MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Project MAC

Machine Structures Group Memo No. 23 March 1966

Parenthesis Grammars

by

Robert McNaughton

In this paper context-free languages that are very unambiguous and very deterministic will be studied. These are, in other words, languages whose terminal strings wear their syntactic structure on their sleeves. They are the languages that have a grammar in which exactly one pair of parentheses are introduced at every application of every rule, giving the terminal strings (and non-terminal strings too for that matter) a highly structured appearance. These parentheses, however, are of just one species, and have no subscripts. Thus the parentheses do not tell which rule introduced them, or even which non-terminal they came from. Ginsburg and Harrison (in an unpublished paper which I have not seen) have studied languages in which subscripts on parentheses go further towards indicating the structure of a terminal string.

The main result of this paper is that the equivalence problem for parenthesis languages (given by their grammars) is solvable. In other words, there is a decision procedure to tell whether two such parenthesis grammars generate the same language. Incidentally this shows that the inclusion problem is also solvable: for to determine whether the language of one grammar is included in the language of the second, one can take the grammar for the union of the two languages (obtainable by an easy and well known technique) and test for equality with the second.

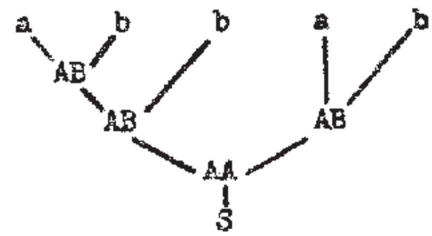
A context-free grammar is as usually defined, except that possibly several initial non-terminals are allowed. (An initial non-terminal is a symbol that may begin a derivation.) The usual technique for ridding a grammar of all but one of its initial non-terminals (usually "S") will not work on parenthesis languages since every new rule that is introduced must have a new set of parentheses.

It is assumed that the reader is familiar with the usual terminology of context-free grammars. Nevertheless, a few words on terminology and notation will be helpful. A string Ω of terminals and non-terminals is <u>derivable</u> from a non-terminal A in a grammar if there is a sequence of strings $\Omega_1, \ldots, \Omega_n$, where Ω_1 is A, Ω_n is Ω and, for every i, Ω_{i+1} follows from Ω_i by one of the rules; if in addition A is an initial non-terminal of the grammar we say simply that Ω is <u>derivable</u> in the grammar.

In the literature, derivation trees have been studied extensively, since in many cases they are more meaningful than a derivation, which is a mere sequence of strings. In a derivation tree a string is not completely rewritten when a rule is applied. Rather the replacement is written over the replaced symbol. Thus an <u>unambiguous language</u> is conveniently defined as one in which every terminal string has at most one derivation tree. For example, a derivation tree for abbab in the grammar

$$
\begin{aligned}
S &\rightarrow AA \\
A &\rightarrow AB \\
A &\rightarrow a \\
B &\rightarrow b
\end{aligned}
$$

is as follows:



Although there is no other derivation tree for this terminal string, there are several derivations that result from the tree, since the order of replacing non-terminals currently appearing in a string of the derivation is arbitrary.

Capital Roman letters are used for non-terminals and often also as variables ranging over non-terminals. Thus it will be convenient to say "for every non-terminal A," etc. Small Roman letters are used for terminal letters; in parenthesis grammars the open parenthesis (and close parenthesis) are also terminal symbols. "Ω" is used as a variable over strings.

The language of a grammar is the set of terminal strings derivable. Two grammars are equivalent if their languages are equal. It is to be noted that, in talking about algorithms, there is no way an infinite language can be given except in the way of a finite representation such as a grammar, or a formula or a finite description of the corresponding push-down store. In this paper confusing phrases, like "given a language" are avoided in favor of more precise phrases, like "given a grammar for a language." Push-down stores are not discussed here. As for formulas, although some context-free languages can be described rather neatly by them, e.g., $1^k 0^n 1^m 0^n 1^k$, many other languages that we wish to discuss do not have conveniently available formulas. In this paper then languages are given by grammars, when algorithms are discussed.

A <u>parenthesis grammar</u> is a context-free grammar all of whose rules are of the form

$$A \longrightarrow (\Omega),$$

where Ω contains no occurrence of (or of). A backwards-deterministic parenthesis grammar is one in which no two rules have the same right side. A (backwards-deterministic) parenthesis language is one that has a (backwards-deterministic) parenthesis grammar.

Note that a backwards-deterministic parenthesis grammar is unambiguous in that every terminal string has at most one derivation tree. What is more, such a grammar has an even stronger property concerning parsing: i.e., given any string, we can pick out certain phrases of limited size (namely, the innermost parenthesized parts) and replace them by a non-terminal without any regard to what occurs to the left or to the right and be confident that we have not made any mistake -- which is the reason for the choice of the phrase "backwards-deterministic." Thus these languages are the context-freest of the context-free languages. At the end of this paper a more general definition is given of "backwards deterministic" applying to languages without parentheses. (The term "backwards-deterministic" and the concept come from some unpublished work by K. Speierman.)

Theorem 1. Every parenthesis grammar has an equivalent backwards-deterministic parenthesis grammar effectively obtainable from it.

Proof: Let G be a parenthesis grammar for a parenthesis language L. Suppose that $A_1, \ldots, A_n$ are the non-terminals of this grammar, one of them having the function of S. Let the stencil of a rule be the right side of that rule with blank spaces in place of the non-terminals. Thus, e.g., the stencil of the rule

$$A_1 \longrightarrow (A_2 bcA_3 A_4)$$

is (bc).

A backwards-deterministic parenthesis grammar G′ is now constructed having 2^n-1 non-terminals, one corresponding to each non-empty set of non-terminals of G. Let these be $B_1, \ldots, B_{2^n-1}$, and let $\sigma(B_1), \ldots, \sigma(B_{2^n-1})$ be the corresponding sets.

$$B_{i_1} \rightarrow \left(B_{i_2}\, bc\, B_{i_3}\, B_{i_4}\right)$$

is a rule of G′ if and only if $\sigma(B_{i_1})$ is precisely the set of non-terminals A_{j_1} of G such that, for some $A_{j_2} \in \sigma(B_{i_2})$, $A_{j_3} \in \sigma(B_{i_3})$ and $A_{j_4} \in \sigma(B_{i_4})$,

$$A_{j_1} \rightarrow \left(A_{j_2}\, bc\, A_{j_3}\, A_{j_4}\right)$$

is a rule of G. And similarly for other stencils. The initial non-terminals of G′ are all those B_i such that there is at least one initial non-terminal of G in $\sigma(B_i)$. Thus the set of stencils of rules of G′ is exactly the set of stencils of rules of G.

That G′ is a backwards-deterministic parenthesis grammar is clear. We must show that its language is that of G. The proofs of Lemmas 1 and 2 below complete the proof of Theorem 1.

<u>Lemma 1.</u> Every derivation D of a terminal string in G can be converted into a derivation D′ of the same terminal string in G′.

Proof: Rewrite the derivation backwards, line by line. The last replacement in D might be an application of the rule

$$A_1 \rightarrow (ab)\ .$$

The rule in G′ to use is the unique rule

$$B_i \rightarrow (ab)$$

with the stencil (ab). This determines the penultimate line of D′, and clearly $A_1 \in \sigma(B_i)$. Now suppose the $(x+1)^{th}$ line of D′ has been determined. Suppose as an inductive hypothesis, that the $(x+1)^{th}$ line of D′

is like the $(x + 1)^{th}$ line of D except that for each occurrence of each non-terminal A_x of G, there is in the corresponding position a non-terminal B_y of G′, where $A_x \in \sigma(B_y)$. (Different occurrences of A_x may have different B_y's.) The x^{th} line of D′ can be determined from the $(x + 1)^{th}$ line of D′ by considering the rule used in going from the x^{th} to the $(x + 1)^{th}$ line in D. Without loss of generality, suppose this rule is

$$A_1 \rightarrow (A_2 \text{ be } A_3 \; A_4) \; .$$

Suppose there is corresponding to $(A_2 \text{ be } A_3 \; A_4)$ in the $(x + 1)^{th}$ line of D the phrase $(B_{i_2} \text{ be } B_{i_3} \; B_{i_4})$ in the $(x + 1)^{th}$ line of D′ as already constructed, where $A_2 \in \sigma(B_{i_2})$, $A_3 \in \sigma(B_{i_3})$ and $A_4 \in \sigma(B_{i_4})$. There is a unique B_{i_1} such that

$$B_{i_1} \rightarrow (B_{i_2} \text{ be } B_{i_3} \; B_{i_4})$$

is a rule of G′, where $A_1 \in \sigma(B_{i_1})$. We thus make the x^{th} line of D′ like the $(x + 1)^{th}$ line of D′ except that B_{i_1} replaces $(B_{i_2} \text{ be } B_{i_3} \; B_{i_4})$ in its occurrence corresponding to the replacement in the $(x + 1)^{th}$ line of D. Clearly the x^{th} line of D′ will have the property that it is a rewrite of the corresponding line of D, with each occurrence of each non-terminal A_x of G rewritten as some B_y of G′ such that $A_x \in \sigma(B_y)$, since it is given that the $(x + 1)^{th}$ line of D′ has this property.

If D′ is constructed backwards in this manner, the first line will be B_y where, if A_x is the first line of D, $A_x \in \sigma(B_y)$. B_y will be initial in G′ since A_x is initial in G. Thus D′ is a derivation of the same string in the grammar G′.

<u>Lemma 2.</u> Every derivation D′ of a terminal string in G′ can be converted into a derivation D of the same terminal string in G.

Proof: This time the derivation is to be rewritten forward. If the first line of D' is the initial non-terminal B_y there exists an A_x such that $A_x \in \sigma(B_y)$ and A_x is an initial non-terminal of G. (If there are several such A_x's it does not matter which one is selected.) As an inductive hypothesis, suppose that the x^{th} line of D has been constructed so that it is like the x^{th} line of D' except for containing in place of each B_y an occurrence of some A_x in the corresponding place, where $A_x \in \sigma(B_y)$. (Different occurrences of B_y may have corresponding to them different A_x's.) Suppose, without loss of generality, that the rule

$$B_{i_1} \longrightarrow \left(B_{i_2}\ bc\ B_{i_3}\ B_{i_4}\right)$$

is the rule by means of which the $(x+1)^{th}$ line of D' is obtained from the x^{th} line of D', and suppose that A_{j_1} occurs in the x^{th} line of D corresponding to that occurrence of B_{i_1} in the x^{th} line of D'. Then $A_{j_1} \in \sigma(B_{i_1})$ and hence there exist non-terminals $A_{j_2} \in \sigma(B_{i_2})$, $A_{j_3} \in \sigma(B_{i_3})$ and $A_{j_4} \in \sigma(B_{i_4})$ such that

$$A_{j_1} \longrightarrow \left(A_{j_2}\ bc\ A_{j_3}\ A_{j_4}\right)$$

is the rule of G that gives us the $(x+1)^{th}$ line of D satisfying the inductive hypothesis. In this manner D is constructed so that it is a derivation of G and its last line (since there are no non-terminals) is exactly the same as the last line of D'. This concludes the proof of Theorem 1.

Where A is a non-terminal of a grammar G let $\Delta_G(A)$ be the set of all strings derivable from A (including those with non-terminals).

<u>Theorem 2.</u> If A and B are distinct non-terminals of a backwards-deterministic parenthesis grammar G then $\Delta_G(A)$ and $\Delta_G(B)$ are disjoint.

Proof. A string in a backwards-deterministic parenthesis grammar has at most one derivation tree. Hence no string could have both A at the bottom of a derivation tree and B at the bottom of another.

A backwards-deterministic parenthesis grammar may have two non-terminals whose combined role can be played by a single non-terminal. If such is the case we shall say that the grammar is not "reduced," in a sense to be defined more precisely below.

A context β is a string with one blank. $\beta[A]$ is the context with the blank filled in with the non-terminal A. Let Σ be a set of non-terminals of a grammar G. Σ is n-distinguishable in G if there is a context β such that, for every $A \in \Sigma$, $\beta[A]$ is derivable in n lines or less in G from an initial non-terminal, but for no non-terminal B of G outside of Σ is $\beta[B]$ derivable in n lines or less. Σ is distinguishable in G if it is n-distinguishable, for some n. Note that, in a parenthesis grammar, the length of a derivation is one more than the number of parenthesis pairs in the string. Thus all derivations of a string of the form $\beta[A]$ must be of the same length.

Theorem 2. If, in a parenthesis grammar, there is a Σ that is (n + 2)-distinguishable but not (n + 1)-distinguishable then there is a Σ_0 that is (n + 1)-distinguishable but not n-distinguishable.

Proof: Let β_{n+2} be a context with n + 1 or fewer parenthesis pairs such that Σ is precisely the set of all non-terminals A such that $\beta_{n+2}[A]$ is derivable in G. The blank in β_{n+2} must be inside a parenthesized phrase, say

$$(A_1 \, bc \ldots A_3) \, .$$

Let β_{n+1} be the context which is like β_{n+2} except for having a single blank in place of this whole parenthesized phrase. Without loss of generality, we can confine our attention to derivations of $\beta_{n+2}[A]$, for any A, in which this parenthesized phrase is the last to appear. Thus the line before will be $\beta_{n+1}[B]$, for some B. Clearly, the number of parenthesis pairs in β_{n+1} is n or less.

Let Σ_0 be the set of all non-terminals B such that β_{n+1} [B] is derivable. The proof of Theorem 3 is complete if we can show that Σ_0 is (n + 1) distinguishable (which is obvious), but not n distinguishable.

Suppose now that Σ_0 is n-distinguishable. There would be a context β'_n with n - 1 or fewer parenthesis pairs such that Σ_0 is the set of all B such that β'_n [B] is derivable in G. Recall that Σ is the set of all non-terminals A such that

$$B \longrightarrow (A_1 \; bc \; A \; A_3)$$

is a rule of G for some $B \in \Sigma_0$. Taking β'_{n+1} to be the context

$$\beta'_n \; [(A_1 \; bc \; \_ \; A_3)] \; ,$$

Σ is the set of all non-terminals A such that β'_{n+1} [A] is derivable. From the fact that there are n or fewer parenthesis pairs in β'_{n+1}, it would follow that Σ is (n + 1)-distinguishable, a contradiction. Q.E.D.

Two non-terminals A_1 and A_2 of a parenthesis grammar G are equivalent if, for every context β, either both $\beta(A_1)$ and $\beta(A_2)$ are derivable in G or neither are. Clearly, this is an equivalence relation. Also, A_1 and A_2 are equivalent if and only if there is no distinguishable Σ having one of these without the other. A grammar is reduced if no two distinct non-terminals are equivalent, and if it has no useless non-terminals: a non-terminal A is useless in G if there is no context β such that β (A) is derivable or if there is no string of terminals derivable from A. It is left to the experienced reader to convince himself (1) that any useless non-terminal, and every rule containing that non-terminal, can be discarded from the grammar without changing the language of the grammar, and (2) that there is an algorithm to determine the useless non-terminals.

(The reader who is familiar with switching theory will recognize the concept of a reduced grammar as resembling the concept of a reduced state graph. The reduction procedure of Theorem 4 will be like the reduction procedure for state graphs. Theorem 3 is, in effect, a lemma for Theorem 4, which corresponds in the reduction procedure for sequential machines to the lemma that if two states are distinguishable by an experiment of length $n + 2$, but not by an experiment of length $n + 1$, then there are two states that are distinguishable by an experiment of length $n + 1$ but not by an experiment of length n.)

Theorem 4. There is an algorithm to obtain a reduced backwards-deterministic parenthesis grammar with the same language as a given backwards-deterministic parenthesis grammar.

Proof: Let G be a backwards-deterministic parenthesis grammar. Without loss of generality, assume G has no useless non-terminals. By Theorem 3, if G has n non-terminals, any set Σ of non-terminals is distinguishable if it is 2^n-distinguishable. The list of all distinguishable sets can be determined by looking at all derivations of length 2^n and less, of which there are finitely many. But this information will also give us the equivalence sets of non-terminals. Let these be, without repetition, $\Sigma_1, \ldots, \Sigma_m$. Form the grammar G' with the non-terminals $B_1, \ldots, B_m$ and such that for every rule of G

$$A_1 \rightarrow (A_2 \; bc \; A_3 \; A_4)$$

use instead the rule

$$B_{i_1} \rightarrow \left(B_{i_2} \; bc \; B_{i_3} \; B_{i_4}\right)$$

where $A_1 \in \Sigma_{i_1}$, $A_2 \in \Sigma_{i_2}$, $A_3 \in \Sigma_{i_3}$ and $A_4 \in \Sigma_{i_4}$. B_i is an initial non-terminal if and only if there is an $A \in \Sigma_i$ which is an initial non-terminal in G. Note that if $A \in \sum_i$ is an initial non-terminal of G then every $A' \in \sum_i$ is initial. We now need two lemmas.

Lemma 1. Every derivation in G of a terminal string can be converted into a derivation of the same string in G'.

Proof: Simply rewrite the derivation with the non-terminal B_j in place of A_i where $A_i \in \sum_j$, for each A_i in the derivation. The rewriting is specified uniquely, all replacements will be according to valid rules of G', and the new derivation will have the same terminal string.

Lemma 2. Let Ω' be a string of terminals and non-terminals derivable in G'. If Ω is like Ω' except for having for each occurrence of a non-terminal B_j a non-terminal $A_i \in \sum_j$ (different occurrences of B_j being replaced possibly by different A_i's), then Ω is derivable in G.

The proof is by induction on the length of the derivation of Ω' in G'. If the length is 1, clearly Ω' must be an initial non-terminal B_j, and every $A_i \in \sum_j$ must be initial by a remark above. Assume that Lemma 2 is true for all strings derivable in x lines, and let Ω'_{x+1} be derivable in x + 1 lines. Suppose Ω'_{x+1} results from Ω'_x in a derivation in G' by means of the rule

$$B_{i_1} \rightarrow \left(B_{i_2} \text{ bc } B_{i_3} \; B_{i_4}\right).$$

By inductive hypothesis every Ω_x related to Ω'_x as described in Lemma 2 is derivable in G. We know that, for some $A_1 \in \sum_{i_1}$, $A_2 \in \sum_{i_2}$, $A_3 \in \sum_{i_3}$, $A_4 \in \sum_{i_4}$,

$$A_1 \rightarrow (A_2 \text{ bc } A_3 \; A_4)$$

is a rule of G. Consider the class K_x of all Ω_x such that (1) Ω_x is related to Ω'_x as described in Lemma 2, and (2) A_1 in Ω_x takes the place in G. Note that if $A \in \sum_i$ is an initial non-terminal of G then every $A' \in \sum_i$ is initial. We now need two lemmas.

Lemma 1. Every derivation in G of a terminal string can be converted into a derivation of the same string in G'.

Proof: Simply rewrite the derivation with the non-terminal B_j in place of A_i where $A_i \in \sum_j$, for each A_i in the derivation. The rewriting is specified uniquely, all replacements will be according to valid rules of G', and the new derivation will have the same terminal string.

Lemma 2. Let Ω' be a string of terminals and non-terminals derivable in G'. If Ω is like Ω' except for having for each occurrence of a non-terminal B_j a non-terminal $A_i \in \sum_j$ (different occurrences of B_j being replaced possibly by different A_i's), then Ω is derivable in G.

The proof is by induction on the length of the derivation of Ω' in G'. If the length is 1, clearly Ω' must be an initial non-terminal B_j, and every $A_i \in \sum_j$ must be initial by a remark above. Assume that Lemma 2 is true for all strings derivable in x lines, and let Ω'_{x+1} be derivable in x + 1 lines. Suppose Ω'_{x+1} results from Ω'_x in a derivation in G' by means of the rule

$$B_{i_1} \rightarrow \left(B_{i_2} \text{ bc } B_{i_3}\ B_{i_4}\right).$$

By inductive hypothesis, every Ω_x related to Ω'_x as described in Lemma 2 is derivable in G. We know that, for some $A_1 \in \sum_{i_1}$, $A_2 \in \sum_{i_2}$, $A_3 \in \sum_{i_3}$, $A_4 \in \sum_{i_4}$,

$$A_1 \rightarrow (A_2 \text{ bc } A_3\ A_4)$$

is a rule of G. Consider the class K_x of all Ω_x such that (1) Ω_x is related to Ω'_x as described in Lemma 2, and (2) A_1 in Ω_x takes the place of B_{i_j} in the noted occurrence in Ω'_x. From each $\Omega_x \in E_x$ we can obtain an Ω_{x+1} by the above rule from Ω_x so that Ω_{x+1} will be related to Ω'_{x+1} as described in Lemma 2 and will have $(A_2 \text{ bc } A_3\ A_4)$ in place of $\left(B_{i_2} \text{ bc } B_{i_3}\ B_{i_4}\right)$ in the noted occurrence; let E_{x+1} be the set of all such Ω_{x+1}.

Now consider an arbitrary string which is related to Ω'_{x+1} as described in Lemma 2; let us designate this string as $\Omega_{x+1}\ [(A'_2 \text{ bc } A'_3\ A'_4)]$ to indicate that $(A'_2 \text{ bc } A'_3\ A'_4)$ occurs in the noted occurrence. We know from what is proved in the above paragraph that $\Omega_{x+1}\ [(A_2 \text{ bc } A_3\ A_4)]$ is derivable in G. But then it follows that $\Omega_{x+1}\ [(A'_2 \text{ bc } A_3\ A_4)]$ is derivable, since A'_2 and A_2 are equivalent in G. From this it follows that $\Omega_{x+1}\ [(A'_2 \text{bc} A'_3 A_4)]$ and finally $\Omega_{x+1}\ [(A'_2 \text{ bc } A'_3\ A'_4)]$ are derivable. Q.E.D.

It follows easily from Lemmas 1 and 2 that G′ is equivalent to G. Clearly G′ is a parenthesis grammar. It remains to show that G′ is (1) backwards-deterministic and (2) reduced.

(1) Suppose G′ is not backwards-deterministic. Then G′ has two rules,

$$B_{i_1} \to \left(B_{i_2} \text{ bc } B_{i_3}\ B_{i_4}\right)$$

$$B_{i'_1} \to \left(B_{i_2} \text{ bc } B_{i_3}\ B_{i_4}\right),$$

where $i_1 \neq i'_1$. Then G must have two rules

$$A_{j_1} \to \left(A_{j_2} \text{ bc } A_{j_3}\ A_{j_4}\right)$$

$$A_{j'_1} \to \left(A_{j'_2} \text{ bc } A_{j'_3}\ A_{j'_4}\right),$$

where $A_{j_1} \in \Sigma_{i_1}$, $A_{j'_1} \in \Sigma_{i'_1}$, A_{j_2}, $A_{j'_2} \in \Sigma_{i_2}$, A_{j_3}, $A_{j'_3} \in \Sigma_{i_3}$, A_{j_4}, $A_{j'_4} \in \Sigma_{i_4}$. A_{j_1} and $A_{j'_1}$ can not be equivalent. But, since A_{j_2}, A_{j_3} and

A_{j_4} are, respectively, equivalent to $A_{j_2'}$, $A_{j_3'}$, and $A_{j_4'}$, for any context β.

$$\beta\left[\left(A_{j_2} \text{ bc } A_{j_3}\ A_{j_4}\right)\right]$$

is derivable if and only if

$$\beta\left[\left(A_{j_2'} \text{ bc } A_{j_3'}\ A_{j_4'}\right)\right]$$

is derivable. But the phrase $\left(A_{j_2} \text{ bc } A_{j_3}\ A_{j_4}\right)$ must come from A_{j_1} and $\left(A_{j_2'} \text{ bc } A_{j_3'}\ A_{j_4'}\right)$ from $A_{j_1'}$, since G is backwards-deterministic, and we can suppose that these rules were used last in any derivation. So $\beta\left[A_{j_1}\right]$ is derivable if and only if $\beta[A_{j_1'}]$ is derivable, for any context β; from which we conclude that $\Sigma_{i_1} = \Sigma_{i_1'}$, contrary to the stipulation that $i_1 \neq i_1'$. Thus G′ is backwards-deterministic.

(2) Suppose G′ is not reduced. It is clear that, since G has no useless non-terminals, G′ has no useless non-terminals. Thus, for some i and j, $i \neq j$ and, for every context β', $\beta'\left[B_i\right]$ is derivable in G′ if and only if $\beta'\left[B_j\right]$ is derivable. Suppose $A_1 \in \Sigma_i$, $A_2 \in \Sigma_j$. Then, by lemmas 1 and 2, it follows that, for every context β in G, $\beta\ [A_1]$ is derivable if and only if $\beta[A_2]$ is derivable, which means that A_1 and A_2 are equivalent, and $\Sigma_i = \Sigma_j$, contradicting the stipulation that $i \neq j$. Thus G′ is reduced, which concludes the proof of Theorem 4.

Two grammars are <u>isomorphic</u> if they have the same set of terminals, and if there is a one-to-one correspondence between the sets of non-terminals, initial non-terminals in one grammar corresponding to initial non-terminals in the other grammar; and there is a one-to-one correspondence between rules of one and rules of the other, where a rule of one grammar can be obtained from the corresponding rule of the other by replacing each non-terminal in the second grammar by the corresponding non-terminal in the first.

Theorem 5. Two equivalent reduced backwards-deterministic parenthesis grammars are isomorphic.

Proof: Let G, G' be grammars satisfying the hypothesis of Theorem 5. Let A_1 be a non-terminal of G and let Δ be the set of terminal strings derivable from A_1. Δ is not empty (lest A_1 be useless); let $\Omega \in \Delta$. Ω must be a part of a word of the language (again, since A_1 is not useless) beginning and ending with a pair of mated parentheses and hence Ω must be derivable from some A_1' in G'. Let Δ' be the set of terminal strings derivable from A_1' in G'.

Now suppose $\Delta \neq \Delta'$. Then there would be a terminal string, say Ω_0, in one but not in the other. Suppose $\Omega_0 \in \Delta - \Delta'$. Then Ω_0 must be derivable from some A_2' in G', where $A_2' \neq A_1'$. Now Ω_0 and Ω are not derivable from any other non-terminal in G other than A_1, by Theorem 2. Hence for any terminal context β, $\beta[\Omega]$ is in the language if and only if $\beta[\Omega_0]$ is. From this fact it follows that, for every context β, terminal or otherwise, $\beta[A_1']$ is derivable in G' if and only if $\beta[A_2']$ is derivable; for, since G' has no useless non-terminals, a terminal string must be derivable from every derivable string. But this result contradicts the assumption that G' is reduced. Similarly, from the supposition that $\Omega_0 \in \Delta' - \Delta$ we can conclude that G is not reduced. Thus we infer that $\Delta = \Delta'$.

From this argument we conclude that each non-terminal in G has a corresponding non-terminal in G' from which the same set of terminal strings can be derived. This shows that there is a one-to-one correspondence between non-terminals. It remains to show that there is the appropriate one-to-one correspondence between rules of G and rules of G'.

Suppose $A_1 \longrightarrow (A_2 \text{ bc } A_3 A_4)$ is a rule of G. Let Ω_2, Ω_3 and Ω_4 be terminal strings derivable from A_2, A_3 A_4, respectively. $(\Omega_2 \text{ bc } \Omega_3 \Omega_4)$ must be derivable from A_1. By what has been proved, Ω_2, Ω_3, Ω_4 and $(\Omega_2 \text{ bc } \Omega_3 \Omega_4)$ are terminal strings derivable in G' from A_2', A_3', A_4' and A_1', respectively. But, because G' is a backwards-deterministic parenthesis grammar, the second line in a derivation of $(\Omega_2 \text{ bc } \Omega_3 \Omega_4)$ from A' in G' must be $(A_2' \text{ bc } A_3' A_4')$. It follows then that

$$A_1' \longrightarrow (A_2' \text{ bc } A_3' A_4')$$

is a rule of G'. This argument goes both ways, and suffices to show that there is the appropriate one-to-one correspondence between the two sets of rules, and that the two grammars are isomorphic.

The objective of this paper has now been achieved namely the proof of the following, which clearly follows from Theorems 1 through 5. The corollary following it is justified in a remark at the beginning of this paper.

Main Theorem. There is a decision procedure to determine whether the languages of two given parenthesis grammars are equal.

Corollary. There is a decision procedure to determine whether one of two such languages is included in the other.

A concealed parenthesis grammar is a grammar that is not a parenthesis grammar but generates a parenthesis language. The problem of how to recognize whether a given context-free grammar is a concealed parenthesis grammar and the problem of converting one into a parenthesis grammar are open. That these are not simple problems is illustrated in the following examples. A grammar whose language is not a parenthesis language, although it might seem so at first glance, is

$S \rightarrow (A)$
$A \rightarrow AA$
$A \rightarrow (a)$.

An example of a concealed parenthesis grammar, but not obviously so, is

$S \rightarrow AB$
$A \rightarrow ((a)\ A)$
$A \rightarrow (c$
$B \rightarrow (B(b))$
$B \rightarrow d)$.

The parenthesis grammar equivalent to it is

$S \rightarrow (cd)$
$S \rightarrow (cZW)$
$S \rightarrow (XYd)$
$S \rightarrow (XYZW)$
$X \rightarrow (a)$
$Y \rightarrow (XY)$
$Y \rightarrow (c)$
$Z \rightarrow (ZW)$
$Z \rightarrow (d)$
$W \rightarrow (b)$.

In all of these grammars the only initial non-terminal is S.

Finally, it should be remarked that the concepts of this paper apply to some grammars that do not have parentheses at all. The well-known parenthesis-free notation is an example. Consider, for example, the following grammar for a fragment of Łukasiewicz[4] propositional calculus (in which S is the only non-terminal and is initial):

$S \rightarrow CSS$
$S \rightarrow NS$
$S \rightarrow p$
$S \rightarrow q$
$S \rightarrow r$

A terminal string in this language can be parsed in only one way; and in fact there is a method of parsing that assures us that we can proceed without making any false starts. The reason is that the grammar is backwards-deterministic, in a sense that can be defined precisely as follows: a context-free grammar is <u>backwards-deterministic</u> (in the general sense) if (1) it has no two distinct rules that have $\Omega_1\Omega_2$ and $\Omega_2\Omega_3$ as the respective right sides, for any strings Ω_1, Ω_2, Ω_3 where Ω_2 is non-null, and (2) it has no two distinct rules having Ω_2 and $\Omega_1\Omega_2\Omega_3$ as the respective right sides for any Ω_1, Ω_2, Ω_3. Thus in any string of terminals and non-terminals, there are no two parsing possibilities that interfere with each other. It is easy to see that for parenthesis grammars, this definition of "backwards-deterministic" is equivalent to the one used above.

Thus we can say that the grammar given above for the fragment of Łukasiewicz' propositional calculus is backwards-deterministic. Indeed it is possible to say that the essence of Łukasiewicz' contribution is the construction of an approximately backwards-deterministic language without parentheses. (There are problems converting the whole of a parenthesis-free propositional calculus into a backwards-deterministic language. For example, the problem of providing for an infinite number of propositional variables. These problems are not unsurmountable, but it does not seem worthwhile to discuss them here.)

It is not difficult to verify that Theorems 2, 3 and 4 hold for such grammars. Unfortunately, Theorem 5 is false. For the grammar for the fragment of Łukasiewicz' propositional calculus is backwards-deterministic and reduced. But the following equivalent reduced backwards-deterministic grammar is not isomorphic to its

$$S \rightarrow XS$$
$$X \rightarrow CS$$
$$S \rightarrow BS$$
$$S \rightarrow p$$
$$S \rightarrow q$$
$$S \rightarrow r$$

The equivalence problem, therefore, for general backwards-deterministic context-free languages is open.