Syntactic Analysis for Transformational Grammars

by

S. R. Petrick

IBM T. J. Watson Research Center

If one wishes to obtain a syntactic analysis algorithm for some class of grammars, it is, of course, essential to characterize that class of grammars completely and precisely. If we merely tie down those details for which there exists abundant linguistic justification and leave unspecified those aspects of a linguistic theory which have not been so thoroughly worked out, it may be possible to propose small sets of rules to generatively account for certain linguistic phenomena but it is more difficult to give meaningful consideration to the problem of syntactic analysis. It is likely that the existence of any general algorithm for syntactic analysis depends upon the specification of the incomplete aspects of the model in question. It should be noted in this regard that the requirement that a language be recursive can offer some help in making certain decisions with respect to the construction of a linguistic theory which would otherwise be arbitrary.

As is well known, transformational theory has been changing rapidly from its inception up to the present time. There is disagreement as to the basic mechanisms that should be allowed (e.g., conventions on transformational applicability, allowable structures, primitive transformations, etc.) and as to the use to which those mechanisms should be put (e.g., lexical or transformational treatment of certain sentences).

A person who wishes to produce an algorithm for transformational syntactic analysis is faced then with a difficult task; he must on the one hand completely specify a class of transformational syntactic components for which an analysis procedure can be found, and he must on the other hand so define that class as to make it a reasonable model of contemporary transformational

descriptive practice. After reading this paper the reader can judge for himself the extent to which I have met the latter requirement while at the same time satisfying the former requirement.

There are several alternatives to the course I have chosen that are open to anyone wishing to work on "transformational syntactic analysis". First, he can talk about the theoretical requirements of transformational analysis without actually working out the complete details of an analysis algorithm for any class of grammars. Such work can be valuable, especially if it contributes to our knowledge of the precise nature of transformational rules and conventions. The more assumptions we can build in, and consequently the tighter we can make our model without impairing its facility to describe language, the better that model is, and the closer we are to saying something about a discovery procedure.

A second alternative (followed by the MITRE Corporation < 1>) is to seek an analysis algorithm for a particular grammar rather than for a class of grammars. There are several objections I would raise to such a course of action. First of all, even though linguists tend to be quite tentative and cautious about the properties and details of a class of grammars they propose as models of natural languages, they are certainly even more tentative about endorsing the likelihood that the particular fragmentary grammars they produce will stand up with the passage of time. My second objection to the consideration of particular grammars concerns the difficulty of producing an analysis procedure for a particular grammar. While it would appear an easier task than for a class of grammars, it is very hard to extract the necessary properties that

one needs for a proof from a particular grammar without in the process specifying a class of grammars that have those properties. The situation is not unlike that in mathematics where a generalization is often easier to prove than a more restricted result. I had an excellent professor in Theory of Functions (William Ted Martin) who delighted in providing such examples. Whenever we would bog down in obtaining a needed proof we would hear his familiar advice to "Ask for more when the required result is too specific."

A third alternative which has been followed by most people who characterize their work in syntactic analysis as "transformational" in nature is to define a "transformational-like" linguistic theory based upon some algorithm for syntactic analysis, not upon the usual generative transformational apparatus. The deep structures produced by such programs often appear to be very close to those that are assigned to the same sentences by current transformational grammars, and the rules, which are variously called "transformations" or "inverse transformations" or sometimes just "rewriting rules", often bear names and functions similar to the transformations of generative transformational grammar theory. Efforts I would classify as being of this type have been undertaken by Kay <2>, Simmons <3>, Moyne <4>, Thorne <5>, Fraser and Bobrow <6>, Woods <7>, Winograd <8>, and Kellogg <9>, to name a few. The most compelling argument for such systems is their efficiency for natural language processing projects relative to existing parsers for generative transformational grammars. Surprisingly, relatively little is made of this by the proponents of these systems. The argument often given, on the other hand, namely that of suitability as a perceptual model, has been totally unconvincing

in my opinion. The most important thing to note is that whatever the merits or shortcomings of such systems, they are linguistic theories which are unrelated to generative transformational grammar theory and as such their proponents face the task of independently establishing their adequacy for linguistic descriptive and explanatory purposes - their capacity for expressing significant linguistic generalizations. Unfortunately, many of the proponents of such systems have not given enough attention to this task, basing the justification of their linguistic theories not upon their ability to account for specific linguistic data, but rather upon their tenuous relationship to generative transformational theory.

Rather than discussing these alternatives further I will instead discuss my own work on transformational syntactic analysis. Let us begin by sketching briefly the transformational analysis algorithm of my thesis.

The model of transformational theory in question is roughly that which was in vogue prior to Aspects of the Theory of Syntax <11>. The base component is a context-free grammar with certain restrictions on recursiveness and sentence embedding. Transformational applicability is specified by a structural index. This structural index is satisfied by a proper analysis which is a sequence of subtrees that constitutes a single cut through a tree. The modification to a tree by a transformation is specified by a structural change. For our simple model this is limited to substituting strings of trees for each of the trees of the proper analysis that satisfies the structural index. A number of restrictions on derivations are necessary to guarantee that the language generated is recursive.

Our analysis algorithm is based upon a reversal of the procedure used

for generating a given string. This reverse procedure makes use of inverse transformations, which are mechanically computed—one for each generative transformation. In analyzing a sentence S, inverse transformations are applied in the reverse order of that in which the corresponding generative transformations are applied in deriving S.

To understand inverse transformations let us examine their generative counterparts. We observe first of all that the structural change of a transformation references a sequence of nodes that occur in the structural index, interspersed possibly with additional morphemes. We call this sequence the inverse structural index of the transformation in question. The structural change of course gives more information than is contained in the inverse structural index, but the latter provides the basis for our analysis algorithm.

To give an example of an inverse structural index, let us consider a passive transformation whose structural index is (NP AUX V X NP X BY PASS) and whose structural change is (5 2 (BE EN 3) 4 0 6 7 1). The inverse structural index is (NP AUX BE EN V X X BY NP) because 5 denotes NP, 2 denotes AUX, etc.

For a transformation to be applicable to a tree T there must be a proper analysis of T that satisfies the structural index of that transformation. The structural changes that may be performed by transformations as we have formalized them are limited to the substitution of a sequence of trees (including possibly the null sequence) for a single tree, a process which is followed by erasure of all nonterminal nodes that dominate no terminal symbol. Hence, for the tree resulting from application of a transformation, there must exist a proper analysis that satisfies the inverse structural index of that transformation

We make use of this fact in the following way. If a string of morphemes s is the terminal string of a tree produced by the action of a transformation *t,* then it must be possible to segment s such that the ith segment can be analyzed as the ith node of the inverse structural index of t with respect to a context-free grammar consisting of the original base component rules augmented by rules reflecting structure that can be produced transformationally. It is possible to mechanically derive an augmented context-free grammar that includes all rules reflecting structure that might be formed in the transformational derivation of a given sentence. Hence, we have a necessary test that a given string of morphemes s was produced by a transformation t.

Sufficient information is given in a transformation to permit the computation of a function we will call the corresponding inverse transformation. This function maps a sequence of trees satisfying the inverse structural index of some transformation into a sequence of trees satisfying the structural index of that transformation. More precisely, if a transformation t performed on a tree T yields a tree T', we denote by P' the proper analysis of T' that satisfies the inverse structural index of t. Now the inverse transformation t' corresponding to t maps the proper analysis P' into a sequence of trees whose debracketization is the terminal string of T. For the previously considered transformation the inverse transformation can be specified in terms of the inverse structural index (NP AUX BE EN V X X BY NP) and the inverse structural change (9 2 5 6 1 7 8 PASS). Note that there is no requirement that the inverse structural index and the inverse structural change have the same number of terms. The inverse transformation, as we define it, is not a true inverse transformation for which t' t T = T.

Let us now see how an analysis procedure can be based upon our inverse transformations. We take up the analysis of a sentence S with respect to a given context-free grammar G and an ordered set of transformations t_1, t_2 . . . t_n. To simplify our exposition we begin with a grammar containing no binary transformations (i.e.,transformations are not applied in a cyclic fashion).

Using one of the methods given by Petrick <10> we determine an augmented context-free grammar G' that contains rules reflecting all structure that can be produced in the derivation of S. In reversing the generative procedure we first consider t_n. If t_n was performed in deriving S, it must be possible to segment S such that with respect to G', the ith segment has an analysis as a tree dominated by the ith term of the inverse structural index of t_n. If such a segmentation is possible, and if t_n' is performed on the sequence of trees provided by this segmentation, then the debracketization S' of the resulting sequence of trees must be the terminal string of the tree that existed just before application of t_n. (Complete debracketization turns out to be unnecessary. Repeated debracketization of outermost structure until no derived constituent structure remains is all that is required.) If the analysis of S and S' (if it exists) are separately considered with respect to the original grammar with only transformations t_1, t_2, ... t_{n-1} , then the problem consists of one or more instances of essentially the original problem of analyzing S with respect to t_1 , t_2, ..., t_n. If we carry out this procedure for each of the remaining n-1 inverse transformations we obtain a set of debracketizations (S, S', .. .). Further reversing the generative procedure, it remains only to determine which elements of this set are analyzable as the sentence symbol

with respect to G. Every deep structure of S with respect to the given transformational grammar must be included in the set of trees thus obtained. With each tree it is also possible to associate the sequence of transformations used in obtaining it.

The analysis procedure becomes more complicated when binary transformations are included, as would be expected. Performance of an inverse binary transformation must always insert two occurrences of the sentence boundary symbol SENTB. The sequence of trees lying between these two SENTB markers corresponds to the constituent sentence, and the sequence of trees lying outside these two markers corresponds to the matrix sentence. Let us call the debracketization of the former sequence the constituent sentence continuation; and let us call the debracketization of the latter sequence, with the symbol COMP inserted to divide the left and righthand sections, the matrix sentence continuation.

It is clear that the constituent sentence continuation could arise from repeated application of the transformational cycle. Hence, the problem of determining the underlying deep structure of this derived string is another instance of the original problem. In other words, inverse transformations must be applied to the constituent sentence continuation in reverse generative order, as we already have discussed. If, eventually, no binary transformation applies on some inverse cycle, the recursion terminates; the structure thus found is of course dominated by the sentence symbol S1, and in the complete structural description of the given sentence this S1-dominated tree is attached under the COMP symbol of the matrix sentence continuation (by the rule COMP $\rightarrow$ SENTB S1 SENTB).

The generative transformational cycle works in such a way that no singulary transformations apply to the matrix sentence structure before a binary transformation has been applied to it and the constituent structure it dominates. Hence, the matrix sentence continuation resulting from an inverse binary transformation need not be subjected to the entire inverse transformational cycle. More than one embedded constituent sentence structure can be dominated by a single matrix sentence structure, however (as, for example, when both subject and object contain relative clauses), so the matrix sentence continuation must be subjected to repeated applications of the same or other binary transformations. The resulting matrix sentence continuation must finally be analyzed with respect to the base component G to see if an analysis as an S1 is possible. Every underlying structure assigned by the given grammar to S must be included in the set of structures thus obtained.

A brief reflection on why we begin the analysis of a sentence by applying inverse singulary transformations is in order. Although it is true that singulary transformations precede binary transformations in a given cycle, when the last binary transformation has been performed for the last time it is still possible for the singulary rules to apply to the result of this final embedding. Once the last singulary has been applied, however, generation is complete because no further binary transformation can be performed. The last singulary transformation for generation is therefore the first transformation whose corresponding inverse is to be applied in recognition.

As we have already observed, the exhaustive procedure we have describe must find all underlying structures assigned by a given transformational gramme

to a sentence. It is possible, however, that one or more spurious structures will also be found. There are several sources of slack in our procedure that could cause such a situation to occur. One of these is related to our use of so-called "auxiliary" phrase structure rules, which reflect structure that can be transformationally derived. These rules are required in order to ensure the application of every inverse transformation necessary to reverse the generative derivation. The use of these rules, however, raises the possibility that an inverse transformation will be applied at some point where it should not apply, from the point of view of reversing a valid generative derivation. If the continuation resulting from this wrong application of an inverse transformation is not subsequently blocked, an invalid underlying structure may result.

Three other sources of incorrect "structural descriptions" are possible. All can be eliminated by suitable modification of the basic procedure we have presented. The first deals with the use of obligatory transformations. The procedure we have described finds all underlying structural descriptions of a sentence with respect to a grammar in which all the transformations are taken to be optional. It also yields all correct structural descriptions for a grammar in which only some of the transformations are obligatory, but it may in addition give erroneous structures.

The second source of unwanted structures is related to supplementary conditions that may be imposed on the applicability of a transformation. For example, the basic procedure we have described could not test to ensure that trees satisfying terms of a structural index are dominated by prescribed higher nodes.

The third source of error is related to trees that are reduplicated by a transformation. In recognition it is of course necessary to ensure that two trees are identical. It would be easy enough to mark such transformations and make the necessary tests for equality, thus eliminating this source of incorrect structures.

Incorrect structures arising from any source may be discarded during the final synthesis phase. In this phase, structures produced by the analysis process are viewed as instructions to produce sentences. The base tree and the list of transformations constitute commands defining the appropriate phrase structure and transformational rules to apply to generate a sentence. The resultant string is compared with the original input string. The structural descriptions that yield strings matching the input string are those that constitute structural descriptions of the input string s. All other structural descriptions, which yield either nonmatching strings or no strings at all, are discarded. In practice, bogus recognitions are rare. In theory, their possible occurrence renders synthesis a necessary part of an effective recognition procedure of the type we have sketched.

A thorough understanding of this analysis algorithm requires more precise definitions and some concrete examples. The reader is referred to references <10>, <12>, and <13> for these.

In the past three years an effort has been made to extend the class of grammars to more adequately reflect current linguistic theory. The principal extensions made have been provisions for handling: (1) complex symbols (nodes with features), (2) a generalized structural condition of transformational

applicability, (3) stateable additional conditions of transformational applicability, (4) Chomsky adjunction, (5) the use of coordination-reduction rule schemata, (6) precyclic and postcyclic transformational components, and (7) obligatory as well as optional transformations.

Considering these extensions individually, the addition of complex symbols presents several problems. First, I have not faced the problem of lexical selection and its inverse so I have assumed that input sentences consist of strings of feature bundles. Second, I have restricted features to lexical items in the manner of Aspects <11>. The more general case of allowing features to be associated with nonterminal nodes has been considered, but there remain unsolved problems of deriving the features to be associated with the nonterminals of transformationally derived structure. Finally, certain feature-sensitive rules can give rise to nondeterministic inverse transformations. For example, if a transformation of the type [+A] $\rightarrow$ [-B] is used, it is indeterminate whether the reverse transformation should leave -B as it is, change it to +B, or delete it entirely. Separate continuations resulting from all three possibilities must be followed, and a rule of the type

$$[+A1] \rightarrow \begin{matrix} +A2 \\ +A3 \\ \cdots \\ +An \end{matrix}$$

would lead to 3^n independent continuations.

The generalized structural condition of transformational applicability and the additional conditions of applicability were allowed by making a basic

change to the analysis algorithm. In the old algorithm, as we sketched it, intermediate derivational stages were not completely known; only proper analyses of intermediate trees were required and found. As an alternative to directly determining proper analyses that satisfy an inverse structural index it is possible to parse a continuation string resulting from the application of an inverse transformation up to the sentence symbol, using the augmented CF grammar; the resulting structures can be examined to insure that they satisfy the labelled bracketting structural condition and any other conditions of the forward transformation in question; and that forward transformation can actually be applied to insure the validity of the inverse transformational step. The mechanics of such a step are illustrated by the diagram which appears at the end of the paper.

It must be noted that even though a general labelled bracketting structural condition is allowed, a structural index must still be identified by means of that labelled bracketting, and a structural change is specified only through the use of that structural index, as before.

Chomsky adjunction presents no particular problem because an inverse structural index and inverse structural change may still be mechanically computed.

The enrichment of a transformational grammar to include the use of coordination-reduction rule schemata is discussed in reference <14>. Fortunately, this enrichment has an associated analysis algorithm which is deterministic.

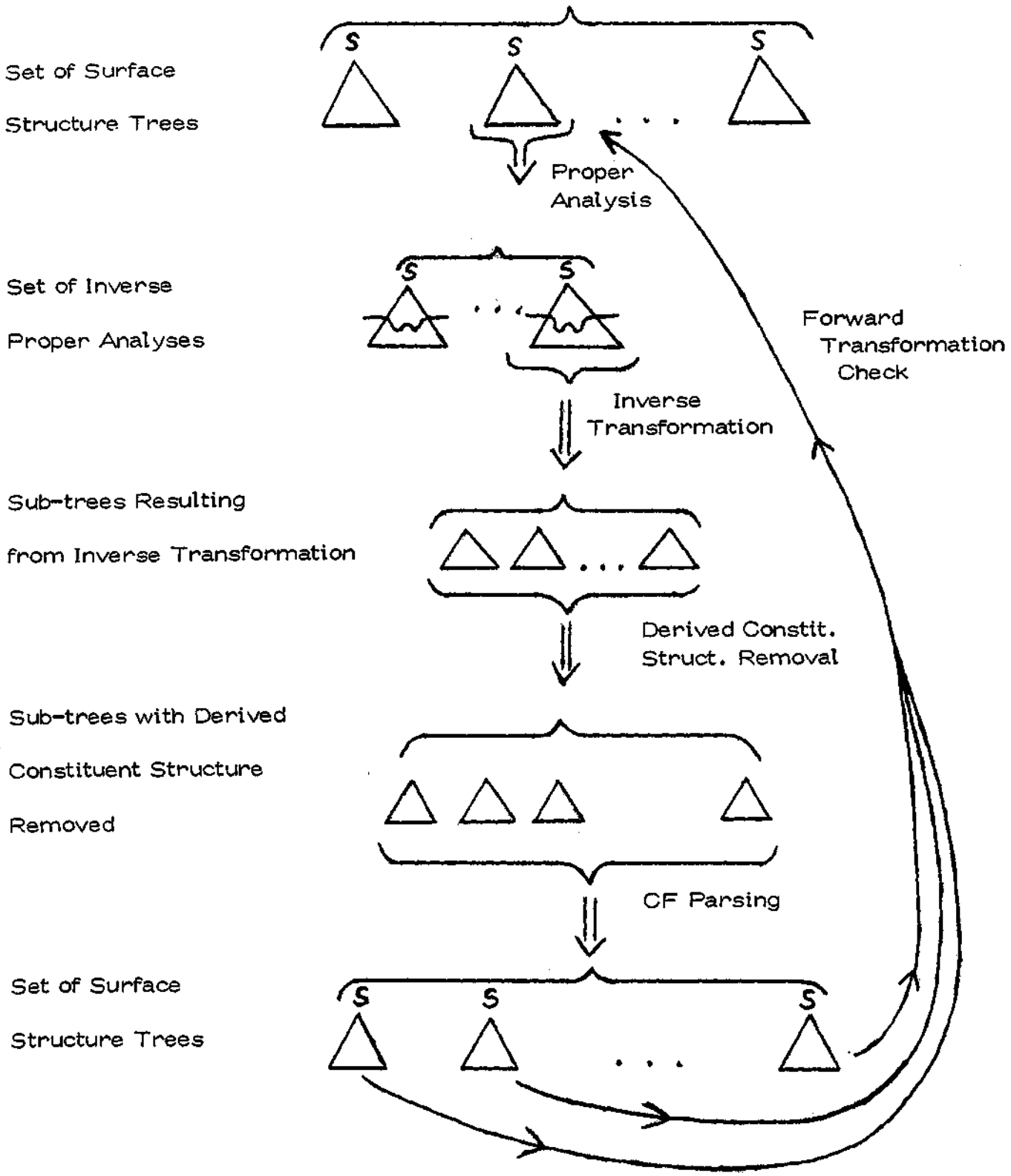
Precyclic and postcyclic components present no new theoretical analysis

problems. They do, however, offer enormous opportunities for the proliferation of spurious continuations. This, in turn, will probably require even more careful modification and tuning of a grammar to keep the analysis time within acceptable bounds.

Finally, the addition of obligatory transformations was fully discussed in reference <10> but was not programmed at that time. A presently existing transformational analysis program incorporates those considerations. This program has not yet been extensively tested and hence must be considered to be in a "debugging" state. For this reason it has not yet been described in the literature. Because it has not yet been tested on any sizeable grammar it is not possible to estimate running times. It is safe to say that the analysis of sentences with respect to a good-sized transformational grammar currently under development at the IBM T. J. Watson Research Center will undoubtedly require careful analysis-dictated modification of that grammar. In addition, the analysis procedure itself may well have to be significantly modified. The principal hope is that by actually performing forward transformational tests on the fly, spurious continuations can be avoided before they exponentially proliferate.

It is clear that at this time it is possible to produce transformational grammars (perhaps not uniformly well-motivated linguistically) which exceed the computational capabilities of the existing program. This, of course, limits the usefulness of the program for investigating the claims inherent in a given grammar (e.g., what structures are assigned to specific sentences?). It remains unknown, however, whether the current generation of transformational

grammars can be modified so as to permit syntactic analysis in a reasonable time for experimental purposes. For the purpose of such applications as question answering systems, information retrieval systems and natural language programming systems this may be less of a problem than the problem of writing a grammar that specifies a sufficiently large subset of English.



THE OPERATION OF A SINGLE INVERSE

TRANSFORMATIONAL STEP

References

<1> Zwicky, A., Friedman, J., Hall, B., and Walker, D. The Mitre syntactic analysis procedure for transformational grammars, Proc. Fall Joint Computer Conference, 1965, Spartan Books, Washington, D. C., pp. 317-326.

<2> Kay, M., Experiments with a powerful parser, Proc. Deuxième Conference International sur le Traitement Automatique des Langues, Grenoble, Aug. 1967, Paper No. 10.

<3> Simmons, R. F., Burger, J. F., and Long, R. E., An approach toward answering English questions from text, Proc. 1966 Fall Joint Computer Conference, 1966, pp. 357-363.

<4> Moyne, J. A., Loveman, D. B., and Tobey, R. G. Cue: A preprocessor system for restricted, natural English, Proc. Symp. on Information Storage and Retrieval, Univ. of Maryland, April 1971, pp. 47-60.

<5> Thome, J., Bratley, P., and Dewar, H. The syntactic analysis of English by machine, Machine Intelligence 3, D. Michie (ed.), American Elsevier, New York, 1968.

<6> Bobrow, D. G. and Fraser, J. B. An augmented state transition network analysis procedure, Proc. Internat. Joint Conf. on Artificial Intelligence, Washington, D. C., 1969, pp. 557-567.

<7> Woods, W. A. Transition network grammars for natural language analysis, Comm. ACM 13 (Oct. 1970), pp. 591-606.

<8> Winograd, T. Procedures as a representation for data in a computer program for understanding natural language, Rept. A1-TR1, Artificial Intelligence Laboratory, MIT, 1971.

<9> Kellogg, C., Burger, J., Diller, T., and Fogt, K. The converse natural language data management system: Current status and plans, Proc. Symp. on Information Storage and Retrieval, Univ. of Maryland, April 1971, pp. 33-46.

<10> Petrick, S. R. A recognition procedure for transformational grammars, Ph.D. thesis, MIT, 1965.

<11> Chomsky, N. Aspects of the theory of syntax, The M.I.T. Press, Cambridge, Mass., 1965.

<12> Petrick, S. R. A program for transformational syntactic analysis, Air Force Camb. Res. Labs. Research Paper No. 278, AFCRL-66-698, Oct. 1966.

<13> Keyser, S. J. and Petrick, S. R. Syntactic analysis, Air Force Camb. Res. Labs. Research Paper No. 324, AFCRL-67-0305, May 1967.

<14> Petrick, S. R., Postal, P. M., and Rosenbaum, P. S. On coordination reduction and sentence analysis, Comm. ACM 12 (April 1969), pp. 223-233.