Syntactic Analysis of Phrase Structure Languages

MAKOTO NAGAO*

The most important problem for the construction of a compiler is to construct a program for the syntactic analysis of a string generated by the grammar. This is usually done by an experienced programmer but it takes much time and is often accompanied by errors. It is shown in this paper that a purely mechanical procedure to construct a compiler can be realized for a class of phrase structure grammars. Algol 60 language [1], [2] is written by the so-called "Backus notation" [3], and the most part of it can be modified to the grammar of this kind.

1. *Phrase Structure Grammar and Characteristic Symbol.*

We will express the phrase structure grammar in the following way [4]:

$$\left.\begin{aligned} \beta_1 &= (\varphi_{11}, \varphi_{12}, \cdots, \varphi_{1m_1}), \\ \beta_2 &= (\varphi_{21}, \varphi_{22}, \cdots, \varphi_{2m_2}), \\ &\cdots\cdots\cdots \\ \beta_l &= (\varphi_{l1}, \varphi_{l2}, \cdots, \varphi_{lm_l}), \end{aligned}\right\} \tag{1}$$

where $S=(\beta_1, \beta_2, \cdots, \beta_l)$ is a set of non-terminal symbols. Each φ_{ij} is called a syntactic unit and is in the form of

$$\varphi_{ij}=\rho_{\nu_1}\rho_{\nu_2}\cdots\rho_{\nu_m}, \tag{2}$$

where ρ_{ν_k} may be a terminal symbol or a non-terminal symbol. Each expression of (1) states that a non-terminal symbol on the left of the equality sign can be replaced by any syntactic unit in the parentheses on the right. Here the following two restrictions are imposed.

(i) In the expression (2)

$$\varphi_{ij}=/\delta/,^{**} \quad \delta\in D \quad \text{for} \quad m\geq 2.$$

(ii) The same δ appearing in a syntactic unit φ_{ij} can not be an element of another syntactic unit. That is to say, δ is proper to only one syntactic unit.

From the restriction (ii) δ is a terminal symbol. Thus the definition of the set D is defined as a subset of the set T of terminal symbols. The set $T-D$ is denoted as V. The phrase structure grammar thus defined

This paper first appeared in Japanese in Joho Shori (the Journal of the Information Processing Society of Japan), Vol. 4, No. 4 (1963), pp. 186–193.

* Faculty of Engineering, Kyoto University.

** $/x/$ means that the element x is a constituent of a syntactic unit.

will be called grammar G.

We can introduce the concept of partial order relation to the set $S \cup V$. When ρ can be found in a syntactic unit φ_{ij} belonging to β_i, where $/\rho/=\varphi_{ij}$, $\rho \notin D$, then the following order is set up.

$$\beta_i < \rho .$$

In general the ordering property is not in general satisfied by $<$ of a phrase structure grammar, but the following fact remains because the grammar G is a set of sentence generation rules.

By deleting some proper syntactic units φ_{ij} from the grammar G, the set $S \cup V$ can be made a partial order set. Here a phrase structure grammar G' is made from G by deleting the minimum number of syntactic units in order that G' can obtain the property of partial order.

This deletion operation can be done for any phrase structure grammar with the restriction that G can produce at least one terminal string.

As each element of D is proper to a syntactic unit φ_{ij}, the subset $D_{\beta i}$ of D can be defined as the elements of D appearing in φ_{ij} for $j=1, 2, \cdots, m_i$. In the family of subsets $(D_{\beta_1}, D_{\beta_2}, \cdots, D_{\beta_l})$ we can introduce the order the same as that in $(\beta_1, \beta_2, \cdots, \beta_l)$ defined above. This partial order among the set (D_{β_i}) has the following meaning.

In a terminal string where δ_i and δ_j appear,

(i) $\delta_i > \delta_j$ means the syntactic unit containing δ_i is to be merged before that of δ_j.

(ii) When δ_i and δ_j can not be compared, it means that the both syntactic units containing δ_i and δ_j can be processed independently.

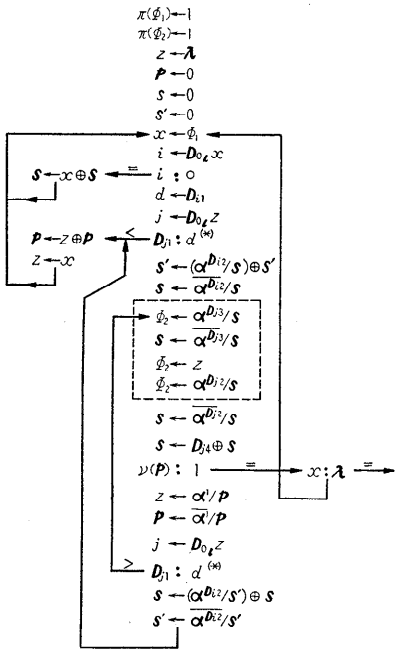
Each element of D is proper to a syntactic unit and, when the length of φ_{ij} is longer than 1, it has at least one element of D by the restriction. So the syntactic analysis of a terminal string can be done depending exclusively on the precedence relation of characteristic symbols and there is no need to pay any attention to the other symbols.

Fig. 1 shows the syntactic analysis of terminal strings generated by the grammar G'. When a syntactic unit is such as

$$\varphi_i = /\delta_{i_1}/\delta_{i_2}/$$

then δ_{i_1} and δ_{i_2} has the same precedence and their treatment must be done by some other additional conditions such as the meaning attached to the unit. In Fig. 1 the part encircled by a dotted line is that for the processing of a syntactic unit. This corresponds to the generation of the object program for the programming languages such as Algol, and the word selection, word ordering and additional insertion etc. for the mechanical translation. In the figure the process is shown to transfer the syntactic unit to the output file. The element of D in the stack p is, from the top to the bottom,

$$z > \delta_{p_1} > \delta_{p_2} > \cdots ,$$



Φ_1: Input file. A given terminal string is stored. It is assumed that the terminal partition λ is attached to the end of the string.

Φ_2: Output file.

p: Stack for the elements of D.

s: Stack for the elements of V.

s': Auxiliary stack for s.

D: Matrix of the grammar G'.

D_0: Column vector of $D \cup \lambda$.

D_1: Column vector indicating the partial order relation of the elements of D.

D_2: Column vector of $\nu(\Delta H)$ where $\Delta H \rho_i \Delta T = \varphi_{ij}$.

: " " " $\nu(\Delta T)$ " " " .

: Column vector of β_j where $\rho_i \in D_{\beta j}$.

(*): When D_{j1} and d can not be compared, the sequence is in the normal order.

Fig. 1*

so that when an element δ of D appearing after z has the relation $z \geq \delta$ or can not be compared with z, the z is locally maximum in regard to the precedence relation, and the syntactic unit containing z can be processed on the spot.

2. *Explicitly Recursive Grammar*

Grammar G', from its formation, can not produce infinite terminal strings. This is not preferable. So here a grammar having recursive structure is treated. A grammar G does not have the partial order

* The expressions in this figure are due to K. E. Iverson [5].

relation that holds in a restricted grammar G' obtained from G. However, it will be expected that G has the similar property.

The explicit recursion is here defined as a rule

$$\beta_i \rightarrow \varphi_{ij} \quad \text{and} \quad \varphi_{ij}=/\beta_i/ .$$

In this case the recursion is local to the non-terminal symbol β_i and does not effect the other part of the grammar, so that the partial order relation defined in the previous section holds as a whole. But the property of the characteristic symbols of β_i should be investigated further more. For the simplicity each syntactic unit belonging to β_i is restricted to have only one characteristic symbol:

$$\varphi_{ij}=/\delta/ , \qquad \delta \in D_{\beta_i} .$$

The following notation is defined. When a terminal string is of the form

$$\Delta H_1 \delta \Delta H_2 \delta \Delta H_3$$

where ΔH_2 does not contain characteristic symbols;

δ on the left is denoted by $'\delta$ and

δ on the right is by δ'.

The precedence relation between $'\delta$ and δ' is as follows:

(i) when $\varphi_{ij}=/\delta/\beta_i/$

the successive replacement shows that

$$\varphi_{ij}=/\delta/(/\delta/\beta_i/)/=/'\delta/\delta'/\beta_i/ ,$$

from which $'\delta<\delta'$ is concluded;

(ii) when $\varphi_{ij}=/\beta_i/\delta/$

the successive replacement shows that $'\delta>\delta'$ is concluded;

(iii) when $\varphi_{ij}=/\delta/\beta_i/\beta_i/ , \quad \varphi_{ij}=/\beta_i/\delta/\beta_i/ ,$ etc.,

the precedence relation between $'\delta$ and δ' can not be determined uniquely;

(iv) when $\varphi_{ij}=/\delta_{i1}/\beta_i/ , \quad \varphi_{ik}=/\delta_{i2}/\beta_i/ ,$

then $'\delta_i <_1 \delta'_{i2}$, $'\delta_{i2} < \delta'_{i1}$ are concluded.

However, in the more intricate rules such as $\varphi_{ij}=/\delta_{i1}/\beta_i/$, $\varphi_{ih}=/\beta_i/\delta_{i2}/$, the precedence cannot be determined uniquely. We can say thence that, when the explicit recursion is of the form (i), (ii), (iv), the right and left precedence of δ can be introduced and the syntactic analysis can be done according to that of characteristic symbols alone.

The interpretation in semantics will help clarify the treatment of syntactic units such as $\varphi=\delta\beta\beta'$ etc., where it is not known which one of β and β' is to be treated first.

3. *General Recursive Grammar*

A grammar has in general the rules of the form

$$\beta_i \to \varphi_{ij}, \quad \varphi_{ij}=/\beta_{k_1}/; \quad \beta_{k_1} \to \varphi_{k_1 j_1}, \quad \varphi_{k_1 j_1}=/\beta_{k_2}/; \quad \cdots;$$
$$\beta_{k_n} \to \varphi_{kl}, \qquad \varphi_{kl}=/\beta_i/,$$

which is named as implicit recursion. The precedence relation among the elements of D varies according to their relative situation in a string. Therefore the precedence relation for the right and left sides is to be clarified among all elements of D.

(A) A square matrix A is constructed from the grammar G in the following way. All the elements of $T \cup S^*$ are associated to the row and column of A. The element A_{ij} which refers to row ρ_i and column ρ_j is defined as

$$\begin{cases} A_{ii}=1, \\ A_{ij}=1, \text{ for all } (\rho_i, \rho_j), \text{ provided } \rho_i \in S \text{ and a syntactic unit } \varphi \text{ belonging} \\ \qquad \text{to } \rho_i \text{ is the form } \varphi=/\rho_j/. \\ A_{ij}=0, \text{ otherwise.} \end{cases}$$

Here, $$A^2=A \cap A, \quad A^3=A^2 \cap A, \quad \cdots, \quad A^n=A^{n-1} \cap A$$

are calculated successively and when a matrix A^k is obtained which satisfies $A^k=A^{k+1}$, it is denoted by A^∞. The matrix A^∞ shows the following property. When the elements belonging to the row β_i has value 1, the corresponding column elements can be components of β_i. This can easily be seen from the construction of matrix A and the production process of A^k's.

(B) The Boolean matrix B having the same row and column elements as A is formed as follows.

$$\begin{cases} B_{ii}=1, \\ B_{ij}=1, \text{ for all } (\rho_i, \rho_j), \text{ provided } \rho_i \in S \text{ and a syntactic unit } \varphi \text{ belonging} \\ \qquad \text{to } \rho_i \text{ has for its left-most element } \rho_j \ (\varphi=/\rho_i/), \\ B_{ij}=0, \text{ otherwise.} \end{cases}$$

The Boolean matrix C is formed as follows.

$$\begin{cases} C_{ii}=1, \\ C_{ij}=1, \text{ for all } (\rho_i, \rho_j), \text{ provided } \rho_i \in S \text{ and a syntactic unit } \varphi \text{ belonging} \\ \qquad \text{to } \rho_i \text{ has for its right-most element } \rho_j \ (\varphi=/\rho_j), \\ C_{ij}=0, \text{ otherwise.} \end{cases}$$

Furthermore a matrix D of the following property is formed.

In a syntactic unit of the grammar (1) if $\varphi=/\rho_i\rho_j/$, then the element D_{ij} correrponding to (ρ_i, ρ_j) is one ($D_{ij}=1$), and otherwise zero ($D_{ij}=0$). Then the following Boolean product is successively calculated.

$$DB=D \cap B, \quad DB^2=DB \cap B, \quad \cdots, \quad DB^n=DB^{n-1} \cap B.$$

When $DB^k=DB^{k+1}$ can be obtained this is denoted by DB^∞. This matrix DB^∞ shows that when the element (i, j) of DB^∞ is 1, ρ_j can be juxta-

* $C=A \cup B$ means $C_{ij}=A_{ij} \cup B_{ij}$,
$D=A \cap B$ means $D_{ij}=\cup_k (A_{ik} \cap B_{kj})$.

posed immediately to the right of ρ_i with higher priority ($\rho_i < \rho_j$).

In the same way the matrix D^tC^∞ is produced. This means that when $(i, j)=1$, ρ_j can be juxtaposed immediately to the left of ρ_i with the higher priority ($\rho_j > \rho_i$).

From these two matrices

$$E=DB^\infty \cap (D^tC^\infty)^t$$

is formed. This means that, when $(i, j)=1$, ρ_j can be contiguously juxtaposed to the right of ρ_i. These properties of terminal strings generated from the grammar G can be used for checking a given string if it can belong to that of grammar G.

Example :—

$$\delta_1=(\uparrow), \quad \delta_2=(\times, /), \quad \delta_3=(+, -),$$
$$\delta_4=(\,(\,), \quad \delta_5=(\,)\,), \quad \alpha=(n, v),$$
$$\beta_1=(\alpha, \delta_4\beta_4\delta_5),$$
$$\beta_2=(\beta_1, \beta_2\delta_1\beta_1),$$
$$\beta_3=(\beta_2, \beta_3\delta_2\beta_2),$$
$$\beta_4=(\beta_3, \beta_4\delta_3\beta_3).$$

The matrices of Fig. 2 are obtained. DB^∞, for instance, shows that δ_4 can be juxtaposed to the immediate right of δ_1, δ_2, δ_3, δ_4 and δ_4 has the priority. This corresponds to the sequences $\uparrow($, $\times($, $+($, $(($ and the part beginning with (should be processed before the operation of $\uparrow$, $\times$, $+$, $'($.

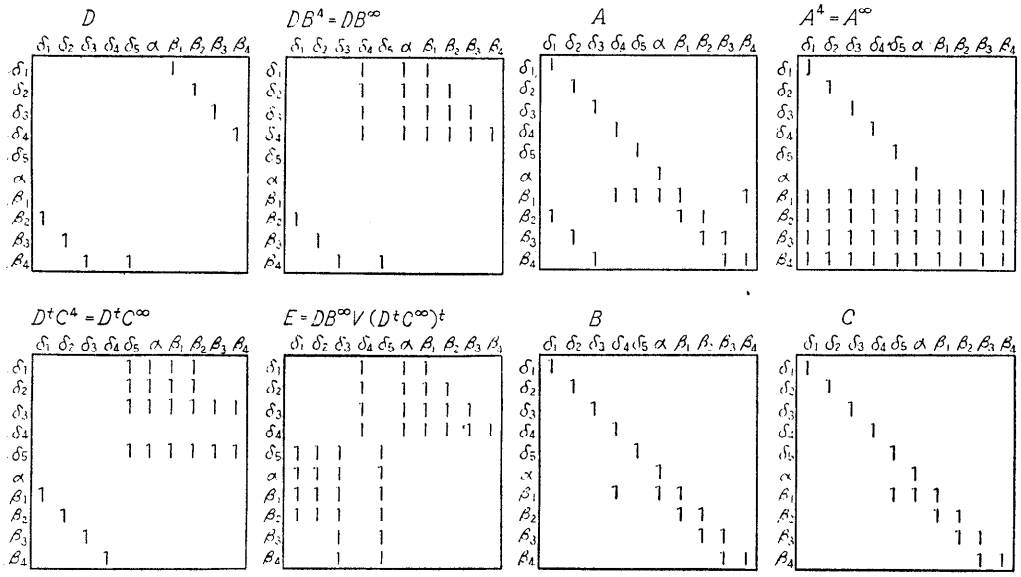


Fig. 2

(C) The matrix F is constructed by the following procedure.

(i) When $\rho_i \in T$, put $F_{\rho_i\rho_i}=1$, $F_{\rho_i\rho_j}=0$ $(i \neq j)$.

(ii) When $\rho_i \in S$, a syntactic unit belonging to ρ_i is denoted as $\varphi_{ij}=\rho_{\nu_1}\rho_{\nu_2}\cdots\rho_{\nu_n}$.

When $n=1$, putting $F_{\rho_i\rho_{\nu_1}}=1$ go to (vi).

When $n \neq 1$, supposing ρ_{ν_k} is the left-most element of D and making $l=1$, go to (v).

(iii) When $\rho_{\nu_l} \in S$:

If a terminal string of V can be generated belonging to ρ_{ν_l}, putting $F_{\rho_i\rho_l}=1$, go to (iv).

If it cannot be generated, putt $F_{\rho_i\rho_{\nu_l}}=1$ and terminating the repetition regarding to l, go to (vi).

(iv) $l+1 \to l$ and, if $l<k$, go back to (iii).

(v) If $l=k$, put $F_{\rho_i\rho_{\nu_l}}=1$ and terminating the repetition regarding l, go to (vi).

(vi) For all syntactic units of the grammar the above operation is to be done. Go to (ii).

From this matrix F, the matrix DF^{∞} is formed in the same way as in (B). If, in this matrix,

$$F_{\rho_i\rho_j}=1 \quad \text{and} \quad \rho_i, \rho_j \in D,$$

ρ_j can be to the immediate right of ρ_j (the elements of V can be between ρ_i and ρ_j) and the priority is $\rho_i < \rho_j$ (that is $'\rho_i < \rho_j'$).

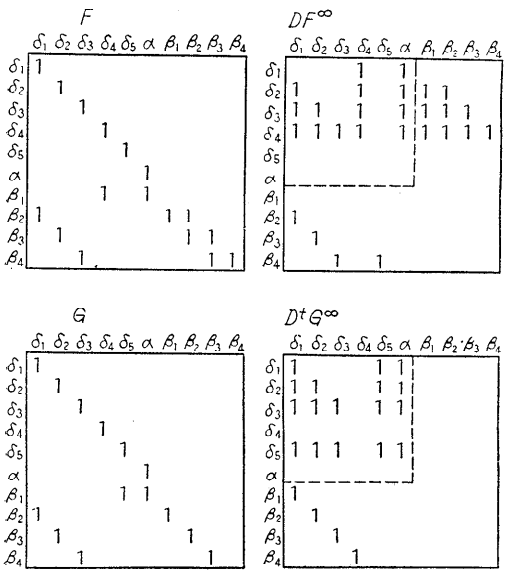
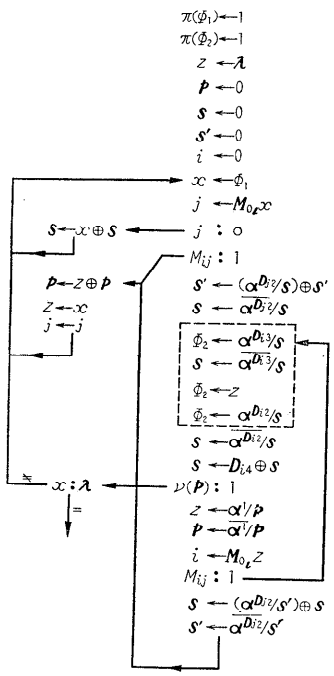


Fig. 3



M: Submatrix formed from DF^{∞} by deleting rows and columns coresponding to $S \cup V$ and adding the row of λ. The priority of λ is the lowest.

Fig. 4

If the above procedure (i)-(vi) is done with $l=n$ instead of $l=1$, and $l-1\rightarrow l$ instead of $l+1\rightarrow l$, another matrix G is obtained. When DG^{∞} is calculated from D and G, this matrix means that, if $F_{\rho_i\rho_j}=1$ and ρ_i, $\rho_j\in D$, ρ_j can be to the immediate left of ρ_i (the elements of V can be between ρ_i and ρ_j), and the priority is $\rho_j>\rho_i$ (that is $'\rho_j>\rho_i'$). The matrices F, DF^{∞} G and DG^{∞} for the above example are shown in Fig. 3.

Using these matrices DF^{∞} and DG^{∞} the syntactic analysis of the terminal strings generated from the grammar G can be done as shown in Fig. 4, depending exclusively on the characteristic symbol. When a syntactic unit has more than one characteristic symbols, such as $\varphi=/\delta_1/\delta_2/$, the sequence of treatment among these symbols is to be determined semantically and these relations are to be added to the matrix DF^{∞}.

4. *Concluding Remarks*

The property of a class of phrase structure language is explained, which has characteristic symbols in each syntactic unit. However the treatment developed in § 3 and the resulting matrices do not depend on the characteristic symbols, so that they are effective for the phrase structure in general. The mechanical analysis shown in Fig. 4 can be constructed in hardware with the matrix of the grammar stored in a fixed memory.

References

[1] NAUR, P. (Ed.), Report on the algorithmic language ALGOL 60. *Comm. ACM, 3* (May 1960), 299-314.

[2] NAUR, P. (Ed.), Revised report on the algorithmic language ALGOL 60. *Comm. ACM, 6* (Jan. 1963), 1-17.

[3] BACKUS, J. W., *ACM-GAMM Conference*, ICIP, June 1959.

[4] CHOMSKY, N., AND M. P. SCHÜTZENBERGER, The algebraic theory of contex-free languages. *Computer Programming and Formal Systems*, North Holland Pub. Co., 1963, 118-161.

[5] IVERSON, K. E., *A Programming Language*. John Wiley & Sons, New York, 1962.