順位関数をもつ順位文法*

浅　井　　清**

Abstract

In this paper the author has shown existence of a nontrivial family of precedence grammars that have precedence functions. The family of the grammars is nontrivial in the sense that its structure is an well-approximation of grammars of current programming languages such as FORIRAN Ⅳ and ALGOL-like language PL 360. Comparative results for these grammars are also given.

要旨　本論文において筆者は，順位関数をもつ順位文法の族が存在すること，およびこの文法族のもつ構造が，現存のプログラミング言語 FORTRAN Ⅳ と ALGOL-like 言語 PL 360 を順位文法で規定した場合の文法の構造をよく近似していることを示した．第1節では順位関数の有用性について，第2節では順位文法の定義と定理について述べる．第3節では順位関数をもつ順位文法の存在を示し，付録においてこの文法族の構造と FORTRAN Ⅳ，および PL 360 の文法の構造との比較結果を示す．

1. はじめに

プログラミング言語を順位文法で記述できれば，その言語の解析は簡単な手順で実行されることが Floyd[1], Wirth and Weber[2], Colmeraner[3] などによって示されている．このとき解析に使用される順位表は，解析される文法の文法要素の数を N として $N\times N$ 行列で表現される．ところが実用規模のプログラミング言語では，この N の値はかなり大きい．たとえば，FORTRAN Ⅳ では N～500 であるから，この順位行列を小さくする方法[4]，$N\times N$ 行列の代わりに $2\times N$ 個の値をとる2個の順位関数 f, g を順位表として使用する方法などが考えられている[1),2)]．いま V_N を変数，V_T を端記号，S をプログラム，P を生成規則の集合とする順位文法 G を $G=(V_N, V_T, S, P)$ として，$V=V_N\cup V_T$，または $V=V_T$ である V の二つの要素 A, B の間に順位が存在するとき，それらの順位は，$\gtrdot$, $\lessdot$, または $\doteq$ のいずれか1つだけが許される G は単純順位文法と呼ばれている．この単純順位文法でプログラミング言語 $L(G)$ を解析する場合に考えられる方法は，次の3種に大別することができるであろう．

（1）　入力された V_T の要素の列の個々の端記号について，その端記号とほかの端記号との前後関係を調べるなどの意味論的，局所的な解析を行ない，その後端記号間のみで定義された順位関係に従って $L(G)$ を解析する．

（2）　入力された V_T の要素の列に上の(1)で述べた前処理を行ない，その後 $V=V_N\cup V_T$ の要素間で定義された順位関係に従って $L(G)$ を解析する．

（3）　入力された V_T の要素の列について，前処理なしで $V=V_N\cup V_T$ の要素間で定義された順位関係に従って $L(G)$ を解析する．

上記の3種の方法のうち，(1)は通常のコンパイラが利用している方法である．この方法では解析の単位が V_T の要素に限定されてしまうために，個々の生成規則についての解釈則を作ることがむずかしくなり，解析過程の全体の流れが見失われやすい．

(3)の方法によることにして文法 G を構成すると，$V=V_N\cup V_T$ の個々の要素について，細部にいたるまで厳密な記述が必要となるために生成規則 P の数が多くなるが，V の大きさは変わらないので，同一の右辺をもつ生成規則の数もまた増大する．同一の右辺をもつ生成規則を消去しようとすれば，入力された V_T の要素の列について，ある程度の前処理が必要となり結局(3)の方法は(2)に近くなる．しかし，(2)の方法も解析の実行効率，計算機メモリ占有量などの点で大きな問題を含んでいる．この問題点を例によって示そう．FORTRAN IV 語を定義している順位文法 G

* Precedence Grammars with Precedence Functions, by Kiyoshi Asai (Japan Atomic Energy Research Institute)

** 日本原子力研究所

のうちから，式（expression）を定義している V_N', V_T', expr, P' を抜き出して順位文法 $G'=(V_N', V_T', \text{expr}, P')$ を作ると，V_N', V_T', P' の要素の数はそれぞれ約 70, 30, 150 である．また $V'=V_N'\cup V_T'\ni A, B$ の順位関係 (A, B) の総計は約 1500 となる．この順位関係を 100×100 の順位行列で表現するのはむだが多いので，通常のコンパイラでは入力される記号は，すべてに V_T 属していることに着目して，順位行列を 100×30 の大きさにすることもできる．しかし，このように小さくされた順位行列を使っても，FORTRAN Ⅳ 語の順位文法のすべての順位関係を保存するには 10,000 語以上のメモリを必要とする．一つの順位関係を3ビットに圧縮するとメモリ所要量は大幅に減少するが，乗除算，シフト命令などのために順位関係 (A, B) へのアクセスが遅くなる．また，よく知られているように方法(2)，あるいは(3)においては解析している入力記号の列中に誤りが存在しても，この誤りに対して適切なエラー・メッセージを出力すること，誤りの部分を切り離して解析を続行することなどは相当面倒な仕事である．そこで従来は入力記号の列に対し，意味論的，局所的な解析を十分に行ない，その後 V_T に属する記号 $a, b, \cdots\cdots, c$ などの間に一次的な順位関係 $a\lessdot b\lessdot\cdots\cdots\lessdot c$ を設定し，この関係に従って翻訳を行なうという(1)の方法がとられてきた．もし，対象となる方法 G について順位関数 f, g が存在するならば，(1)と(2)の利点をもとに生かすことができる．一般に順位文法は順位関数をもたないが，順位関数をもつ文法の族が存在すれば，任意の文脈に依存しない文法（CF 文法と呼ぶ）を順位文法に変形し[5]，そこで現われた同一の右辺をもつ生成規則を意味論的，局所的な解析によって消去し[6),7)]，これによって得られた文法を順位関数をもつ文法に変形するという一連の操作の繰り返しが文法の定義とコンパイラ設計の段階において可能となってくるであろう．第3節においてわれわれは常に順位関数をもつトリビアルではない順位文法の族が存在することを示し，あわせて現存するプログラミング言語を順位文法で規定した場合に，その文法のもつ構造がこの順位関数をもつ順位文法の構造に非常に近いことを付録で示そう．

2. 順位文法の定義

まず，順位文法の定義と定理について述べよう．**2.1** から **2.3** までの記法と定理は A. Colmerauer[3] による．

2.1 対関係

集合 E の上の対関係の部分集合を ρ で表わし，$a, b\in E$, $(a, b)\in\rho$ を $a\rho b$ と略記する．対関係の集合を $E\times E$ で，これについての ρ の補集合を $\bar{\rho}=E\times E-\rho$ で表わす．E についての2つの対関係 ρ, σ の積 $\rho\sigma$ は，次のように定義される：

$$a\rho\sigma b\equiv[\exists c\in E, a\rho c\cap c\sigma b].$$

ρ の閉包 ρ^+ は

$$\rho^i=\rho\rho^{i-1},\quad a\rho^0 b\equiv[a=b]$$

として

$$\rho^+=\bigcup_{i=1}^{\infty}\rho^i$$

と表わされるが，E の要素の数が $n<\infty$ であるときは

$$\rho^+=\bigcup_{i=1}^{\infty}\rho^i=\bigcup_{i=1}^{n}\rho^i$$

となる．

2.2 *CF* 文法

文法 $G=(V_N, V_T, S, P)$ が CF 文法であるとは，$V_N\cap V_T=\phi$, $S\in V_N$ となる有限集合 V_N, V_T と，$V_N\cup V_T$ の要素の有限列（長さ0の列も含む）の集合 $(V_N\cup V_T)^*$ の上で定義された有限個の対関係 $\rightarrow$ の集合 P が存在し，長さ1の x と $y\in(V_N\cup V_T)^*$ なる y について $x\rightarrow y$ であるときは $x\in V_N$ であり，また $x, y\in(V_N\cup V_T)^*$, $Z\in V_N$, $x=uZv$, $y=uwv$ なる x, y, Z について $x\rightarrow y$ ならば，$Z\rightarrow w$ となる $u, v, w\in(V_N\cup V_T)^*$ が存在するときをいう．ここで V_N, V_T, P をそれぞれ変数，端記号，生成規則の集合と呼び，$x\in V_N, y_1, \cdots\cdots, y_n\in(V_N\cup V_T)^*$ なる x と $y_1, \cdots\cdots, y_n$ について $x\rightarrow y_1\rightarrow\cdots\cdots\rightarrow y_n$ となるときは $x\Rightarrow y_n$ と略記する．このとき文法 G によって生成される言語 $L(G)$ は

$$L(G)=\{t\in V_T^*\mid S\Rightarrow^+ t\}$$

と表現される．

2.3 順位文法

CF 文法 $G=(V_N, V_T, S, P)$ の任意の要素 $A, B\in(V_N\cup V_T)$ について，対関係 $\gtrdot$, $\lessdot$, $\doteq$, または ϕ（空なる対関係）のいずれか1つが存在するとき，文法 G は単純順位文法であるという．これ以後の議論においてはすべて単純順位文法を取り扱い，それを順位文法と略記する．次に CF 文法 $G=(V_N, V_T, S, P)$ の任意の $A, B\in(V_N\cup V_T)$ について対関係 α, λ, ρ を次のように定義する．

$$A\alpha B\equiv[\exists U\in V_N, x, y\in(V_N\cup V_T)^*, U\rightarrow xABy],$$

$A\lambda B\equiv(\exists y\in(V_N\cup V_T)^*, A\to By)$,

$A\rho B\equiv(\exists x\in(V_N\cup V_T)^*, B\to xA)$.

この α, λ, ρ を組み合わせて得られる $V_N\cup V_T$ 上の新しい対関係

$\doteq=\alpha$, $\lessdot=\alpha\lambda^+$, $\gtrdot=\rho^+\alpha\cup\rho^+\alpha\lambda^+$

を Wirth-Weber 形の単純順位関係と呼ぶ[2),3)]. 文法 G についてこの順位関係が定義されているとき, G があいまいさのない単純順位文法であるとは

(1) $X\to u$, $Y\to u$ なる $X=Y$,

(2) $(\lessdot\cap\gtrdot)\cup(\lessdot\cap\doteq)\cup(\doteq\cap\gtrdot)=\phi$

なる条件を満たすときをいう. ここで ϕ は空集合を表わす. A. Colmerauer は上の条件(2)が次の条件(2′)と同値であること, および文法 G の順位関係 $\lessdot$, $\doteq$, $\gtrdot$ が条件(3)を満たせば, その文法は順位文法として解析できることを示した[3)].

(2′) $(\alpha\lambda^+\cap\alpha)\cup(\rho^+\alpha\cap\alpha)\cup(\rho^+\alpha\lambda^+\cap\alpha)\cup(\rho^+\alpha\cap\alpha\lambda^+)=\phi$,

(3) $\alpha\subset\doteq$, $\alpha\lambda^+\subset\lessdot$, $\rho^+\alpha\subset\gtrdot$, $\rho^+\alpha\lambda^+\subset\lessdot\cup\gtrdot$.

3. 順位関数を持つ順位文法

この節では順位関係をもつ順位文法の族が存在することを示そう.

定義 1. CF 文法 $G=(V_N, V_T, S, P)$ において, 任意の $S_i\in(V_N\cup V_T)$ が生成規則の右辺に一度だけ出現するとき, P は単純であるという.

定義 2. $G=(V_N, V_T, S, P)$ が単純順位文法であるとき $S_i, S_j\in(V_N\cup V_T)$ の順位関係 (S_i, S_j) を (i, j) 要素とする行列を G の順位行列と呼ぶ.

G が単純順位文法であれば, 任意の $S_i, S_j\in(V_N\cup V_T)$ について順位関係 $R=\rho^r\alpha\lambda^q$, あるいは $R=\phi$ が存在するが, P が単純である CF 文法については次の補題が成立する.

補題 1. P が単純である CF 文法は, あいまいさのない単純順位文法である.

証 G は **2.3** の条件(1), (2′)を満たしている.

補題 2. 上の補題1の文法の任意の $S_i, S_j\in(V_N\cup V_T)$ について $S_i\rho^r\alpha\lambda^q S_j$ となる p, q は, S_i と S_j が順位関係をもたない場合を除き一意的である.

証 S_i と S_j との間に順位関係が存在するなら, P が単純であるから $S_i\rho^r\alpha\lambda^q S_j$ を満たす (p, q) の組はただ一つである〔証終〕.

上記の $\rho^r\alpha\lambda^q$ は S_i と S_j の順位関係を定義し, G の順位行列の要素となる. この p, q を使って順位行列に対応する pq 行列を定義しよう.

定義 3. 補題2の p, q について対 (p, q) または空集合 ϕ を要素とする行列を文法 G の pq 行列と呼ぶ.

定義 4. 単純順位文法 G の任意の $S_i, S_j\in V_N\cup V_T$ について

$S_i\lessdot S_j$ なら $f(S_i)<g(S_j)$,

$S_i\doteq S_j$ なら $f(S_i)=g(S_j)$,

$S_i\gtrdot S_j$ なら $f(S_i)>g(S_j)$

となるそれぞれが N 個の値をとる関数 f と g が存在するとき, この f と g を文法 G の順位関数と呼ぶ. ここで, N は集合 $V=V_N\cup V_T$ の要素の数である.

定義 5. $\varphi_i\in P$, $x\in V_N$, $y_i\in(V_N\cup V_T)^*$, $(1\leq i\leq n)$ について

$\varphi_1: x\to y_1, \varphi_2: y_1\to y_2, \cdots\cdots, \varphi_n: y_{n-1}\to y_n$

となる規則の列が存在するとき, $\varphi_n\varphi_{n-1}\cdots\cdots\varphi_1$ を $\varphi=\varphi_n\cdots\cdots\varphi_1$ と略記し, 上の対関係の列を

$\exists\varphi: x\Rightarrow y_n, \varphi\in P^*$ で表わす.

補題 3. CF 文法 G の生成規則 P が単純であれば, G の pq 行列について次の(1)～(4)が成立する.

pq 行列の行ベクトルを K_i, 列ベクトルを L_j, (i, j) 要素を $(p_{i,j}, q_{i,j})$ で表わすことにしよう. この時

(1) $\exists j, (p_{i,j}, q_{i,j})=(0, 0)$ ならば

$K_i\ni(p_{i,t}, q_{i,t})$, $p_{i,t}>0$.

証 順位行列でいえば, これは **Fig. 1** の場合が存在しないことと同じである. もし存在するなら

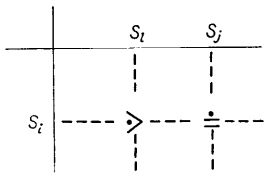

Fig. 1 This is not the case if P is simple or semi-simple.

$\exists\varphi_1: U\to x_1S_iS_jy_1$, $\varphi_1\in P$,

$\exists\varphi_2: V\to x_2S_kS_ly_2$, $\varphi_2\in P$,

$\exists\varphi_3: S_k\Rightarrow x_3S_i$, $\varphi_3\in P^*$,

または

$\exists\varphi_4: Y\to xABy$, $\varphi_4\in P$,

$\exists\varphi_5: A\Rightarrow xS_i$, $\varphi_5\in P^*$,

$\exists\varphi_6: B\Rightarrow S_ly$, $\varphi_6\in P^*$

となる $\varphi_1, \varphi_2, \varphi_3, \varphi_4, \varphi_5, \varphi_6$ が存在することになり, P が単純であるという仮定と矛盾する.

(2) $\exists i$, $(p_{i,j}, q_{i,j})=(0, 0)$ ならば,

$L_j\ni(0, q_{k,j})$, $q_{k,j}>0$.

証 もし **Fig. 2** の場合が成立するなら

$\exists\varphi_1: U\to x_1S_iS_jy_1$, $\varphi_1\in P$,

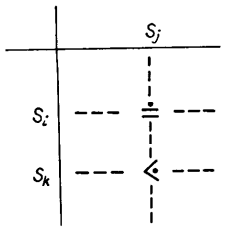

Fig. 2 This is not the case if P is simple.

$\exists\varphi_2$: $V\to x_2S_kCy_2$, $\varphi_2\in P$

$\exists\varphi_3$: $C\Rightarrow S_jy_3$, $\varphi_3\in P^*$

となる $\varphi_1,\varphi_2,\varphi_3$ が存在することになり，P が単純であるという仮定に矛盾する．

(3) $\exists j$, $(0,q_{i,j})$, $q_{i,j}>0$ ならば

$K_i\ni(p_{i,l},q_{i,l})$, $p_{i,l}>0$.

証　もし **Fig. 3** が成立すれば $S_i\lessdot S_j$ だから

$\exists\varphi_1$: $U\to x_1S_iVy_1$, $\varphi_1\in P$

$\exists\varphi_2$: $V\Rightarrow S_jy_2$, $\varphi_2\in P^*$

となる φ_1,φ_2 が存在し，また $S_i>S_l$ から

$\exists\varphi_3$: $W\to x_3XS_ly_3$, $\varphi_3\in P$

$\exists\varphi_4$: $X\Rightarrow x_4S_i$, $\varphi_4\in P^*$

となる φ_3,φ_4 が存在するか，または

$\exists\varphi_5$: $Y\to x_5ABy_5$, $\varphi_5\in P$

$\exists\varphi_6$: $A\Rightarrow x_6S_i$, $\varphi_6\in P^*$

$\exists\varphi_7$: $B\Rightarrow S_ly_7$, $\varphi_7\in P^*$

となる $\varphi_5,\varphi_6,\varphi_7$ が存在することになる．しかし，φ と φ_4,φ_1 と φ_6 は，それぞれ P が単純であるという仮定に反する．

(4) $\exists i,j,k$, $L_j\ni(p_{k,j},q_{k,j})$, $p_{k,j}>0$ かつ

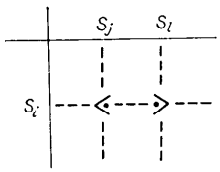

Fig. 3 This is not the case if P is simple or semi-simple.

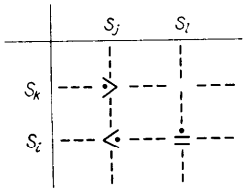

Fig. 4 This may be the case if P is simple or semi-simple.

$L_j\ni(0,q_{i,j})$, $q_{i,j}>0$ ならば

$L_l\ni(p_{i,l},q_{i,l})=(0,0)$ が許される．

証　**Fig. 4** の列 L_j の順位関係が成立するためには

$\exists\varphi_1$: $U\to x_1VS_jy_1$, $\varphi_1\in P$,

$\exists\varphi_2$: $V\Rightarrow x_2S_k$, $\varphi_2\in P^*$

となる φ_1,φ_2 が存在するか，あるいは

$\exists\varphi_3$: $U\to x_3ABy_3$, $\varphi_3\in P$

$\exists\varphi_4$: $A\Rightarrow x_4S_k$, $\varphi_4\in P^*$

$\exists\varphi_5$: $B\Rightarrow S_jy_5$, $\varphi_5\in P^*$

となる $\varphi_3,\varphi_4,\varphi_5$ が存在し，その上に

$\exists\varphi_6$: $W\to x_6S_iCy_6$, $\varphi_6\in P$,

$\exists\varphi_7$: $C\Rightarrow S_jy_7$, $\varphi_7\in P^*$

となる φ_6,φ_7 が存在しなければならない．P が単純であるという仮定から，$\varphi_5\equiv\varphi_7$, $B\equiv C$, $A\equiv S_i$, $\varphi_3\equiv\varphi_6$ となるから，$S_i\doteqdot B$ となる B が存在する．この B が S_l である〔証終〕．

補題 4.　CF 文法 $G=(V_N,V_T,S,P)$ の生成規則 P が単純であれば，任意の $S_i,S_j,S_k\in(V_N\cup V_T)$ について

$$f(S_i)=\max_l(p_{i,l}),\quad(p_{i,l},q_{i,l})\in K_i$$

$$g(S_j)=\begin{cases}\dfrac{1}{2}, & \exists k,\ L_j\ni(0,q_{k,j}),\ q_{k,j}>0,\\ 0, & \text{otherwise}\end{cases}$$

と定義した f,g は，文法 G の順位関数である．

証

(1) $S_i\lessdot S_j$ のとき

$V_N\cup V_T$ の要素の数を N とすると，補題3の(3)によって

$$f(S_l)=\max(p_{i,l})=0,\quad l=1,\cdots\cdots,N.$$

また補題3の(2)から $S_k\doteqdot S_j$ となる S_k は存在しない．$S_i\lessdot S_j$ という仮定から $L_j\ni(0,q_{k,j})$, $q_{k,j}>0$ であるから $g(S_j)=1/2$. したがって $f(S_i)\lessdot g(S_j)$. この $g(S_j)$ は補題3の(4)の $f(S_k)>g(S_j)$ を満たしている．なぜなら $f(S_k)=\max_l(p_{k,l})\geq 1$.

(2) $S_i\doteqdot S_j$ のとき

補題3の(1)によって $K_i\ni(p_{i,l},q_{i,l})$, $p_{i,l}>0$ だから

$$f(S_i)=\max_l(p_{i,l})=0,\quad l=1,\cdots\cdots,N.$$

補題3の(2)によって $L_j\ni(0,q_{k,j})$, $q_{k,l}>0$ だから $g(S_j)=0$. したがって $f(S_i)=g(S_j)$.

(3) $S_k>S_j$ のとき

補題3の(3)によって

$f(S_k)=\max_l(p_{k,l})\geq 1.$

一方，$g(S_j)\leq 1/2$ であるから $f(S_k)>g(S_j)$ 〔証終〕.

このようにして，生成規則 P が単純であれば，文法 G は順位関数をもつことがわかった．ここで，われわれはさらに広い順位文法の族が順位関数をもつことを示そう．

定義 6.　順位文法 G の生成規則の集合 P が次の条件を満たすとき，P は半単純であるという．

（1）　任意の $S_i, S_j\in(V_N\cup V_T)$ の $S_i\rho p_k\alpha\lambda q_l S_j$ となる p_k について，$^{\exists}k_0, p_{k_0}>0$ ならば $^{\forall}k$ について $p_k>0$.

（2）　pq 行列の (i, j) 要素 (p, q) を

$p=\max_k(p_{i,k}),\quad k=1, 2, \cdots\cdots$

$q=\max_l(q_{j,l}),\quad l=1, 2, \cdots\cdots$

と定義したとき，上の pq 行列は補題3の条件（1），(3)を満たす．

ここで，$p_{i,K}, q_{j,l}$ はそれぞれ順位関係 $S_i\rho p_{i,1}\alpha\lambda q_{j,1}\cdot S_j$，$S_i\rho p_{i,2}\alpha\lambda q_{j,2}S_j, \cdots\cdots$ を満たす非負の整数である．S_i，または S_j が回帰的 (recursive) に出現する場合は $p_{i,k}$，または $q_{j,l}$ の値は1とする．P は単純ではないので S_i, S_j について $p_{i,k}, q_{j,l}$ は一意的ではない．そこで上のように p と q を定義して pq 行列の要素を定める．

（3）　上の pq 行列について

$K_i\ni(p_{i,j}, q_{i,j})=(0, 0), K_k\ni(p_{k,j}, q_{k,j})=(0, 0)$

$K_j\ni(p_{i,l}, q_{i,l})=(0, 0)$

であるときは

$K_k\ni(p_{k,l}, q_{k,l})=(0, 0)$，または $=\phi$.

（4）　上の（2）の pq 行列について

$L_j\ni(p_{i,j}, q_{i,j})=(0, 0),$

$L_j\ni(p_{k,j}, q_{k,j})=(0, q_{k,j}), q_{k,j}>0,$

$L_l\ni(q_{m,l}, q_{m,l})=(0, 0),$

$L_l\ni(q_{n,l}, q_{n,l})=(0, q_{n,l}), q_{n,l}>0,$

ならば $i=m$，かつ $k=n$ である．

順位文法 G の P が半単純であるときは，順位行列

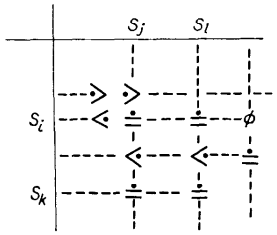

Fig. 5　This may be the case if P is semi-simple.

は **Fig. 5** のようになり，次の定理が成立する．

定理　生成規則が半単純である順位文法は順位関数をもつ．

証　生成規則が半単純である順位文法 $G=(V_N, V_T, S, P)$ の $S_i, S_j\in(V_N\cup V_T)$ について

$$f(S_i)=\begin{cases}1/4, {}^{\exists}j, k, K_i\ni(p_{i,j}, q_{i,j})=(0, 0)\text{ かつ}\\ \quad L_j\ni(0, q_{k,j}), q_{k,j}>0\text{ のとき},\\ 1/4, {}^{\exists}j, K_i\ni(p_{i,j}, q_{i,j})=(0, 0)\text{ かつ}\\ \quad g(S_j)=1/4\text{ のとき},\\ \max_j(p_{i,j}),\ \text{otherwise},\end{cases}$$

$$g(S_j)=\begin{cases}1/4, {}^{\exists}i, k, L_j\ni(p_{i,j}, q_{i,j})=(0, 0)\text{ かつ}\\ \quad L_j\ni(0, q_{k,j}), q_{k,j}>0\text{ のとき},\\ 1/4, {}^{\exists}i, L_j\ni(p_{i,j}, q_{i,j})=(0, 0)\text{ かつ}\\ \quad f(S_i)=1/4\text{ のとき},\\ 1/2, {}^{\exists}k, L_j\not\ni(0, 0), L_j\ni(0, q_{k,j}),\\ \quad q_{k,j}>0\text{ のとき},\\ 0,\ \text{otherwise}\end{cases}$$

と定義した f, g で G の順位関数となるものが存在する．$V_N\cup V_T$ のすべての要素について，上記の定義に従って f と g の値を決定する．このとき一度定義された $f(S_i)$ と $g(S_j)$ の値が再び定義されるのは，$f(S_i)=g(S_j)=0$ か $f(S_i)=g(S_j)=1/4$ となる場合である．上述の f と g の定義式では有限回の操作で f, g の値を決定することができる．このようにして定義された f と g が順位関数となっていることは，補題4の証明と同じ手順で示すことができる．

（1）　$S_i\lessdot S_j$ のとき

$S_i\lessdot S_j$ であるから，$(p_{i,j}, q_{i,j})=(0, q_{i,j}), q_{i,j}>0$. 定義6の(2)によって $K_j\not\ni(p_{i,l}, q_{i,l}), p_{i,l}>0$ だから $\max_j(p_{i,j})=0$. したがって $f(S_i)\leq 1/4$. また $q_{i,j}>0$ から $g(S_j)\geq 1/4$. いま $f(S_i)=g(S_j)=1/4$ と仮定すれば，この等式を成立させる条件の組合せは，次の(a)～(d)である．

（a）　$f(S_i)$, $g(S_j)$ の値がそれぞれ他の $g(S_l)$, $f(S_k)$ の値に依存しないで決定された場合：　このときは，f と g の定義から $K_i\ni(0, 0)$, $L_j\ni(0, 0)$ かつ $L_j\ni(0, q_{k,j}), q_{k,j}>0$, $i\neq k$ となる行 K_i と列 L_j が存在することになる（**Fig. 7**）．しかし，定義6の(4)によって，このような場合は起こり得ない．

（b）　$f(S_i)$ の値のみが他の $g(S_l)$ によって決定された場合：　このときは $g(S_l)$ の定義によって，次の2つの場合に分けられる．

(b-1)　$g(S_l)$ が他の $f(S_m)$ の値に依存しないとき：このときは $g(S_l)$ の定義から，$^{\exists}k, l, L_l\ni(p_{i,l}, q_{i,l})$

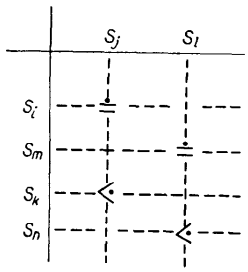

Fig. 7　This is not the case if P is semi-simple.

$=(0,0)$, $L_i \ni (0, q_{k,l}), q_{k,l}>0$ である．また $g(S_j)=1/4$ かつ $g(S_j)$ は他の $f(S_k)$ に依存しないという仮定から，$^{\exists}m$, $L_j \ni (p_{m,j}, q_{m,j})=(0,0), i \neq m$，ところが，これは定義6の(4)に矛盾する．

(b-2)　$g(S_l)$ が他の $f(S_m)$ によって決定されているとき：　$f(S_m)$ が他の $g(S_u)$ に依存しているのでなければ，上述の(a)の場合になり，このような $g(S_l)$ は存在しない．$g(S_l)$ が他の $f(S_m)$ に依存しているとき，$f(S_i), g(S_l)$ の定義から $i \neq m$ である．同様に $l \neq u$．この操作を有限回続けると f，または g がそれぞれ他の g，または f に依存しないで決定された場合に到達する．これらの g と f の値はすべて 1/4 であるから，(b-1)と同様に定義6の(4)に矛盾する．

(c)　$g(S_j)$ の値のみが他の $f(S_k)$ によって決定された場合：　上述の(b)と同様に $f(S_i)=g(S_j)=1/4$ という仮定が矛盾を引き起こす．

(d)　$f(S_i), g(S_j)$ ともに他の $g(S_l), f(S_k)$ によって決定された場合：　このときは，$f(S_i) \to g(S_l) \to \cdots$ となる関数列と $g(S_j) \to f(S_k) \to \cdots$ となる関数列がある．定義6の(3)によって，この2つの関数列が合流することはない．すなわち，$f(S_i) \to g(S_l) \to f(S_k) \to \cdots$ となることはない (**Fig. 6** 付録参照)．したがって2つの関数列はともに終点に到達し，上述の(a)の場合となる．

以上から $S_i \ll S_j$ のときに $f(S_i)=g(S_j)=1/4$ となることはない．ゆえに，$f(S_i)<g(S_j)$.

(2)　$S_i \doteqdot S_j$ のとき

$S_i \doteqdot S_j$ であるから $(p_{i,j}, q_{i,j})=(0,0)$．したがって定義6の(2)によって $K_i \ni (p_{i,l}, q_{i,l}), p_{i,l}>0$ だから $\max_j(p_{i,j})=0$．$L_j \ni (0, q_{k,j}), q_{k,j}>0$ ならば定義から $f(S_i)=g(S_j)=1/4$．$f(S_i)=1/4$ かつ $L_j \ni (0, q_{k,j}), q_{k,j}>0$ のときは $f(S_i)$ は $g(S_i)$ に依存する．$g(S_j) \neq 1/4$ であるときは $f(S_i)=g(S_i)$ が成立する．そこで $L_j \ni (0, q_{k,j}), q_{k,j}>0$ という仮定から $g(S_j)=0$ としよう．このとき $g(S_j)$ の値を決定する過程において，$g(S_j)=0$ を $g(S_j)=f(S_i)=1/4$ と変化させても矛盾は生じない．なぜなら $S_k \doteqdot S_j$ となる S_k は定義6の(3)から $S_k \ll S_i$ となることはない．したがってこの $f(S_k)$ は，等式 $f(S_i)=g(S_i)=1/4$ を変化させない．同様にして f と g の計算の過程において $f(S_i)=0, g(S_j)=1/4$ となっているとき，$f(S_i)$ の値を $f(S_i)=g(S_j)=1/4$ と決定しても矛盾が生じないことを示すことができる．これ以外の場合は $f(S_i)=g(S_j)=0$ である．ゆえに $f(S_i)=g(S_j)$.

(3)　$S_i \gg S_j$ のとき

定義6の(2)から $K_i \ni (0, q_{i,l}), q_{i,l} \geq 0$．したがって $f(S_i)=\max_l(p_{i,l}) \geq 1$，かつ $g(S_j) \leq 1/2$．ゆえに $f(S_i)>g(S_j)$ 〔証終〕.

この定理の結果として，次の二つの系が得られる．

系 1.　定理の pq 行列を首座小行列，他の要素は ϕ とする pq 行列に対応する順位文法は順位関数をもつ．

この系の成立は自明のことであるが，応用上は重要な意味をもつ．この系を利用すれば，順位関数をもつ小規模な文法をふくらませて大きくすることができる．たとえば expression を定義する骨組みだけの文法を作っておき (**Fig. 8**)，ここで現われる変数を型宣言を考慮して増加させ，もとの文法を大きくする．この操作でも順位関数の存在は変わらない．

系 2.　一般の単純順位文法のなかで順位関数をもつ順位文法の族が存在する．

定義5で述べている生成規則の集合 P についての特性は，その順位関係が Wirth-Weber 形の場合であるが，これを一般の場合の順位関係に適用することができる．このとき順位関係は

$$\alpha \subset \doteqdot,\ \alpha\lambda^+ \subset \ll,\ \rho^+\alpha \subset \gg,\ \rho^+\alpha\lambda^+ \subset \ll \cup \gg$$

で定義されているから(**2.3**)，$A\rho^p\alpha\lambda^q B$ となる文法要素 A, B について $A \gg B$ なら $A\rho^p\alpha\lambda^q B$ を $A\rho^p\alpha B$ とみなし，$A \ll B$ なら $A\alpha\lambda^q B$ とみなして定義5の pq 行列を作ればよい．

プログラミング言語の場合には，backtracking を少なくする Wirth-Weber 形の順位関係で，文法が規定されているほうが生成規則についての解釈規則を作りやすく便利である．

4. おわりに

前節の定理によって規定される順位文法の族は，そ

```
1.  'EXPRESSION'          ...= 'ARITHMETIC EXPR'
2                         ...= 'LOGICAL EXPR'
3   'ARITHMETIC EXPR'     ...= 'ARITHMETIC EXPR*'
4   'ARITHMETIC EXPR*'    ...= 'TERM'
5                         ...= + 'TERM'
6                         ...= - 'TERM'
7                         ...= 'ARITHMETIC EXPR*' + 'TERM'
8                         ...= 'ARITHMETIC EXPR*' - 'TERM'
9   'TERM'                ...= 'TERM*'
10  'TERM*'               ...= 'FACTOR'
11                        ...= 'TERM*' * 'FACTOR'
12                        ...= 'TERM*' / 'FACTOR'
13  'FACTOR'              ...= 'FACTOR*'
14  'FACTOR*'             ...= 'PRIMARY'
15                        ...= 'FACTOR*' ** 'PRIMARY'
16  'PRIMARY'             ...= 'PRIMARY.1'
17                        ...= ( 'ARITHMETIC EXPR' )
18  'PRIMARY.1'           ...=  CONSTANT
19                        ...= 'VARIABLE'
20                        ...= 'FUNCTN DESIGNATOR'
21  'VARIABLE'            ...=  IDENTIFIER
22                        ...=  ARRAY-IDENTIFIER
23                        ...=  ARRAY-IDENTIFIER ( 'SUBSCRIPT LIST' )
24  'SUBSCRIPT LIST'      ...= 'SUBSCRIPT LIST*'
25  'SUBSCRIPT LIST*'     ...= 'SUBSCRIPT'
26                        ...= 'SUBSCRIPT LIST*' , 'SUBSCRIPT'
27  'SUBSCRIPT'           ...= 'ARITHMETIC EXPR*'
28  'LOGICAL EXPR'        ...= 'LOGICAL EXPR*'
29  'LOGICAL EXPR*'       ...= 'LOGICAL TERM'
30                        ...= 'LOGICAL EXPR*' .OR. 'LOGICAL TERM'
31  'LOGICAL TERM'        ...= 'LOGICAL TERM*'
32  'LOGICAL TERM*'       ...= 'LOGICAL FACTOR'
33                        ...= 'LOGICAL TERM*' .AND. 'LOGICAL FACTOR'
34  'LOGICAL FACTOR'      ...= 'LOGICAL PRIMARY'
35                        ...=  .NOT. 'LOGICAL PRIMARY'
36  'LOGICAL PRIMARY'     ...= 'PRIMARY.1'
37                        ...= 'RELATIONAL EXPR'
38                        ...= ( 'LOGICAL EXPR' )
39  'RELATIONAL EXPR'     ...= 'ARITHMETIC EXPR*' 'REL OP' 'ARITHMETIC EXPR*'
40  'ARITHMETIC EXPR*'    ...= 'ARITHMETIC EXPR*'
41  'REL OP'              ...= .LT.
42                        ...= .LE.
43                        ...= .EQ.
44                        ...= .NE.
45                        ...= .GT.
46                        ...= .GE.
47  'FUNCTN DESIGNATOR'   ...=  FUNCTN-IDENTIFIER ( 'ACTUAL ARG LIST' )
48  'ACTUAL ARG LIST'     ...= 'ACTUAL ARG LIST*'
49  'ACTUAL ARG LIST*'    ...= 'ACTUAL ARGUMENT'
50                        ...= 'ACTUAL ARG LIST*' , 'ACTUAL ARGUMENT'
51  'ACTUAL ARGUMENT'     ...= 'ARITHMETIC EXPR*'
52                        ...= 'LOGICAL EXPR*'
53                        ...= EXT-FUN-NAME
```

Fig. 8 Skeletonized grammar for expression.

の範囲が狭すぎてプログラミング言語に適用するには不向きのようにみられるかも知れない．しかし，現存するプログラミング言語を生成する順位文法の生成規則の集合 P が半単純に近くなっていることを FORTRAN Ⅳ, ALGOL-like 言語 PL 360[8] を例にとって示しておこう（付録参照）．

実際，筆者が順位文法解析プログラム[9]を使って FORTRAN Ⅳ や PL 360 を解析し順位関数を求めているときに，定義される文法が定理で規定されているものに近くなっていることに気付いたのが，本論の出発点となっている．

順位関数が存在しない場合には，通常は新しい端記号 a を導入し，文法 $G=(V_N, V_T, S, P)$ を $G'=(V_N, V_T \cup \{a\}, S, P')$ と変形することによって P' を半単純に近づけうる．たとえば，左かっこ（を LP と書換えて P' を作るなどの操作がこれに相当する．これは入力 $L(G)$ が意味論的，局所的に前処理されて $L(G')$ となっているときに可能となる．このような操作を一般化して変形可能な文法族の範囲をある程度まで規定することも可能である．

謝辞 FORTRAN Ⅳ の文法を Backus 記法で詳細に記述して筆者の作業を助けて下さった日本原子力研究所計算センターの中村康弘氏に感謝する．

付 録

われわれの順位文法解析プログラムでは，生成規則の数が 300 を越える文法の順位関係を計算することができない．そこで，FORTRAN Ⅳ については第1節の主旨に従って文法を簡略化し（簡略化した例を式について **Fig. 8** で示す），いくつかの部分に分割したもので順位関係と順位関数を求めた．すなわち**Table 1** のように，FORTRAN Ⅳ は主制御部分，宣言文，入出力文，書式文，式の5つのそれぞれが独立な部分に分割し，各文法の順位関係を計算している．

Table 1 の記号について説明すると，各文法を $G=(S, P, V_N, V_T)$ として

Table 1 Precedence relations of FORTRAN IV and PL 360 languages.

	S	P	V_N	V_T	T_P	T_f	K	R	GT	EQ	LT	$GT2$	$LT2$
FORTRAN	主制御	104	49	65	4	1	1	366	143	85	138	128	31
	宣言文	60	30	24	0	2	7	157	74	43	40	26	4
	書式文	49	24	30	1	1	1	433	243	37	153	126	81
	入出力文	54	24	25	2	1	4	149	47	37	65	43	6
	式	53	26	22	0	0	6	536	322	31	183	250	79
	PL 360	154	65	64	0	0	13	1531	975	117	439	498	183

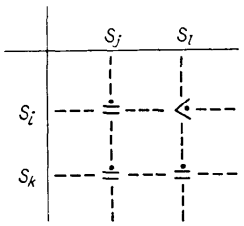

Fig. 6 This is not the case if P is semi-simple.

T_p: Gを順位文法にするために新しく導入した端記号 $\in V_T$ の数.

T_f: Gが順位関数をもつように新しく導入した端記号 $\in V_T$ の数.

K: Fig. 1, または Fig. 3 の形をとる順位行列の行の数.

R: 順位関係の総数.

GT: $(V_N \cup V_T) \ni A, B$ について $A \gtrdot B$ となる順位関係の数.

EQ: $A \doteq B$ となる順位関係の数.

LT: $A \lessdot B$ となる順位関係の数.

$GT2$: $A \gtrdot B$, $A \in (V_N \cup V_T)$, $B \in V_T$ となる順位関係の数.

$LT2$: $A \lessdot B$, $A \in (V_N \cup V_T)$, $B \in V_T$ となる順位関係の数.

Table 1 の注

(1) 主制御部分において

$T_p = \{,,), \text{statement-no}, DO\}$, $T_f = \{,\}$, K はすべて Fig. 3 の形.

(2) 宣言文において

$T_f = \{/\}$ (/を2つの端記号 $a, b \in V_T$ とした.) K は Fig. 1 の形.

(3) 書式文において

$T_p = \{(\}$, $T_f = \{)\}$, K は Fig. 1 の形.

(4) 入出力文において

$T_p = \{,\}$ (, を2つの端記号 $a, b \in V_T$ とした.) $T_f = \{)\}$, K は Fig. 1 の形.

(5) 式において

K は Fig. 1 の形.

(6) PL 360 において

Fig. 1 の形の K の数は 10, Fig. 3 の形の K の数は3.

Fig. 1 と Fig. 3 の形の順位関係になると, 一度定義した $f(S_i)$ と $g(S_i)$ を再定義するという操作が必要となり順位関数が存在しない場合もある. 上記の順位文法の順位関数は, Floyd-Wirth[10] の方法によって求めた. その際, 関数が存在しない文法を変形して順位関数をもつ文法に近づける基準として本論の定理は有用であった[11].

参考文献

1) Floyd, R. W.: Syntactic analysis and operator precedence. J. ACM 10, 3, July, 1963, pp. 316-333.
2) Wirth, N. and Weber, H. Euler: A generalization of ALGOL and its formal definition: Part I. Comm. ACM 9, 1, Jan, 1966, pp. 13-23, 25.
3) Colmerauer, A.: Total precedence relations. J. ACM 17, 1, Jan. 1970, pp. 14-30.
4) 井上謙蔵, 右順位文法, 情報処理 11, 8, 昭和45年8月, pp. 449-456.
5) Learner, A. and Lim, A. L.: A note on transforming context-free grammars to wirth-weber precedence form. The Computer Jour. 13, 2, May, 1970, pp. 142-144.
6) Floyd, R. W.: Bounded context syntactic analysis. Comm. ACM 7, 2, Febr., 1964, pp. 62-67.
7) Knuth, D. E.: On the translation of languages from left to right. Infor. and Control 8, 1965, pp. 607-639.
8) Wirth, N.: PL 360, A programming language for the 360 computers. J. ACM 15, 1, Jan., 1968, pp. 37-74.
9) 藤村, 浅井: PRECEDENCE: Precedence grammars を求めるプログラム. JAERI-memo. 4003, 日本原子研究所, 1970年5月.
10) Wirth, N.: Algorithm 265, Find precedence functions. Comm. ACM 8, 10, Oct. 1965, pp. 604-605.
11) 浅井 清: 順位関数を持つ順位文法, JAERI-memo. 4168, 日本原子研究所, 1970年10月.