_k(A)$は次のように定義される。

文脈自由文法$G=(N, \Sigma, P, S)$において

$S \overset{*}{\Rightarrow} u_i A \xi_i$なる導出に対し

$FOLLOW_k(A)=\cup_i FIRST_k(\xi_i)$

［定義４］ L_1、L_2をΣ^*の部分集合とするとき、演算子$\oplus_k$は以下のように定義される。

$$L_1 \oplus_k L_2 = \{w \mid \text{ある} x \in L_1, y \in L_2 \text{に対して、} \| xy \| \leq k \text{ならば} w=xy, \| xy \| > k \text{ならば} w=u, \text{ただし} xy=uv, \| u \| =k\}$$

［定義５］ 文脈自由文法$G=(N, \Sigma, P, S)$において、$A \rightarrow \alpha$、$A \rightarrow \beta$を相異なる生成規則とするとき

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

に対して

$$(FIRST_k(\alpha) \oplus_k FIRST_k(v)) \cap (FIRST_k(\beta) \oplus_k FIRST_k(v)) = \phi$$

が成り立つとき、GはＬＬ(ｋ)文法であるという。

［定義６］ 文脈自由文法$G=(N, \Sigma, P, S)$において、$A \rightarrow \alpha$、$A \rightarrow \beta$を相異なる生成規則とするとき

$$(FIRST_k(\alpha) \oplus_k FOLLOW_k(A)) \cap (FIRST_k(\beta) \oplus_k FOLLOW_k(A)) = \phi$$

が成り立つとき、Gは強ＬＬ(２)文法であるという。

［定義７］ 集合Q、Rは以下で定義される。

$Q=\{(A, p) \mid A \overset{*}{\Rightarrow} \alpha \overset{*}{\Rightarrow} \varepsilon\}$

$R=\{(A, a, p) \mid A \overset{*}{\Rightarrow} \alpha \overset{*}{\Rightarrow} a\}$

3 文法の書き換え

3.1 書き換えの必要性の検討

与えられる文法はＬＬ(２)文法であるとする。この中にはすでに強ＬＬ(２)文法の条件を満足しているものとそうでないものがある。前者については、書き換えの必要がないのは当然である。そこで、書き換えの必要のない場合をいちいち強ＬＬ(２)文法の定義に当てはめることなく、検出するために次のような[定理]を用意した。

An Algorithm for the conversion of a LL(2) Grammer into a Strong-LL(2)Grammer
†Takuya Hirose
‡Yoshiko Takeuchi
†Keiichi Yoshida
†Graduate School of Science and Engineering, Shizuoka University
‡Hamamatsu Polytechnic college

［定理］　ＬＬ(２)文法$G=(N、\Sigma、P、S)$において文法Gが強ＬＬ(２)文法となるのは

$A\rightarrow\alpha、A\rightarrow\beta、\alpha\neq\beta、\alpha\overset{*}{\Rightarrow}w、\beta\overset{*}{\Rightarrow}v$

とするとき、$|w|\geqq 2$かつ$|v|\geqq 0$、もしくは$|w|=1$かつ$|v|=1$の場合に限られる。

［証明］　省略

この［定理］を書き換えアルゴリズムの中に組み込むことにより書き換え効率を上げることができる。

３．２　書き換えアルゴリズム

$w=a$、$v=\varepsilon$としたとき強ＬＬ(２)の定義を適用するとその左辺は

$$(FIRST_2(\alpha)\oplus_2 FOLLOW_2(A))\cap(FIRST_2(\beta)\oplus_2 FOLLOW_2(A))$$
$$=(\{a\})\oplus_2 FOLLOW_2(A))\cap FOLLOW_2(A)$$

となる。ここで前述の定理から左辺がϕでないときのみ書き換えを行えばよい。ここで書き換えの必要性がある場所を判別するために必要な情報は上式から

- $Q=\{(A、p)\mid A\overset{*}{\Rightarrow}\alpha\overset{*}{\Rightarrow}\varepsilon\}$
- $R=\{(A、a、p)\mid A\overset{*}{\Rightarrow}\alpha\overset{*}{\Rightarrow}a\}$
- *FOLLOW*集合

の三つに限定することができる。本研究では上記のQ集合とR集合、*FOLLOW*表を用いて書き換えが必要な場所を特定する。

本稿では、書き換えアルゴリズムで用いる*FOLLOW*表を以下のように定義する。

［定義８］　*FOLLOW*表は$N\times(N\cup\Sigma)$型の表としA行B列を$FL(A、B)$で表す。$FL(A、B)$は要素として$X(p)$を持つ。ここで$X\in FOLLOW_1(B)$でありpは$A\overset{*}{\Rightarrow}\alpha\overset{*}{\Rightarrow}X$となる場合の生成規則番号である。

以下に書き換えアルゴリズムを示す。

```
begin
  i←0;
  j←0;
  for each A→Y1Y2...Yn do
    if Yi∈N then
      if Yi∈R∩Q
      such that R∋(Yi, ah, p), Q∋(Yi, q)
        h←0;
        for each ah do
          if
            ({ah} ⊕2FOLLOW2(A))
                  ∩FOLLOW2(A)=φ
          then
            h←h+1;
            continue;
          else
            Yij = Yi;
            Aj = A;
            h←h+1;
            j←j+1;
            continue;
          endif
        endfor
      endif
    endif
    i←i+1;
  endfor
```

以上のアルゴリズムを用いて書き換えプログラムを実装し、いくつかのサンプル文法に対し、ＬＬ(２)文法から強ＬＬ(２)文法への書き換えを行うことができた。

４　評価

本研究では以下のことが達成できた。

- 前述の定理の発見とその証明
- 書き換え対象となる非終端記号の発見
- 書き換えアルゴリズムの構築とその実装

今後の課題として以下のことが挙げられる。

- 本質的にＬＬ(２)文法であるもの、つまり強ＬＬ(２)文法に書き換えできないものについての検討
- テーブルを用いずに書き換えを行う場合との比較

５　参考文献

［1］ 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」第31号，第6月号

［2］ 吉田・竹内：準LL(2)文法に対する解析表の構造と解析アルゴリズム，情報処理学会論文誌」第31号，第9月号

［3］ 井上謙蔵：コンパイラ「プログラム言語処理の基礎」pp77-117　丸善

［4］ Aho, A. V. and Ullman, J. D. : *The Theory of Parsing, Translation and Compiling, Vol. 1 pp334-361* Prentice-Hall(1972)