Reprinted from:
Journal of Cybernetics, 1972, 2, 3, pp. 39–50

A Two-Stage BNF Specification of Natural Language

Naomi Sager†
New York University

Abstract

This paper presents a method of using BNF to specify natural language in such a way that a relatively small grammar of English can express the major grammatical constraints of the language and can be refined without undue proliferation of the rules. The results show that the departures of natural language from a context-free language are of a very restricted kind. The analysis obtained for sentences of the scientific literature is relevant for information processing.

1. Introduction

The field of computer science has benefited theoretically and practically from considering language-like systems, e.g. symbolic programming languages, within the framework of natural language description. In this view, a computer program is a text in a given language. The language is formally specified by a constructive grammar which characterizes the well-formed strings (sentences) of the language and which can be applied algorithmically to provide a structural description of each wellformed input string. This structural description is then the point of departure for further processing, such as translating the sentence (symbolic instruction) into computer code.

A hierarchy of formal grammars, each associated with a particular class of recognition devices, has been intensively investigated in recent years [9]. Within this hierarchy, those formal grammars whose rules are "context-free", or equivalently, those whose rules can be written in Backus Normal Form (BNF) [1], have achieved a special importance. Many of the more powerful programming languages of greatest practical interest (e.g. FORTRAN, ALGOL) can be specified by grammars of this type. At the same time, the form of such grammars is particularly convenient for algorithmic treatment. For example, there is an immediate correspondence between a BNF grammar of a language and the tree-structure representation of a wellformed string in the language. This leads to a natural association of BNF syntax specification with the task of parsing and translating programming languages by syntax-directed compilers, and has led to the development of a number of parsing algorithms for context-free languages [5].

Since the BNF metalanguage has proved to be both simple and powerful, a natural question is to investigate the limits of its power and to look into simple extensions that may increase its power. A test case of prime interest is natural language. Is BNF powerful enough to specify a natural language in a small or moderate number of rules? And if so, how should it be applied? The latter is an important question because, while it may be possible

†Requests for reprints should be sent to Professor Naomi Sager, New York University, Linguistic String Project, 2 Washington Square Village, New York, New York 10012.

to show that natural language can be specified by a straightforward expansion of context-free rules in order to cover all cases of dependencies, the magnitude of the resulting grammar and the lack of perspicacity would rob it of all practical interest. A partial study of such an expansion was made in [14].

This paper presents a method of using BNF to specify natural language in such a way that: 1) BNF is the only tool required, used in two stages of language specification; 2) the resulting grammar is small, easily comprehensible, and capable of refinement without undue proliferation of the rules; 3) a computer program which uses a grammar of English constructed in this way has been shown to be capable of correctly analyzing uncontrived sentences of the scientific literature [13]. The method is as follows: we take a context-free specification of English sentence structures which by itself is insufficient for recognizing wellformedness but has certain desirable and intuitive properties which we can associate naturally with basic sentence structures. We then show that these context-free structures fall into several main syntactic types which are such that a small set of functions defined on the types suffice for computing all the remaining constraints necessary for recognizing sentence-wellformedness. The statement of these constraints in terms of the functions is made in a context-free language whose grammar constitutes the second stage of BNF specification.

The existence of a typology of sentence structures in terms of which a small set of functions can express all grammatical constraints shows that the departures of natural language from a context-free language are of a very restricted kind. It is worth noting here that many contextual conditions are only superficially context-sensitive. For example, A. Joshi, in [10], shows that contextual information, associated with context-sensitive rules and certain transformational rules, when used for analysis gives no more power than a context-free grammar. This result is a generalization of a result of S. Peters [11].

Joshi's method is to define local transformations, which are the rewriting of a single category symbol restricted by Boolean combinations of proper analysis and domination predicates, and to show that these local transformations when used for analysis have no more power than a context-free grammar; yet local transformations suffice for stating many of the constraints needed for natural language. A certain type of grammar which has many local transformations built into its very formulation is called string grammar [8]. It is this type of grammar (in a form slightly different from Joshi's) which is used in this paper.

2. Linguistic Strings

The grammar of English is divided into two parts, a part which is naturally, or conveniently, context-free,[1] and a set of restrictions on the context-free productions. The context-free productions are written as a set of BNF definitions which terminate in atomic symbols ⟨*X⟩ representing major word classes (e.g. ⟨*N⟩ noun, ⟨*V⟩ infinitive verb, ⟨*tV⟩ tensed verb, ⟨*Pro⟩ pronoun, ⟨*A⟩ adjective, ⟨*D⟩ adverb). We distinguish two major types of these definitions: *linguistic strings* and *adjunct sets*. A sample definition of each type, taken from the implemented string grammar of English of [12] and [13], is shown in Fig. 1.

A linguistic string is a single option definition which consists of a squence of "required" elements (e.g. ⟨SUBJECT⟩, ⟨VERB⟩, ⟨OBJECT⟩ in the ⟨ASSERTION⟩ string in Fig. 1) in

[1] While the form is context-free, it should be noted that the linguistic content of the definitions is not that of the familiar phrase structure grammar found in Chomsky [3].

```
Linguistic String

<ASSERTION> ::= <SA><SUBJECT><SA><VERB><SA><OBJECT><RV><SA>.


Adjunct Set

<SA>*R ::= <*INT>|<DSTG><RD>|<PN>|<PA>|<NSTGT>|<RSUBJ>|
           <RNSUBJ>|<LCS><CSSTG>|-<OBJBESA>|-<SOBJBESA>|
           <VINGO>|<VENPASS>|<SAWH>|<TOVO>|<NVSA>.
```

FIG. 1. Sample BNF definitions from the grammar of English.

direct correspondence with an elementary word string in the sentence,[2] interspersed with elements of the adjunct set type. An adjunct set definition, e.g. ⟨SA⟩*R in Fig. 1, is a multi-option definition whose options are a particular subset of the linguistic strings, those which have optional occurrence (i.e. as modifier strings) at stated points in other linguistic strings. E.g. the set ⟨SA⟩ of adjunct strings are modifiers of a whole string or sentence and can occur in an ⟨ASSERTION⟩ at any of the four ⟨SA⟩ positions named in ⟨ASSERTION⟩. This set includes interjections ⟨*INT⟩, e.g. *however, moreover,* adverbial strings ⟨DSTG⟩ ⟨RD⟩, e.g. *clearly, quite clearly, clearly enough,* prepositional phrases ⟨PN⟩, e.g. *in this case.* ⟨SA⟩*R is an abbreviation for a recursive definition of SA, and indicates that the adjunction of ⟨SA⟩-strings is repeatable: *However, clearly in this case . . . .*

In addition to the main types of definitions there are auxiliary definitions which group together linguistically related classes of strings or string-segments, such as those which occur in the same element-position, or adjunct strings which constitute a convenient subset of a given adjunct set. Thus, the definition ⟨SUBJECT⟩ : : = ⟨NSTG⟩ | ⟨VINGSTG⟩ | ⟨SN⟩ | ⟨*NULLWH⟩ names subsets of positional variants for the subject position; and ⟨SA⟩*R includes several options which are subsets of adjunct strings, e.g. ⟨LCS⟩ ⟨CSSTG⟩, the set of subordinate conjunction strings with their possible left adjuncts (*just because it rained*). Also, the left and right adjunct strings of a given atomic class ⟨*X⟩ which occur adjacent to ⟨*X⟩ are conveniently treated by a standard definition: ⟨LXR⟩ : : = ⟨LX⟩ ⟨*X⟩ ⟨RX⟩, where ⟨LX⟩ | ⟨RX⟩ is the set of left/right adjunct strings of ⟨*X⟩. These auxiliary definitions are not essential to the definitions formulating the grammar. The types of interest are the linguistic strings and the adjunct sets, and the lists of atomic symbols and their attributes.

To give an idea of the size of the BNF part of the implemented English grammar, there are currently about 180 definitions[3] of which about 100 are linguistic string definitions, 25 are adjunct set definitions, and about 20 are the LXR type. There are 24 atomic symbols and about 120 attributes.

The BNF definitions used in analyzing a particular sentence can be displayed in a tree diagram of the sentence. For example, the parse tree for the simple sentence *They*

[2] An elementary word string in a sentence S is a (possibly interrupted) word sequence S′ in S which has the property that if an arbitrary member of S′ is excised, then all the members of S′ must also be excised if S, thus reduced, is to remain a wellformed sentence. This procedural definition cannot of course be applied to one sentence alone. In practice, a controlled excision procedure is used to obtain tentative classes of elementary word-strings having the same word-class form and occurring in similar positions in respect to other word class sequences which are similarly obtained. Families of these word-class sequences become the linguistic strings of the BNF grammar.

[3] These do not include conjunctional strings, which are treated in somewhat special way.

experiment (Fig. 2) shows that a series of choices among positional variants of ⟨SUBJECT⟩ leads to the atomic node ⟨*PRO⟩, which is satisfied by the sentence word *they;* and the choices among positional variants of ⟨VERB⟩ lead to ⟨*TV⟩, which is satisfied by *experiment* in the sentence. The words *they experiment* are an elementary word string in the sentence; the elements ⟨SUBJECT⟩ ⟨VERB⟩, (and ⟨OBJECT⟩ which here is satisfied by ⟨*NULLOBJ⟩[4]) are the corresponding required elements in the linguistic string of the BNF grammar.

The second part of the grammar is a set of restrictions (about 180 in the present English grammar). The need for restrictions arises as follows. The BNF part of the grammar defines a set of word-class sequences, called sentence formulas, which correspond to the wellformed sentences of the language, e.g. ⟨*Pro⟩ ⟨*TV⟩ ⟨*D⟩ (*They experiment frequently*). However, as is well known of natural language, not every combination of word values which satisfies a sentence formula yields a wellformed sentence of the language; e.g. *Them experiment frequently* also satisfies the sentence formula ⟨*Pro⟩ ⟨*TV⟩ ⟨*D⟩.

Generally speaking what restrictions do is to check that the sentence-words which correspond to a given instance of a wellformed word-class sequence have compatible attributes (i.e. compatible word subclassifications). For example, a restriction associated with the subject requires that the sentence word associated with ⟨*Pro⟩ in ⟨LPROR⟩ under ⟨SUBJECT⟩ not have the attribute ACCUSATIVE (except in stated strings, as in the object string of *want* in *We want him to go,* where *him* is identified as the subject of *go*). Further examples of restrictions are those that check for number agreement (e.g. *They experiment,* ∄ *They experiments*), and those that test for the appropriate predicates for nominalized sentences: *That he is here surprises us,* ∄ *That he is here surrounds us.*

3. Locating Relations

The crucial part of the grammar is clearly the restrictions; and what is unique is not the statements of the particular grammatical constraints but the algorithmic ability to apply

[4] Intransitive verbs do not require an object, but to regularize the syntax we posit that a null object satisfies the OBJECT position in this case.

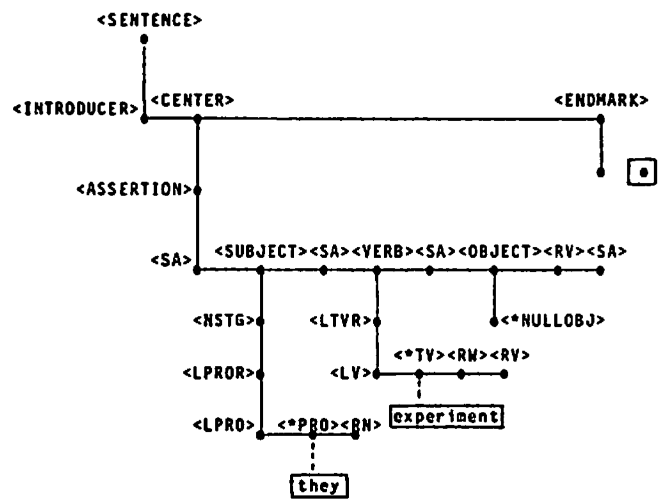

FIG. 2. Parse tree for the sentence *They experiment.*

them to a sentence. The problem of computable grammar in general is not so much one of stating grammatical constraints formally as of locating the arguments of the constraints in any sentence in which they apply. The solution here is to define a small set of "locating relations" which are such that if any two words in a sentence have any grammatical constraint operating between them, they will be related by one of the locating relations or a particular product of them. This leads to a hypothesis–namely that grammatical constraints not already expressed in terms of these locating relations can be so expressed.

The locating relations are defined in terms of the types of definitions noted above, and the type "atomic" of the BNF grammar. Their computable form can thus be stated as operations on types of tree nodes, referred to a generalized tree module. The generalized tree module (Fig. 3) has as its root a node of type "string," and extends down each branch to the first node which is either of type "atomic" or of type "string." The bounding string nodes are each the root of a similar module. The parse tree for any given sentence is composed entirely of such modules.

The essential locating relations[5] are listed in Fig. 4, illustrated by statements applying to the tree module of Fig. 3. A locating relation $\alpha R \beta$ holds between two nodes α, β, if an associated locating routine R′ can be executed starting at α and ending at β. For example, the CORE routine descends the branch originating at α, searching each level until it finds a node which is of the type "atomic" or "string," but it may not pass through nodes of the type "adjunct set" ("a" in Fig. 3); it then checks that the node reached is β. β need not be named; the CORE routine will assign the status of "CORE" to the node reached by executing the routine.

Some important points about the tree modules and locating relations are as follows: The fact that the locating relations are defined with respect to types of grammatical variables instead of the individual grammatical variables overcomes the practical difficulty due to the detailed differences between different sentences in which the same constraint applies. For example, the constraint that the subject and verb agree in number can be stated without regard to whether the sentence in hand has a noun or pronoun or a whole string as subject, by referring only to the core of the subject.

There clearly is no problem of "action at a distance" since the relations of words within one string module are independent of how many sentence words are subsumed by string modules originating at its boundaries. That such a constraint as subject-verb agreement is

[5] All restrictions could be written using only the locating relations called essential. For convenience, other locating relations have been defined.

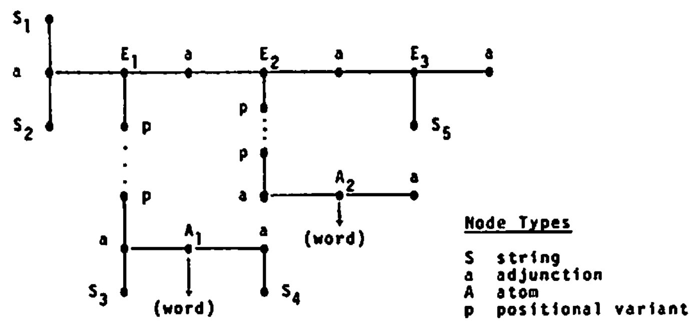

FIG. 3. Tree module.

ELEMENT	S_1 has element E_1
	E_1 is element of S_1
COELEMENT	E_1 has coelement E_2, E_3
CORE (atom)	E_1 has core A_1
CORE (string)	E_1 has core S_5
LEFT ADJUNCT	A_1 has left adjunct S_3
RIGHT ADJUNCT	A_1 has right adjunct S_4
SENTENCE ADJUNCT	S_1 has pre-E_1 sentence adjunct S_2
HOST STRING	S_1 is the host string of S_2
HOST	A_1 is the host of $S_3 . S_4$

FIG. 4. Essential locating relations (examples refer to Fig. 3). (Under CORE (string): E_1 is error for E_3.)

applied to the proper noun and verb in the sentence (regardless of how many words separate them) is insured by their being the cores of elements in the same linguistic string. Lastly, an important fact is that except for a few cases which will be described presently, grammatical constraints operate within one module or two adjacent modules related by adjunction. This means that constraints are largely local.

The grammatical constraints which apply to words connected by a product of locating relations which reaches beyond two adjacent modules are quite special and usually involve "zeroing".[6] Each connecting path (routine) iterates a particular small set of locating relations. The main cases are illustrated by the sets of sentences and sentence fragments below.

The routines which handle each case are named, but not further discussed here.

(1) Routines: DEEPEST VERB/ULTIMATE SUBJECT
John will leave on Sunday.
John plans to try to arrange to leave on Sunday.
John is known to have tried to arrange to leave on Sunday.
(subject of *leave* is *John*)
the man whom I met ( )
the man whom I plan to try to arrange to meet ( )
(object of *met* and *meet* is *man*)

(2) Routines: DEEPEST S/ULTIMATE OBJECT
He will go and she can ( ) too.
He will go and she thinks you said that she can ( ) too.
He will go and she thinks you said you were willing to try to get John to agree that she can ( ) too.
(*go* is zeroed after *can*)

(3) Routines: DEEPEST PN/ULTIMATE HOST
the book he wrote the outline of ( )
the book he wrote the outline of the abstract of ( )
the book he wrote the outline of the abstract of the first chapter of ( )
(the object of *of* is *book*)

[6] "Zeroing" refers to implicit occurrences of words which can be reconstructed in a regular way: *He left and she too (left).*

It should be noted finally that the locating relations are not a device which just happens to work. First of all, it may be assumed that a method effective for computing language structure would necessarily reflect the relevant properties of language itself. In this case the tree module and the locating relations are based on a fundamental property of language. Since there is no *a priori* distance defined for natural language (analagous to bars and rests in music), it follows that if language can have a constructive grammar then there must exist some characterization of the wellformed sentences which is based on purely contiguous relations [7]. In this characterization, all grammatical events can be stated as occurring between particular elements at a point in the constructive history, or derivation, of the sentence when these elements are adjacent. Linguistic string are units of syntactic description for which this property of contiguity of operations holds. Inside the string the contiguity consists in placing element next to element. Among strings, the contiguity consists in adjoining a string to the left or right or at an interior point of a host string. Since the tree modules represent linguistic strings, they preserve the property of contiguity of operations.

4. Restriction Structure and the Restriction Language

Having established a means of locating the arguments of a grammatical constraint, we can now state what a restriction is: A restriction is a statement of grammatical requirements at positions in the sentence which are related by one of the defined locating relations, or by a product of such relations. Since the grammatical constraints usually involve subclass checks, the computable form of a restriction statement is typically a sequence of locating routines followed by attribute-checking routines applied at the end points. A complete restriction may consist of logical combinations of such statements.

To illustrate, consider the restriction which checks the case of a subject pronoun: *They experiment*, ∄ *Them experiment*. A simplified version of the restriction might simply require that a subject pronoun not be accusative.[7] The restriction test would consist of the operations: START AT SUBJECT; GO TO CORE; CHECK THAT CORE DOES NOT HAVE ATTRIBUTE ACCUSATIVE. Figure 5 shows the sequence of routine-calls for this restriction. Below the sequence of routine-calls (labeled OBJECT RECORD in Fig. 5) is a trace of the restriction as it is being executed in the sentence *They experiment*. Each routine gives rise to an amount of more detailed code which carries the burden of the generality built into the locating relations.

Because restrictions are built in a regular way we can specify which sequences of routines constitute wellformed restrictions. This syntax specification (along with that of BNF itself) defines a metalanguage for natural language. The metalanguage has been called the Restriction Language.

[7] The actual restriction WPOS5 accepts an accusative pronoun in subject position in certain object strings: ⟨FORTOVO⟩ we prefer *for them to go*, ⟨NTOVO⟩ we want *them to go*, ⟨SOBJBE⟩ we thought *them responsible*, ⟨SASOBJBE⟩ we view *them as responsible*, ⟨SVO⟩ we hold *them responsible*, ⟨SVEN⟩ we saw *them refused admission*, ⟨STOVO-N⟩ he has *them to provide for*. The source text of the restriction reads:

```
WPOS5 = IN LPROR AFTER PRO: IF LPROR IS OCCURRING IN SUBJECT X1, THEN BOTH $NOM
        AND $ACC ARE TRUE.
$NOM  = IF PRO IS NOMINATIVE THEN IT IS NOT THE CASE THAT $INSN IS TRUE.
$ACC  = IF PRO IS ACCUSATIVE THEN $INSN IS TRUE.
$INSN = AT X1, SUBJECT IS AN ELEMENT OF FORTOVO OR NTOVO OR SOBJBE OR SASOBJBE OR
        SVO OR SVEN OR STOVO-N.
```

```
WTEST = IN SUBJECT: CORE IS NOT ACCUSATIVE .

OBJECT RECORD:
      ( WTEST      [ 17408 ]  , EXECUTE     [ ( CORERT  ) ]  , NOT     [
              ( ATTRB    [ ( ACCUSATIVE ) ] ) ] )


RESTRICTION BEING EXECUTED IS WTEST     IN SUBJECT
 ( CORERT:  ORPTH  ( IS ( ATOM  ) - DNTRN ( ATOM ) ( 0 ) ( ADJSET )  +  )  +
                                     )  +  NOT ( ATTRB ( ACCUSA )  -  )  +
```

FIG. 5. Example of a restriction.

For reasons of user-convenience and for its linguistic interest each restriction is written, not as a sequence of routine calls directly, but as a sentence of English whose parts are in 1 : 1 correspondence with the routine calls of the executable restrictions.[8] The metalanguage is then a subset of English (one which constitutes a sublanguage of English[9]) sufficient to state the grammar of English. This sublanguage is context-free, and its grammar, the Restriction Language Syntax (RLS), is contained in the second set of BNF definitions referred to in the introduction.

The RLS consists of about 175 BNF definitions (of which 75 are conjunctional expansions). A portion of the RLS used in parsing, WTEST = IN SUBJECT: CORE IS NOT ACCUSATIVE, is shown in Fig. 6. Only a few of the options of ⟨DEFTYP⟩, ⟨RTEST⟩, ⟨PREDICATE⟩ and ⟨BEPRED⟩ are shown. The non-null options used in WTEST are underlined, and the parse tree for the restriction sentence is shown in Fig. 7.

The basic construct in restrictions is the option ⟨STATEMENT⟩ of ⟨RTEST⟩; the other options of ⟨RTEST⟩ are logical tests whose arguments are RTEST's. A ⟨STATEMENT⟩ is either a preset phrase (e.g. "THE RARE SWITCH IS ON"), called a ⟨MONOSENT⟩, or a restriction subject ⟨RSUBJ⟩ followed by a ⟨PREDICATE⟩. ⟨RSUBJ⟩ can be any one of the locating relations of the restriction language or one of the atomic types ⟨*NODE⟩, ⟨*ATTRIBUTE⟩, or a register ⟨*REG⟩. Examples of restriction statements using the different values of ⟨RSUBJ⟩ are shown in Fig. 8. The restriction statements in Fig. 8 also illustrate most of the options of ⟨PREDICATE⟩; for reasons of space these were not listed in Fig. 6.

Among the elements of the definiens in many definitions are calls on code-generating routines. (These are preceded by "."; the notation has been adapted from Cocke and Schwartz [4].) As a restriction written in the Restriction Language is parsed, the corresponding code is generated; i.e. sentences in the Restriction Language are translated into code by a syntax driven compiler. The RLS is also compiled by a syntax-driven compiler, so that the entire system is a compiler-compiler, the work of Ralph Grishman described in [6].

[8] The sentences of the Restriction Language are sentences of English if the names of the grammatical elements in the computable English grammar are accepted as names, e.g. NTOVO, and if *a* and *the* are inserted before count nouns, such as SUBJECT. In the Restriction Language, for brevity, we have permitted the optional dropping of articles.

[9] This subset of English sentences constitutes a sublanguage of English [ref. 7, §5.9] in that it is closed under those operations of English grammar which are included in the subset, e.g. the restricted operations of conjunction and certain paraphrastic transformations, e.g. A "has-R" B ↔ B "is-R-of" A, for R = element, coelement, core, host, etc. It is also assumed that BNF definitions can be read as sentences of English; e.g. "The syntactic type ⟨ASSERTION⟩ is defined to be the sequence consisting of the syntactic type ⟨SA⟩ followed by the syntactic type ⟨SUBJECT⟩ followed by. . .".

```
<RESTSENT>     ::=  <INITIALIZER><RTEST>+.+  .
<INITIALIZER>::=  <INDEF> ( <AFTERL>/<REOPT>/<NULL>.HOUSE.(0) ) +:+ .
<INDEF>        ::=   IN,PUSH <DEFTYP><CDEFTYP*> .GENLIST.(1) .
<DEFTYP>       ::=  <*DEF> .LOC.(1) / ...
<STATEMENT>    ::=  <MONOSENT>/<RSUBJ><PREDICATE>  .
<RSUBJ>        ::=  <*NODE><REGST>.ROUTX.(STARTAT).ISRP/
                    <*ATTRIBUTE><REGST>.ROUTX.(ISIT).ISREG/
                    <*REG .LOOK>/<ELEMX>/<COELX>/<CORE>/
                    <LADJOF>/<RADJOF>/<PREPOSTSA>/<HOST>/
                    <HOSTRING>/<ULTIML>/<NTHL>/<PRESENT>/
                    <PREFOL>/<VALOF>/<IMMEDSTG>/<SENTWD>/
                    <NODEWD>.
<CORE>         ::=     CORE  <REGST>[OF  OPCORE  ] .ROUTE
                    .(CORERT).ISREG  .
<PREDICATE>    ::=  IS (( <BEPRED>/OCCURRING <OCCPRED> )<REGST>
                    .ISREG /
                    NOT .MARK(<BEPRED>/OCCURRING<OCCPRED> )) .PUT
                    .NOT / ...
<BEPRED>       ::=  <ATOMAT>/<ATTRB>/ ...
<ATTRB>        ::=  <*ATTRIBUTE> .MARK .GENSYM.(6) .ATTRBX
                    <ATT*>(<ATRAOR>/<NULL>.UNMARK) .
```

FIG. 6. A portion of the RLS.

The power of this simply-structured metalanguage can be judged from the fact that a grammar of moderate size is effective in parsing scientific texts (not perfectly, but with good performance, [2]). As an example, the output parse for a sentence from a

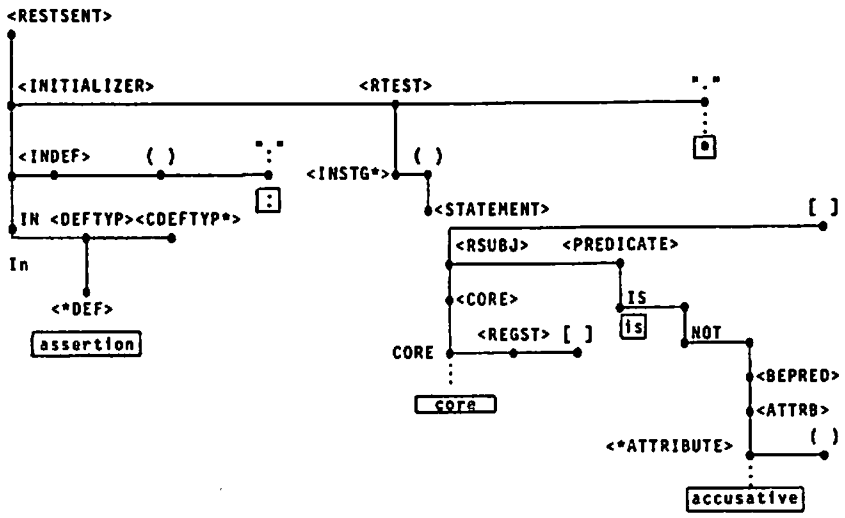

FIG. 7. Parse tree of a restriction sentence.

<*NODE>	THE OBJECT HAS THE VALUE NULLOBJ.
<*ATTRIBUTE>	ASENT1 HAS THE ATTRIBUTE ATHAT.
<*REG>	X1 IS SINGULAR.
<ELEMX>	THE ELEMENT SUBJECT OF ASSERTION HAS THE CORE N.
<COELX>	THE COELEMENT VERB HAS THE CORE TV.
<CORE>	THE CORE OF THE COELEMENT OBJECT IS N:SINGULAR.
<LADJOF>	THE LEFT ADJUNCT OF N IS EMPTY.
<RADJOF>	THE RIGHT ADJUNCT OF TV HAS THE CORE D.
<PREPOSTSA>	THE PRE-OBJECT SENTENCE ADJUNCT OF THE IMMEDIATE STRING IS OF THE TYPE CENTERLIKE-STRING.
<HOST>	THE HOST OF PN IS OCCURRING AS OBJECT.
<HOSTRING>	THE HOST-STRING OF SUB1 IS ASSERTION.
<ULTIML>	THE ULTIMATE SUBJECT IS AN ELEMENT OF AN N-OMITTING-WH-STRING.
<NTHL>	THE SECOND ELEMENT OF PN HAS THE CORE PRO.
<PRESENT>	THE PRESENT ELEMENT IS CONJOINED BY AN AND-STRING.
<PREFOL>	THE PREVIOUS ELEMENT IS STRING INITIAL.
<VALOF>	THE VALUE OF THE CENTER IS ASSERTION OR QUESTION.
<IMMEDSTG>	THE IMMEDIATE STRING OF SUBJECT IS ASSERTION.
<SENTWD>	THE CURRENT WORD IS N OR VING.
<NODEWD>	THE UPCOMING NODE WORD IS NOT +OF+

FIG. 8. Examples of ⟨RSUBJ⟩.

pharmacology text[10] is shown in Fig. 9. The output does not show the full BNF tree. A line of output is assigned to each linguistic string in the sentence; below each element E of the string the core of E is written directly: if the core is atomic the sentence word corresponding to the atomic node is written; if the core is a string, the output line number of that string appears under E. Thus the output displays the decomposition of the sentence into its component elementary word strings. These units are basic both to the effectiveness of the grammatical formulation and to the informational content of the sentence.

The excision method of defining grammatical strings (*cf.* fn. 2) is based on the close relation which exists between the words in an elementary word string in the sentence, and secondarily on the relation between contiguous strings. These inter- and intra-string relations are preserved in the BNF definitions and their types, and in the locating relations which operate in terms of those types. The elementary word strings, which are so closely tied together that no one word of the string can exist in the sentence without the others, are also the most closely bound information units of the sentence. It is for this reason that the string decompositions can be used as the first step in a mechanical breakdown of the sentence into its informational components. However, the amount of ambiguity and complexity of this process is such that it can only be carried out in a practical way in more restricted subject-matter areas (e.g. a subfield of science) where word-usage is more precise and where additional constraints on wellformedness (over and above those for English as a whole) can be stated. Such an application in the field of pharmacology is in progress. It has been found that a more restricted grammar for the pharmacology sentences, viewed as a sublanguage,

[10] Glynn, I. M., "The Action of Cardiac Glycosides on Ion Movements," *Pharm. Review 16*, 1964.

```
THE EFFECTS CANNOT BE EXPLAINED SATISFACTORILY BY SUPPOSING
THAT CARDIAC GLYCOSIDES SIMPLY DISCONNECT THE ENERGY SUPPLY .
PARSE 1

1.  SENTENCE  = INTRODUCER CENTER ENDMARK
                           2.

2.  ASSERTION = SA SUBJECT      CA2 SA VERB      CA4 SA OBJECT
                   3. EFFECTS       CANNOT BE            4.
                CA6 RV              SA
                   SATISFACTORILY    5.

3.  LN        = TPOS QPOS APOS NSPOS NPOS
                THE

4.  VENPASS   = LVSA LVENR      SA PASSOBJ RV SA
                    EXPLAINED

5.  SUB4      = CS4 VINGSTG
                BY    6.

6.  NSVINGO   = TPOS VINGO
                      7.

7.  VINGO     = LVSA LVINGR    SA OBJECT RV SA
                    SUPPOSING      8.

8.  THATS     = +THAT+ ASSERTION
                THAT     9.

9.  ASSERTION = SA SUBJECT          CA2 SA VERB             CA4 SA
                 10. GLYCOSIDES          SIMPLY DISCONNECT
                OBJECT       CA6 RV SA
                 11. SUPPLY

10. LN        = TPOS QPOS APOS     NSPOS NPOS
                          CARDIAC

11. LN        = TPOS QPOS APOS NSPOS NPOS
                THE                    ENERGY

1 SUCCESSFUL PARSE  COMPLETED
NO MORE PARSES
```

FIG. 9. Output parse.

can be stated, and that this sublanguage grammar has considerable semantic sharpness. A particularly simple illustration of the point can be seen by considering the RLS, the grammar of the sublanguage of English in which computable grammars of natural languages can be written. The RLS types (e.g. CORE, HOST) and the contents of the major construct ⟨STATEMENT⟩ are a concise and precise summary of the contents of the small subfield, computable string grammar.

References

[1] J. W. Backus, "The syntax and semantics of the proposed international algebraic language of the Zurich ACMGAMM conference," *Proc. International Conf. on Information Processing*, UNESCO, pp. 125-132, 1960.

[2] B. Bookchin, "Computer outputs for sentence decomposition of scientific texts," in *String Program Reports 3*, Linguistic String Project, New York University, 1968.

[3] N. Chomsky, *Syntactic Structures*. The Hague: Mouton & Co., 1957.

[4] J. Cocke and J. T. Schwartz, *Programming Languages and Their Compilers*. New York: Courant Institute of Mathematical Sciences, New York University, 1969.

[5] J. Feldman and D. Gries, "Translator Writing Systems," *CACM 11*, pp. 77-113, 1968.

[6] R Grishman, "The Implementation of the String Parser," in *Courant Computer Science Symposium 8: Natural Language Processing*, R. Rustin, Ed. New York: Algorithmics Press, 1973

[7] Z. S. Harris, *Mathematical Structures of Language*. New York: John Wiley Interscience Tracts in Pure and Applied Mathematics 21, 1968.

[8] Z. S. Harris, *String Analysis of Sentence Structure*. The Hague: Mouton & Co., 1962.

[9] J. E. Hopcroft and J. D. Ullman, *Formal Languages and Their Relation to Automata.* Reading, Mass.: Addison-Wesley, 1969.
[10] A. K. Joshi, "A brief survey of some mathematical results relevant to natural language processing," in *Courant Computer Science Symposium 8: Natural Language Processing,* R. Rustin, Ed. New York: Algorithmics Press, 1973.
[11] S. Peters, in *Proc. 1969 ACM Symposium on Theory of Computing,* May 1969.
[12] N. Sager, *A Formal Grammar of English and its Computer Applications.* New York: Gordon & Breach Series on Mathematics and its Applications (in preparation).
[13] N. Sager "The string parser for scientific literature," in *Courant Computer Science Symposium 8: Natural Language Processing,* R. Rustin, Ed. New York: Algorithmics Press, 1973.
[14] M. Salkoff and N. Sager, "The elimination of grammatical restrictions in a string grammar of English," 2ème Conférence International sur le Traitement Automatique des Langues, Grenoble, August 1967.

This work was supported in part by Research Grants R01-LM 00720-01, -02 from the National Library of Medicine, and in part by Research Grants GS 2462 and GS27925 from the National Science Foundation, Division of Social Sciences. I wish to thank Aravind K. Joshi and Jerry Hobbs for helpful comments on this paper.

Received May 1972