数理解析研究所講究録
第618巻 1987年 150-169

Equational Computation of LALR(1) Look-Ahead Sets

安 在 弘 幸

Hiroyuki Anzai

Kyushu Institute of Technology

Tobata-ku, Kitakyushu-shi, 804 Japan

Summary

An efficient method computing LALR(1) Look-Ahead sets from a given grammar was given by DeRemer and Pennello in 1982. Instead of their method based on the traditional graph theoretical approach, this paper presents a formalized and more efficient method based on a new methodology, that is, linear algebra-like approach called semi-linear algebra.

For a given BNF or extended BNF, Tixier's Standard Right Linear equation which is transformed from it and made empty-free is used in this method. For the equation, First sets are equationally defined and solved. Then, a formalized method obtaining an LR(0) automaton from the equation is given, and Follow sets concerned with nonterminal state transitions of the automaton are equationally defined and solved. Finally LALR(1) Look-Ahead sets are obtained as unions of some Follow sets.

1. INTRODUCTION

In order to construct an LALR(1) parser from a given grammar, it is necessary to construct an LR(0) automaton and to compute LALR(1) Look-Ahead sets from the automaton. Many

researchers gave methods doing so. An efficient method was shown by DeRemer and Pennello[1] in 1982.

The author has been studied linear algebra-like approach to language and automata theory called semi-linear algebra[4]. This approach applying to the above problem is shown in this paper. In other words, this paper presents a formalized efficient method based on the linear algebra-like approach instead of the graph theoretical one used traditionally until now. This method is efficient because of depending on mainly operation of Boolean matrices instead of set operation used in usual methods.

For a given BNF or extended BNF, Tixier gave a parametric representation equivalent to it, that is, a system of simultaneous right linear equations, and called it Standard Right Linear equation (system). In the equations, each coefficient is a set of terminal and nonterminal symbols and each constant term is either an empty set or an empty string set.

Chapter 2 prepares some definitions necessary in this paper and introduces Tixier's SRL equation. For the equation, if some of the coefficients contain empty strings, Chapter 3 offers a method modifying the equation so as to preserve the effect of the empty strings even if they are removed or neglected. So in the following chapters, it is assumed that none of the coefficients of the equation contains the empty string.

In Chapter 4, for each nonterminal X, the first symbol set First(X), a set of symbols appeared at the first position in sentential forms derived from X, is equationally defined and solved.

Chapter 5 presents a method constructing an LR(0) automaton from the SRL equation. In Chapter 6, for each nonterminal state transition (I_k,X) of the LR(0) automaton given in the previous chapter, the follow set $Follow(I_k,X)$ is equationally defined and solved. Finally in Chapter 7, each LALR(1) Look-Ahead set is obtained as a union of some Follow sets.

2. STANDARD RIGHT LINEAR EQUATION

Some preparatory definitions and Tixier's SRL equation[3] as a description of context-free grammars are introduced.

A set of terminals and a set of nonterminals are denoted by T and N, respectively. A union of T and N is written by V. The empty set is represented by ϕ. The empty string and the empty string set are both denoted by λ. A union of V and λ is represented by V^{λ}. For notational convensions, we use t as an element of T ; X, Y and Z as elements of N ; and s as an element of V. And furthermore, for the summing operator $\sum$, $\sum_{t \in T}$ is abbreviated as $\sum_t$; $\sum_{X \in N}$ as $\sum_X$; and $\sum_{s \in V}$ as $\sum_s$.

For a given set S, a matrix $A=(a_{ij})$, $a_{ij} \subseteq S$ and a vector $v=(v_i)$, $v_i \subseteq S$, are called an S-matrix and an S-vector, respectively.

For a set $S \in V^{\lambda}$ and a symbol $\sigma \in V^{\lambda}$, a function ∂_σ is defined as

$$\partial_\sigma S = \lambda \quad \text{if } \sigma \in S,$$
$$= \phi \quad \text{otherwise.}$$

For a matrix $A=(a_{ij})$, the definition is extended as

$$\partial_\sigma A = (\partial_\sigma a_{ij}).$$

For a nonterminal symbol $X \in N$, a function ρ is defined as

$\rho X = \lambda$ if X derives λ , i.e. $X \overset{*}{\Rightarrow} \lambda$,

$= \phi$ otherwise.

And furthermore, the definition is extended for a set S and a matrix $A = (a_{ij})$, as follows:

$\rho S = \lambda$ if $\lambda \in S$ or $\exists X \in S: \rho X = \lambda$,

$= \phi$ otherwise.

$\rho A = (\rho a_{ij})$.

Example 1. For a set $S = \{X, t\}$, where $\rho X = \lambda$ and $Y \neq X$, it holds that $\rho S = \lambda$, $\partial_\lambda S = \phi$, $\partial_X S = \lambda$, $\partial_t S = \lambda$ and $\partial_Y S = \phi$.

For a given BNF or extended BNF, Tixier gave its parametric representation called Standard Right Linear or SRL equation. This is an extension of representing a given regular expression as a right linear equation of which the 1st component of the minimal solution is equivalent to the regular expression. Namely, for each nonterminal X in the BNF or extended BNF, the X-defining equation:

$$X ::= f(Y, \cdots, X, \cdots, Z) \tag{2.1}$$

is unfolded by means of introducing appropriate intermediate parameters $x_{X1}, x_{X2}, \cdots, x_{Xn_X}$ as follows:

$$X = x_{X1}$$

$$x_{Xi} = \sum_{j=1}^{n_X} a_{Xij} x_{Xj} + c_{Xi}$$

where $a_{Xij} \subseteq V^\lambda$, $c_{Xi} \subseteq \lambda$

$i=1, 2, \cdots, n_X$

or (2.2)

$$\mathbf{x}_X = A_X \mathbf{x}_X + \mathbf{c}_X$$

where $\mathbf{x}_X = (x_{Xi})$, $\mathbf{c}_X = (c_{Xi})$, $A_X = (a_{Xij})$.

In the BNF or extended BNF Eq.(2.1), the right part

$f(Y,\cdots,X,\cdots,Z)$ is able to be regarded as a regular expression defined on a space V^*, which is a set of strings generated from an alphabet V^{λ} . On the other hand, the SRL equation Eq.(2.2) is able to be regarded as defined on an n_X-dimensional V^*-vector space. In the space, it has the minimal solution $A_X{}^* c_X$, of which the 1st component is equivalent to the right part $f(Y,\cdots,X,\cdots,Z)$ of Eq.(2.1). We call the solution on V^* the semi-solution.

In the following discussions, we use the SRL equation Eq.(2.2) instead of the BNF or extended BNF Eq.(2.1), and there, the right three n_X-dimensional λ-vectors associated with Eq.(2.2) will be often used.

$$i_X = (\lambda, \phi, \ldots, \phi)$$
$$\phi_X = (\phi, \phi, \ldots, \phi)$$
$$\Lambda_X = {}^t(\lambda, \lambda, \ldots, \lambda)$$

Example 2.

Fig.1 is an example grammar written by BNF. Fig.2 is a state transition diagram representing the multiple finite automata system derived from Fig.1.

S → G #

G → E = E | f

E → T | E + T

T → f | T * f

Fig.1 An example grammar quoted from the book written by Tremblay and Sorenson[2].

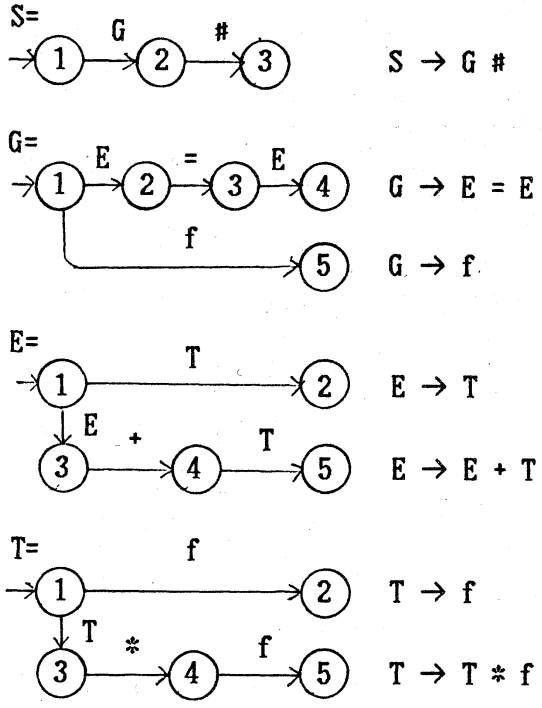

Fig.2 Finite automata system derived from Fig.1

The coefficient matrices and the constant term vectors of the SRL equations describing the automata in Fig.2 are shown below.

$$A_S = \begin{bmatrix} \phi & G & \phi \\ \phi & \phi & \# \\ \phi & \phi & \phi \end{bmatrix} \qquad c_S = \begin{bmatrix} \phi \\ \phi \\ \lambda \end{bmatrix}$$

$$A_G = \begin{bmatrix} \phi & E & \phi & \phi & f \\ \phi & \phi & = & \phi & \phi \\ \phi & \phi & \phi & E & \phi \\ \phi & \phi & \phi & \phi & \phi \\ \phi & \phi & \phi & \phi & \phi \end{bmatrix} \qquad c_G = \begin{bmatrix} \phi \\ \phi \\ \phi \\ \lambda \\ \lambda \end{bmatrix}$$

$$A_E = \begin{bmatrix} \phi & T & E & \phi & \phi \\ \phi & \phi & \phi & \phi & \phi \\ \phi & \phi & \phi & + & \phi \\ \phi & \phi & \phi & \phi & T \\ \phi & \phi & \phi & \phi & \phi \end{bmatrix} \qquad c_E = \begin{bmatrix} \phi \\ \lambda \\ \phi \\ \phi \\ \lambda \end{bmatrix}$$

$$A_T = \begin{bmatrix} \phi & f & T & \phi & \phi \\ \phi & \phi & \phi & \phi & \phi \\ \phi & \phi & \phi & * & \phi \\ \phi & \phi & \phi & \phi & T \\ \phi & \phi & \phi & \phi & \phi \end{bmatrix} \qquad c_T = \begin{bmatrix} \phi \\ \lambda \\ \phi \\ \phi \\ \lambda \end{bmatrix}$$

Fig.3

3. EMPTY-FREE EQUATION

An SRL equation $\mathbf{x}_X = A_X\mathbf{x}_X + \mathbf{c}_X$ (2.2) is said to be empty-free if its minimal semi-solution is preserved even if each empty string contained in A_X are removed or neglected. This chapter presents a method making a given SRL equation empty-free without changing its minimal semi-solution. For this, we must obtain the value of ρX, for each $X \in N$.

For a given BNF or extended BNF, it is not so difficult to evaluate ρX, $X \in N$, by inspection. A method doing so by calculation is given as follows:

As the preparation, interprete the concatination operator "·" and the selection operator "|" (or addition operator "+") as shown in Fig.4. Then, for each $X \in N$, initialize ρX as $\rho X = u$, where u means "undefined". Note

·	λ	u	φ
λ	λ	u	φ
u	u	u	φ
φ	φ	φ	φ

|	λ	u	φ
λ	λ	λ	λ
u	λ	u	u
φ	λ	u	φ

Fig.4 Interpretation of · and | .

that the function ρ is homomorphism. Namely, it holds that $\rho(s + s') = \rho s + \rho s'$, $\rho(ss') = \rho s \rho s'$ and $\rho(s^*) = (\rho s)^*$.

Then for each X-defining equation:

$$X ::= f(Y, \cdots, X, \cdots, Z),$$

such that $\rho X = u$, evaluate

$$\rho X = f(\rho Y, \cdots, \rho X, \cdots, \rho Z) \qquad (3.1)$$

recursively until values of every ρX, $X \in N$, become stable.

Example 3. Evaluate ρX and ρY in Fig.5.

$$\rho X = \rho(aX + Y)$$
$$= \rho a \rho X + \rho Y$$
$$= \rho Y \qquad (\because \rho a = \phi)$$
$$Y = \rho(aY + \lambda) = \rho a \rho Y + \rho\lambda = \lambda$$
$$X = \lambda$$

$X \rightarrow Y \mid aX$

$Y \rightarrow aY \mid \lambda$

Fig.5 An example of BNF.

On the other hand, in order to evaluate ρX by using the X-defining SRL equation Eq.(2.2), the following equation can be used in the above manner.

$$\rho X = \mathbf{i}_X(\partial_\lambda A_X + \textstyle\sum_Y \rho Y \partial_Y A_X)^* \mathbf{c}_X \qquad (3.2)$$

If the value of each ρX, $X \in N$, is thus determined as whether λ or ϕ, we are able to make each the SRL equation empty-free by means of the following transformation.

$$\mathbf{x}_X = (\rho A_X)^*((A_X - \partial_\lambda A_X)\mathbf{x}_X + \mathbf{c}_X) \qquad (3.3)$$

Eq.(3.3) preserves the semi-solution of Eq.(2.2). Furthermore, Eq.(3.3) is equivalent to Eq.(2.2) concerned with the semi-solution even if each nonterminal symbol in the coefficients of Eq.(3.3) would not contain λ. In other words, the transformation Eq.(3.3) completely realizes in Eq.(3.3) itself the effect of all λ's contained in A_X of Eq.(2.2).

This fact means that, for Eq.(3.3), if there exists a nonterminal containing λ, we may neglected it or remove it from the nonterminal except for the starting nonterminal X_0 without loss of its effect. Namely, we may assign ϕ to c_{X1}, the 1st component of $\mathbf{c}_X$ in Eq.(3.3) defining X except for X_0.

For a given grammar, parsing techniques generally require to add a padding rule defining a new starting nonterminal X'_0 instead of the original X_0, as follows:

$$X'_0 ::= X_0 \ \#$$

where # is a newly introduced terminal which is not contained in the original terminal set T and is used as the end-marker of input sequences.

We are thus able to make a given SRL equation system empty-free. Therefore in the following chapters, we assume without loss of generality that a given SRL equation system is empty-free.

Example 4. A system of SRL equations transformed from the BNF in Fig.5 is shown below. Make each constituent equation empty-free.

$$S = s_1, \quad \begin{bmatrix} s_1 \\ s_2 \\ s_3 \end{bmatrix} = \begin{bmatrix} \phi & X & \phi \\ \phi & \phi & \# \\ \phi & \phi & \phi \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ s_3 \end{bmatrix} + \begin{bmatrix} \phi \\ \phi \\ \lambda \end{bmatrix} \tag{3.4}$$

$$X = x_1, \quad \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \phi & a & Y \\ \phi & \phi & X \\ \phi & \phi & \phi \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} \phi \\ \phi \\ \lambda \end{bmatrix} \tag{3.5}$$

$$Y = y_1, \quad \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} \phi & a & \lambda \\ \phi & \phi & Y \\ \phi & \phi & \phi \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} + \begin{bmatrix} \phi \\ \phi \\ \lambda \end{bmatrix} \tag{3.6}$$

From Eq.(3.3), Eq.(3.4) becomes

$$S = s_1,\quad \begin{bmatrix} s_1 \\ s_2 \\ s_3 \end{bmatrix} = \begin{bmatrix} \phi & \lambda & \phi \\ \phi & \phi & \phi \\ \phi & \phi & \phi \end{bmatrix}^{*} \left(\begin{bmatrix} \phi & X & \phi \\ \phi & \phi & \# \\ \phi & \phi & \phi \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ s_3 \end{bmatrix} + \begin{bmatrix} \phi \\ \phi \\ \lambda \end{bmatrix} \right)$$

$$= \begin{bmatrix} \lambda & \lambda & \phi \\ \phi & \lambda & \phi \\ \phi & \phi & \lambda \end{bmatrix} \left(\begin{bmatrix} \phi & X & \phi \\ \phi & \phi & \# \\ \phi & \phi & \phi \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ s_3 \end{bmatrix} + \begin{bmatrix} \phi \\ \phi \\ \lambda \end{bmatrix} \right)$$

$$= \begin{bmatrix} \phi & X & \# \\ \phi & \phi & \# \\ \phi & \phi & \phi \end{bmatrix} \begin{bmatrix} s_1 \\ s_2 \\ s_3 \end{bmatrix} + \begin{bmatrix} \phi \\ \phi \\ \lambda \end{bmatrix} \qquad (3.7)$$

Similarly, Eq.(3.5) and Eq.(3.6) becomes

$$X = x_1,\quad \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \phi & a & Y \\ \phi & \phi & X \\ \phi & \phi & \phi \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} \phi \\ \lambda \\ \lambda \end{bmatrix} \qquad (3.8)$$

$$Y = y_1,\quad \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} \phi & a & \phi \\ \phi & \phi & Y \\ \phi & \phi & \phi \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} + \begin{bmatrix} \phi \\ \lambda \\ \lambda \end{bmatrix} \qquad (3.9)$$

In each Eq.(3.8) and (3.9), the 1st constituent of the constant term was λ after computation. However they were changed to ϕ without loss of the effect.

4. FIRST SYMBOL SETS

For a nonterminal X in a production grammar, a set of symbols (terminals and nonterminals) appeared at the 1st position of sentential forms directly derived from X is defined as

$$DFirst(X) = \{ s \in V \mid X \Rightarrow sw,\ w \in V^{*} \}$$

and is called a Directly First (symbol) set.

Similarly, a set of symbols appeared at the first position of sentential forms derived from X is defined as

$$First(X) = \{ s \in V \mid X \overset{*}{\Rightarrow} sw,\ w \in V^{*} \}$$

and is called a First (symbol) set.

They are, in our equation Eq.(2.2), defined as

$$DFirst(X) = i_X A_X \Lambda_X = \sum_{j=1}^{n_X} a_{X1j} \tag{4.1}$$

$$First(X) = \sum_Y i_X \partial_Y A_X \Lambda_X First(Y) + DFirst(X) \tag{4.2}$$

Now, in order to solve them, put

$\mathbf{u} = (u_X)$, $u_X = First(X)$,

$\mathbf{d} = (d_X)$, $d_X = DFirst(X)$,

$\Gamma = (\gamma_{XY})$, $\gamma_{XY} = i_X \partial_Y A_X \Lambda_X = \partial_Y (i_X A_X \Lambda_X) = \partial_Y d_X$

then, Eq.(4.2) becomes

$$\mathbf{u} = \Gamma \mathbf{u} + \mathbf{d} =_s \Gamma^* \mathbf{d} \tag{4.3}$$

Where, $=_s$ shows that its right-hand side is the minimal solution of the equation in its left-hand side.

Example 5 Obtain all First (symbol) sets in Example 2.

From Eq.(4.1) and Fig.3, we have

$$d_S = \{ G \},\quad d_G = \{ E, f \},\quad d_E = \{ E, T \},\quad d_T = \{ T, f \}.$$

From Eq.(4.3), the desired First sets are obtained as follows:

$$\Gamma = \begin{bmatrix} \partial_S d_S & \partial_G d_S & \partial_E d_S & \partial_T d_S \\ \partial_S d_G & \partial_G d_G & \partial_E d_G & \partial_T d_G \\ \partial_S d_E & \partial_G d_E & \partial_E d_E & \partial_T d_E \\ \partial_S d_T & \partial_G d_T & \partial_E d_T & \partial_T d_T \end{bmatrix} = \begin{bmatrix} \phi & \lambda & \phi & \phi \\ \phi & \phi & \lambda & \phi \\ \phi & \phi & \lambda & \lambda \\ \phi & \phi & \phi & \lambda \end{bmatrix},\quad \mathbf{d} = \begin{bmatrix} d_S \\ d_G \\ d_E \\ d_T \end{bmatrix} = \begin{bmatrix} \{ G \} \\ \{E,f\} \\ \{E,T\} \\ \{T,f\} \end{bmatrix}$$

$$\mathbf{u} = \begin{bmatrix} First(S) \\ First(G) \\ First(E) \\ First(T) \end{bmatrix} = \Gamma^* \mathbf{d} = \begin{bmatrix} \lambda & \lambda & \lambda & \lambda \\ \phi & \lambda & \lambda & \lambda \\ \phi & \phi & \lambda & \lambda \\ \phi & \phi & \phi & \lambda \end{bmatrix} \begin{bmatrix} \{ G \} \\ \{E,f\} \\ \{E,T\} \\ \{T,f\} \end{bmatrix} = \begin{bmatrix} \{G,E,T,f\} \\ \{\ E,T,f\} \\ \{\ E,T,f\} \\ \{\ \ \ T,f\} \end{bmatrix}$$

5. LR(0) AUTOMATON

We have studied the traditional method to derive an LR(0) automaton from a given grammar and found that it is an extended variation of the subset construction method used in the case of

transformation from a nondeterministic finite automaton into a deterministic one. This chapter shows a formalized method constructing an LR(0) automaton from a given SRL equation(2.2).

For any nonterminals Z and X, a function f_{ZX} is defined as

$$f_{ZX} = \partial_X(\{ Z \} \cup \mathrm{First}(Z)). \tag{5.1}$$

Each state of the LR(0) automaton, the so-called LR(0) term, is specified by a bit vector of which the length (i.e. dimension) is $\sum_X n_X$, as follows:

$$I_k = (I_{kX} , \ldots , I_{kX}, \ldots , I_{kZ}), \quad k = 0, 1, \ldots , K.$$

where each I_{kX}, $X \in N$, is an n_X-dimensional bit vector associated with the X-defining SRL equation: $\mathbf{x}_X = A_X\mathbf{x}_X + \mathbf{c}_X$.

The initial state of the automaton is defined as

$$I_0 = (I_{0X_0} ,\ldots, I_{0X} ,\ldots, I_{0Z}),$$

where $I_{0X} = f_{X_0X}i_X , \quad X \in N,$

X_0 is the starting nonterminal.

The state transition function of the automaton is defined as

$$\mathrm{Goto}(I_k, s) = \mathrm{Closure}(\mathrm{Trans}(I_k, s)) \tag{5.2}$$

where

$$\mathrm{Trans}(I_k,s) = \mathrm{Trans}((I_{kX} , \ldots , I_{kX}, \ldots , I_{kZ}),s)$$

$$= (\mathrm{Trans}(I_{kX_0},s),\ldots, \mathrm{Trans}(I_{kX},s),\ldots,\mathrm{Trans}(I_{kZ},s)) \tag{5.3}$$

where for any $X \in N$, $\mathrm{Trans}(I_{kX},s) = J_{kX} = I_{kX}\partial_s A_X$ (5.4)

$$\mathrm{Closure}(J_k) = \mathrm{Closure}(J_{kX_0}, \ldots , J_{kX}, \ldots , J_{kZ})$$

$$= (\mathrm{Closure}(J_{kX_0}),\ldots, \mathrm{Closure}(J_{kX}),\ldots, \mathrm{Closure}(J_{kZ})) \tag{5.5}$$

where for any $X \in N$,

$$\mathrm{Closure}(J_{kX}) = J_{kX} + \sum_Y\sum_Z J_{kY}\partial_Z A_Y \Lambda_Y f_{ZX} i_X \tag{5.6}$$

If a state I_k contains at least one I_{kX} such that $I_{kX}\mathbf{c}_X = \lambda$, then it is a reduce state of the automaton.

Example 6. From Example 2,

obtain the LR(0) automaton.

$I_k=(\ I_{kS}\quad I_{kG}\quad I_{kE}\quad I_{kT}\ )$

$I_0=(\lambda\phi\phi\ \ \lambda\phi\phi\phi\phi\ \ \lambda\phi\phi\phi\phi\ \ \lambda\phi\phi\phi\phi)\qquad =I_0,$

$\mathrm{Trans}(I_0,G)=(\phi\lambda\phi\ \ \phi\phi\phi\phi\phi\ \ \phi\phi\phi\phi\phi\ \ \phi\phi\phi\phi\phi)=\mathrm{Goto}(I_0,G)=I_1,$

$\mathrm{Trans}(I_0,E)=(\phi\phi\phi\ \ \phi\lambda\phi\phi\ \ \phi\phi\lambda\phi\phi\ \ \phi\phi\phi\phi\phi)=\mathrm{Goto}(I_0,E)=I_2,$

$\mathrm{Trans}(I_0,T)=(\phi\phi\phi\ \ \phi\phi\phi\phi\phi\ \ \phi\lambda\phi\phi\phi\ \ \phi\phi\lambda\phi\phi)=\mathrm{Goto}(I_0,T)=I_3^{\#},$

$\mathrm{Trans}(I_0,f)=(\phi\phi\phi\ \ \phi\phi\phi\phi\lambda\ \ \phi\phi\phi\phi\phi\ \ \phi\lambda\phi\phi\phi)=\mathrm{Goto}(I_0,f)=I_4^{\#},$

$\mathrm{Trans}(I_1,\#)=(\phi\phi\lambda\ \ \phi\phi\phi\phi\phi\ \ \phi\phi\phi\phi\phi\ \ \phi\phi\phi\phi\phi)=\mathrm{Goto}(I_1,\#)=I_5^{A},$

$\mathrm{Trans}(I_2,=)=(\phi\phi\phi\ \ \phi\phi\lambda\phi\phi\ \ \phi\phi\phi\phi\phi\ \ \phi\phi\phi\phi\phi)$

$\Rightarrow(\phi\phi\phi\ \ \phi\phi\lambda\phi\phi\ \ \lambda\phi\phi\phi\phi\ \ \lambda\phi\phi\phi\phi)=\mathrm{Goto}(I_3,=)=I_6,$

$\mathrm{Trans}(I_2,+)=(\phi\phi\phi\ \ \phi\phi\phi\phi\phi\ \ \phi\phi\phi\lambda\phi\ \ \phi\phi\phi\phi\phi)$

$\Rightarrow(\phi\phi\phi\ \ \phi\phi\phi\phi\phi\ \ \phi\phi\phi\lambda\phi\ \ \lambda\phi\phi\phi\phi)=\mathrm{Goto}(I_3,+)=I_7,$

$\mathrm{Trans}(I_3,*)=(\phi\phi\phi\ \ \phi\phi\phi\phi\phi\ \ \phi\phi\phi\phi\phi\ \ \phi\phi\phi\lambda\phi)=\mathrm{Goto}(I_4,*)=I_8,$

$\mathrm{Trans}(I_6,E)=(\phi\phi\phi\ \ \phi\phi\phi\lambda\phi\ \ \phi\phi\lambda\phi\phi\ \ \phi\phi\phi\phi\phi)=\mathrm{Goto}(I_6,E)=I_9^{\#},$

$\mathrm{Trans}(I_6,f)=(\phi\phi\phi\ \ \phi\phi\phi\phi\phi\ \ \phi\phi\phi\phi\phi\ \ \phi\lambda\phi\phi\phi)=\mathrm{Goto}(I_6,f)=I_{10}^{\#},$

$\mathrm{Trans}(I_7,T)=(\phi\phi\phi\ \ \phi\phi\phi\phi\phi\ \ \phi\phi\phi\phi\lambda\ \ \phi\phi\lambda\phi\phi)=\mathrm{Goto}(I_7,T)=I_{11}^{\#},$

$\mathrm{Trans}(I_8,f)=(\phi\phi\phi\ \ \phi\phi\phi\phi\phi\ \ \phi\phi\phi\phi\phi\ \ \phi\phi\phi\phi\lambda)=\mathrm{Goto}(I_8,f)=I_{12}^{\#},$

$\mathrm{Trans}(I_6,T)=\mathrm{Goto}(I_6,T)=I_3^{\#},\quad \mathrm{Trans}(I_7,f)=\mathrm{Goto}(I_7,f)=I_{10}^{\#},$

$\mathrm{Trans}(I_9,+)=\mathrm{Goto}(I_9,+)=I_7,\quad \mathrm{Trans}(I_{11},*)=\mathrm{Goto}(I_{11},*)=I_8,$

	1	2	3	4	5	6
	I_0	I_0	I_0	I_6	I_6	I_7
	S	G	E	T	E	T	T
I_0	1	1	1	1	.	.	.
I_1	2	.	.	.	.	.	.
I_2	.	2	3	.	.	.	.
$I_3^{\#}$	.	.	2	3	2	3	.
$I_4^{\#}$	.	5	.	2	.	.	.
I_5^{A}	3	.	.	.	.	.	.
I_6	.	3	.	.	1	1	.
I_7	.	.	4	.	4	.	1
I_8	.	.	.	4	.	4	4
$I_9^{\#}$	.	4	.	.	3	.	.
$I_{10}^{\#}$	.	.	.	.	.	2	2
$I_{11}^{\#}$	.	.	5	.	5	.	3
$I_{12}^{\#}$	.	.	.	5	.	5	5

The embedding matrix

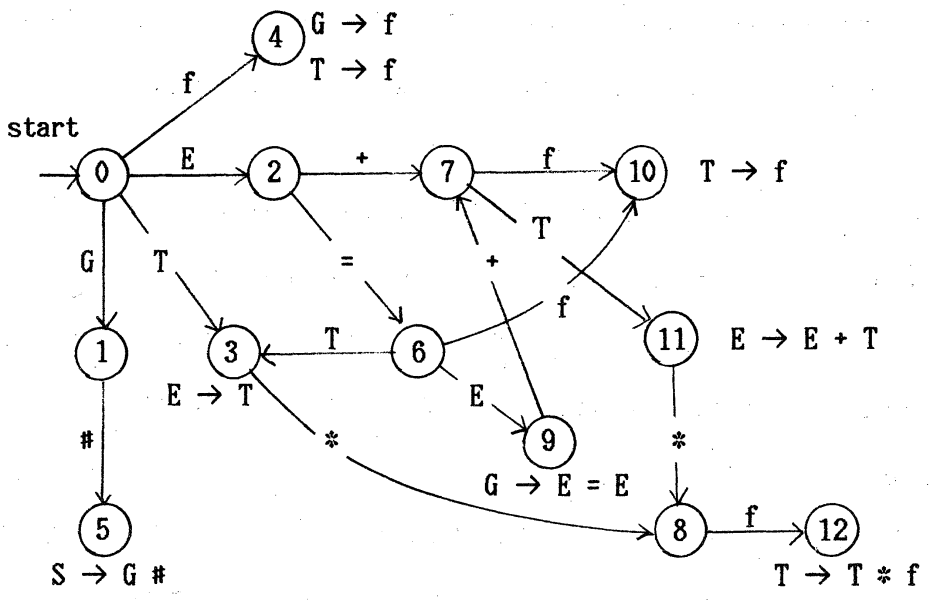

Fig.6

The above method constructs an LR(0) automaton from a given SRL equation (2.2) so as to embed once or more times each one of finite automata described by the SRL equation into the LR(0) automaton. In the above example, this is shown by a matrix in the right-hand side. In the matrix, each column corresponds to one of occurrences of the embedded automata and, for each state I_k of the LR(0) automaton, each element of the I_k-th row contains λ if there is, in the embedded automata, a state which is embedded in I_k , otherwise ϕ . (In Fig.6, the state number is written instead of λ). In this case, automata S and G were each embedded from state I_0, automaton E was embedded from states I_0 and I_6, and automaton T was embeded from states I_0, I_6 and I_7.

This matrix is called an embedding matrix and each element is denoted by $E[I_k,(I_\ell,Y)]$, where I_k is the row number associated with the state I_k of the derived LR(0) automaton and (I_ℓ,Y) indicates an occurrence of embedding the automaton Y from state I_ℓ in the LR(0) automaton. For each nonterminal transition (I_ℓ,Y), there exists one and only one occurrence of embedding the automaton Y from state I_ℓ, and it suffices that $I_\ell Y^t i_Y = \lambda$.

6. FOLLOW SETS

For the LR(0) automaton, Follow set concerned with each nonterminal (state) transition (I_k,X) is defined as

$$\text{Follow}(I_k,X) = \{ t \in T \mid X_0 \overset{+}{\underset{R}{\Rightarrow}} \theta X t w \text{ and } \theta \text{ access state } I_k \}$$

where X_0 is the starting nonterminal and $w \in T^*$.

DeRemer and Penello derived from the definition two types of inclusive relations related to Follow sets and gave a method to obtain the sets concretely on the basis of the inclusive

relations in the manner of recursive computation, which needs repeative computation of set union and set comparison taking much time.

Our method shown below takes not so much time because of depending on mainly closure operation of an n x n Boolean matrix instead of the recursive computation on sets, where n is the number of nonterminal transitions in the LR(0) automaton.

Generally, a production grammar has two types of rules such as $Y \to \alpha Xs\beta$, $\alpha,\beta \in V^*$ and $Y \to \alpha X$, $\alpha \in V^*$, which introduce two types of situations in the derivation sequences, as follows:

For $Y \to \alpha Xs\beta$, there is $X_0 \overset{+}{\underset{R}{\Longrightarrow}} \delta Yw \underset{R}{\Longrightarrow} \delta\alpha Xs\beta w$

and for $Y \to \alpha X$, there is $X_0 \overset{+}{\underset{R}{\Longrightarrow}} \delta Yw \underset{R}{\Longrightarrow} \delta\alpha Xw$.

Similarly, in the related LR(0) automaton, two types are derived as shown in Fig.7 and Fig.8.

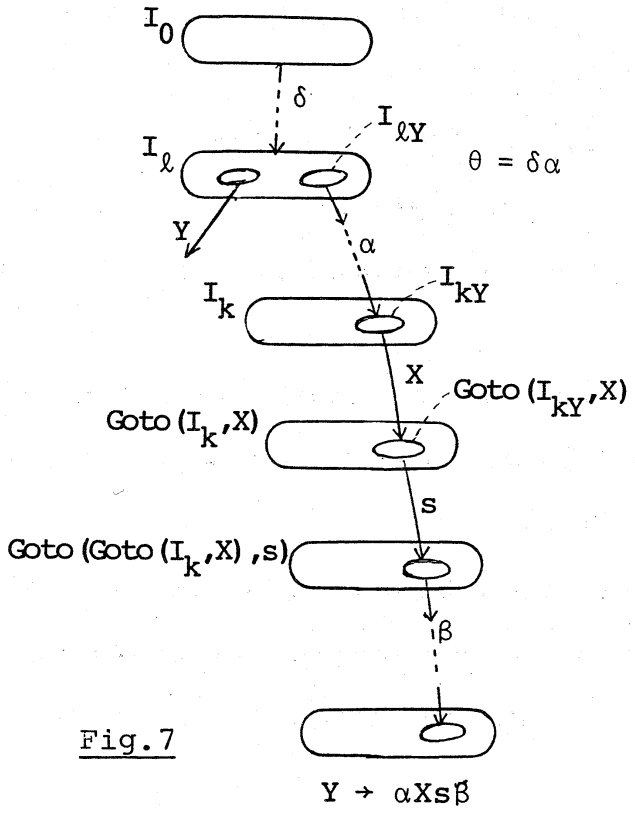

Fig.7

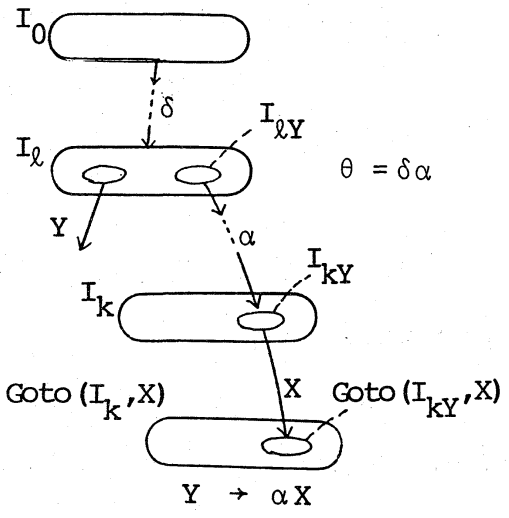

Fig.8

According to the 1st type, We have the following.

PROPOSITION 6.1 The following 2 conditions are all equivalent.

- There exists a rule $Y \rightarrow \alpha X s \beta$, $\alpha, \beta \in V^*$.
- There exist an occurrence of embedding an automaton Y and a state I_k of the LR(0) automaton such that

$$I_{kY}\partial_X A_Y \partial_s A_Y \Lambda_Y = \lambda \tag{6.1}$$

Fig.7 shows this situation and its relation to the definition of Follow sets. Namely, if Eq.(6.1) holds, we can expect to have the following relation.

$$\text{Follow}(I_k, X) \supseteq \text{First}_T(s) \tag{6.2}$$

where $\text{First}_T(s) = s$, for $s \in T$,

$= T \cap \text{First}(s)$, for $s \in N$.

The above conditional relation is simply shown as follows:

$$\text{Follow}(I_k, X) \supseteq I_{kY}\partial_X A_Y \partial_s A_Y \Lambda_Y \text{First}_T(s) \tag{6.3}$$

This relation for any $Y \in N$ and for any $s \in V$ is given as

$$\text{Follow}(I_k, X) \supseteq d[(I_k, X)] \tag{6.4}$$

$$\text{where } d[(I_k, X)] = \sum_Y I_{kY}\partial_X A_Y b_Y \tag{6.5}$$

$$b_Y = \sum_s \partial_s A_Y \Lambda_Y \text{First}_T(s) \tag{6.6}$$

Note that $I_{kY}\partial_X A_Y$ is a λ-vector $\text{Trans}(I_{kY}, X)$ given in the previous chapter.

Associated with the 2nd type, the following is given.

PROPOSITION 6.2 The following 3 conditions are all equivalent.

- There is a rule $Y \rightarrow \alpha X$, $\alpha \in V^*$.
- There are two nonterminal transitions (I_k, X) and (I_ℓ, Y) and there is an occurrene of embedding an automaton Y of which the initial state is embedded in I_ℓ and of which a final state is embedded in state $\text{Goto}(I_k, X)$.

- There are two nonterminal transition (I_k,X) and (I_ℓ,Y) such that $E[I_k,(I_\ell,Y)] = \lambda$ and there is a state $Goto(I_k,X)$ reducible concerned with Y, that is, it holds that $I_{kY}\partial_X A_Y c_Y = \lambda$.

Fig.8 shows the above situation and relations to the definition of Follow sets. Namely, if the above condition holds, we can expect to have the following relation

$$Follow(I_k,X) \supseteq Follow(I_\ell,Y) \tag{6.7}$$

The above conditional relation is simply shown as

$$Follow(I_k,X) \supseteq \sum_{(I_\ell,Y)\in\Omega} \gamma[(I_k,X),(I_\ell,Y)]\cdot Follow(I_\ell,Y) \tag{6.8}$$

where

Ω is an ordered set of all nonterminal transitions.

$\gamma[(I_k,X),(I_\ell,Y)] = \lambda$ if the condition of PROPOSITION 6.2 holds,

$= \phi$ otherwise.

From Eqs.(6.4) and (6.8), Follow sets are, for a given LR(0) automaton, defined as follows:

$$Follow(I_k,X) = \sum_{(I_\ell,Y)\in\Omega} \gamma[(I_k,X),(I_\ell,Y)]\cdot Follow(I_\ell,X) + d[(I_k,X)] \tag{6.9}$$

In order to compute Eq.(6.9), put

$$\mathbf{u} = (\, u[(I_k,X)]\,),\ u[(I_k,X)] = Follow(I_k,X)$$

$$\Gamma = (\, \gamma[(I_k,X),(I_\ell,Y)]\,),$$

$$\mathbf{d} = (\, d[(I_k,X)]\,).$$

Then, the desired solution is obtained as follows:

$$\mathbf{u} = \Gamma\mathbf{u} + \mathbf{d} =_s \Gamma^*\mathbf{d} \tag{6.10}$$

Example.7 Obtain Follow sets in the LR(0) automaton in Example 6.

Because there are six allowable nonterminal transitions (I_0,G), (I_0,E), (I_0,T), (I_6,E), (I_6,T) and (I_7,T) in the LR(0) automaton, it is necessary to obtain a six-dimensional vector d and an n × n matrix Γ. d needs b which is composed of b_S, b_G, b_E and b_T obtained from Eq.(6.5) and Example 5.

$$b = \begin{bmatrix} b_S \\ b_G \\ b_E \\ b_T \end{bmatrix}, \text{ where } b_S = \begin{bmatrix} \{ f \} \\ \{ * \} \\ \phi \end{bmatrix}, \; b_G = \begin{bmatrix} \{ f \} \\ \{ = \} \\ \{ f \} \\ \phi \\ \phi \end{bmatrix}, \; b_E = \begin{bmatrix} \{ f \} \\ \phi \\ \{ + \} \\ \{ f \} \\ \phi \end{bmatrix}, \; b_T = \begin{bmatrix} \{ f \} \\ \phi \\ \{ * \} \\ \{ f \} \\ \phi \end{bmatrix}$$

$$d = \begin{bmatrix} d[(I_0,G)] \\ d[(I_0,E)] \\ d[(I_0,T)] \\ d[(I_6,E)] \\ d[(I_6,T)] \\ d[(I_7,T)] \end{bmatrix} = \begin{bmatrix} Trans(I_0,G) \\ Trans(I_0,E) \\ Trans(I_0,T) \\ Trans(I_6,E) \\ Trans(I_6,T) \\ Trans(I_7,T) \end{bmatrix} b = \begin{bmatrix} \{ \# \} \\ \{=, +\} \\ \{ * \} \\ \{ + \} \\ \{ * \} \\ \{ * \} \end{bmatrix}, \quad \Gamma = \begin{bmatrix} \phi & \phi & \phi & \phi & \phi & \phi \\ \phi & \phi & \phi & \phi & \phi & \phi \\ \phi & \lambda & \phi & \phi & \phi & \phi \\ \lambda & \phi & \phi & \phi & \phi & \phi \\ \phi & \phi & \phi & \lambda & \phi & \phi \\ \phi & \lambda & \phi & \lambda & \phi & \phi \end{bmatrix}$$

Thus, we have the desired Follow sets as follows:

$$u = \begin{bmatrix} Follow(I_0,G) \\ Follow(I_0,E) \\ Follow(I_0,T) \\ Follow(I_6,E) \\ Follow(I_6,T) \\ Follow(I_7,T) \end{bmatrix} = \Gamma^* d = \begin{bmatrix} \lambda & \phi & \phi & \phi & \phi & \phi \\ \phi & \lambda & \phi & \phi & \phi & \phi \\ \phi & \lambda & \lambda & \phi & \phi & \phi \\ \lambda & \phi & \phi & \lambda & \phi & \phi \\ \lambda & \phi & \phi & \lambda & \lambda & \phi \\ \lambda & \lambda & \phi & \lambda & \phi & \lambda \end{bmatrix} \begin{bmatrix} \{ \# \} \\ \{=, +\} \\ \{ * \} \\ \{ + \} \\ \{ * \} \\ \{ * \} \end{bmatrix} = \begin{bmatrix} \{ \# \} \\ \{=, +\} \\ \{=, +, *\} \\ \{\#, +\} \\ \{\#, +, *\} \\ \{\#, =, +, *\} \end{bmatrix}$$

7. LOOK-AHEAD SETS

LALR(1) parsers use sets called Look-Ahead sets to make parse deterministic. The sets are defined for each reduce state I_k concerned with a nonterminal X, i.e. a rule $X \to \alpha$, as follows:

$$LA(I_k, X \to \alpha) = \{ t \in T \mid X_0 \overset{+}{\underset{R}{\Longrightarrow}} \delta X t w \text{ and } \delta\alpha \text{ access } I_k \}$$

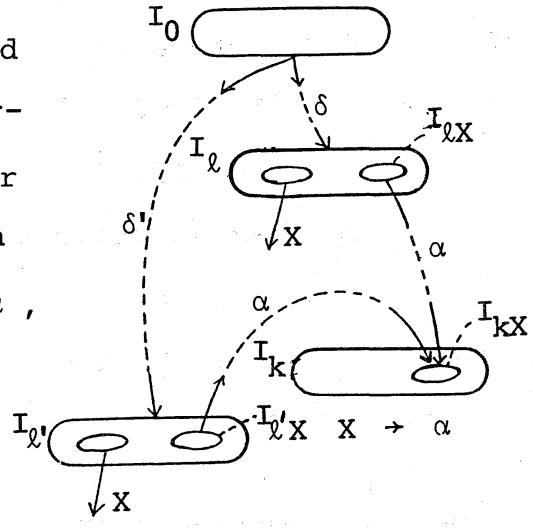

Fig.9

$= \{ \text{Follow}(I_\ell, X) \mid I_\ell \text{ transits to } I_k \text{ by } \alpha \}$

Fig.9 shows the situation in the LR(0) automaton concerned with the definition. So we can expect to have the following.

PROPOSITION 7.1 The following 3 conditions are all equivalent.

- I_ℓ transits to I_k by α for the rule $X \rightarrow \alpha$.
- There exists an occurrence of embedding an automaton X of which the initial state is embedded in I_ℓ and of which a final state is embedded in I_k and the initial state transits to the final state by α in the automaton X.
- $E[I_k,(I_\ell,X)] = \lambda$ and $I_{kX}c_X = \lambda$ (7.1)

Here, we obtain a table called Look-Back table from the embedding matrix in such a manner that if I_k is not a reduce state, then the I_k-th row is ommitted. Note that only states satisfying Eq.(7.1) are left. There may be a state reducible concerned with two or more nonterminals. Accordingly, we expand the LB table to the row-direction such that if I_k is reducible concerned with nonterminals X, X',..., X", then we make reducible state-nonterminal pairs (I_k,X), (I_k,X'),..., $(I_k,X")$ and expand the I_k-th row to the (I_k,X)-th, the (I_k,X')-th ,..., the $(I_k,X")$-th rows so as to satisfy the following condition. The obtained is a matrix called Look-Back matrix of which the $[(I_k,X),(I_\ell,Y)]$ element is defined as

$$\Delta[(I_k,X),(I_\ell,Y)] = LB[I_k,(I_\ell,Y)] \quad \text{if } X = Y$$
$$= \phi \quad \text{otherwise.}$$

Note that the Look-Back matrix is obtained directly from the embedding matrix as follows:

$$\Delta[(I_k,X),(I_\ell,Y)] = E[I_k,(I_\ell,Y)], \text{ if } X = Y$$

$$= \phi \qquad \text{otherwise.} \qquad (7.2)$$

Then for each reducible state-nonterminal pair $(I_k,\ X \to \alpha)$, the Look-Ahead set $LA(I_k,\ X \to \alpha)$ is defined equationally as follows:

$$LA(I_k,\ X \to \alpha) = \sum_{(I_\ell, Y) \in \Omega} \Delta[(I_k,X),(I_\ell,Y)] \cdot Follow(I_\ell,Y) \qquad (7.3)$$

In order to compute them, put

$$\mathbf{v} = (LA(I_k,\ X \to \alpha)),$$

$$\Delta = (\ \Delta[(I_k,X),(I_\ell,Y)]\ ),$$

$$\mathbf{u} = (\ u[(I_\ell,Y)]\ ),\quad u[(I_\ell,Y)] = Follow(I_\ell,Y)$$

Then, from Eqs.(7.3) and (6.9), the desired Look-Ahead sets are obtained, as follows:

$$\mathbf{v} = \Delta\mathbf{u} = \Delta\Gamma^{*}\mathbf{d} \qquad (7.4)$$

Note that it is not necessary to obtain Follow sets concretely in order to compute Look-Ahead sets.

Example 8 Obtain Look-Ahead sets of the LR(0) automaton in Example 6.

$$u = (\ LA(I_k,\ X \to \alpha)\ ) = \Delta\Gamma^{*}\mathbf{d}$$

$$= \begin{bmatrix} LA(I_3,\ E\to T) \\ LA(I_4,\ G\to f) \\ LA(I_4,\ T\to f) \\ LA(I_9,\ G\to E{=}E) \\ LA(I_{10},\ T\to f) \\ LA(I_{11},\ E\to E{+}T) \\ LA(I_{12},\ T\to T{*}f) \end{bmatrix} = \begin{bmatrix} \phi\ \lambda\ \phi\ \lambda\ \phi\ \phi \\ \lambda\ \phi\ \phi\ \phi\ \phi\ \phi \\ \phi\ \phi\ \lambda\ \phi\ \phi\ \phi \\ \lambda\ \phi\ \phi\ \phi\ \phi\ \phi \\ \phi\ \phi\ \phi\ \phi\ \lambda\ \lambda \\ \phi\ \lambda\ \phi\ \lambda\ \phi\ \phi \\ \phi\ \phi\ \lambda\ \phi\ \lambda\ \lambda \end{bmatrix} \begin{bmatrix} \lambda\ \phi\ \phi\ \phi\ \phi\ \phi \\ \phi\ \lambda\ \phi\ \phi\ \phi\ \phi \\ \phi\ \lambda\ \lambda\ \phi\ \phi\ \phi \\ \lambda\ \phi\ \phi\ \lambda\ \phi\ \phi \\ \lambda\ \phi\ \phi\ \lambda\ \lambda\ \phi \\ \lambda\ \lambda\ \phi\ \lambda\ \phi\ \lambda \end{bmatrix} \begin{bmatrix} \{\ \#\ \} \\ \{=,+\} \\ \{\ *\ \} \\ \{\ +\ \} \\ \{\ *\ \} \\ \{\ *\ \} \end{bmatrix} = \begin{bmatrix} \{\ \#,\ =,\ +\ \} \\ \{\ \#\ \} \\ \{\ =,\ +,\ *\ \} \\ \{\ \#\ \} \\ \{\ \#,\ =,\ +,\ *\ \} \\ \{\ \#,\ =,\ +\ \} \\ \{\ \#,\ =,\ +,\ *\ \} \end{bmatrix}$$

8. CONCLUSION

For a given BNF or extended BNF, Tixier's SRL equation equivalent to it was used. The SRL equation, which describes multiple finite automata system, was at first made empty-free. First sets were equationally defined on the empty-free SRL equation and solved.

A method that derives an LR(0) automaton from a given SRL equation was given. Follow sets were equationally defined on the LR(0) automaton and solved. LALR(1) parse on the LR(0) automaton uses Look-Ahead sets composed of Follow sets.

Until now, concrete computation of the above sets has been done in the recursive manner on the equations. Instead, this paper gave direct and efficient computation using the solutions.

This work was supported by the Spec. Res. Grant. Min. Educ. of Japan in 1986. This assistance is acknowledged.

REFERENCES

1) F.DeRemer and T.Pennello: "Efficient computation of LALR(1) look-ahead sets," ACM. Trans. Programming Languages and Systems, Vol.4, pp.615-649(1982).

2) J.Tremblay, P.G.Sorenson: The Theory and Practice of Compiler Writing, McGraw-Hill, New York(1985).

3) V.Tixier: "Recursive functions of regular expressions in language analysis," Tech. Rpt. CS58, Comp. Sci. Dept., Stanford U.(1968).

4)H.Anzai, et al.: "Semi-linear algebra, 1 to 8," Papers of Tech. Group Automaton and Languages, I.E.C.E., Japan, AL80-60,60,62; AL81-4,5,38,39,72(1981).(in Japanese)