THEO JANSSEN, GERARD KOK AND LAMBERT MEERTENS

ON RESTRICTIONS ON TRANSFORMATIONAL GRAMMARS REDUCING THE GENERATIVE POWER

ABSTRACT. Various restrictions on transformational grammars have been investigated in order to reduce their generative power from recursively enumerable languages to recursive languages.

It will be shown that any restriction on transformational grammars defining a recursively enumerable subset of the set of all transformational grammars, is either too weak (in the sense that there does not exist a general decision procedure for all languages generated under such a restriction) or too strong (in the sense that there exists a recursive language that cannot be generated by any transformational grammar thus restricted). In addition, some related problems will be discussed.

1. MOTIVATION

Chomsky (1965), Ch. I, § 6, states that a theory of linguistic structure should aim at descriptive adequacy. To provide for this aim the theory must contain, among other things:

(A) a definition of 'generative grammar';

(B) a method for determining the structural description of a sentence, given a grammar.

Requirement (A) can be interpreted as the requirement that we have a procedure to decide whether a given text describes a possible generative grammar. Chomsky, however, gives a more liberal formulation:

(A′) the theory must provide for an enumeration $G_1, G_2, \ldots$ of possible generative grammars.

When Chomsky describes how a descriptively adequate theory would attempt to account for language learning, it appears that (B) is to be interpreted as:

(B′) a method for determining, given a grammar, whether a given sentence can be generated by that grammar, and if so, what structural description is assigned to it.

Thus (B′) implies that:

(C) the theory provides for a method to decide whether a given sentence can be generated by a given grammar.

This last statement (C) is often formulated as '*natural languages are recursive*'.[1] For arguments concerning (C) see, besides Chomsky (1965), also Peters and Ritchie (1973) and Putnam (1961).

[1] A language L is called *recursive* if there exists an effective procedure to decide whether a given sentence belongs to L or not.

Linguistics and Philosophy **1** (1977) 111–118. *All rights reserved.*

Chomsky has introduced transformational grammars to serve as a model for the linguistic structure of natural languages. A formal definition can be found in Ginsburg and Partee (1969) or Peters and Ritchie (1973); knowledge of the definition, however, is not needed for our purposes. Peters and Ritchie (1973) have proved that the generative power of transformational grammars is of Chomsky type 0 (recursively enumerable).[2] So, in the light of requirement (C), it appears that the system of transformational grammars is too powerful, since not every transformational grammar is a possible grammar for natural language. Hence, the aim of descriptive adequacy is not fulfilled. Therefore, the obvious thing to do is to search for a restriction on *TG* (the set of all transformational grammars) that defines a subset *RTG* of *TG* such that only recursive languages will be generated.

In accordance with (A′) we require *RTG* to be recursively enumerable. On the other hand, it must be possible to describe every possible natural language by a grammar determined by the theory.

Requirement (C) states that natural languages should be recursive. In order to escape the risk of excluding possible natural languages by the restriction we suggest:

(D) for every recursive language there is a grammar in *RTG*.

In this situation it is interesting to search for the 'ideal' restriction: a restriction on *TG* satisfying requirements (A′), (C) and (D).

Several proposals have been investigated previously. Peters and Ritchie (1971) have investigated restrictions on the base-component of transformational grammars. Even drastic restrictions do not reduce their generative power. The same authors introduce the cycling function of a transformational grammar. The generated language is recursive if and only if this function is recursive (i.e. effectively computable). But they define the cycling function in a non-effective way; hence, the restriction of having a recursive cycling function cannot be checked by an algorithm. In the same article they discuss a proposal of Putnam (1961) and show that the generative capacity is reduced to that of context-sensitive grammar, thus showing this restriction to be too strong. So it appears that the restrictions investigated are not recursive themselves, or if they are, that they either reduce too strongly, or that they do not reduce the generative capacity at all.

Since we conjectured that this was not due to a lack of good ideas but to mathematical necessity, we investigated the matter and succeeded in showing that an ideal restriction does not exist. Futhermore the proof suggested some interesting new problems, which are also investigated. The problems

[2] A language *L* is called *recursively enumerable* if there exists an effective procedure for enumerating the sentences of *L*.

under consideration can also be formulated in terms of Recursion Theory (the branch of mathematics that studies recursively enumerable sets, recursive functions, etc.). It turned out, as could be expected, that these problems had already been dealt with in this field.

In fact, the result mentioned above was proved by Dekker (1953). Although our result is not new, we hold the opinion that it is useful to present our proof because, from our point of view, it is an important result which seems not to be known outside the context of Recursion Theory; moreover, the proof is not complex. We have formulated it in terms of transformational grammars, but it applies analogously for any formal-language-describing system. For example, although we deal with grammars only as *generative* systems, the results are equally valid for *accepting* systems.

2. Preliminaries

We suppose that all grammars of *TG* generate languages over the same finite alphabet V (e.g., all symbols on all typewriters in the world). So we have an enumeration $z_1, z_2, \ldots$ of all sentences[3] over V, arranged on length, and for every length in some 'alphabetical' order. In case V is infinite, but enumerable, we can prove analogous results. In that case, it suffices to encode all symbols in some finite alphabet (e.g., 0–1 code) and to prove the theorems for the encoded languages. The language generated by a grammar G will be denoted by $\mathrm{L}(G)$.

3. Results

3.1. *The Ideal Restriction*

THEOREM 1. *There exists no subset RTG of TG which satisfies the following requirements*:

(A′) *RTG is recursively enumerable*;

(C) *there is a method to decide whether a given sentence can be generated by a given grammar of RTG*;

(D) *for every recursive language there is a grammar in RTG.*

Proof. Assume that all three requirements are satisfied.

Let $z_1, z_2, \ldots$ be the enumeration of all sentences over V and let $G_1, G_2, \ldots$ be an enumeration of the grammars of *RTG*.

Let the language H be defined by:

$$z_i \text{ belongs to } H \text{ if and only if } z_i \text{ does not belong to } \mathrm{L}(G_i).$$

[3] In the literature also the terms *words* or *strings* are used with the same meaning.

Because of the requirement (C), there is a decision procedure to test whether z_i belongs to $\mathrm{L}(G_i)$, and, therefore, whether z_i belongs to H. So H is recursive. Because of requirement (D), there is a grammar in *RTG*, say G_h, such that $H = \mathrm{L}(G_h)$.

Now we have the following contradiction for z_h:

> z_h belongs to $\mathrm{L}(G_h)$ if and only if z_h belongs to H, that is, if and only if z_h does not belong to $\mathrm{L}(G_h)$. □

3.2. *A Weaker Restriction*

In the above theorem, (C) implied the existence of a method providing for a decision procedure for each language generated by a grammar of *RTG*. As we shall see, the situation changes dramatically if (C) is replaced by the weaker requirement:

($\mathrm{C_w}$) none of the languages generated by a grammar of *RTG* is non-recursive.[4]

In order to explain the difference between (C) and ($\mathrm{C_w}$), consider the language L defined as follows: The decimal expansion of π, 3.141592653 . . . , may be viewed as an infinite string of digits. L contains exactly all strings of seven digits occurring in the decimal expansion of π. Obviously, L is a recursively enumerable language (with enumeration 3141592, 1415926, 4159265, . . .) so there exists a transformational grammar G generating L. Since no effective procedure is (yet) known to decide whether, e.g., 1234567 belongs to L, it is obvious that G could not belong to an *RTG* which has been shown to satisfy (C). However, G might well belong to an *RTG* which has been shown to satisfy ($\mathrm{C_w}$): Since all finite languages are recursive, a non-recursive language cannot be finite, i.e. it must be infinite. But L is certainly not infinite: there are but 10^7 strings of seven digits, not even considering occurrence in the decimal expansion of π. Consequently, L cannot be non-recursive either.

The consequence of replacing (C) by ($\mathrm{C_w}$), is that, surprisingly, the construction of an *RTG* satisfying (A′), ($\mathrm{C_w}$) and (D) does become possible. For Recursion Theory, this case has also been studied by Dekker (1953). We give a slightly sharpened version of this theorem, presented in linguistic terminology. Unlike Theorem 1, in which *TG* may be replaced by any

[4] This formulation is preferred to ***each language generated by a grammar of RTG is recursive***, since this would be interpreted by constructivists as being equivalent to (C) rather than to ($\mathrm{C_w}$). In view of the fact that the expressions ***natural languages are recursive*** and ***no natural language is non-recursive*** are normally understood to have the same meaning, it should be recommended to formulate (C) as ***natural languages are recursive*** only if it is clear that this is to be interpreted in a constructive way.

formal-language-describing system, this theorem applies only to Chomsky-type-0 systems such as transformational grammars, Turing machines or Van Wijngaarden grammars. Our result is slightly sharper in that it exhibits the existence of a *recursive* restriction, whereas Dekker merely shows the existence of a *recursively enumerable* restriction.

Only a sketch of the proof is given.

THEOREM 2. *There exists a subset RTG of TG which satisfies the following requirements*:

(A″) *RTG is recursive*;

(C_w) *none of the languages generated by a grammar of RTG is non-recursive*;

(D) *for every recursive language there is a grammar in RTG.*

Sketch of Proof. Peters and Ritchie (1971) have shown constructively that any language enumerated by a Turing machine is generated by some transformational grammar which, as it were, simulates that Turing machine. We may, therefore, give a construction in terms of machines. The enumeration $z_1, z_2, \ldots$ of all sentences over V, as introduced in the preliminaries, defines an ordering of the sentences over V: $z_1 < z_2 < \ldots$. An *enumerator* is a Turing machine which enumerates a sequence $y_1, y_2, \ldots$ of sentences over V; it is called *ascending* if $y_1 < y_2 < \ldots$.

LEMMA 1. *For each recursive language, there exists an ascending enumerator.*

Proof. Enumerate 'internally' all sentences over V in ascending order, but emit only the sentences belonging to the language and discard all others (which can be tested using the decision procedure for that language). □

LEMMA 2. *An infinite language for which an ascending enumerator exists is recursive.*

Proof. In order to decide whether a given sentence y belongs to the language, enumerate its sentences $y_1, y_2, \ldots$ in ascending order until a y_i is met such that $y_i \geqslant s$. The sentence s belongs to the language if and only if $y_i = s$. □

Note that this method does not work for finite languages, since, if at some moment only sentences $y_i < s$ have been enumerated, there is no general way to tell whether this constitutes the full language or not, so one simply has no choice but to wait and see if more is coming. This waiting might continue indefinitely.

The following transformation turns an arbitrary enumerator into an ascending one: Enumerate the sentences, remembering the sentence last emitted. If the next sentence is higher in order, it is emitted; otherwise it is discarded. Obviously, this construction transforms already ascending enumerators into equivalent ones.

RTG consists of all grammars that can be obtained as follows: Start with an arbitrary enumerator, transform it into an ascending enumerator and take the grammar which, according to the construction of Peters and Ritchie 'simulates' this ascending enumerator.

We will show that *RTG*, thus defined, satisfies (A″), (C_w) and (D).

As for (A″). Obviously, *RTG* is recursively enumerable, since it is possible to enumerate all enumerators, applying the transformation and the construction of Peters and Ritchie to each enumerator. In fact it is possible to decide by inspection whether a grammar of *TG* may be obtained in this way, much in the same sense in which it is possible to decide whether a given sentence may have been obtained by a transformation which replaces all occurrences of 's' in some sentence by 'f'.

As for (C_w). Let *L* be a language generated by a grammar of *RTG*. Suppose that *L* is non-recursive. Clearly, *L* cannot be finite. But *L* is enumerated by an ascending enumerator, so by Lemma 2, *L* cannot be infinite either.

As for (D). For each recursive language, there exists by Lemma 1 an ascending enumerator for that language, which may be transformed into an equivalent one. Consequently, there is a grammar of *RTG* for that language. □

3.3. *Restriction to Infinite Languages*

The crux of Theorem 2 lies in the proof of (C_w). If we know that the language described by a given grammar is infinite, we have (by Lemma 2) a decision procedure. If the language happens to be finite, however, this procedure fails, but in that case the language is recursive because all finite languages are. Since there is no procedure for deciding whether such a language is finite or not, the theorem, although of some theoretical interest, is, in the opinion of the authors, of no practical value.

Since all 'interesting' languages are infinite, one might wonder if in this theorem it is essential that finite languages play such an elusive role. This is indeed the case, as has been shown (in terms of Recursion Theory) by Van Emde Boas and Vitányi (1975) who proved:

THEOREM 3. *There exists no subset RTG of TG which satisfies the following requirements*:

(A′) *RTG is recursively enumerable*;

(D_i) *for every infinite recursive language, there is a grammar in RTG describing that language*;

(E) *every language described by a grammar of RTG is infinite.*

Proof. Assume that all three requirements are satisfied. Let $G_1, G_2, \ldots$ be an enumeration of the grammars of *RTG*. The following enumerator enumerates two sequences of sentences $x_1, x_2, \ldots$ and $y_1, y_2, \ldots$ simultaneously, in ascending order: In order to obtain x_k, enumerate the sentences of $L(G_k)$, until a sentence is met which is higher in order than all previously emitted sentences; emit this sentence as x_k (such a sentence must occur in the enumeration of $L(G_k)$, since $L(G_k)$ is infinite). In order to obtain y_k, proceed with the enumeration of $L(G_k)$ until again a sentence is met which is higher in order than all previously emitted sentences; this sentence is emitted as y_k (so $x_1 < y_1 < x_2 < y_2 \ldots$).

In this way two infinite languages are obtained: X, consisting of the sentences $x_1, x_2, \ldots$ and Y, consisting of $y_1, y_2, \ldots$. By the construction, there exists an ascending enumerator for X and for Y, so X and Y are infinite recursive languages. Moreover, X and Y are disjoint, i.e., no sentence of X belongs to Y and vice versa. By (D_i) there is a grammar in *RTG*, say G_x, such that $X = L(G_x)$.

Now we have the following contradiction for y_x:

> y_x belongs to $L(G_x) = X$, but y_x also belongs to Y, which is impossible by the construction of X and Y. □

4. Linguistic consequences

In the light of the foregoing theorems we see two possibilities for descriptive linguistics:

(1) To describe language in a system which is essentially more powerful than is necessary for the description of all recursive languages. Transformational grammars are an example of such a system; there are more such systems, all with equal power: every general-purpose programming language is one. The justification of the choice for *TG* as descriptive mechanism can then only be its convenience as a tool.

(2) To postulate a constructive restriction which excludes not only nonrecursive but also some recursive languages (i.e., a restriction like context-free or context-sensitive); such a restriction should be based on a new hypothesis concerning the character of natural languages.

Mathematisch Centrum, Amsterdam
Instituut voor Cognitie Onderzoek, University of Amsterdam

BIBLIOGRAPHY

Chomsky, N.: 1965, *Aspects of the Theory of Syntax*, MIT Press, Cambridge, Mass.

Dekker, J. C. E.: 1953, 'The Constructivity of Maximal Dual Ideals in Certain Boolean Algebras', *Pacific Journal of Mathematics* **3**, 73–101.

Van Emde Boas, P. and Vitányi, P. M. B.: 1975, *A Note on the Recursive Enumerability of Some Classes of Recursively Enumerable Languages* (= *M.C. Report* IW 37/75).

Ginsburg, S. and Partee, B.: 1969, 'A Mathematical Model of Transformational Grammars', *Information and Control* **15**, 297–334.

Peters, Jr., P. S. and Ritchie, R. W.: 1971, 'On Restricting the Base Component of Transformational Grammars', *Information and Control* **18**, 483–501.

Peters, Jr., P. S. and Ritchie, R. W.: 1973, 'On the Generative Power of Transformational Grammars', *Information Sciences* **6**, 49–83.

Putnam, H.: 1961, 'Some Issues in the Theory of Grammar', In Roman Jacobson (ed.), *Proc. Symp. in Appl. Mathematics*, Vol. XII: *The Structure of Language and its Mathematical Aspects*, A.M.S., Providence, R.I.