Proc. Indian Acad. Sci., Vol. 88 A, Part 3, Number 1, January 1979, pp. 1-19,

On the covering of syntax-directed translations for context-free grammars

R K SHYAMASUNDAR
National Centre for Software Development and Computing Techniques,
Tata Institute of Fundamental Research, Bombay 400 005

MS received 13 February 1978

Abstract. A necessary and sufficient set of conditions is obtained that relates any two context-free grammars G_1 and G_2 with the property that whenever G_2 left—or right—covers G_1, the syntax-directed translations (SDT's) with underlying grammar G_1 is a subset of those with underlying grammar G_2. Also the case that G_2 left—or right—covers G_1 but the SDT's with underlying grammar G_1 is not a subset of the SDT's with underlying grammar G_2 is considered; in this case an algorithm is described to obtain the syntax-directed translation schema (SDTS) with underlying grammar G_2 to the given SDTS with underlying grammar G_1, if it exists.

Keywords. Chomsky normal form; context-free grammar; covering of grammars; semantics; syntax-directed translation schema.

1. Introduction

Context-free grammars have been used extensively for describing the syntax of programming languages and natural languages. Parsing algorithms for context-free grammars consequently play a useful role in the implementation of compilers and interpreters for programming languages. Many of the parsing algorithms require that the grammar under consideration be in some normal form or have some special property such as not having left recursion for use with top-down parsing techniques.

We can consider a grammar G_2 to be similar from the point of view of parsing, with respect to G_1 if $L(G_2)=L(G_1)$ and we can express the left and/or right parse of a sentence generated by G_1 in terms of its parse in G_2. In such a case, we say that G_2 covers G_1; hence the parses in G_1 can be recovered from those in G_2 by suitable mapping. Similar grammars play an important role in the construction of translators. Different grammars for the same language, which are similar, can be used at different stages of translation depending upon the requirements of the particular task.

The importance of syntax-directed translation is dealt with in detail by Aho and Ullman [1] and Lewis and Stearns [5]. Intuitively, a syntax-directed translation schema (SDTS) is simply a grammar in which translation elements are attached to each production. Whenever a production is used in the derivation of an input sentence, the translation element is used to help complete a portion of the output sentence associated with the portion of the input sentence generated by that production.

An interesting problem relating the covering of grammars and the SDTS of the grammars is the following: (*cf.* Aho and Ullman [1]).

Problem: Is it true that whenever G_2 left—or right—covers G_1, every SDTS with G_1 as underlying grammar is equivalent to an SDTS with G_2 as underlying grammar? The answer to this is in the negative (*cf.* Aho and Ullman[1]).

In this paper we give a necessary and sufficient set of conditions relating G_1 and G_2 so that whenever G_2 left—or right—covers G_1, the SDT with underlying grammar G_1 is a subset of the SDT with underlying grammar G_2. Also an algorithm is described to obtain the SDTS with underyling grammar G_2 equivalent to the given SDTS with underlying grammar G_1 if it exists.

1.1. *Syntax-directed translations*

Definition 1.1: SDTS is a 5-tuple,

$T = (V_N, V_T, \Delta, R, S)$, where V_N is a finite set of nonterminal symbols, V_T is a finite input alphabet, Δ is a finite output alphabet, S is a distinguished nonterminal in V_N, the start symbol and R is a finite set of rules of the form $A \rightarrow \alpha, \beta$, where α is in $(V_N \cup V_T)^*$, β is in $(V_N \cup \Delta)^*$, and the nonterminals in β are a permutation of the nonterminals in α.

Let $A \rightarrow \alpha, \beta$ be a rule. To each nonterminal of α there is associated an identical nonterminal of β. If a nonterminal B appears only once in α and β, then the association is obvious. If B appears more than once, we use integer subscripts to indicate the association. This association is an intimate part of the rule.

A translation form of an SDTS, T, is defined as follows:

(i) (S, S) is a translation form and the two S's are said to be associated.

(ii) If $(\alpha A \beta, \alpha' A \beta')$ is a translation form, in which the two explicit instances of A are associated, and if $A \rightarrow \gamma, \gamma'$ is a rule in R, then $(\alpha\gamma\beta, \alpha'\gamma'\beta')$ is a translation form. The nonterminals of γ and γ' are associated in the translation form exactly as they are associated in the rule. The nonterminals of α and β are associated with those of α' and β' in the new translation form exactly as in the old. The association will again be indicated by superscripts when needed, and this association is an essential feature of the translation form.

If the translation forms $(\alpha A \beta, \alpha' A \beta')$ and $(\alpha\gamma\beta, \alpha'\gamma'\beta')$, together with their associations, are related as above, then we write $(\alpha A \beta, \alpha' A \beta') \Longrightarrow_T (\alpha\gamma\beta, \alpha'\gamma'\beta')$.

Definition 1.2: The translation defined by T, denoted $\tau(T)$, is the set of pairs

$$\{(x, y) \mid (S, S) \Longrightarrow^* (x, y),\ x \in V^* \text{ and } y \in \Delta^*\}$$

If T is an SDTS, then $\tau(T)$ is called SDT.

Definition 1.3: An SDTS is *semantically unambiguous* if there are no two distinct rules of the form $A \rightarrow \alpha, \beta$ and $A \rightarrow \alpha, \gamma$.

Covering of Grammars

Let $G_1 = (V_{N_1}, V_T, P_1, S_1)$ and $G_2 = (V_{N_2}, V_T, P_2, S_2)$ be context-free grammars (CFGs) such that $L(G_1) = L(G_2)$.

Definition 1.4: We say that G_2 left-covers G_1 if there is a homomorphism h from P_2 to P_1 such that

(i) If $S_2 \Longrightarrow^{\pi} w$, then $S_1 \Longrightarrow^{h(\pi)} w$, and

(ii) For all π such that $S_1 \Longrightarrow^{\pi} w$ there exists π' such that $S_2 \Longrightarrow^{\pi'} w$ and $h(\pi') = \pi$.

Terms and symbols not specifically defined here are employed with their usual meaning as in Aho and Ullman [1].

Definition 1.5: *Terminating property*

We say that G_2 and G_1 satisfy the terminating property whenever

(i) G_2 left-or right-covers G_1,

(ii) $V_{N_2} = V^{(1)}_{N_2} \cup V^{(2)}_{N_2}$

where $V^{(1)}_{N_2} = \left\{ A \in V_{N_2} \mid i{:}A \to \alpha, \text{ is in } P_2, \text{ such that for all rules in } P_2 \text{ with } A \text{ as the left-side, } h(i) \neq \lambda \right\}$, and $V^{(2)}_{N_2} = V_{N_2} - V^{(1)}_{N_2}$,

i.e., $V^{(1)}_{N_2} \cap V^{(2)}_{N_2} = \phi$, and

(iii) If $A \in V^{(1)}_{N_2}$ then if $A \Longrightarrow_{\pi_1} x \in V_T^*$ then

$$B \underset{\pi_2\, h(\pi)}{\Longrightarrow} x' \in V_T^* \text{ where } \pi_2 = h(\pi_1).$$

In the following we assume that the grammars G_2 and G_1 are proper and nonredundant and hence the associated SDTS with the underlying grammars G_2 and G_1 will be free of useless nonterminals*.

Proposition: If G_2 left—or right—covers G_1 (both G_2 and G_1 are non-redundant) and $i{:}B \to x_1 Bx_2$ where $x_1, x_2 \in V_T^*$, is in G_2, then $h(i{:}B \to x_1 Bx_2) \neq \lambda$.

Proof: We have $S_2 \Longrightarrow_{\pi} x$ where π is a sequence of productions (i.e., $\pi = \pi_1 \pi_2 \ldots \pi_i \pi_{i+1} \ldots \pi_r$) and $x \in V_T^*$, then by definition of covering, we should have

$$S_1 \Longrightarrow_{h(\pi)} x.$$

* In a CFG, $G = (V_N, V_T, P, S)$ a symbol X in $V_N \cup V_T$ is said to be *inaccessible* if X does not appear in any sentential form.

A nonterminal is said to be *useless* if it cannot generate a terminal string.

A CFG is said to be *cycle-free* if there is no derivation of the form $A \Longrightarrow^{*} A$.

A CFG is said to be *proper*, if it is cycle-free and free of useless symbols.

A CFG is said to be *nonredundant* if it is proper and free of inaccessible symbols.

Now, consider $S_2 \Longrightarrow_{\pi'} x'$ where $\pi' = \pi_1 \ldots \pi_i \pi_{i+1} \ldots \pi_r$ and $x' \in V_T^*$. Then, $S_1 \Longrightarrow x'$ since $h(\pi')=h(\pi)$. This contradicts the definition of covering and hence the proposition follows.

Section 1.2

Let $G_2=(V_{N_2}, V_T, P_2, S_2)$ left–or right–cover, $G_1=(V_{N_1}, V_T, P_1, S_1)$. Without loss of generality, it can be assumed that $V_{N_1} \cap V_{N_2}=\phi$. Let I and J denote the set of labels of the production rules of G_2 and G_1 respectively.

Let $i{:}A \to \alpha$ be a production rule belonging to G_2. In this section, corresponding to the nonterminal A occurring on the leftside of the ith rule associate a set of strings over $(V_{N_1} \cup V_T)^*$ through the following steps: (In the following, associate a string with the ith rule and associate a string with the nonterminal occurring on the left-side of the ith rule, are used interchangeably, as it will be clear from the context).

Step 1: To begin with, associate with the nonterminal occurring on the left-side of the ith rule in G_2 for which $h(i)=j$, $j \neq \lambda$, the nonterminal belonging to V_{N_1} occurring on the left side of the jth rule in G_1. This is done for all $i \in I$ such that $h(i) \neq \lambda$. Form the set,

$$H = \{ A \in V_{N_2} \mid h(i{:}A \to \alpha) = j, j \neq \lambda \}.$$

Step 2: Consider the terminal rules in G_2

(a) Let $h(i)=\lambda$. In this case associate with the ith rule in G_2, the string belonging to V_T^* occurring on the right-side of the ith rule.

(b) Let $h(i)=j$ and $j \neq \lambda$. In this case, in addition to the string belonging to V_T^*, appearing on the right-side of the ith rule, the nonterminal belonging to V_{N_1} occurring on the left-side of the jth rule in G_1 is associated with the ith rule in G_2.

Step 3: Consider the nonterminal rule in G_2

(a) Form the set,

$$H_1 = \{ i \in I \mid h(i) = \lambda \}.$$

First, choose a rule from H_1 that has a right side consisting of nonterminals of the following two types only:

(1) nonterminals (a subset of V_{N_2}) belonging to H.

(2) nonterminals that have been already processed in step 2.

The existence of such a rule is assured from the fact that G_2 covers G_1, and from the fact that G_1 and G_2 are nonredundant grammars.

Associate with such a rule (say ith rule) strings over $(V_{N_2} \cup V_T)^*$ obtained by replacing each nonterminal occurring on the right-side of the ith rule, by its respective associated sets.

Next, consider the rule belonging to H_1 whose right-side consists of nonterminals that are already processed; associate with this rule the set of strings obtained by replacing each nonterminal by its respective associated sets.

Repeat this step for all members belong to H_1.

(*b*) Consider the set of nonterminal rules for which $h(i) = j, j \neq \lambda$.

Choose a rule that has in its right-side, nonterminals that have been already processed and associate a set of strings with this rule as indicated earlier.

Repeat this step till all the rules are exhausted.

From the above procedure, a set of strings over $(V_{N_1} \cup V_T)^*$ has been associated with every rule. Note that if $i_1: A \to \alpha$ and $i_2: A \to \beta$ are two rules of G_2, then two different sets will be associated with A, one corresponding to rule i_1 and the other to rule i_2.

The following example illustrates the procedure.

Example 1.1: Let G_1 be the grammar

$$G_1:\ 1:\ S' \to 0\,S'\,1;\quad 2:\ S' \to 0\,1$$

and let G_2 be the Chomsky normal form (CNF) grammar equivalent to G_1

$$G_2:\ 1:\ S \to AB;\ 2:\ S \to AC;\ 3:\ B \to SC;\ 4:\ A \to 0;\ 5:\ C \to 1.$$

It can be observed that G_2 left-covers G_1 under the homomorphism $h(1) = 1$, $h(2) = 2$, and $h(3) = h(4) = h(5) = \lambda$.

From step 1, the first rule is associated with S' and the second rule also is associated with S'.

From step 2(a), rule 4 is associated with 0 and rule 5 is associated with 1.

From step 3(a), rule 3 is associated with $S'1$.

From step 3(b), rule 2 is associated with 0 1 and rule 1 is associated with $0S'1$.

Section 1.3

In the following, a sequence of production rules belonging to I^*, called the parse set, is associated with each member of the set of strings associated with each production rule of G_2. The meaning of parse set is made clear by the following example:

e.g., Consider example 1.1.

Rule 5 is associated with the string 1. Since the string 1 associated with this rule is the right-side of the rule 5, associate a parse π_5 (the suffix indicates the rule number). For convenience, the production number is represented as a suffix of π.

Consider rule 3. The string associated with this is $S'1$. Let us trace the steps involved during the process of association of the string $S'1$ with rule 3.

1. Use rule 3.

2. Substitute for C the associated string (the corresponding parse associated with this string in π_5).

3. S is replaced by S'. This step does not involve the application of any production rule G_2. It is obtained from the homomorphism h of I onto J.

Therefore we associate a parse $\pi_3\pi_5$ with the string $0S'1$ associated with rule 3.

Consider rule 1 of G_2. (i.e., $S \rightarrow AB$). The string $0S'1$ is associated with this rule. The sequence of steps associating $0S'1$ with rule 1 is given below:

$S \rightarrow AB$; the parse associated with S is π_1.

A is replaced by '0'; the parse associated with A is π_4.

B is replaced by $S'1$; the parse associated with B is $\pi_3\pi_5$. Hence, associate $\pi_1\pi_4\pi_3\pi_5$ $0S'1$ associated with rule 1.

In this manner associate a parse set with each element of the associated string of each production rule in G_2.

Now, obtain the following sets:

For those rules in G_2 for which $h(i)=j$, $j \neq \lambda$ obtain the set of associated strings with rule i. Let it be represented by q_{ij}. Let p_{ij} be the corresponding parse set associated with the members of q_{ij}.

$$\text{Let } p_j = \{p_{i_1 j}, p_{i_2 j}, \ldots, p_{i_k j}\} \text{ and}$$

$$q_j = \{q_{i_1 j}, q_{i_2 j}, \ldots, q_{i_k j}\} \text{ where } h(i_r) = j,$$

for $r=1, 2, \ldots, k$ and $p_{i_r j}$ is the parse set associated with rule i_r for which $h(i_r)=j$.

Let $P=\{p_1, p_2, \ldots, p_m\}$, where m is the number of production rules in G_1.

Definition 1.6: Define two homomorphisms h_1 and h_2 as follows:

$$h_1 : (V_{N_1} \cup V_T) \rightarrow V_{N_1}$$

$$h_1(x) = \begin{cases} \lambda \text{ if } x \in V_T \\ x \text{ if } x \in V_{N_1} \end{cases}$$

Now h_2 is defined for strings associated with the production rules of G_2. The strings associated with each production are strings over $(V_{N_1} \cup V_T \cup \overline{V}_N)^*$ where $\overline{V}_N=\{A \in V_{N_2} \mid \text{for all rules with } A \text{ as the subject such that, } h(A \rightarrow \alpha)=\lambda\}$.

[Here note that for $A \rightarrow x_1 A x_2, x_1, x_2 \in V_T^*$, $h(A \rightarrow x) \neq \lambda\}$.

$$h_2 : (V_{N_1} \cup V_T \cup V_N) \rightarrow V_{N_1} \cup \{\lambda\}$$

$$h_2(x)=\lambda \text{ if } x \in V_T,\ \lambda \text{ if } x \in \overline{V}_N \text{ and } x \text{ if } x \in V_{N_1}$$

Before obtaining the required conditions we prove the following lemma:

Lemma 1.1: If $h(i{:}A \rightarrow \alpha)=j, j \neq \lambda$, then $h_1(\alpha)$ matches with at least one element of $h_2(q_j)$.

Proof: Consider a rule in G_2, $i : A \rightarrow \alpha$ such that $h(i)=j$(let $j : B \rightarrow \gamma$).

Consider an element associated with the ith rule in G_2, and let $h(i)=j, j \neq \lambda$. From the process of association given earlier, it is clear that the parse associated with every element of q_j has atmost one production rule belonging to G_2 which does not map to λ.

Let an element associated with the *i*th rule in G_2 be represented by,

$$\alpha_1 A_1 \alpha_2 \ldots A_m \alpha_{m+1},$$

where $h_2(\alpha_i)=\lambda$, for all $1 \leqslant i \leqslant m+1$, and $A_1, \ldots, A_m$ are nonterminals in V_{N_2} such that P_2 contains atleast one rule with A_i as the subject for which the mapping h is nonempty.

We have,

$$A \overset{\bullet}{\Longrightarrow}_{G_2}^{*} \alpha_1 A_1 \alpha_2 \ldots A_m \alpha_{m+} \qquad (1)$$

Consider the derivation of a terminal string from A such that

$$A_i \underset{\pi_{A_i}\ \pi_i}{\Longrightarrow} x_i \in V_T^* \text{ where } h(\pi_{A_i}) \neq \lambda.$$

Since by the definition of covering,

$A \Longrightarrow_{G_2} x \in V_T^*$, implies

$B \Longrightarrow_{G_2} x' \in V_T^*$,

there should be a nonterminal in γ corresponding to every A_i in (1). [Here 'correspondence' is understood in the following way: If $h(i)=j$ and $i: A \rightarrow \alpha$ and $j: B \rightarrow \gamma$, then B corresponds to A]

Hence the lemma follows.

Section 1.3.1

Let $M_j=\{x \in p_{i_r j} \mid h(i_r)=j$, if the jth rule in G_1 is of the form $j: A' \rightarrow \alpha'$ then the element in $q_{i_r j}$ corresponding to x, say β, satisfies the property, $h_2(\beta)=h_1(\alpha')\}$.

Let $M=\{M_1, M_2, \ldots, M_m\}$, where m is the number of rules in G_1.

Example 1.2: Consider example 1.1. The grammar G_1 has the production rules, $1{:}S' \rightarrow 0S'1$ and $2{:}\ S' \rightarrow 01$. It may be observed that the right side of rule 1 in G_1, viz., $0S'1$, matches with the string associated with rule 1 of G_2 in the sense of lemma 1.1. Similarly, the right side of rule 2 in G_2, viz., 01, matches with the string associated with rule 2 of G_2; in fact, in this case the strings are identical.

Section 1.4

In this section, we obtain a sufficient set of conditions for the problem cited in section 1.

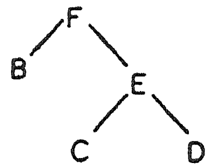
Remark 1.1: Consider a rule $A \rightarrow BCD$ (1)

and two other rules: $F \rightarrow BE$ (2)

and $E \rightarrow CD$ (3)

Here, A, B, C, D, E and F are all nonterminals. Now, the direct derivation of rule 1 can be represented by

and corresponding derivation with rule 2 and 3 together can be represented by



The frontiers of these two trees are identical. Now, consider a translation, CDB, associated with rule 1; the corresponding translation can be achieved by associating the translations EB and CD respectively, with rules 2 and 3. However, if we associate a *translation* CBD with rule 1, then it is impossible to obtain the same translation from the set of rules 2 and 3.

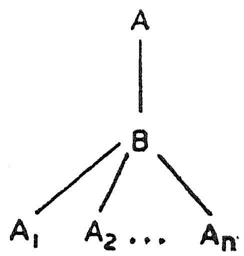
Theorem 1.1: If G_2 left-or right-covers G_1 satisfying the terminating property (definition 1.5), then the following conditions form a sufficient set of conditions for the problem cited in section 1.

(i) each member of M_i, $1 \leqslant i \leqslant m$ (m is the number of rules in G_1), consists of at most one multiple-length production* (MLP) rule and

(ii) $M_i \cap M_j$ $(i \neq j)$ consists of at most of only simple length production (SLP) rules for all $i \neq j$, so that the SDTS (semantically unambiguous) with the underlying grammar G_1 is a subset of the SDTS with underlying grammar G_2.

Proof : Consider condition (*i*)

Since the elements in M_i satisfy condition (i), a typical structure for an element in M_i will have the form shown below:



Let α be the element of the rule $D \rightarrow \alpha$ in G_1 corresponding to this element in M_i that satisfies lemma 1.1.

$$\text{Let } \alpha = x_1 D_1 x_2 \ldots x_r D_r x_{r+1}, \text{ where } x_i \in V_T^*, \text{ and } D_i \in V_{N_1}.$$

*A production is said to be an MLP if the number of symbols occurring on the right-handside of the production is more than one; a production is said to be an SLP if there is only one symbol on the right-hand side.

Then

$$h_1 (x_1 D_1 x_2 \ldots D_r x_{r+1}) = h_2 (A_1 A_2 \ldots A_n)$$

i.e., $D_1 D_2 \ldots D_r = A_i \ldots A_{i'}$

where $i' - i + 1 = r$.

Now the translations that can be associated with the rule $D \to \alpha$ in G_1 corresponds to the permutations of $D_1 D_2, \ldots, D_r$. To get a corresponding translation in G_2, we have to obtain a corresponding permutation of $A_i, \ldots, A_{i'}$. This is always possible since $A_i, \ldots, A_{i'}$ are direct descendants of the node B (except for chains). Thus all translations defined* by,

$$x'_1 D_{p(1)} x'_2 D_{p(2)} \ldots x'_r D_{p(r)} x'_{r+1}$$

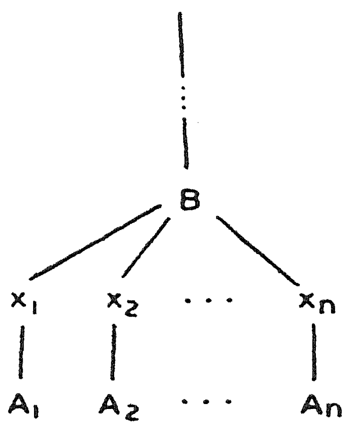
associated with the rule $D \to \alpha$ in G_1, can be achieved by associating the corresponding translations with B.

Consider condition (ii) assuming that condition (i) holds.

Case (a): Let $(M_i \cap M_j) = \phi$. In this case it is obvious that there is no chance of two distinct translations being associated with any rule of G_2.

Hence sufficiency holds in this case.

Case (b): Let $(M_i \cap M_j)$ consist of SLP rules. Here again the translation associated with G_2 will be unambiguous; this is illustrated below by considering the typical tree structure shown below:



Now, since the translation associated with the rules in G_2 is associated with the node B (subject of MLP's) and not with the subject of SLP's there is no chance of ambiguous translation being associated with any rule.

In deriving the trees of the SDT's with underlying grammar G_2 we essentially replace a node and its direct descendants in the tree of G_1 by a corresponding tree structure and thus the equivalence of the two SDT's follows since G_2 left—or right—covers G_1, and G_2 and G_1 satisfy the terminating property.

Hence the theorem follows.

---

*p is a permutation mapping.

Section 1.5

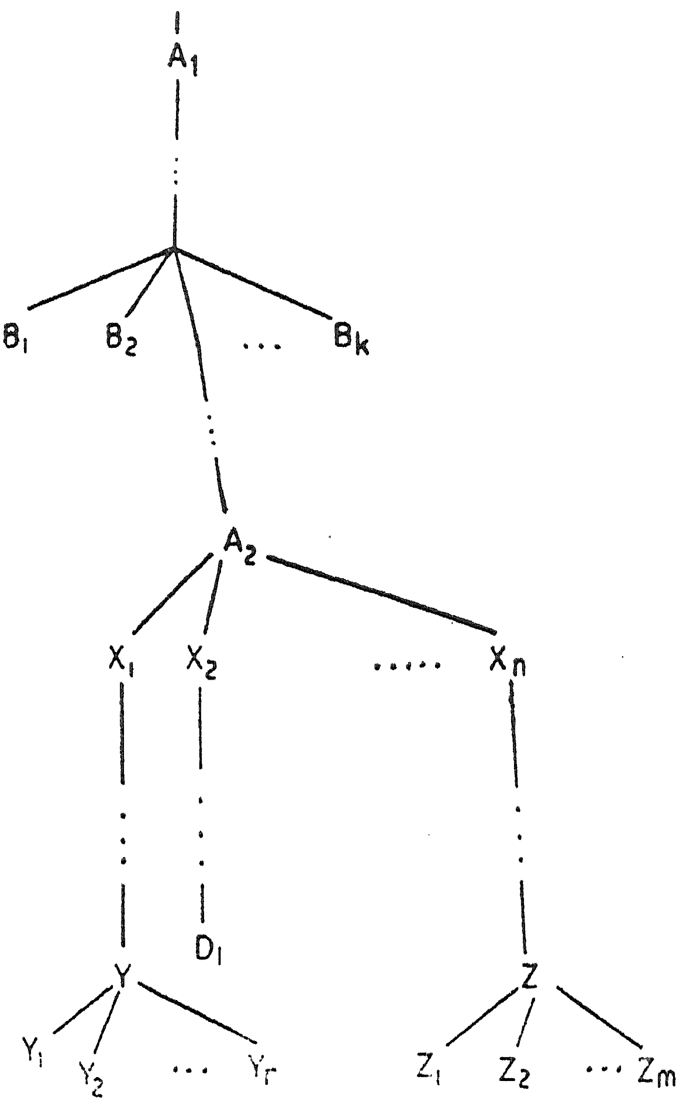
In this section the necessary set of conditions are developed by refining conditions (i) and (ii) of theorem 1.1.

Claim 1: Condition (i) of theorem 1.1 can be replaced by the condition that each member of M_i, $1 \leqslant i \leqslant m$ satisfies.

(a) The condition (i) specified in theorem 1.1, or

(b) The condition that each member of M_i, $1 \leqslant i \leqslant m$ consists of one MLP rule [say $A \to a$] for which $|h_2(a)| \geqslant 2$ and many MLP rules for which $|h_2(a_j)| \leqslant 1$.

Proof: A typical tree structure corresponding to an element satisfying the above condition will have the form shown below:



Now consider the tree structure after pruning all the leaves whose image under h_2 is λ. After this operation, the tree structure will take the form shown below:

This tree structure has the same form as discussed in theorem 1.1. Hence the claim follows.

Note: Without any ambiguity, we will be in a position to associate the translation with the node which has descendants more than or equal to two after taking the image of the leaves under the mapping h_2. [i.e., $|h_2(\alpha)| \geqslant 2$]; note that an element of M_i does not consist of more than one element of the form π_i for which

$$h(\pi_i) \neq \lambda.$$

Claim 2: Condition (i) can be replaced by the condition that for each element in M_i, all the distinct translations that can be effected on a corresponding element of G_1 can be obtained by the corresponding translations on the elements of M_i.

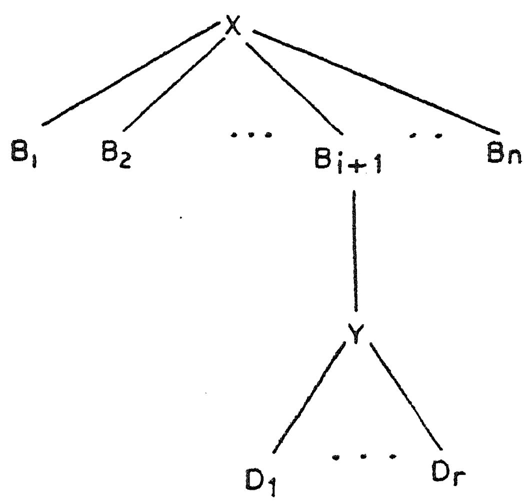
Consider a parse structure corresponding to an element in M_i. Let the frontier of the tree be represented by $X_1 X_2, \ldots, X_n$. Let $h_2(X_1 X_2, \ldots, X_n) = X_i, \ldots, X_{i'}$. Let the corresponding element in G_1 be α [i.e., the right side of the rule h (i)]. In that case if $\alpha = x_1 D_1 x_2, \ldots, D_r x_{r+1}$ then from lemma 1.1 we have

$$h_1(\alpha) = D_1, \ldots, D_r = h_2(X_1 X_2, \ldots, X_n) = X_i, \ldots, X_{i'}$$

and $$i' - i + 1 = r.$$

Translation on $D_1, \ldots, D_r$ corresponds to the permutations of $D_1, \ldots, D_r$. What claim 2 says is that all *distinct* permutations of $D_1, \ldots, D_r$ can be obtained for $X_i, \ldots, X_{i'}$. The proof of claim 2 follows from theorem 1.1. However, the condition is illustrated below:

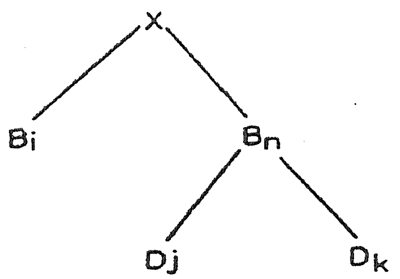
Let $h(i) = j$ and $j : D \rightarrow \alpha$ in G_1. Let $h_1(\alpha) = A_1 A_2 A_1$. Let the corresponding element in M_i have the parse structure shown below:



Now from lemma 1.1 it follows that,

$$h_2(B_1 B_2 \ldots B_i D_1 \ldots D_r \ldots B_{i+2} \ldots B_n)$$

consists of only three elements, say $B_i\ D_j\ D_k$. Then, it will have the structure



Now $A_1A_2A_1$ and $B_i\ D_j\ D_k$ are in correspondence i.e., the first A_1 corresponds to B_i, A_2 corresponds to D_j and second A_1 corresponds to D_k. The possible translations that can be associated with $A_1A_2A_1$ are the permutations of $A_1A_2A_1$, i.e., $A_1A_2A_1$, $A_1A_1A_2$, $A_2A_1A_1$ and that is all: these can be achieved by the set of translations

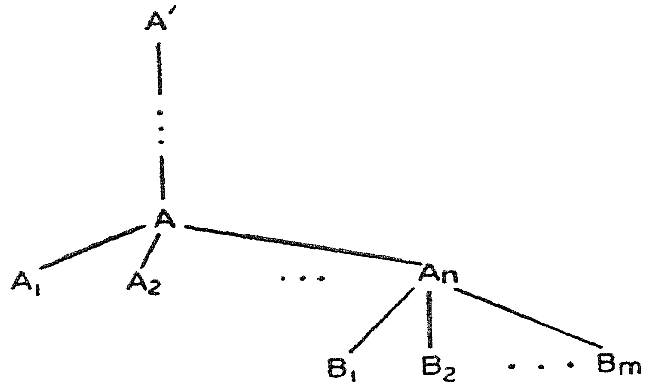
(a) $X \to B_iB_n,\ B_iB_n$ and $B_n \to D_jD_k,\ D_jD_k$,

(b) $X \to B_iB_n.\ B_iB_n$ and $B_n \to D_jD_k,\ D_kD_j$ and

(c) $X \to B_iB_n,\ B_nB_i$ and $B_n \to D_jD_k,\ D_jD_k$, respectively.

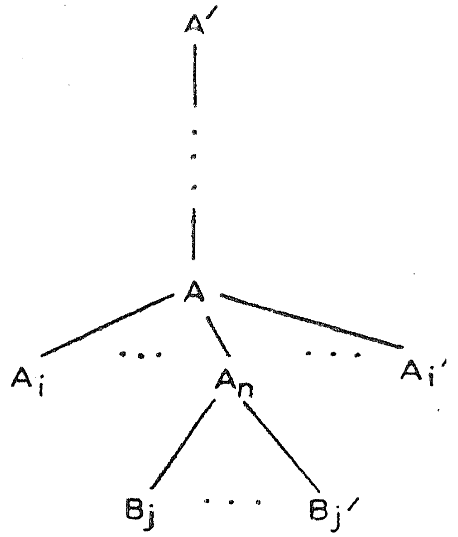
Claim 3: The condition specified by claim (2) is also necessary.

Proof: Now let an element of M_i has the parse consisting of more than or equal to two MLPs for which the length of their images under h_2 is greater than or equal to 2. Then a typical structure of such an element has the form shown below:



Let $h_2\ (A_1A_2, \ldots A_{n-1}B_1B_2, \ldots, B_m)$ match with a string α over $(V_N \cup V_T)^*$ which is a right side of a rule in G_1 corresponding to $M_{i'}$. Let $\alpha = x_1D_1x_2, \ldots, D_rx_{r+1}$.

Now $|h_2(A_1A_2, \ldots, A_{n-1}B_1B_2, \ldots, B_m)| \geqslant 3$, and let it be equal to $D_1, \ldots, D_{r'}$. In other words, the tree structure after obtaining the Hh_2 image of the eaves will have the form shown below where $i'-i \geqslant 1$ and $j'-j \geqslant 1$.



Now from remark 1.1 it follows that if all D_i's are distinct, a permutation exists, i.e., $D_{p(1)}, \ldots, D_{p(r)}$ for which corresponding elements in G_2 will be of the form,

$$A_K \ldots A_K{}' B_p \ldots A_r{}' B_q \ldots B_q{}'$$

This type of translation cannot definitely be obtained in G_2 (cf. remark 1.1).

Hence the necessity of the above condition follows.

Claim 4: Condition (ii) of theorem 1.1 can be replaced by the following condition:

M_i, $1 \leqslant i \leqslant j$ satisfy the condition of claim 3 and $M_i \cap M_j$, $i \neq j$, consist of MLP rules for which the length of the homomorphic image (i.e., h_2) is less than or equal to 1.

Proof: If $M_i \cap M_j$ consist of MLP rules for which $|h_2(\alpha)| \geqslant 2$, then it should be obvious from the argument of theorem 1.1, that this gives rise to elements of the form $A \rightarrow \alpha, \gamma'$ and $A \rightarrow \alpha, \gamma$ where $\gamma \neq \gamma'$ with the SDTS with underlying grammar G_2; in that case the SDTS defined will be ambiguous. This establishes the necessity.

The sufficiency of condition of claim 4 is established in the following:

Now, let $M_i \cap M_j$ consist of MLP for which $|h_2(\alpha)| = 1$ and $|\alpha| \geqslant 1$

In this case it follows that the effective translation will be associated with the MLP rule for which $|h_2(\alpha)| \geqslant 2$ and with the MLP rules (say $B \rightarrow \beta$), for which $|h_2(\beta)| \leqslant 1$, we associate a translation of $h_2(\beta)$ itself. Thus no contradiction results; the case $|h_2(\beta)| \leqslant 1$ and $|\beta| = 1$ has already been treated in theorem 1.1. Thus the sufficiency of the condition is established.

The above case analysis leads us to the following theorem.

Theorem 1.2: The conditions

(A) M_i, $1 \leqslant i \leqslant m$, have elements consisting of at most one MLP rule *or* consists of one MLP rule for which $|h_2(\alpha)| \geqslant 2$ and many MLP rules for which $|h_2(\alpha)| \leqslant 1$ *or* consists of MLP rules which can effect all the distinct translations that can be effected on the rule in G_1 corresponding to the element considered in M_i,

(B) $(M_i \cap M_j)$ for $i \neq j$ consists only rules for which $|h_2(\alpha)| \leqslant 1$, are both necessary and sufficient so that the SDTS with underlying grammar G_1 is a subset of the SDTS with the underlying grammar G_2, whenever G_2 left—or right—covers G_1, and G_2 and G_1 satisfy the terminating property.

Example 1.3

G_3	G_4
1: $S \rightarrow 0A1$	1: $S' \rightarrow 01A'1,\ 01A1$
2: $A \rightarrow 1A1$	2: $A' \rightarrow 1A'1,\ 1A1$
3: $A \rightarrow 11$	3: $A' \rightarrow 1,\ 1$

Here G_4 left-covers G_3 (also G_1 left-covers G_2) under the homomorphism $h(1) = 1$, $h(2)=2$, $h(3)=3$. Also, G_4 and G_3 satisfy the terminating property (definition 1.5). It can be observed from the strings associated with G_4, that theorem 1.1 holds in this case (The SDTS with G_3 is a subset of the SDTS with G_2).

The grammars G_1 and G_2 of example 1.1 also satisfy theorem 1.1 Example 1.3 is given to illustrate that the strings associated with the rules of G_4 do not exactly match with the right sides of G_3. In the next section we consider the practicability of the conditions derived in this section.

Section 1.6

Before describing the practicability of the condition we comment on the terminating property mentioned in definition 1.5.

It should be noted that many of the normal form of grammars that cover their original grammars satisfy the terminating property; e.g., grammars in two canonical form, invertible context-free grammars (cf. Gray and Harrison [4]); note that a grammar need not be covered by any grammar in Greibach normal form

In most formal treatments of parsing, the parser must enumerate all the nodes of the parse tree. In programming practice, certain nodes of the parse tree have no semantic significance and do not need to be present in a similar grammar. For example, consider the generation tree shown in figure 1, which occurs in EULER given in Wirth and Weber [6]; see also Gray and Harrison [4].

The chain expr - → lamda is typical of what happens in grammars for programming languages. Chains exist to enforce precedence among operators and to collect several categories of syntactic types.

Chain productions rarely have semantic significance. In our running example, only the following productions have nontrivial semantics.

expr — → var ← expr —

var — → lambda

primary → var

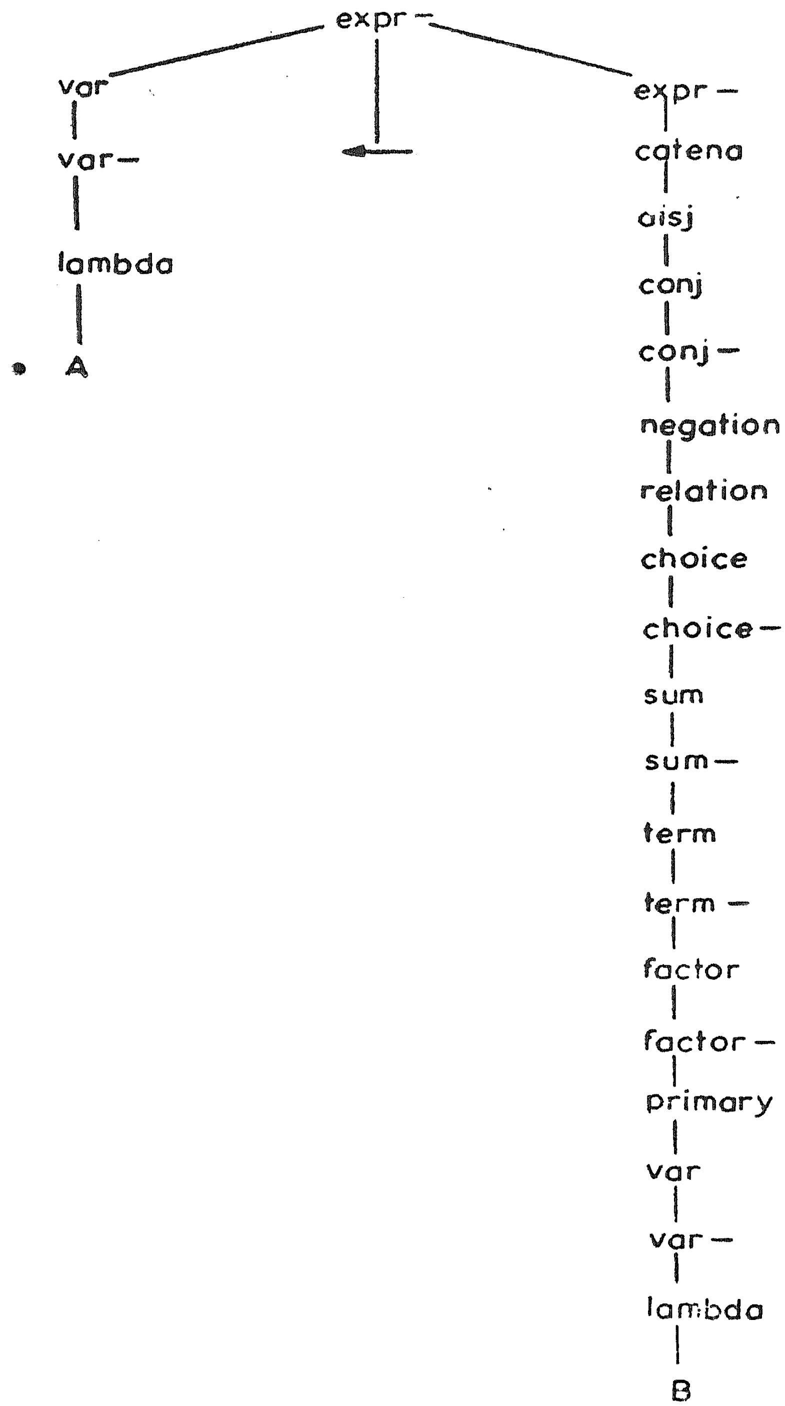


Figure 1.

lambda -> A

lambda -> B

For the purposes of code generation the tree shown below is as satisfactory as the tree shown earlier.

In view of the above facts, the conditions obtained seem to be quite interesting.

Section 1.7

In this section a general algorithm is described that yields the SDTS (semantically unambiguous) with underlying grammar G_2 equivalent to the given SDTS (semanti-

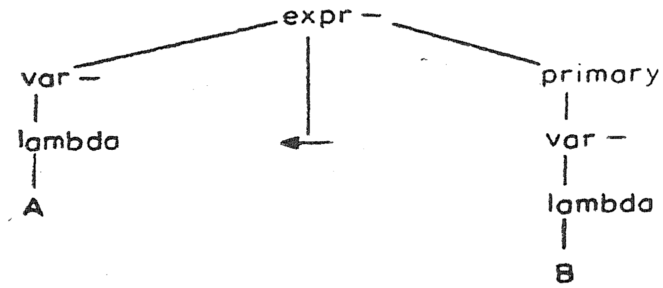


Figure 2.

cally unambiguous) with underlying grammar G_1 (Note that G_2 left—or right—covers G_1, and G_2 and G_1 satisfy the terminating property). Thus it is not necessary that the conditions stipulated in theorems 1.1. and 1.2 should be satisfied by G_2 and G_1 if the algorithm is to be applicable.

Algorithm:

Input: Grammars G_2 and G_1 (G_2 left—or right—covers G_1) and SDTS with underlying grammar G_1.

Output: The SDTS with underlying grammar G_2 equivalent to the given SDTS with underlying grammar G_1, if it exists; otherwise ' NO '.

Method:

Step 1(a): Find the sets M_i, $1 \leqslant i \leqslant m$, where m indicates the number of production rules in G_1.
Set $i=1$; let n_i indicate the number of elements in the set M_i i.e, $h(i_r)=i$, $1 \leqslant i_r \leqslant n_i$.

Step 1(b): Set $n=1$.

Step 2: For the nth element in M_i where $1 \leqslant n \leqslant n_i$, construct the derivation tree. The frontier of this tree is expressed in terms of the bracket language of the grammar G_2 (*cf.* Chartres and Florentin [2]). We illustrate this through an example.

$$\text{e.g.,} \quad G: \begin{array}{lcl} S & \rightarrow & AB \\ A & \rightarrow & a \\ A & \rightarrow & S \\ B & \rightarrow & b \\ B & \rightarrow & SS \end{array}$$

To obtain the bracket language the grammar is augmented as follows: For each nonterminal A which has, say, r different productions $A \rightarrow \phi_k$, $k=1, 2, \ldots, r$, introduce left and right brackets $A(_k$ and $A)_k$, $k=1, 2, \ldots, n$. Let L and R denote the sets of left and right brackets, respectively, so introduced. The kth production $A \rightarrow \phi$ in G for A, now carries the subscript k. It actually represents $A \rightarrow A(_k \phi A)_k$.

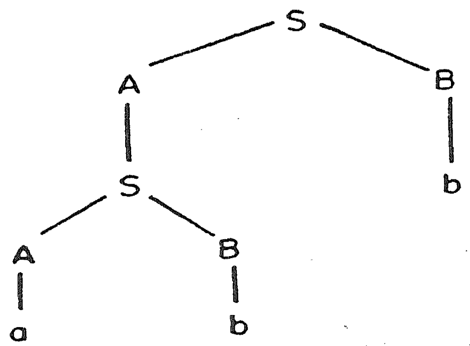
The grammar G is converted into $G' = (V_N, V_T, S, L, R, P')$, where

$$L = \{S_1{}^{(}, A_1{}^{(}, A_2{}^{(}, B_1{}^{(}, B_2{}^{(}\}$$

$$R = \{S_1{}^{)}, A_1{}^{)}, A_2{}^{)}, B_1{}^{)}, B_2{}^{)}\}$$

The sentence abb belonging to $L(G)$ having the structure shown corresponds to the sentence in $L(G')$,

$$S_1{}^{(}\ A_2{}^{(}\ S_1{}^{(}\ A_1{}^{(}\ aA_1{}^{)}\ B_1{}^{(}\ bB_1{}^{)}\ S_1{}^{)}\ A_2{}^{)}\ B_1{}^{(}\ bB_1{}^{)}\ S_1{}^{)}$$



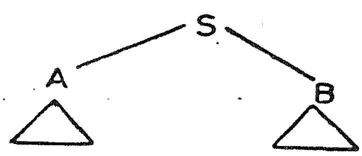
Step 3: Let $h(i_k) = i$, $i \neq \lambda$, $i_k \leqslant n_i$.

Let the frontier of the derivation tree of an element of M_i be of the form $X^{(}, \ldots, X^{)}$. Let the next immediate leftmost symbol belonging to L be $Y_1{}^{(}$. Find the corresponding (leftmost) element belonging to R viz., $Y_1{}^{)}$. Consider the symbols from $Y_1{}^{(}$ to $Y_1{}^{)}$ as forming a single group. Next find out whether there is any symbol belonging to L beyond $Y_1{}^{)}$ (i.e., beyond the group obtained earlier). Let it be $Z^{(}$. Then find the corresponding $Z^{)}$. The sequence of symbols from $Z^{(}$ to $Z^{)}$ forms the next group. In this way, the grouping of sequences of symbols is done till $X^{)}$ is reached. The number of such groups obtained represents the direct descendants of X, in the same order (each descendant is represented by the first symbol of each group belonging to L). The groupings obtained, form the first level of partition of the concerned frontier of the derivation tree.

e.g., $S^{(}A^{(}\ X_1\ X_2\ A^{)}\ B^{(}X_3\ X_4\ B^{)}\ S^{)}$ is grouped as

$$S^{(}[A^{(}X_1\ X_2\ A^{)}]\ [B^{(}X_3X_4B^{)}]\ S^{)}$$

The tree corresponding to this is shown below:



Thus there are two direct descendants from node S. Omitting the symbols belonging to $L \cup R$, we can consider the string $X_1X_2X_3X_4$ to be partitioned to two groups consisting of X_1X_2 and X_3X_4 respectively.

Now, in the same way partition of each of the groups is obtained.

Note 2: A terminating rule expressed in the bracket language will have the form $A^{(} b_1, \ldots, b_n A^{)}$; hence no such grouping is possible.

Step 4: Since the homomorphic image of the frontier of the derivation tree is in correspondence with the homomorphic image of the right hand side of the ith rule in G_1, the right-hand side of the ith rule can be partitioned in an identical way ignoring the terminal sequence.

E.g., Let $X_1X_2X_3X_4$ be the frontier of the derivation tree grouped as $\overline{X_1X_2}\,\overline{X_3X_4}$. Let the right-hand side of the ith rule in G_1 be $\alpha = Y_1Y_2Y_3Y_4$. The corresponding grouping in this case is $\overline{Y_1Y_2}\,\overline{Y_3Y_4}$.

Let the translation associated with this be given by $i : A \to \alpha, \beta$ (belonging to G_1). Now group the symbols of β corresponding to the groups of α, level by level using (h_1). This is illustrated through an example.

E.g., Let $h_1(\alpha) = \overline{X_1\,\overline{X_2\,X_3}\,X_4}\;\overline{X_5\,X_6}$

$$h_1(\beta) = X_6\,X_5\,X_3\,X_2\,X_1\,X_4$$

α consists of two level groupings.

First level of groupings of $h_1(\alpha)$: $\overline{X_1\,X_2\,X_3\,X_4}\;\overline{X_5\,X_6}$

Corresponding groupings of $h_1(\beta)$: $\overline{X_6\,X_5}\;\overline{X_3\,X_2\,X_1\,X_4}$

Second level of groupings of $h_1(\alpha)$: $\overline{X_1\,\overline{X_2\,X_3}\,X_4}\;\overline{X_5\,X_6}$

Corresponding groupings of $h_1(\beta)$: $\overline{X_6\,X_5}\;\overline{\overline{X_3\,X_2}\;X_1\,X_4}$

If such a partition of β is not possible at any of the grouping of α, go to step 7; otherwise the translation is associated with the rules of G_2, corresponding to the translation associated with the ith rule of G_1, in the following way:

Let there be r first level groups in α. Therefore there will be r first level groups in the frontier of the corresponding derivation tree. Let the groupings be represented by the first element of each group belonging to L viz., the grouping of α is represented by $w_1{}^{(}\,A_1\,w_2\,A_2{}^{(}, \ldots, A_r{}^{(}\,w_{r+1}$, where $A_i{}^{(}$ represent the groupings $1 \leqslant i \leqslant r$ and the frontier is denoted by

$$X_k{}^{(}\,w_1\,A_1{}^{(}\,\gamma_1\,A_1{}^{)}\,w_2\,A_2{}^{(}\,\gamma_2\,A_2{}^{)}, \ldots, A_r{}^{(}\,\gamma_r\,A_r{}^{)}\,w_{r+1}\,X_k{}^{)}$$

where $X_k \in V_{N_1}$ and is the root of this tree, γ_i, $1 \leqslant i \leqslant r$ belongs to $(V_{N_1} \cup V_T)^*$. Now, by the definition of translation, γ can be grouped* as,

---

*p is a permutation mapping.

$\bar{w}_1 \bar{A}^{(}_{p(1)} \bar{w}_2 \bar{A}^{(}_{p(2)}, \ldots, \bar{A}^{(}_{p(r)} \bar{w}_{r+1}$, where $\bar{w}_i$, $1 \leqslant i \leqslant r+1$ belongs to $(V_{N_1} \cup V_T)^*$ and the groups $\bar{A}_{p(i)}$, $1 \leqslant i \leqslant r$, consists of sequences over $(V_{N_1} \cup V_T)^*$ [since the lengths of symbols in $L \cup R$ are considered as 0, we write $(V_{N_1} \cup V_T)^*$ instead of $(V_{N_1} \cup V_T \cup L \cup R)^*$], and consists of the same nonterminals that occur in the group represented by some $A_i^{(}$, $1 \leqslant i \leqslant r$. Construct $A'^{(}_{p(i)}$, $1 \leqslant i \leqslant r$, corresponding to $\bar{A}^{(}_{p(i)}$, $1 \leqslant i \leqslant r$ so that $A'^{(}_{p(i)}$ consists of sequences of $(V_{N_2} \cup V_T)^*$ after renaming of the nonterminals by the corresponding mapping used for the elements M_i. Now associate the translation,

$\bar{w}_1 A'^{(}_{p(1)} \ldots A'^{(}_{p(r)} \bar{w}_{r+1}$ with the node X_k (thus with the associated rule i_k). Repeat this for each of the groups $A_i^{(}$, $1 \leqslant i \leqslant r$.

Step 5: If the given SDTS with underlying grammar G_1 has to be covered by a semantically unambiguous SDTS with underlying grammar G_2, then go to step 6(a); otherwise go to step 6(b).

Step 6(*a*): Find whether the translations associated with the rules are ambiguous (i.e., there are translations of the form $A \to \alpha, \beta$; $A \to \alpha, \gamma$); if so, go to step 7; otherwise go to step 6(b).

Step 6(*b*): $n = n+1$; if $n \leqslant n_i$, go to step 2; otherwise go to step 8.

Step 7: The given SDTS with underlying grammar G_1 cannot be covered by any SDTS with underlying grammar G_2; halt.

Step 8: $i = i+1$; if $i \leqslant m$, go to step 1(b); otherwise the SDTS obtained with G_2 as underlying grammar is the one required, i.e., the obtained SDTS with G_2 covers the given SDTS with underlying grammar G_1; stop.

Acknowledgement

The author is thankful to the referee for his suggestions for improving the clarity of the paper.

References

[1] Aho A V and Ullman J D (1972) Theory of *Parsing, Translation and Compiling, Vol I: Parsing*, Prentice-Hall, Englewood Cliffs, N.J.
[2] Chartres B A and Florentin J J (1968) *J. ACM* 15 pp 447-464
[3] Ginsburg S and Harrison M A (1967) *J. Computer and System Sciences* pp 1-23
[4] Gray J N and Harrison M A (1972) *J. ACM* 19 pp 675-698
[5] Lewis II P M and Stearns R E (1968) *J. ACM* 15 pp 465-588
[6] Wirth N and Weber H (1966) *Comm. ACM* 11 pp 13-23