ショート・ノート

パージングテーブルによる弱順位パーザの補足*

山本米雄** 青江順一** 島田良作**

Abstract

This letter points out the unsuitable and redundant versions of weak precedence parsers using parsing tables, and presents the methods for improving them. The improved algorithms for parsing and constructing parsers become more efficient.

1. まえがき

パージングテーブルによる弱順位パーザ[1]（パーザ π と呼ぶ）は弱順位パーザの連続還元動作の特長を利用し，スパーズな弱順位行列を表形式に縮小したものである．しかし，このパーザ π を実用のプログラミング言語に適用する場合，いくつかの問題点が生じる．従って，本論文の目的はその問題点の指摘と改善法を与えることにある．

2. パージングテーブルの改善

まず，用語の説明を行うが，その詳細については文献 1) を参照されたい．

弱順位文法を $G=(V_N, V_T, P, S)$ で表わす．ここで，V_N, V_T, P はそれぞれ非終端語，終端語，生成規則の有限集合を表わし，$S(\in V_N)$ は始記号を表わす．但し，以後は $G=(V_N \cup \{S_0\}, V_T \cup \{\$\}, P \cup \{S_0 \to S\})$ として使用する．\$ は端記号 (endmarker) である．順位関係 $\doteq$, $<$, $>$ に対して，$\leqq = \doteq \cup <$ を使用する．この順位関係を求めるためにグラフ RG, LG が文献 1) で使用されているが，ここでは RG を特に使用するので，その説明を行う．RG のノード $B(\in V_N)$ の子孫の集合を $R^+(B)$ で表わし，そのノード B の右側には $\mu_R(B)=\{b \mid a \leqq b,\ b \in V_T\}$ なる隣接集合が加えられている．そして，$a \in R^+(B)$ 且つ $b \in \mu_R(B)$ ならば，$a > b$ なる順位関係が成立する．

パーザ π はシフト動作か連続還元動作かを決定するパージングテーブル $T_S(G)$ と連続還元動作を実行するパージングテーブル $T_R(G)$ を使用する．

まず，$T_S(G)$ の問題点とその改善法を説明する．文献 1) では $T_S(G)$ の状態 S の内容を

〔入力記号，入口状態，出口状態，行先状態〕

で表わしている．但し，入口，出口状態はそれぞれ $T_R(G)$ で連続還元動作を実行するための初期状態と終了状態である．パーザ π はこの連続還元動作の後必ず1回のシフト動作を実行して $T_S(G)$ の状態に帰るが，この状態が行先状態である．また，連続還元動作かシフト動作の決定は入口，出口状態が存在するか否かで行う．しかし，$T_S(G)$ のすべての状態は V_T の要素に対応して存在し，また，$T_R(G)$ の入口状態も $a > b$ なる関係をもつ V_T の要素 a に対応して存在するので，この入口，行先状態はスタックのトップ記号の情報によって決定できる．従って，入口，行先状態の項を状態 S の内容より除く，また，文献 1) の $T_S(G)$ の構成手順 (1-1) では $a \in R^+(B)$, $b \in \mu_R(B)$ ならば，$S_a=(b, R_a, R_B', S_b)$ とし，出口状態 R_B' が一意的に決定できない場合，それらの出口状態を1つにまとめるとしているが，**Fig. 1** のような RG の場合，パーザ π は誤った解析に陥いる．

すなわち，スタックのトップ記号が c，入力記号が

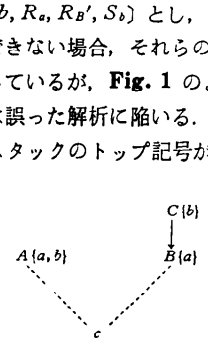

Fig. 1 Example of *RG*.

* Supplement to Weak Precedence Parsers Using Parsing Tables by Yoneo YAMAMOTO, Jun-ichi AOE, and Ryousaku SHIMADA (Faculty of Engineering, Tokushima University).

** 徳島大学工学部情報工学科

b で連続還元動作が始まったとすれば，スタックのトップ記号が A あるいは C になるまで還元動作が連続するべきであるが，手順 (1-1) では出口状態が $R_A'=R_B'=R_C'$ とまとめられるので，スタックのトップ記号が B になって還元動作が終了する場合がある．従って，この矛盾を避けるために，次態 S の内容は

〔入力記号，出口状態の集合〕　(1)

とすべきである．

次に，$T_R(G)$ の問題点と改善法について説明する．文献 1) では $T_R(G)$ の状態 R の内容を

〔スタック記号，動作，出力，行先状態〕

で表わしている．但し，動作には→，←A，↔A があって，それぞれスタック記号を pop up すること，記号 A を push down すること，スタック記号を pop up して記号 A を push down することを表わす．出力は生成規則番号，行先状態は $T_R(G)$ で次に進むべき状態を表わす．実用のプログラミング言語の生成規則の右辺は数個の記号列で構成されている場合が多いので，各スタック記号に対して，$T_R(G)$ の状態を作るのはかえって記憶量の増加をまねく．従って，本論文での状態 R では pop up すべきスタック記号列に対して，出力，行先状態を与える方法をとり，状態 R の内容を

〔スタック記号列，出力，行先状態〕　(2)

で表わす．

次に，改善された $T_S(G)$, $T_R(G)$ の構成法を示すが，$T_S(G)$ の状態 S と $T_R(G)$ の状態 R はそれぞれ上記 (1), (2) の内容を要素にもつ集合として記述する．また，p 番目の生成規則を $A_p\to\alpha_pX_p$ で表わし，$\boldsymbol{R}=\lessdot\cup\gtrdot$ とする．

(手順A)　$T_S(G)$ と $T_R(G)$ の構成法

(A-1)　$a\in V_T$ なるすべての a に対して，状態 S_a を作り，$a\boldsymbol{R}b$ なるすべての b に対して，〔b, $N_{a,b}$〕$\in S_a$ とする．ここで，$N_{a,b}$ は出口状態の集合であって，$N_{a,b}=\{R_A \mid a\in R^+(A)$ 且つ $b\in\mu_R(A)\}$ として構成される．

(A-2)　$A_p\to\alpha_pX_p\in P$ なるすべての $A_p\to\alpha_pX_p$ に対して，〔α_p, p, R_{A_p}〕$\in R_{X_p}$ とする．(手順終)

次に，改善されたパージングテーブルによるパーザ π のパージングアルゴリズムを **Fig. 2** に示す．但し Fig. 2 の TOS, IN はそれぞれスタックのトップ記号と入力記号を表わし，SHIFT, REDUCE はそれぞれ入力記号をスタックに push down し，入力を1つ左へ動かす操作と $T_R(G)$ における pop up 操作を表わす．R_{ENT} は $T_S(G)$ から $T_R(G)$ へ状態が遷移する場合の入口状態を表わす．また，$T_R(G)$ の状態において，pop up すべきスタック記号列は最長一致法に従うものとする．空集合を ϕ で表わす．

```
repeat; {begin with TOS=$}
    repeat SHIFT until [IN, N_TOS,IN]∈S_TOS and N_TOS,IN≠φ
      if [IN, N_TOS,IN]∉S_TOS then shift error
      else
        begin
          TEMP:=N_TOS,IN; {since REDUCE changes TOS}
          R_ENT:=R_TOS
          pop up TOS
            repeat REDUCE
            until R_X∈TEMP; {R_X is current state in T_R(G)}
          push down X
        end
until TOS=S_0
```

Fig. 2　Parsing algorithm.

手順 (A-1) より，$a, b\in V_T$ なる a, b に対して，次の対応関係が明らかに成立する．

$$N_{a,b}=\phi\rightleftarrows a\lessdot b,\quad N_{a,b}\neq\phi\rightleftarrows a\gtrdot b$$

従って，$T_S(G)$ は $V_T\times V_T$ 上の正しい順位関係の情報を含む．また，弱順位文法では $\lessdot\cap\gtrdot=\phi$ であるから，Fig. 2 の REDUCE ループは $N_{TOS,IN}$ の情報によって正しいループとなる．

〈例〉 次の生成規則をもつ文法 G_1 の $T_S(G_1)$ と $T_R(G_1)$ の状態の内容を示す．但し，ε は空記号を表わす．

0. $E_0\to E$, 1. $E\to E+T$, 2. $E\to T$, 3. $T\to T*F$, 4. $T\to F$, 5. $F\to(E)$, 6. $F\to a$

$S_a=\{(*, \{R_T\}), (), \{R_E\}), (+, \{R_E\}), (\$, \{R_{E_0}\})\}$

$S_*=\{(a, \phi), ((, \phi)\}$

$S_)=S_a$, $S_(=S_+=S_\$=S_*$

$R_E=\{(\varepsilon, 0, R_{E_0})\}$

$R_T=\{(E+, 1, R_E), (\varepsilon, 2, R_E)\}$

$R_F=\{(T*, 3, R_T), (\varepsilon, 4, R_T)\}$

$R_)=\{((E, 5, R_F)\}$

$R_a=\{(\varepsilon, 6, R_F)\}$

3. むすび

以上の改善により，文献 1) のパージングテーブルにおける冗長な状態およびその内容が除去された．また，このパージングアルゴリズムでは $T_R(G)$ の連続還元動作の際，生成規則の右辺の右端の記号をスタックに入れなくてよいので，$T_R(G)$ における動作を pop up だけに限定できる．従って，$T_R(G)$ の構成，操作法の簡単化および情報量の軽減にも役立っている．

弱順位行列に相当する $T_S(G)$ の状態数は V_T の要

Table 1 Each value of grammar G_2 and G_3.

各値 / 文法	Pの要素数	V_N の要素数	V_T の要素数	n_{IN}	n_{EXT}
G_2	118	61	47	4.8	1.4
G_3	126	73	44	5.9	1.3

Table 2 Data structure of $T_S(G)$.

(a) シフト状態

b_1	b_2	.....	b_n

(b) シフト還元状態

b	R_{X1}	R_{X2}	.....	R_{Xm}

素数（n_T とする）と同じであるが，$T_S(G)$ の占有する記憶量は各状態に現れうる入力記号の平均数を n_{IN}，各入力記号に対する空でない出口状態の集合の平均要素数を n_{EXT} とすれば，$n_T \times n_{IN} \times n_{EXT}$ のオーダとなる．この点に対して，我々は XPL[3]（文法 G_2 とする），ALGOL-JIS 3000[2]（文法 G_3 とする）の n_{IN}, n_{EXT} を計算した（**Table 1** 参照）．

Table 1 より文法 G_2, G_3 の $n_{IN} \times n_{EXT}$ はそれぞれ 6.7, 7.7 程度となるので，$T_S(G_2), T_S(G_3)$ の規模は n_T の線形のオーダになるといえる．

$[a, N_{a,b}] \in S$ なるすべての $N_{a,b}$ が空である状態 S をシフト状態と呼び，それ以外の $T_S(G)$ の状態をシフト還元状態と呼ぶとき，それぞれの状態のデータ構造を **Table 2** のように考える*．但し，シフト状態は入力記号 $b_1, b_2, \cdots, b_n$；$n \geq 1$ をもつ場合であり，シフト還元状態は $[b, \{R_{X_1}, R_{X_2}, \cdots, R_{X_m}\}]$；$m \geq 0$** なる1エントリーの場合を示す．

Table 2 のデータ構造の1項目を8ビットで与え，各シフト還元状態の同一エントリーは共有できるものとして，$T_S(G_2), T_S(G_3)$ の記憶量を計算するとそれぞれ 3072 ビット，2752 ビットとなった．文法 G_2, G_3 に対する弱順位行列がそれぞれ 10152 ビット，10296 ビットであるから，$T_S(G_2), T_S(G_3)$ はかなり縮小されていることが分かる．

また，$T_R(G)$ における解析時間も n_{EXT} の値に関係しているが***，文法 G_2, G_3 の n_{EXT} は1に近い値となっているので，あまり問題とはならない．

* このように $T_S(G)$ の状態を区別すれば，シフト状態は入力記号の配列だけを示せばよいので，都合がよい．なお，$T_S(G_2), T_S(G_3)$ に対するシフト状態の数はそれぞれ 28, 14 となった．

** $m=0$ の場合は出口状態の集合が空であるという情報を R_{X_1} の場所に入れるものとする．

*** 出口状態の確認時間が n_{EXT} の値によって決まる．

参 考 文 献

1) 青江，山本，原田，島田：パージングテーブルによる弱順位パーザの構成法，情報処理，Vol. 18, No. 5, pp. 438～444 (1977).

2) 海尻，打浪，手塚：拡張弱順位関数，情報処理，Vol. 18, No. 6, pp. 542～549 (1977).

3) W. M. Mckeeman, J. J. Horning and D. B. Wortman : A compiler generator, Prentice-Hall (1970).

(昭和 52 年 10 月 22 日受付)
(昭和 53 年 1 月 30 日再受付)