論　文

拡張弱順位関数*

海尻賢二*** 打浪清一** 手塚慶一**

Abstract

Weak precedence grammars are useful tools for description of the syntax of programming languages. We have considered the weak precedence functions (WPF) with an error relation (?), and this method is a modified version of Aho's error detecting simple precedence functions. We represent the three weak precedence relations ($\lesssim, \gtrsim, ?$) by two pairs of functions, and we call them the Extended Weak Precedence Functions (EWPF).

We define the semi-strong equivalence (SSE) to reduce the number of error relations, and show an efficient algorithm to compute the EWPF in SSE level. We have shown an algorithm which transforms an unambiguous context free grammar to a grammar which has the EWPF. This algorithm is easy to implement.

As an example, we have shown the EWPF for JIS ALGOL 3000 using it.

1. まえがき

順位パーザ[1]を実際にコンパイラに利用するという観点から，誤り検出を考慮した順位関数の研究がAhoらによってなされている[7]．しかしAhoらの方法では依然として1対の順位関数で3種の順位関係を表現しようとするために関数化の能力は低い．またそこで提案されている2つの等価性の間には飛躍がある．そこで筆者らは文献6)において1対の順位関数では2種の関係のみを表現し，2対の順位関数の組み合わせとして4種の単純順位関係（$\lessdot$, $\gtrdot$, $\doteq$, 誤り）を表現する方法を提案し，またその実現のためにAhoらの提案した2つの等価性の中間のものとして準強等価を導入し，関数の実現法を示した．

本論文においてはこの方法を更に発展させ，弱順位パーザ[3),4)]に対して適用する．弱順位パーザでは3種の弱順位関係（$\lesssim$, $\gtrsim$, 誤り）を2対の関数で表現する（拡張弱順位関数）．その実現のために文献6)と同様に弱順位パーザ間の等価性として準強等価を導入し，そのもとでの拡張弱順位関数の実現法を示す．また準強等価で拡張弱順位関数が存在しないとき，極小の誤り要素を無視して関数化を可能にする方法を述べ，あわせて関数を使って還元を早く行う方法についても述べる．最後に以上の方法に基づいて作成したALGOL 3000[2)]の弱順位パーザについて述べ，弱順位行列中の組み合わせ禁止でない誤り要素の内，78%を保存して関数化が可能なことを示す．

2. 弱順位パーザ，及びその等価性

本章では弱順位文法，弱順位パーザについて述べ，あわせてパーザ間の誤り検出能力に関する等価性についても述べる．なお特に定義しない記号及び記法は文献8)に拠る．

[定義 1] 弱順位文法

文法 $G=\langle N, T, P, S\rangle$ が次の条件を満足するとき，弱順位文法と呼ぶ．但し$\lessdot$, $\doteq$, $\gtrdot$はwirthとweberの3つの順位関係とする[1)]．また T, N, P, S はそれぞれ終端語の集合，非終端語の集合，生成規則の集合及び始記号を表わす．

（1） 順位関係($\lessdot$, $\doteq$)と($\gtrdot$)は互いに両立しない．

（2） $A\to\alpha X\beta$, $B\to\beta$ が G の2つの生成規則と

* Extended Weak Precedence Functions by Kenji KAIJIRI (Faculty of Engineering, Shinshu University), Seiichi UCHINAMI and Yoshikazu TEZUKA (Department of Communication Engineering, Faculty of Engineering, Osaka University)

** 大阪大学工学部通信工学科
*** 信州大学工学部

すると $X \leqq B$ は成立しない．($X \leqq B$ は $X<B$ もしくは $X \doteqdot B$ を表わす．) ☒

弱順位パーザは弱順位関係*と生成規則の集合 P により駆動され，その姿態は次の2つ組で表わされる．

$$[\alpha, \beta]$$

α: スタックの内容　　$\alpha \in \$(N \cup T)^*$

β: 未だ走査していない入力例　　$\beta \in T^*\$$

但し　$\$ \notin (N \cup T)$

パーザ π は各弱順位関係に従い次の動作を行う．

$X \leqq Y$　　$[\alpha X, Y\beta] \vdash_\pi [\alpha XY, \beta]$

$X \geqq Y$　　$[\alpha_1 \alpha_2 X, Y\beta] \vdash_\pi [\alpha_1 A, Y\beta]$

但し $A \to \alpha_2 X$ は G の生成規則の中で，その右辺が $\alpha_1\alpha_2 X$ の postfix として含まれる最長の規則．

[定義 2] 正準弱順位行列，弱順位行列

弱順位文法 G 上の順位関係より以下の対応で作った行列 M_c を G の正準弱順位行列と呼ぶ．

$$X \leqq Y \longleftrightarrow M_c[XY]=4$$
$$X \geqq Y \longleftrightarrow M_c[XY]=5$$
$$X ? Y \longleftrightarrow M_c[XY]=6$$

? は ($\leqq$, $\geqq$) 以外の関係，即ち誤り関係である．また誤り関係について特にその値を規定しない行列を単に弱順位行列と呼ぶ．☒

弱順位パーザを弱順位行列 M と生成規則の集合 P によって (M, P) と表わすならば，(M, P) の誤り検出能力は M によって異なる．そこでこの M による誤り検出能力の判定規準を与えるために，準強等価を定義し，(M_c, P) と (M, P) が準強等価であるための M の必要十分条件について述べる．

[定義 3] 準強等価

π_1 と π_2 を弱順位文法 G に対する2つのパーザとする．次の2つの条件が満足されるならば π_1 と π_2 は準強等価であるという．

(i) $[\$, w\$] \vdash_{\pi_1} Q_1 \vdash_{\pi_1} \cdots \vdash_{\pi_1} Q_n \vdash_{\pi_1}$ 受理ならば，$[\$, w\$] \vdash_{\pi_2} Q_1 \vdash_{\pi_2} \cdots \vdash_{\pi_2} Q_n \vdash_{\pi_2}$ 受理であり，また逆も真である．但し Q_i はパーザの途中の姿態である．

(ii) $[\$, w\$] \vdash_{\pi_1} Q_1 \vdash_{\pi_1} \cdots \vdash_{\pi_1} Q_n \vdash_{\pi_1}$ 誤りならば，$[\$, w\$] \vdash_{\pi_2} Q_1 \vdash_{\pi_2} \cdots \vdash_{\pi_2} Q_n \vdash_{\pi_2}$ 誤りとなり逆も真である．☒

準強等価なパーザは同じ姿態で誤りをみつける．但し誤りの種類は必ずしも同じではない．

[定理 1] $G=\langle N, T, P, S\rangle$ を弱順位文法，M_c を G の正準弱順位行列，M を G のある弱順位行列とする．(M, P) が (M_c, P) に準強等価であるための必要十分条件は次の(1)～(4)の条件が成り立つことである．

(1) 任意の $X, Y (X \in (N \cup T \cup \{\$\}), Y \in (T \cup \{\$\}))$ に対して $M_c[XY] \neq 6$ ならば $M[XY]=M_c[XY]$ である．

(2) 任意の $a, b \in (T \cup \{\$\})$ に対して $M_c[ba]=6$ ならば $M[ba]=6$ もしくは $A \to ab$ なる形の生成規則が存在せず，かつ $M[ba]=5$ である．

(3) 任意の $B \in N$, $a \in (T \cup \{\$\})$ に対して $M_c[ba]=6$ ならば i) $M[Ba]=6$ もしくは ii) $B \to \alpha C$ なる規則が存在するすべての C に対して $M[Ca] \neq 5$ もしくは iii) $A \to \beta B$ なる形の生成規則が存在せず，かつ $M[Ba]=5$ である．☒

(証明は文献6)の方法に準ずる．)

定理1の(3)の ii) 条件を満す要素の対 (B, a) はいかなるときにもパーザにより参照されない対であり，4, 5, 6 のどの値をとってもかまわない．以後これを組み合わせ禁止と呼び，順位行列では7で表わす．また同様に(2)の後者及び(3)の iii) の条件を満す要素対は5でも6でもよい．これは9で表わす．そしてこの条件を満たす誤り要素をすべて7及び9でおきかえた正準弱順位行列を準強等価弱順位行列と呼び，M_{SSE} で表わす．M_{SSE} は正準パーザに準強等価な弱順位パーザ (M, P) を構成する順位行列 M を一般的に表わしたものであり，その7及び9の値を持つ要素をそれぞれ6か5か4，6か5のどれかの値にかえることにより個々の準強等価な弱順位行列が得られる．

3. 拡張弱順位関数

本章では弱順位行列を関数化するために拡張弱順位関数を定義し，その計算法について述べる．まず従来の順位関係にかえて，順位関数を次のように定義する．なお以下で単に行列というときにはその要素は1と2だけであり，順位行列とは区別する．また特に値をきめる必要のない要素，即ち1でも2でもよい要素は0で表わす．

[定義 4] 順位関数

行列 M に対して次の関係を満たす関数対 $\langle F, G\rangle$ を M の順位関数と呼ぶ．

$$M[XY]=1 \longleftrightarrow F(X) \geqq G(Y)$$
$$M[XY]=2 \longleftrightarrow F(X) < G(Y)$$ ☒

行列 M を有向グラフで表現したとき**，ループに

---

* 弱順位関係は弱順位行列か弱順位関数により表現される．

** 行列 M と有向グラフ H_M の対応は次のようなものである．H_M はそれぞれ M の行及び列に対応する要素を節として持つ．$M[XY]=1$ ならば X から Y へ，$M[XY]=2$ ならば Y から X へ枝を作る．$M[XY]=0$ では XY 間に枝はない．

含まれないすべての枝（即ち M の要素）を除いてできたグラフ（即ち連結グラフの集合）を表わす行列をもとの行列の核行列と呼ぶ．核行列をこのように定義すると次の定理の成立は明らかである．（核行列の計算については文献 6）を参照のこと）

[定理 2] 行列 M が順位関数を持つための必要十分条件は M の核行列が空行列（すべての要素が0）になることである．⊠

[アルゴリズム 1] 行列 M の順位関数 $\langle F, G\rangle$ の計算．（但し M の核行列は空とする．）

（方法）（1） 行列 M の行及び列に対応する各要素を X_i, Y_j $(1\leq i\leq n,\ 1\leq j\leq m,\ n, m$ は行及び列の大きさ）とし，F, G を行及び列に対応した関数とする．（2） $F(X_i)=G(Y_j)=1$ とする．$(i=1\sim n,\ j=1\sim m)$．（3） すべての X_i, Y_j の対に対して，$M(X_iY_j)=1$ でかつ $F(X_i)<G(Y_j)$ ならば，$F(X_i)=G(Y_j)$ とする．（4） すべての X_i, Y_j の対に対して $M(X_iY_j)=2$ かつ $F(X_i)\geq G(Y_j)$ ならば，$G(Y_j)=F(X_i)+1$ とする．（5） (3), (4) で F, G の値が変化しなくなるまで (3), (4) をくりかえす．変化しなくなれば止まる．そのときの F, G の値が M の順位関数の値である．⊠

次に順位行列を2個の要素を持つ行列の対で表現するために拡張弱順位行列を定義する．

[定義 5] 拡張弱順位行列 A, R

2つの集合 S_1, S_2 をそれぞれ $S_1=\{4,5,6\}$，$S_2=\{(1,2)\times(1,2)\}$ とし，S_1 から S_2 への任意の同型写像を Φ とする．弱順位行列 M に対して M と同じ大きさを持ち，次の関係を満たす2つの行列 A, R を M の拡張弱順位行列と呼ぶ．

$$\Phi(M(XY))=(A(XY), R(XY)) \qquad ⊠$$

このように定義すると A, R は先に定義した順位関数で表現できる．そこで，M の拡張弱順位行列 A, R の順位関数をそれぞれ $\langle F, G\rangle$, $\langle H, L\rangle$ とするとき，$(\langle F, G\rangle, \langle H, L\rangle)$ を M の拡張弱順位関数と呼ぶ．

弱順位行列 M の拡張弱順位関数は拡張弱順位行列の選び方により数多く存在するが，反転行列*の考えを導入して考えると，**Table-1 (a)** の3種類の対応で作られる拡張弱順位行列について関数の存在を確かめれば十分である．Table-1 (a) では Table-1 (b) のような対応づけを除外して考えている．これは以下のアルゴリズムの簡単化のためである．このような対応を許すとたとえば $M[XY]=6$ なる (X, Y) の組に対して A, R で (1,2) とするか (2,1) とするかの区別が生じ，アルゴリズムが複雑になる．このため Table-1 (b) のような対応づけは除外して考える．そのため以下の条件は完全な必要十分条件とはならない．

* 行列 A に対してその要素の1と2を入れかえた行列を A の反転行列と呼ぶ．

Table-1 Correspondences between the weak precedence matrix and the extended weak precedence matrices

(a)

M	4	5	6
A_1	1	2	2
R_1	0	1	2
A_2	2	1	2
R_2	1	0	2
A_3	1	2	1
R_3	1	2	0

(b)

M	4	5	6	
A	1	2	1	2
R	1	2	2	1

Table-2 Correspondences in SSE level

M	4	5	6	7	9
A_1	1	2	2	0	2
R_1	0	1	2	0	0
A_2	2	1	2	0	0
R_2	1	0	2	0	2*
A_3	2	2	1	0	0
R_3	1	2	0	0	2*

[定理 3] 弱順位行列 M の拡張弱順位関数が存在するための必要十分条件は Table-1 (a) の (A_i, R_i) $(i=1,2,3)$ の内，少なくとも1組の (A_i, R_i) の核行列がともに空となることである．（証明略）

行列の核行列を求める計算は簡単であるので，与えられた弱順位行列の拡張弱順位関数の存在の有無は上記定理より簡単に判定することが可能である．次に準強等価弱順位行列の拡張弱順位関数の存在の有無の判定，及び求め方について述べる．準強等価弱順位行列は $\{4,5,6\}$ の要素の他に $\{7,9\}$ の2つの要素を持つため，そのままでは定理3は適用できない．しかし7及び9の意味を考えるならば，7及び9に対して拡張弱順位行列では **Table-2** のような値をわりあてればよいことがわかる．

Table-2 において M の9に対する R_2, R_3 の2* は次のような意味である．9は5か6を意味するから R_2 では0か2である．故に A_2 が関数化されたのち

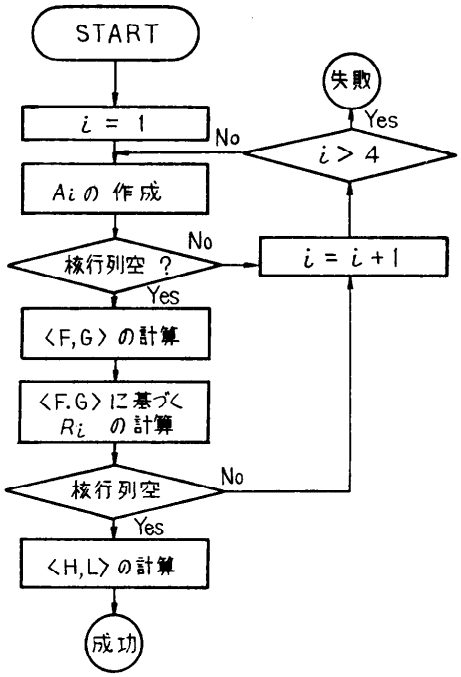


Fig. 1 Algorithm for computing the EWPF from the weak precedence matrix in SSE level.

9に対する A_2 の要素の値が1のときには 0, 2 のときには2でなければならない．このように R_2, R_3 については A_2, A_3 を関数化した際，0の部分にどんな値を持たすかにより R_2, R_3 における 2* の部分の持つべき値が異なる．

[アルゴリズム 2] 準強等価弱順位行列の拡張弱順位関数を求める．詳細は **Fig. 1** に示す．

4. 拡張弱順位関数の実現

本章ではアルゴリズム2で関数ができないとき，関数化を可能にするための方法について述べる．関数のできない原因を次の2つに分け，それぞれについて考察する．

① 誤りをすべて無視した弱順位行列（要素は 4, 5, 7 のみ）の拡張弱順位関数が存在しない．

② 準強等価弱順位行列の6または9の要素が原因となって関数ができない．

①の場合には文法の変形が必要であり，②の場合にはある程度の誤り要素（6または9）を7に書きかえる必要がある．まず①の場合に文法を変形するアルゴリズムを次に示す．

* max は行の大きさをその値とする定数である．

[アルゴリズム 3] 要素として 4, 5, 7 のみを持つ弱順位行列が関数を持つように文法を変形する．

（方法）$(A_1, R_2), (A_3, R_3)$ の組についても同様であるので (A_1, R_1) の組についてのみ述べる．

（1） A_1 の核行列を求める．（2） 順位行列の値を4から5に変更すべき対 (X, Y) を決定する．（3） $X \leqq Y$ を新しい非終端語 Z の導入により $X \gtrdot Y$ とする．（詳細は文献 5)）（4） 以上の操作を (A_1, R_1) に対する核行列が空になるまでくりかえす．⊠

このアルゴリズムによる弱順位行列の変形が有限回で終ること，即ち有限回の変形で A_1 の核行列を空にできることは Presser[5] によりなされた証明とほぼ同様に証明できる．特にこの場合には A_1 の核行列を空にするためにもとの順位行列 M を変更しても R_1 には何の影響も与えないこと．また M の変更は $\leqq \to \gtrdot$ だけでよいことから変更は非常に容易である．

弱順位関数と同様に次の定理が成り立つことは上記のアルゴリズムより自明である．

[定理 4] すべての弱順位文法に対してそれと等価な拡張弱順位関数の存在する（但し誤り要素は無視）弱順位文法が存在する．⊠

次に②の場合に6または9の値を持つ要素をできるだけ少なく7に変えて関数化を可能にする方法について述べる．

[アルゴリズム 4] 極小の誤り要素を無視して関数化を可能にする方法．

拡張弱順位行列としてどれを選ぶかにより方法は少しずつ異なる．ここでは (A_1, R_1) を選んだ場合について述べる．他の場合もほぼ同様である．

（方法）（1） A_1 の核行列 CA_1 を求める．CA_1 が空行列であれば(5)へ行く．（2） CA_1 の各行について $\mathrm{COL}_1(i)$, 各列について $\mathrm{ROW}_1(j)$ を求め，それらのうち最小値をとる行または列を選ぶ．但し $\mathrm{COL}_1(i)$ は $M(ik)=5$ かつ $CA_1(ik)=2$ なる k が存在すれば max*, 存在しなければ $CA_1(ik)=2$ なる k の個数をそれぞれ値としてとる．また $\mathrm{ROW}_1(j)$ は $M(kj)=5$ かつ $CA_1(kj)=2$ なる k が存在すれば max, 存在しなければ $CA_1(kj)=2$ なる k の個数を値としてとる．(3)最小値をとるものを行 j とする．M の行 j の誤り要素 (6 or 9) の内，$CA_2(ji)=2$ となるすべて i について $M(ji)=7$ とする．（4） M より再び A_1 を求め，(1)より繰り返す．（5） R_1 の核行列 CR_1 を求め，CR_1 が空行列であれば止まる．（関数化可能）（6） CR_1 の各行について $\mathrm{COL}_2(i)$, 各列について

$ROW_2(j)$ を求め，最小値をとる行または列を選ぶ．但し $COL_2(i)$ は CR_1 の行 i における値2を持つ要素の個数，$ROW_2(j)$ は同じく列 j における値2を持つ要素の個数である．(7) 最小値をとるものを列 i とする．M の列 i の誤り要素の内，$CR_1(ji)=2$ となるすべての j に対して $M(ji)=7$ とする．(8) M

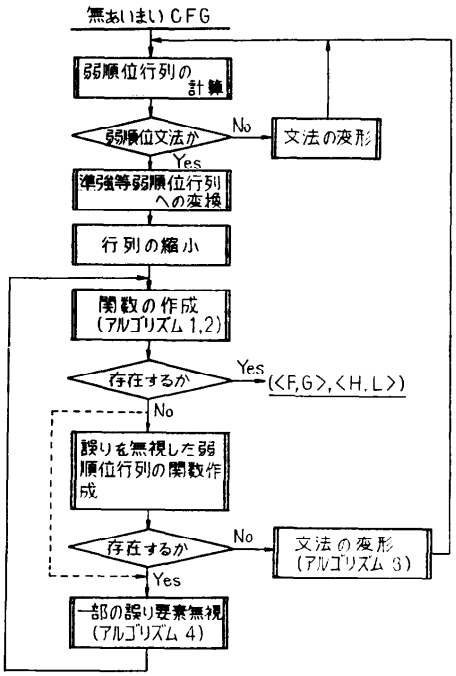
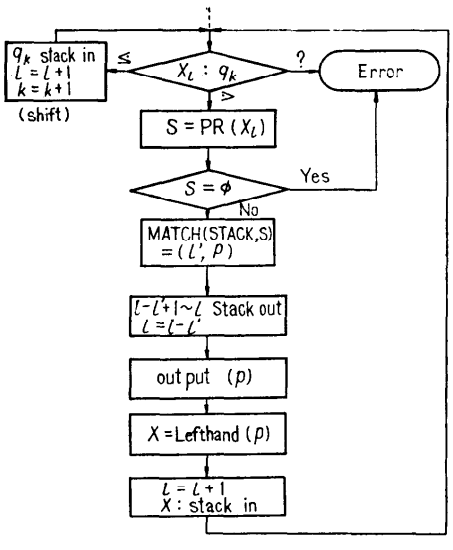


Fig. 2 Computation of the EWPF from an unambiguous context free grammar.



MATCH(STACK,S)ではSの各要素と，STACK内容との最長一致をテストし，一致すれば生成規則の右辺の長さとその番号を出力する．

Fig. 3 Syntax Analysis by the EWPF and PR function.

Table-3 Grammatical Symbols of JIS ALGOL 3000

1	プログラム	57	変数
2	ブロック	58	添字付変数
3	複合文	59	配列名
4	名札のないブロック	60	添字の並び
5	名札	61	添字式
6	名札のない複合文	62	単純変数
7	ブロックの頭	63	変数名
8	複合後部	64	関数呼び出し
9	文	65	手続き名
10	無条件文	66	式
11	基本文	67	変数項
12	名札のない基本文	68	算術式1
13	代入文	69	単純変数1
14	左辺	70	文の区切り
15	飛び越し文	71	文1
16	条件文	72	配列名1
17	部分条件文	73	配列
18	条件節	74	:
19	繰り返し文	75	;
20	繰り返し節	76	begin
21	繰り返し要素	77	end
22	手続き文	78	:=
23	実パラメータ部	79	go to
24	実パラメータの並び	80	else
25	パラメータの区切り	81	if
26	実パラメータ	82	then
27	宣言	83	for
28	型の宣言	84	do
29	型	85	step 1 until
30	型の並び	86	(
31	配列の宣言	87	)
32	配列の並び	88	,
33	配列片	89	real
34	上下限の並び	90	integer
35	上下限	91	array
36	上限	92	[
37	下限	93	]
38	手続きの宣言	94	procedure
39	手続きの本体	95	string
40	手続きの頭	96	value
41	規制部	97	↑
42	規制詞	98	×
43	値の部	99	/
44	名前の並び	100	+
45	仮パラメータ部	101	−
46	仮パラメータの並び	102	<
47	仮パラメータ	103	≦
48	算術式	104	=
49	項	105	≧
50	因子	106	>
51	1次子	107	≠
52	乗除作用素	108	SYMBOL
53	加減作用素	109	UI
54	比較式	110	ID
55	比較作用素	111	UN
56	行先式		

より R_1 を求め(5)より繰り返す．□

アルゴリズム4では(1)～(4)で A_1 の核行列を空行列にし，(5)～(8)で R_1 の核行列を空行列にする．従って誤りをすべて無視した弱順位行列の拡張弱順位関数が存在する限り，極小の誤り要素を無視することにより関数化を可能にする．但しアルゴリズム4は一回の操作での変更を最小にするだけであるから全体として最小になるとは限らない．

任意の無あいまい文法を弱順位文法に変更するアルゴリズムは Presser[5] により示されている．これらを全体としてまとめて無あいまい文法より，それと等価で拡張弱順位関数の存在する弱順位文法をつくるフローを **Fig. 2** に示す．Fig. 2 における“行列の縮小”はプログラミングのためだけであり，本質的な意味はない．

5. 拡張弱順位パーザの実現

弱順位文法ではハンドルの右端の決定は弱順位関係により行うが，左端及び生成規則の決定は生成規則の表を look up することにより行う．従って文法が大きくなるとこのための時間が増加する．特に単純順位パーザと異なり，弱順位パーザは左端の決定まで行わなければならないのでその負担は大きい．そこで本章ではこの操作をより早く行うための方法について述べ，更に以上の方法に基づく JIS ALGOL 3000 の拡張弱順位パーザについて述べる．

Table-4 EWPF for JIS ALGOL 3000

(a)

	F	H	PR		F	H	PR		F	H	PR		F	H	PR
1	1	1		29	1	3		57	1	2	87	85	10	5	
2	1	1	1,4,18	30	5	8	49	58	1	1	104	86	9	5	
3	1	1	2,7,17	31	1	1	47	59	4	2	45,125	87	3	1	39,75,89
4	1	1	3	32	1	8	54	60	1	8		88	9	1	42
5	1	2	101	33	1	1	55,56,58	61	4	1	107,108	89	7	1	50
6	1	1	6	34	1	8		62	1	1	52,53,103,110,122	90	7	1	51
7	1	9		35	1	1	59,60	63	1	1	113	91	14	1	71,126
8	1	1	5,8,12	36	4	1		64	1	1	88	92	10	3	
9	1	1	29,34,66,124	37	1	2		65	2	4		93	2	1	57,105
10	1	1	13,32	38	1	1	48	66	6	1	44	94	14	3	
11	1	1	16,20	39	1	1	64,65	67	1	5		95	14	1	70
12	1	1	19	40	1	6		68	6	10		96	14	3	
13	1	1	21	41	1	2	67	69	2	5		97	10	2	
14	1	5		42	1	3		70	8	6		98	12	1	90
15	1	1	22	43	8	2		71	1	10		99	12	1	91
16	1	1	14,31	44	5	9		72	1	8		100	10	2	92
17	1	7	28	45	5	9		73	1	3		101	10	2	93
18	1	6		46	1	10		74	10	6		102	1	1	95
19	1	1	15,30,35	47	5	1	76,77	75	8	1	68,69,72,123	103	1	1	96
20	1	6		48	1	5	24,37,94,117,121	76	8	6		104	1	1	97
21	3	6		49	1	4	79,80,81	77	5	1	11	105	1	1	98
22	1	1	23	50	1	3	82,83	78	10	1	25,26,120	106	1	1	99
23	1	1	38,115	51	1	1	84	79	16	1		107	1	1	100
24	1	10		52	1	4		80	11	6		108	6	1	43
25	1	5		53	12	4		81	10	5		109	1	1	62,63,85,109,111,112
26	1	1	40,41	54	1	7	118	82	13	1	33	110	1	1	73,74,78,102,106,114,116
27	1	1	9,10	55	1	5		83	14	3		111	3	1	86
28	1	1	46	56	1	1	27,119	84	11	1	36	112	13	6	

(b)

	G	L		G	L		G	L		G	L		G	L
74	2	2	82	4	7	90	9	2	98	4	4	106	4	5
75	6	9	83	14	6	91	8	2	99	4	4	107	4	5
76	14	6	84	4	6	92	5	2	100	11	5	108	10	2
77	6	10	85	4	5	93	5	6	101	11	5	109	11	2
78	3	2	86	13	4	94	9	3	102	4	5	110	15	3
79	14	6	87	7	10	95	9	2	103	4	5	111	13	4
80	6	7	88	7	8	96	9	1	104	4	5	112	6	2
81	12	6	89	9	2	97	4	3	105	4	5			

```
1.  〈プログラム〉::=〈ブロック〉
2.                  ::=〈複合文〉
3.  〈ブロック〉::=〈名札のないブロック〉
4.                  ::=〈名札〉:〈ブロック〉
5.  〈名札のないブロック〉::=〈ブロックの頭〉〈文の区切り〉〈複合後部〉
6.  〈複合文〉::=〈名札のない複合文〉
7.                  ::=〈名札〉:〈複合文〉
8.  〈名札のない複合文〉::=begin〈複合後部〉
9.  〈ブロックの頭〉::=begin〈宣言〉
10.                 ::=〈ブロックの頭〉〈文の区切り〉〈宣言〉
11. 〈複合後部〉::=〈文1〉end
12.                 ::=〈文1〉〈文の区切り〉〈複合後部〉
13. 〈文〉::=〈無条件文〉
14.                 ::=〈条件文〉
15.                 ::=〈繰り返し文〉
16. 〈無条件文〉::=〈基本文〉
17.                 ::=〈複合文〉
18.                 ::=〈ブロック〉
19. 〈基本文〉::=〈名札のない基本文〉
20.                 ::=〈名札〉:〈基本文〉
21. 〈名札のない基本文〉::=〈代入文〉
22.                 ::=〈飛び越し文〉
23.                 ::=〈手続き文〉
24. 〈代入文〉::=〈左辺〉〈算術式〉
25. 〈左辺〉::=〈変数〉:=
26.                 ::=〈手続き名〉:=
27. 〈飛び越し文〉::=go to〈行先式〉
28. 〈条件文〉::=〈部分条件文〉
29.                 ::=〈部分条件文〉else〈文〉
30.                 ::=〈条件節〉〈繰り返し文〉
31.                 ::=〈名札〉:〈条件文〉
32. 〈部分条件文〉::=〈条件節〉〈無条件文〉
33. 〈条件節〉::=if〈比較式〉then
34. 〈繰り返し文〉::=〈繰り返し節〉〈文〉
35.                 ::=〈名札〉:〈繰り返し文〉
36. 〈繰り返し節〉::=for〈変数項〉:=〈繰り返し要素〉do
37. 〈繰り返し要素〉::=〈算術式〉step 1 until〈算術式〉
38. 〈手続き文〉::=〈手続き名〉〈実パラメータ部〉
39. 〈実パラメータ部〉::=(〈実パラメータの並び〉)
40. 〈実パラメータの並び〉::=〈実パラメータ〉
41.                 ::=〈実パラメータの並び〉〈パラメータの区切り〉〈実パラメータ〉
42. 〈パラメータの区切り〉::=,
43. 〈実パラメータ〉::=SYBOL
44.                 ::=〈式〉
45.                 ::=〈配列名〉
46. 〈宣言〉::=〈型の宣言〉
47.                 ::=〈配列の宣言〉
48.                 ::=〈手続きの宣言〉
49. 〈型の宣言〉::=〈型〉〈型の並び〉
50. 〈型〉::=real
51.                 ::=integer
52. 〈型の並び〉::=〈単純変数〉
53.                 ::=〈型の並び〉〈パラメータの区切り〉〈単純変数〉
54. 〈配列の宣言〉::=〈型〉〈配列〉〈配列の並び〉
55. 〈配列の並び〉::=〈配列片〉
56.                 ::=〈配列の並び〉〈パラメータの区切り〉〈配列片〉
57. 〈配列片〉::=〈配列名〉[〈上下限の並び〉]
58.                 ::=〈配列名1〉〈パラメータの区切り〉〈配列片〉
59. 〈上下限の並び〉::=〈上下限〉
60.                 ::=〈上下限の並び〉〈パラメータの区切り〉〈上下限〉
61. 〈上下限〉::=〈下限〉:〈上限〉
62. 〈上限〉::=UI
63. 〈下限〉::=UI
64. 〈手続きの宣言〉::=Procedure〈手続きの頭〉〈手続きの本体〉
65.                 ::=〈型〉Procedure〈手続きの頭〉〈手続きの本体〉
66. 〈手続きの本体〉::=〈文〉
67. 〈手続きの頭〉::=〈手続き名〉〈仮パラメータ部〉;〈値の部〉〈規制部〉
68. 〈規制部〉::=〈規制詞〉〈名前の並び〉;
69.                 ::=〈規制部〉〈型〉〈名前の並び〉;
70. 〈規制詞〉::=string
71.                 ::=〈型〉array
72. 〈値の部〉::=value〈名前の並び〉;
73. 〈名前の並び〉::=ID
74.                 ::=〈名前の並び〉〈パラメータの区切り〉ID
75. 〈仮パラメータ部〉::=(〈仮パラメータの並び〉)
76. 〈仮パラメータの並び〉::=〈仮パラメータ〉
77.                 ::=〈仮パラメータの並び〉〈パラメータの区切り〉〈仮パラメータ〉
78. 〈仮パラメータ〉::=ID
79. 〈算術式〉::=〈項〉
80.                 ::=〈加減作用素〉〈項〉
81.                 ::=〈算術式〉〈加減作用素〉〈項〉
82. 〈項〉::=〈因子〉
83.                 ::=〈項〉〈乗除作用素〉〈因子〉
84. 〈因子〉::=〈1次子〉
85.                 ::=〈因子〉↑UI
86. 〈1次子〉::=UN
87.                 ::=〈変数〉
88.                 ::=〈関数呼び出し〉
89.                 ::=(〈算術式1〉)
90. 〈乗除作用素〉::=×
91.                 ::=／
92. 〈加減作用素〉::=+
93.                 ::=−
94. 〈比較式〉::=〈算術式〉〈比較作用素〉〈算術式〉
95. 〈比較作用素〉::=<
96.                 ::=≤
97.                 ::==
98.                 ::=≥
99.                 ::=>
100.                ::=≠
101. 〈行先式〉::=〈名札〉
102. 〈名札〉::=ID
103. 〈変数〉::=〈単純変数〉
104.                ::=〈添字付変数〉
105. 〈添字付変数〉::=〈配列名〉[〈添字の並び〉]
106. 〈配列名〉::=ID
107. 〈添字の並び〉::=〈添字式〉
108.                ::=〈添字の並び〉〈パラメータの区切り〉〈添字式〉
109. 〈添字式〉::=UI
110.                ::=〈単純変数〉
111.                ::=〈単純変数1〉+UI
112.                ::=〈単純変数2〉−UI
113. 〈単純変数〉::=〈変数名〉
114. 〈変数名〉::=ID
115. 〈関数呼び出し〉::=〈手続き〉〈実パラメータ部〉
116. 〈手続き名〉::=ID
117. 〈式〉::=〈算術式〉
118.                ::=〈比較式〉
119.                ::=〈行先式〉
120. 〈変数項〉::=〈単純変数1〉=
121. 〈算術式1〉::=〈算術式〉
122. 〈単純変数1〉::=〈単純変数〉
123. 〈文の区切り〉::=;
124. 〈文1〉::=〈文〉
125. 〈配列名1〉::=〈配列名〉
126. 〈配列〉::=array
```

Fig. 4 Syntax of JIS ALGOL 3000

文法 G の各要素 $X\in(N\cup T)$ に対して次の集合を定義する.

$PR(X)=\{i\,|\,A\to\alpha X$ は G の i 番目の生成規則$\}$

但し X を右辺の右端とする生成規則が存在しないときには $PR(X)$ は空である. 以下に述べる ALGOL 3000 の場合, $PR(X)$ の要素数の最大値は 7 であり, 平均値は 1.8 である. このことは生成規則を右辺の右端ごとに並べておけば約2回のルックアップで生成規則がきまることを示している. $PR(X)$ は順位関数と同様の形で貯えられる. 以上の考えに基づく弱順位パーザによる解析アルゴリズムを **Fig. 3** に示す.

以上の方法に基づいて作った JIS ALGOL 3000 の拡張弱順位関数が **Table-4*** であり, その文法が **Fig. 4** である. なお Table-4 の第1列の数は Table-3 の第1列の数に対応する. これによると PR 関数の値の個数が最大値を持つ語は識別子 (ID) であるが, これは語い解析のレベルで前もって識別され得るものと考える. そのようなものとして ID, UI, 〈単純変数〉を除外して考えるとかなり減り, ほとんどが3回以内の表検索で生成規則がきまることを示している. また Fig. 4 の文法には弱順位文法の定義1の(2)に反する個所が8カ所あるが文脈情報を導入すれば解決される. **Table-5** に Table-4 の拡張弱順位関数が準強等価弱順位行列の誤り要素 (6 or 9) の内どれだけを保存しているか, 即ち再現しているかを示す. 行列中の誤り要素 1,570 個の内, 1,227個は関数により再現されている. 残りの誤り要素に起因する誤りは還元フェイズもしくは他の場所の誤りとして認識される. 今回は誤りの発生頻度は考えずに, アルゴリズム4に従って関数を作成したが, 誤りの発生頻度を考慮するならば実質的な誤り要素保存率は更に向上するものと考える.

Table-5 Preservation ratio of error entries by EWPF

順位行列	順位関係	4	5	6	9	誤り要素
112×39	総数	147	441	981	589	1,570
	関数による保存数	147	441	682	545	1,227

関数による誤り要素保存率 78%

* F,H はすべての文法記号に対して値を持つが, G,L は終端記号に対してしか値を持たない.

6. まとめ

本論文では誤り要素を考慮した弱順位関数を実現する一方法として, 拡張弱順位関数を提案し, その実現法を示した. まず考慮すべき誤り要素の数を減らすために準強等価を定義し, その条件のもとでの実現法を示し, あわせてパーザの速度を上げるための方法についても述べた. 本方式により JIS ALGOL 3000 の拡張弱順位関数によるパーザを, 準強等価弱順位行列の誤り要素の 78% を保存して実現した. パーザの大きさの大巾な削減から考えると満足のいく値と考える. より大きな文法に対しても同様に適用できるがミニコン等のコンパイラの作成に最も適するものと考える.

本論文で示した準強等価弱順位関数を求める方法はまだ必要十分条件とはなっていない. またアルゴリズム4も最小の誤り要素の変更とはなっていない. これらを実現する能率のよいアルゴリズムの研究が今後の課題である.

参考文献

1) N. Wirth & H. Weber: Euler-a generalization of ALGOL, and its formal definition. Comm. ACM, Vol. 9, No. 1, pp. 13〜23 (1966)
2) 日本工業規格: 電子計算機プログラム用言語 ALGOL (水準 3000) C 6214-1967
3) 井上謙蔵: 右順位文法. 情報処理, Vol. 11, No. 8, pp. 449〜456 (1970)
4) A. V. Aho: Weak and Mixed Strategy Precedence Parsing. JACM, Vol. 19, No. 2, pp. 225〜243 (1972)
5) J. McAfee & L. Presser: An Algorithm for the Design of Simple Precedence Grammar. JACM, Vol. 19, No. 3, pp. 385〜395 (1972)
6) 海尻 他: 順位関数の新しい実現法について, 電子通信学会論文誌 Vol. 59-D, No. 11
7) A. V. Aho & J. D. Ullman: Error Detection in Precedence Parsers. Math. Systems Theory, Vol. 7, No. 2, pp. 97〜113 (1973)
8) A. V. Aho, J. D. Ullman: The Theory of Parsing, Translation, and Compiling. Vol. 1, Prentice-Hall (1972)

(昭和51年5月31日受付)
(昭和51年9月17日再受付)