STICHTING
MATHEMATISCH CENTRUM
2e BOERHAAVESTRAAT 49
AMSTERDAM

REKENAFDELING

MR 86

# TWO ALGORITHMS CONCERNING CONTEXT FREE GRAMMARS

J.W. de Bakker

March 1967

## 1. Introduction

In this note we consider two problems concerning context free grammars.
The first problem is the following:
Let $G = (\mathcal{V}, \Sigma, \mathcal{P}, \sigma)$ be a context free grammar (for definitions see section 2), let $A \in \mathcal{V} - \Sigma$, and let $n$ be a non negative integer.
Let $L(A, n)$ ($L(A, \geq n)$, $L(A, \leq n)$) be the subset of $L(G)$ consisting of all those words of $L(G)$, which have a derivation tree that contains the variable A precisely n (at least n, at most n) times. Equivalently, $L(A, n)$ ($L(A, \geq n)$, $L(A, \leq n)$) is the set of those words of $L(G)$ which have a derivation using a production rule with A as its left hand side precisely n (at least n, at most n) times.
The problem is whether $L(A, n)$, $L(A, \geq n)$ and $L(A, \leq n)$ are context free languages. We prove in section 3 that this is indeed the case, by giving an algorithm for deriving context free grammars for these three sets from the given grammar G.
Section 4 is devoted to the second problem:
Let $G = (\mathcal{V}, \Sigma, \mathcal{P}, \sigma)$ be a context free grammar and let $A, B \in \mathcal{V} - \Sigma$.
Find an algorithm for determining whether the subset of those words of $L(G)$, each derivation tree of which contains A, is equal to the set of words from $L(G)$, whose derivation trees all contain B.
In the case that G is unambiguous, this means that this algorithm determines whether, $L(A, \geq 1)$ equals $(B, \geq 1)$.

## 2. Definitions

In this section we follow the definitions of Ginsburg [1].

An alphabet is a finite non empty set, the elements of which are called symbols. If A is an alphabet, $A^*$ is the set of all finite sequences of elements of A, including the empty sequence.

A context free grammar G is a four-tuple $(\mathcal{V}, \Sigma, \mathcal{P}, \sigma)$, where
a. $\mathcal{V}$ is an alphabet,
b. $\Sigma \subset \mathcal{V}$ is an alphabet,
c. $\mathcal{P}$ is a finite set of ordered pairs (u, v), with $u \in \mathcal{V} - \Sigma$, $v \in \mathcal{V}^*$,
d. $\sigma \in \mathcal{V} - \Sigma$.
The elements of $\mathcal{V} - \Sigma$ are called variables and the elements of $\Sigma$ are called terminals. Elements (u, v) of $\mathcal{P}$ are called productions and are usually written $u \rightarrow v$.

Let $w, y \in \mathcal{V}^*$. We write $w \underset{G}{\Rightarrow} y$ (or $w \Rightarrow y$ if G is understood), if there exist $z_1, z_2, u, v \in \mathcal{V}^*$ such that $w = z_1uz_2$, $y = z_1vz_2$ and $u \rightarrow v$ is in $\mathcal{P}$.
We write $w \overset{*}{\underset{G}{\Rightarrow}} y$ (or $w \overset{*}{\Rightarrow} y$ if G is understood), if either $w = y$, or there exist $w_0 = w, w_1, w_2, \ldots, w_r = y$, such that $w_i \underset{G}{\Rightarrow} w_{i+1}$ for each i.
The sequence $w_0, w_1, \ldots, w_r$ is called a derivation and is denoted by $w_0 \Rightarrow w_1 \Rightarrow \ldots \Rightarrow w_r$.
If $G = (\mathcal{V}, \Sigma, \mathcal{P}, \sigma)$ is a context free grammar, then the subset of $\Sigma^*$, $L(G) = \{w \in \Sigma^* \mid \sigma \overset{*}{\Rightarrow} w\}$ is called a context free language.

We impose the following restrictions on the grammars considered in this note:
a. For each $A \in \mathcal{V} - \Sigma$, there exist $u, v \in \mathcal{V}^*$ such that $\sigma \overset{*}{\Rightarrow} uAv$.
b. For each $A \in \mathcal{V} - \Sigma$, there exists $w \in \Sigma^*$ such that $A \overset{*}{\Rightarrow} w$.
These two restrictions ensure that G contains no "superfluous" variables. Since from each grammar containing superfluous variables a grammar can be constructed without superfluous variables, but which generates the same language [1], this is an inessential restriction, imposed only for convenience.

With each derivation $A \overset{*}{\Rightarrow} w$ a derivation tree can be associated in the usual way.
A derivation tree with root A is denoted by $\tau(A)$.
A node $\nu_1$ is called a direct extension of a node $\nu_2$ if there exists a directed line from $\nu_2$ to $\nu_1$.
A node $\nu_1$ is called an extension of a node $\nu_2$ if it is either a direct extension of $\nu_2$ or an extension of a direct extension of $\nu_2$.
A node $\nu_1$ is called predecessor of a node $\nu_2$ if $\nu_2$ is a direct extension of $\nu_1$.
A derivation tree is called recursive, if it contains a node which has the same node name as one of its extensions. Otherwise, it is called recursion free.

In this paper we consider only derivation trees with the property that each variable occurring therein has at least one direct extension.

## 3. Algorithm 1

**Theorem 1**

Let $G = (\mathcal{V}, \Sigma, \mathcal{P}, \sigma)$ be a context free grammar, let $A \in \mathcal{V}$, and let $n$ be an integer $\geq 0$. Define

$L(A, n) = \{w \in L(G) \mid \exists$ a derivation tree of $w$ which contains $A$ precisely $n$ times$\}$,

$L(A, \geq n) = \bigcup_{i \geq n} L(A, i)$,

$L(A, \leq n) = \bigcup_{i=0}^{n} L(A, i)$.

Then $L(A, n)$, $L(A, \geq n)$ and $L(A, \leq n)$ are context free languages.

**Proof**

1.1 First we construct a context free grammar for $L(A, n)$. The basic idea in this construction is the introduction of "indexed" variables, e.g. $P^{(j)}$, $(0 \leq j \leq n)$, such that each derivation tree with $P^{(j)}$ as its root contains the variable $A$ precisely $j$ times.

We define the grammar $G' = (\mathcal{V}', \Sigma', \mathcal{P}', \sigma')$ as follows:

a. $\Sigma' = \Sigma$,

b. $\mathcal{V}' = \Sigma \cup \bigcup_{P \in \mathcal{V}-\Sigma} \bigcup_{0 \leq i \leq n} \{P^{(i)}\}$, i.e. each variable $P \in \mathcal{V}-\Sigma$ leads to $n+1$ variables $P^{(0)}, P^{(1)}, \ldots, P^{(n)} \in \mathcal{V}'$,

c. $\sigma' = \sigma^{(n)}$,

d. $\mathcal{P}' = \bigcup_{P \to \phi \in \mathcal{P}} \bigcup_{0 \leq i \leq n} \{P \to \phi\}^{(i)}$, where the sets $\{P \to \phi\}^{(i)}$ are defined as follows:

α. $P \neq A$.

If $\phi$ is a terminal sequence, i.e. $\phi \in \Sigma^*$, then $\{P \to \phi\}^{(0)} = \{P^{(0)} \to \phi\}$, and $\{P \to \phi\}^{(i)} = \emptyset$ 1), for $i > 0$. If $\phi$ is not a terminal sequence, then there exist $x_1, x_2, \ldots, x_{m+1} \in \Sigma^*$, $Q_1, Q_2, \ldots, Q_m \in \mathcal{V} - \Sigma$ $(m \geq 1)$ such that $\phi = x_1Q_1x_2Q_2 \ldots x_mQ_mx_{m+1}$. Then we define for each $i \geq 0$:

1) $\emptyset$ denotes the empty set.

$$\{P \to \phi\}^{(i)} = \bigcup_{\substack{i_1+i_2+\ldots+i_m=i \\ i_1,i_2,\ldots,i_m\geq 0}} \{P^{(i)} \to x_1Q_1^{(i_1)} x_2Q_2^{(i_2)} \ldots x_mQ_m^{(i_m)} x_{m+1}\}.$$

β. $P = A$.

If $\phi$ is a terminal sequence, then $\{A \to \phi\}^{(1)} = \{A^{(1)} \to \phi\}$, and $\{A \to \phi\}^{(i)} = \emptyset$, for $i \neq 1$. Otherwise, let $\phi = y_1R_1y_2R_2 \ldots y_pR_py_{p+1}$, with $y_1, y_2, \ldots, y_{p+1} \in \Sigma^*$, and $R_1, R_2, \ldots, R_p \in \mathcal{V}-\Sigma$. Then $\{A \to \phi\}^{(0)} = \emptyset$ and for $i > 0$:

$$\{A\to\phi\}^{(i)} = \bigcup_{\substack{i_1+i_2+\ldots+i_p=i-1 \\ i_1,i_2,\ldots,i_p\geq 0}} \{A^{(i)} \to y_1R^{(i_1)} y_2R^{(i_2)} \ldots y_pR^{(i_p)} y_{p+1}\}.$$

1.2 Example

Let G be $(\{\sigma, P, N, a, +, (,), 0, 1\}, \{a, +, (,), 0, 1\}, \{\sigma \to \sigma + P, \sigma \to P, P \to N, P \to (\sigma), P \to a, N \to 0, N \to 1, N \to 0N, N \to 1N\}, \sigma)$.

G is a grammar for a set of simple arithmetic expressions. A grammar G' for the subset of this set, consisting of all expressions which contain only two digits (i.e. L(N, 2)) is the following:

$G' = (\mathcal{V}', \Sigma', \mathcal{P}', \sigma')$, where

$\mathcal{V}' = \{\sigma^{(2)}, \sigma^{(1)}, \sigma^{(0)}, P^{(2)}, P^{(1)}, P^{(0)}, N^{(2)}, N^{(1)}, N^{(0)}, a, +, (,), 0, 1\}$,

$\Sigma' = \{a, +, (,), 0, 1\}$,

$$\mathcal{P}' = \left\{\begin{array}{llll}
\sigma^{(2)} \to \sigma^{(2)} + P^{(0)}, & \sigma^{(2)} \to P^{(2)}, & P^{(2)} \to (\sigma^{(2)}), & N^{(2)} \to 0N^{(1)} \\
\sigma^{(2)} \to \sigma^{(1)} + P^{(1)}, & \sigma^{(1)} \to P^{(1)}, & P^{(1)} \to (\sigma^{(1)}), & N^{(2)} \to 1N^{(1)}, \\
\sigma^{(2)} \to \sigma^{(0)} + P^{(2)}, & \sigma^{(0)} \to P^{(0)}, & P^{(0)} \to (\sigma^{(0)}), & N^{(1)} \to 0N^{(0)}, \\
\sigma^{(1)} \to \sigma^{(1)} + P^{(0)}, & P^{(2)} \to N^{(2)}, & P^{(0)} \to a, & N^{(1)} \to 1N^{(0)} \\
\sigma^{(1)} \to \sigma^{(0)} + P^{(1)}, & P^{(1)} \to N^{(1)}, & N^{(1)} \to 0, & \\
\sigma^{(0)} \to \sigma^{(0)} + P^{(0)}, & P^{(0)} \to N^{(0)}, & N^{(1)} \to 1, &
\end{array}\right\}$$

$\sigma' = \sigma^{(2)}$.

Two possible derivations are:

$$\sigma^{(2)} \Rightarrow \sigma^{(2)} + P^{(0)} \Rightarrow \sigma^{(1)} + P^{(1)} + P^{(0)} \Rightarrow P^{(1)} + P^{(1)} + P^{(0)} \Rightarrow$$
$$\Rightarrow N^{(1)} + P^{(1)} + P^{(0)} \Rightarrow 0 + P^{(1)} + P^{(0)} \Rightarrow 0 + N^{(1)} + P^{(0)} \Rightarrow$$
$$\Rightarrow 0 + 1 + P^{(0)} \Rightarrow 0 + 1 + a,$$
$$\sigma^{(2)} \Rightarrow \sigma^{(2)} + P^{(0)} \Rightarrow P^{(2)} + P^{(0)} \Rightarrow N^{(2)} + P^{(0)} \Rightarrow$$
$$\Rightarrow 0N^{(1)} + P^{(0)} \Rightarrow 01 + P^{(0)} \Rightarrow 01 + a.$$

## 1.3 Proof that $L(G') = L(A, n)$

### 1.3.1. First we show that $L(G') \subset L(A, n)$.

Let

(1) $\sigma^{(n)} = w_0 \underset{G'}{\Rightarrow} w_1 \underset{G'}{\Rightarrow} \dots \underset{G'}{\Rightarrow} w_r = w$

be a derivation of $w \in L(G')$.

Clearly, deletion of all the indices from the variables occurring in each $w_i (0 \leq i < r)$, gives a derivation

(2) $\sigma \overset{*}{\underset{G}{\Rightarrow}} w$,

hence $w \in L(G)$. In order to prove that the derivation tree corresponding to (2) contains A precisely n times, we introduce the "level" $\lambda$ for each element of $\mathcal{V}'^{*}$ as follows:

a. $\lambda(a) = 0$, for each $a \in \Sigma'$,

b. $\lambda(P^{(j)}) = j$, for each $P^{(j)} \in \mathcal{V}' - \Sigma'$

c. $\lambda(xy) = \lambda(x) + \lambda(y)$, for $x, y \in \mathcal{V}'^{*}$.

In (1), let $w_{i-1} = x\,P^{(j)}\,y$, $w_i = x\phi y \quad (0 \leq i \leq r)$,

where $x, y \in \mathcal{V}'^{*}$, $P^{(j)} \to \phi \in \mathcal{P}'$.

From the construction of $\mathcal{P}'$ it follows that $\lambda(w_i) = \lambda(w_{i-1})$ if and only if $P \neq A$, and $\lambda(w_i) = \lambda(w_{i-1})-1$ if and only if $P = A$. Since $\lambda(\sigma^{(n)}) = n$, and $\lambda(w) = 0$, we conclude that the derivation (2) uses a production from $\mathcal{P}$ with A as its left hand side precisely n times, which means that A occurs n times in the derivation tree of w.

### 1.3.2 Proof that $L(A, n) \subset L(G')$.

Let $w \in L(G)$ have a derivation tree which contains A precisely n times. The following process is executed:

(*) Each node in this tree which is a variable, say P, is supplied with an index, equal to the number of times A occurs in the subtree $\tau(P)$, containing all extensions of P.

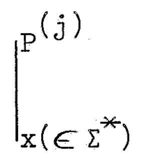
Then the tree which is thus constructed is a derivation tree of w in L(G').

For, consider an arbitrary variable $P^{(j)}$ in this derivation tree.

First suppose $P \neq A$.

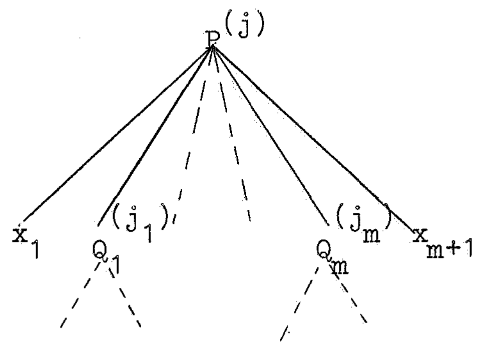
There are two possibilities for the subtree $\tau(P^{(j)})$:

a.



The only direct extension of P is a terminal sequence; hence, $j = 0$. But $P^{(0)} \rightarrow x \in \mathcal{P}'$ by the construction of $\mathcal{P}'$.

b.

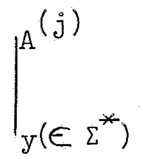


From (*) it follows that $j = j_1 + j_2 + \ldots + j_m$; hence, $P^{(j)} \rightarrow x_1 Q_1^{(j_1)} \ldots x_m Q_m^{(j_m)} x_{m+1} \in \mathcal{P}'$.

Now suppose $P = A$.

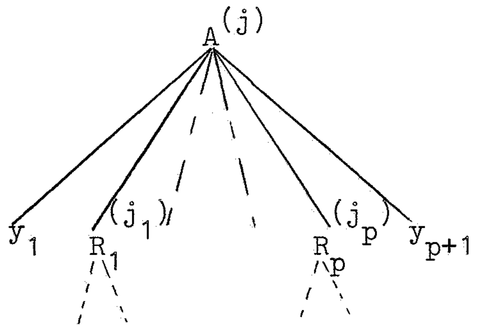
Again there are two cases:

a.



By (*), $j = 1$, and from the construction of $\mathcal{P}'$ we see that $A^{(1)} \rightarrow y \in \mathcal{P}'$.

b



By (*), $j = j_1 + j_2 + \ldots + j_p + 1$; hence, $A^{(j)} \rightarrow y_1 R_1^{(j_1)} \ldots y_p R_p^{(j_p)} y_{p+1} \in \mathcal{P}'$.

This completes the proof that $L(A, n) \subset L(G')$.

2. The grammar G" for $L(A, \geq n)$ is constructed as follows:

   $G'' = (\mathcal{V}'', \Sigma'', \mathcal{P}'', \sigma'')$, where

   $\mathcal{V}'' = \mathcal{V}' \cup \mathcal{V},$

   $\Sigma'' = \Sigma,$

   $\mathcal{P}'' = \mathcal{P}' \cup \mathcal{P} \cup \{A^{(1)} \to A\} \cup \bigcup_{P \in \mathcal{V}-\Sigma} \{P^{(0)} \to P\},$

   $\sigma'' = \sigma^{(n)}.$

   The proof that this grammar generates $L(A, \geq n)$ is very similar to the first proof and is therefore omitted.

3. Since $L(A, \leq n) = \bigcup_{i=0}^{n} L(A, i)$ and since a finite union of context free languages is again a context free language, the proof of theorem 1 is complete.

Remark:

Theorem 1 can easily be generalized to the following

Corollary:

Let $\{A_1, A_2, \ldots, A_m\}$ be a set of variables of a context free grammar G, and let $\{n_1, n_2, \ldots, n_m\}$ be a set of non negative integers. Then: The subset of L(G) consisting of all words which have a derivation tree containing $A_1$ precisely (at least, at most) $n_1$ times, $A_2$ precisely (at least, at most) $n_2$ times, ..., $A_m$ precisely (at least, at most) $n_m$ times, is a context free language. Here each choice of "precisely", "at least", "at most" may be made independently of the others.

## 4. Algorithm 2.

Let A, B be two variables of a context free grammar G. In this section we define an algorithm which determines whether each derivation tree of a word $w \in L(G)$ which contains A, also contains B. This algorithm is given in definition 2. First, a special case is treated in definition 1. There we define an algorithm which determines whether each derivation tree with A as its root contains B.
Although these problems are fairly simple, some care has to be taken in view of recursive use of variables, in order to avoid closed loops in these algorithms. This may be illustrated by the following part of the set of production rules of a context free grammar:

$$\vdots$$
$$A \rightarrow \ldots\ C\ \ldots,$$
$$C \rightarrow \ldots\ D\ \ldots,$$
$$D \rightarrow \ldots\ A\ \ldots,$$
$$\vdots$$

Suppose one used the following scheme: Each derivation tree with A as its root contains B if and only if each production rule with A as its left hand side contains in its right hand side either B, or variable C with the property that each derivation tree with C as its root contains B. Verification of this property for C and next for D would give the result that each derivation tree with A as its root contains B if each derivation tree with A as its root contains B.
In order to avoid such loops it is clearly necessary to remember which variables have already been considered in the course of the above described scheme.
This explains the following definition:

Definition 1

Let $G = (\mathcal{V}, \Sigma, \mathcal{P}, \sigma)$ be a context free grammar, let $A, B \in \mathcal{V}-\Sigma$, and let $\mathcal{W} \subset \mathcal{V}-\Sigma$. The predicate $(A > B, \mathcal{W})$ is defined as follows:
$(A > B, \mathcal{W})$ is true if and only if either

1. $A = B$, or (1)

2. For each rule $A \rightarrow \phi \in \mathcal{P}$ which has A as its left hand side there exist

$\psi_1$, $\psi_2 \in \mathcal{V}^*$, $P \in \mathcal{V} - \Sigma$, such that $\phi = \psi_1 P \psi_2$, and either

2.1. $P \in \mathcal{W}$, or (2)

2.2. $(P > B, \mathcal{W} \cup \{P\})$ (3)

Example: $(A > B, \mathcal{V} - \Sigma)$ is true if and only if either $A = B$, or $\mathcal{P}$ contains no rule $A \rightarrow \phi$, with $\phi$ a terminal sequence.

Theorem 2

Let A, B be two variables of a context free grammar. Then $(A > B, \{A\})$ if and only if each derivation tree with A as its root contains B.

Proof

1. Suppose $(A > B, \{A\})$. We prove that each tree with A as its root contains B.
   First we consider a derivation tree $\tau(A)$ which is recursion free (section 2). Suppose that $B \notin \tau(A)$ 1).
   From the definition of $(A > B, \{A\})$ we conclude that there exists in $\tau(A)$ at least one direct extension of A which is a variable, say P, such that $(P > B, \{A, P\})$; for, (1) does not apply since $B \notin \tau(A)$, and (2) does not apply since $\tau(A)$ is recursion free.
   By the same argument there exists a direct extension Q of P such that $(Q > B, \{A, P, Q\})$ etc.
   However, after a finite number of these steps we reach a variable R, such that $(R > B, \{A, P, Q, \ldots, R\})$, but such that the only extension of R is a terminal sequence, say x.
   This is a contradiction, since the occurence of a rule $R \rightarrow x$ in $\mathcal{P}$ contradicts $(R > B, \{A, P, Q, \ldots, R\})$.

   Next we consider a tree $\tau(A)$ which is recursive. It is easy to see that from such a tree, another derivation tree can be constructed with the same root, which is recursion free and which contains only nodes that occur also in the original tree. We can apply the above argument to this recursion free tree. If it contains B, then so does the originally considered tree $\tau(A)$.

1) $B \in \tau(A)$ means: B occurs as one of the nodes in the tree $\tau(A)$.

2. Suppose that each derivation tree with A as its root contains B.

We prove that then $(A > B, \{A\})$.

Let $\mho - \Sigma = \{P_1, P_2, \ldots, P_n\}$.

Let $E(i)$, $1 \leq i \leq n$, be the following assertion:

$\left\{\begin{array}{l}\text{If } \{P_1, P_2, \ldots, P_i\} \text{ is a subset of } \mho - \Sigma\text{, if } B, Q \in \mho - \Sigma\text{, and if} \\ (Q > B, \{P_1, P_2, \ldots, P_i\}) \text{ is false, then there exists a derivation} \\ \text{tree } \tau(Q) \text{ which does not contain B.}\end{array}\right.$

We shall show that:

a. $E(n)$ holds,

b. $E(i)$ implies $E(i-1)$, for $i = n, n-1, \ldots, 2$.

Assuming a and b, we conclude that $E(1)$ holds, which means that if $(A > B, \{A\})$ is false, then there exists a tree $\tau(A)$, such that $B \notin \tau(A)$.

Proof of $E(n)$:

Suppose that $(Q > B, \{P_1, P_2, \ldots, P_n\})$ is false. From definition 1 we see that there exists at least one production rule $Q \rightarrow \phi$ with $\phi$ a terminal sequence. Hence there exists a derivation tree $\tau(Q)$ which does not contain B.

Proof of "$E(i)$ implies $E(i-1)$".

Suppose that $(Q > B, \{P_1, P_2, \ldots, P_{i-1}\})$ is false, for some subset $\{P_1, P_2, \ldots, P_{i-1}\}$ of $\mho - \Sigma$.

Then, either

a. There exists a rule $Q \rightarrow \phi$, with $\phi$ a terminal sequence, which immediately gives a tree $\tau(Q)$ without B, or

b. There exists a rule

$$Q \rightarrow x_1 R_1 \ x_2 R_2 \ \ldots \ x_m R_m \ x_{m+1} \tag{4}$$

such that for each $R_j$ $(1 \leq j \leq m)$:

$$R_j \notin \{P_1, P_2, \ldots, P_{i-1}\} \tag{5}$$

and

$$(R_j > B, \{P_1, P_2, \ldots, P_{i-1}, R_j\}) \text{ is false.} \tag{6}$$

We can now apply $E(i)$ to (6), which means that there exist derivation trees $\tau(R_j)$ withour B, for each $j (1 \leq j \leq m)$, Together with (4) this yields a derivation tree $\tau(Q)$ which does not contain B.

This completes the proof of theorem 2.

Remark: In the sequel we abbreviate $(A > B, \{A\})$ to $A > B$.

<u>Definition 2</u>

Let A, B be two variables of a context free grammar $G = (\mathcal{V}, \Sigma, \mathcal{P}, \sigma)$ and let $\mathcal{W} \subset \mathcal{V} - \Sigma$. The predicate $(A \rho B, \mathcal{W})$ is defined as follows:

1. $(\sigma \rho B, \mathcal{W}) = \sigma > B$, and for each $A \neq \sigma$:
2. $(A \rho B, \mathcal{W})$ is true if and only if either

   2.1. $A = B$, or $\quad$ (7)

   2.2. For each rule $P \to \phi$, which contains A in its right hand side (i.e. there exist $\psi_1, \psi_2 \in \mathcal{V}^*$, such that $\phi = \psi_1 A \psi_2$), either

   2.2.1. $\phi$ contains a variable C (which may be A itself) for which $C > B$ holds or $\quad$ (8)

   2.2.2. $P \in \mathcal{W}$, or $\quad$ (9)

   2.2.3. $(P \rho B, \mathcal{W} \cup \{P\})$ $\quad$ (10)

<u>Theorem 3</u>

Let A, B be two variables of a context free grammar. Then $(A \rho B, \{A\})$ if and only if each derivation tree $\tau(\sigma)$ which contains A, also contains B.

<u>Proof</u>

1. Suppose $(A \rho B, \{A\})$. We prove that $A \in \tau(\sigma)$ implies $B \in \tau(\sigma)$.
   If $A = \sigma$ then the proof follows from theorem 2.
   If $A \neq \sigma$, we first consider a derivation tree $\tau(\sigma)$ which is recursion free. Suppose $A \in \tau(\sigma)$, $B \notin \tau(\sigma)$. Let P be the predecessor of A in $\tau(\sigma)$. From the definition of the relation $(A \rho B, \{A\})$ it follows that $(P \rho B, \{A, P\})$; for, (7) or (8) does not apply since $B \notin \tau(\sigma)$ and (9) does not apply since $\tau(\sigma)$ is recursion free.
   Again, for the predecessor of P, say Q, we find:
   $(Q \rho B, \{A, P, Q\})$ etc. Finally we conclude that $(\sigma \rho B, \{A, P, Q, \ldots, \sigma\})$, which means that $\sigma > B$; hence, $B \in \tau(\sigma)$, which is a contradiction.
   The general case for recursive trees follows directly.

2. Suppose that $A \in \tau(\sigma)$ implies $B \in \tau(\sigma)$, but that $(A \rho B, \{A\})$ is false. For $A = \sigma$, this is clearly impossible. If $A \neq \sigma$, we conclude from definition 2 that there exists a rule $P \to \phi$, such that:

   a. $A$ occurs in $\phi$,

   b. For no $C$ occuring in $\phi$, $C > B$,

   c. $(P \rho B, \{A, P\})$ does not hold.

   From a and b we see that there exists a tree $\tau(P)$, with $A \in \tau(P)$, $B \notin \tau(P)$.

   From c it follows in the same way that there exists a $Q$, $(\neq A, \neq P)$, and a derivation tree $\tau(Q)$, such that $A \in \tau(Q)$, $B \notin \tau(Q)$ and $(Q \rho B, \{A, P, Q\})$ is false.

   Eventually a tree $\tau(\sigma)$ results, such that $A \in \tau(\sigma)$ but $B \notin \tau(\sigma)$, which is a contradiction.

   This completes the proof of theorem 3.

Corollary

For unambiguous grammars there exists an algorithm which determines whether $L(A, \geq 1) = L(B, \geq 1)$.

Proof

$L(A, \geq 1) = L(B, \geq 1)$ if and only if $(A \rho B, \{A\})$ and $(B \rho A, \{B\})$.

Reference

[1] S. Ginsburg The mathematical theory of context free languages. New York, Mc Graw-Hill, 1966.