A Left-to-Right Generative Grammar of French [*]

A thesis presented

by

David Allen Dinneen

to

The Department of Linguistics

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

in the subject of

Linguistics

Harvard University

Cambridge, Massachusetts

(December, 1962)

[*] This work was supported in part by the National Science Foundation, and in part by the U.S. Army (Signal Corps), the U.S. Air Force (Office of Scientific Research, Air Research and Development Command), and the U.S. Navy (Office of Naval Research).

Acknowledgments

I wish to express my gratitude to John Bennett, Carol Bosche, Muriel Kannel, and Kenneth Knowlton for their help in solving innumerable programming problems, to René Gsell and Edward S. Klima for giving me their time and advice in discussing various problems of French syntax, to the entire Mechanical Translation Group at M.I.T. and the *Centre d'Etudes pour la Traduction Automatique* at Grenoble, directed by Bernard Vauquois, for their encouragement and help, and, of course, to Victor H. Yngve for his constant encouragement and his sympathetic and helpful direction.

Table of Contents

Chapters

Appendices

Introduction

The grammar of French described in the following chapters differs from most other grammars of French in that it synthesizes, rather than analyzes, French sentences. Furthermore, most analytic grammars treat individual syntactic problems separately, without consistently placing them in the context of a sentence, whereas the grammar to be described below is always concerned with complete sentences. By a series of operations, which are described by the rules of the grammar, the final constituents (words) of a sentence are developed from higher level syntactic constituents, which were in turn developed from constituents at still higher levels in the syntactic hierarchy, and so on up to the highest level, "SENTENCE". This type of grammar is called a generative grammar, [1] since it is designed to generate sentences in the language being described. The process by which an upper level syntactic unit produces two (or more) lower level units which are the immediate constituents [2] of this higher unit, is called expansion.

A typical rule of a generative grammar will take a syntactic unit, for example, SENTENCE, and expand it into its immediate constituents, for example, SUBJECT plus PREDICATE. It then continues to expand each unit until it reaches a unit (a word) which cannot be further expanded. The types of rules, the ways in which they are applied, and the order in which they are applied, vary depending upon the type of generative grammar, but the results are, in general, the same: There should be produced a string of words forming a sentence in the language, plus a description of the structure of this sentence in the form either of a list of the rules which produced the sentence or a list of the syntactic structures of which the sentence is composed.

In this generative grammar of French the syntactic structures and words are generated in their normal order: after a constituent is expanded into its immediate constituents, the leftmost constituent is operated on first and the following constituent is not operated on until a terminal symbol, or word, has been generated. There is no reordering of the words of the sentence after the sentence has been generated. Since the operations of this grammar proceed from left to right, from the beginning to the end of the sentence, it is commonly referred to as a "left-to-right" grammar. A complete discussion of "left-to-right" grammars and a description of a left-to-right grammar of English can be found in Professor V.H. Yngve's article, "A Model and an Hypothesis for Language Structure". [3]

Since a generative grammar is intended to generate grammatically correct sentences, the most effective way to test the correctness of the grammar is to write it in a form in which it can be submitted to a computer which will then rigorously follow the rules and generate sentences which can be examined to see if they are grammatically correct. I have written such a grammar in a special, stylized language which permits the linguist-researcher to direct the computer to execute complex logical operations. This programming language, known as COMIT, [4] was developed primarily to aid research workers in linguistics and mechanical translation, though it has been used by research workers in many other fields. By using it, I have been able, with comparative ease, to tell the computer what operations to execute, under various conditions, in order to generate French sentences. The program in Appendix V is a list of the rules of the grammar, written in the COMIT language. When this program, in the form of impressions on a magnetic tape, is submitted to a computer, the computer will generate a set of sentences, printed out in normal form (i.e., just the French words in their correct final form) with each sentence optionally followed by a string of constituent names which can be used to trace its syntactic structure. For a list of sentences generated by the IBM 7090 Computer at the Massachusetts Institute of Technology with this program, see Appendix VI.

Ideally, a generative grammar will produce all and only the sentences of the language. The syntactic structures and lexical items are chosen freely (by a "flip of the coin") by the computer, except for restrictions that arise during the development of the sentence. This process is known as random generation. If a grammar is complete, therefore, and if the machine were allowed to continue this random generation *ad infinitum*, it could theoretically generate all of the sentences of the language. In its present form, this grammar of French will generate several types of sentences comprising many different syntactic forms, but it is not sufficiently complete to produce all of the sentences of French. However, it provides a framework within which further syntactic forms can be added without requiring major changes in the general structure of the grammar.

The programmed grammar has three purposes: First, it represents the final component of a translation process passing from some input language to French. Second, it provides material which is useful in the development of an analytic grammar of French (usually referred to in mechanical translation literature as a "recognition routine"). Third, it serves as a tool for the study of French syntax.

The type of translation process referred to in the statement of the first purpose is the following: 1) A given sentence of the input language is analyzed. One result of this analysis is a description of the syntactic structure of the sentence. 2) The set of relationships between words that is expressed by the syntactic structure of the input sentence is translated into an equivalent set of relationships between the words to be used in the output sentence, by stating the equivalent syntactic structure of the output sentence. 3) Taking the results of the translation of the syntactic structure and the translation of the semantic units, the translator (human or mechanical) writes out the sentence in the output language. [5] I believe that a generative grammar of French will be able to fulfill the third function in a mechanical translation scheme of the type outlined above. The statement of the equivalent syntactic structure (Step 2) would be in the form of a set of directions requiring the choice of particular rules within the generative grammar. By applying these rules in their proper order, the machine would generate the required sentence.

The usefulness of this or any other generative grammar as an aid in the development of an analytic grammar (my second purpose mentioned above) remains to be demonstrated.. However, I am hopeful that the research involved in writing this grammar, and the sentences produced by it, will provide me and other research workers in mechanical translation with material that will help us to write a recognition routine for French sentences. In correcting the grammatical errors in the sentences generated at various stages of the programmed grammar, I have constantly been forced to clarify grammar rules and make them more detailed, thereby developing a grammar which is extremely explicit in its statement of the structure of each sentence generated. I do not suggest that an analytic grammar can be written by simply inverting the rules of a generative grammar, but I do believe that these highly explicit rules, which are necessarily consistent with each other, provide a sound basis for an analytic grammar. I am especially hopeful that the study of complicated sentences produced by this program, with their complete syntactic structures indicated, will provide insights into methods of analyzing sentences.

Finally, there is the question of utilizing a generative grammar as a tool in syntactic research. I have already used the grammar of French under consideration in the study of French syntax. As frequently happens when we approach a problem from a fresh point of view, I was able to gain new insights into a number of problems of French syntax simply by considering how they fit in with the concept I had of this grammar of French. Perhaps more important, the finished product (i.e., the computer program) has been and can be used in the study of specific problems by altering the program for each problem according to the needs of the researcher.

It is this third purpose, the development and use of a syntactic research tool, that will interest us most in this thesis. In the following chapter, in order to demonstrate how a generative grammar is useful simply in the form that it takes, I shall compare it briefly with other types of grammars and describe the steps followed in the writing of my generative grammar of French. Each step will be illustrated fully with structural diagrams representing my analyses of various syntactic forms.

Conception and Development

The first steps toward the development of this grammar were taken before I had any intention of writing a machine-oriented grammar of French. I was simply testing the "left-to-right" hypothesis (which holds that the sentences of a natural language can be generated from left to right, without back-stepping, by a finite state mechanism) by applying it to various French sentences with complicated structures, in particular, sentences with discontinuous constituents. (As a simple example of a discontinuous constituent pair, consider, Je n'y suis pas allé, where the auxiliary, suis, and the past participle, allé, form one constituent but are separated by the negative particle, pas.) As I continued to study this problem, I realized that although the hypothesis seemed to be satisfied in each case, we would require a complete grammar (or at least the complete framework of a grammar) in order to be sure that all the rules were consistent with each other. The danger of inconsistency is evident: I could demonstrate with one set of rules how a particular structure seemed to conform to the hypothesis, and then with another set of rules how a different structure also was in conformity with the hypothesis. However, without a more general framework in which to place the separate sets of rules, I could not test their consistency with each other. It seemed obvious, moreover, that such a grammar would not only provide a more conclusive means of testing the hypothesis but also would be useful in the study of various syntactic problems that I had become interested in while doing research in Mechanical Translation.

Before finally deciding to attempt to write a left-to-right grammar, I reviewed once again certain approaches to the problem of syntax and grammars. Two of them deserve particular attention here: Chomsky's Syntactic Structures and Tesniere's Eléments de Syntaxe Structurale. [6]

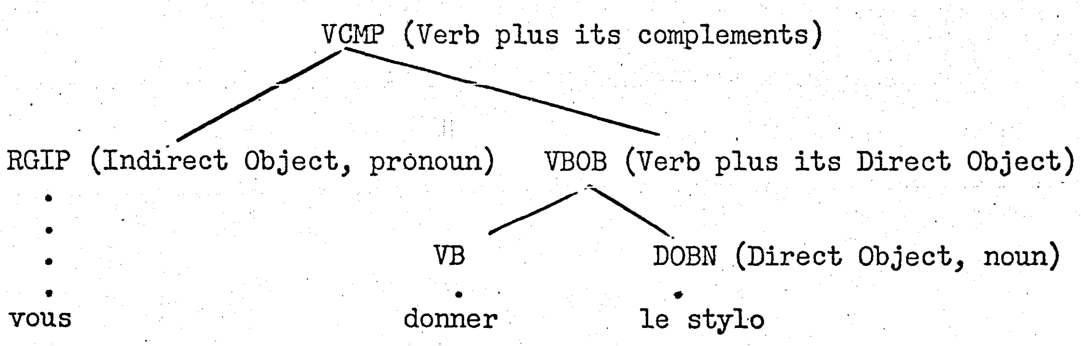
Chomsky's work is pertinent because it presents a strong argument against the possibility of generating the sentences (all and only) of a natural language by means of a finite state mechanism, and because it describes a more sophisticated type of generative grammar, the transformational grammar. His book provided an excellent point of departure for further consideration of the merits of any given type of grammar, and it was in spite of doubts about the validity of the finite state mechanism approach to grammar rather than because of a conviction that Chomsky's arguments were wrong that I went ahead with work on my left-to-right grammar. It should be noted also that the left-to-right grammar I have written is not, strictly speaking, a finite state grammar of the type described by Chomsky: When a structure is generated which necessarily limits (for example, requires agreement in person and number) a structure that is to be generated later in the sentence, this fact is remembered and is used to provide for the generation of the correct form of the later structure. Also, though it is never necessary to return to and alter any syntactic structure once it has been generated, such "backstepping" is necessary in the morphophonematic part of the program (see Chapter six). [7]

Tesniere's attempt to show the semantic as well as the syntactic relationships that obtain between the words of each sentence analyzed was particularly helpful to me in the first stages of my research. Although I did not in fact make any of the semantic ties shown in his analyses, they did guide me in deciding which characteristics of a particular structure or word should be carried over to the following words of the sentence.

Once I had completed reviewing the above-mentioned works and others, [8] and decided that further research based on the notion of a left-to-right grammar would be fruitful, I returned to the French sentences I had been analyzing. The analysis consisted of determining the immediate constituents in each sentence and then illustrating the way these immediate constituents were related within the sentence. The diagrams I used to illustrate these relationships are called phrase structure trees. In order to organize the material into syntactic groups, I classified the sentences according to the principal syntactic categories exemplified in each one, such as negative forms, interrogative forms, relative clauses, complementary infinitive clauses, and analytic tenses.

The use of phrase structure trees fulfilled two functions in my research work. First, a phrase structure tree, as can be seen in the example at the end of the next paragraph, provides a clear, easily read illustration of the relationships that obtain between various pairs of constituents that make up a phrase or sentence. Second, and most pertinent in my attempt to maintain consistency among the various sets of rules of the grammar, the format of these diagrams permitted me to label clearly each syntactic level and thereby check how consistently I was treating structures at higher levels in the syntactic hierarchy. A syntactic level in a phrase structure tree is the point at which two (or more) constituents join to form a higher level constituent (as the auxiliary and past participle join to form the verb). By studying the labeled syntactic levels ("nodes" of the trees), I was able to group the sets of immediate constituents which belonged under the same node, and I was able to isolate the instances where, in different sentences containing similar types of syntactic structures, I had not been consistent in my analysis of a given structure into its immediate constituents.

One result of my solutions to problems such as the preceding, problems characterized by variations in the immediate constituent analyses of certain structures when found in different contexts, was the development of a new set of labels for a few syntactic levels. However, for the most part, I preferred to use the more traditional terms in labeling the nodes of my phrase structure trees. For example, I decided to label the verb, its direct object and its indirect object as follows:

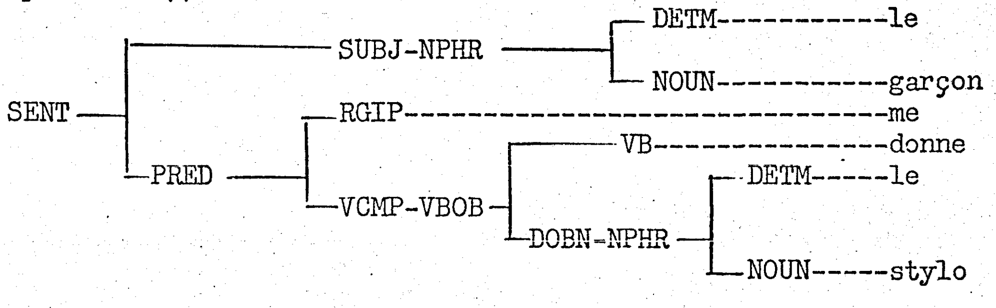

VCMP (Verb plus its complements)
RGIP (Indirect Object, pronoun)
VBOB (Verb plus its Direct Object)
VB
DOBN (Direct Object, noun)
vous
donner
le stylo

This diagram, given simply as an example of the node-labeling procedure, illustrates that the verb plus its direct object is taken as a unit which, when combined with the indirect object, forms a still higher unit in the syntactic hierarchy. These elements are not restricted to this particular order or final composition, and some steps are omitted, as can be seen in the complete program (Appendix V, Rules F0620 to F1025).

The next step in my research preparatory to writing the grammar was to proceed down each phrase structure diagram, writing the rules for the expansion of each syntactic unit into its two components. Taking a simple sentence, such as, *Le garçon me donne le stylo,* we can illustrate the above steps as follows:

1) Draw a phrase structure tree:

((Example No. 1))



2) Write the rules for expansion:

SENT = SUBJ + PRED
SUBJ = NPHR
NPHR = DETM + NOUN
PRED = RGIP + VCMP
VCMP = VBOB
VBOB = VB + DOBN
DOBN = NPHR

DETM = le
NOUN = garçon
= stylo
RGIP = me
VB = donne

It is obvious that the rules given above may correctly form the sentence of Example 1, but are insufficient both for consistently correct expansion of the syntactic elements included in this sentence, and, of course, for the generation of other types of sentences. In fact, in the above form, that of a simple phrase structure grammar, they are subject to all the difficulties exposed by Chomsky. In order to eliminate the difficulties which are a necessary part of such a limited grammar, I decided to analyze many sentences of differing types in order to develop a clearer notion of how much flexibility was required before attempting to write the rules of the grammar. In the following paragraphs I shall discuss some particular weaknesses of the sample rules and then present the analyses of more interesting sentences with comments, particularly pertaining to how these analyses pointed to the development of more flexible rules.

One major inadequacy (not treated by Chomsky) of the sample set of rules given above is that no attempt was made to express and maintain the relationship which obtains necessarily between the determiner and the substantive in each noun phrase. This, of course, must be done if we are to generate correctly nouns which differ in gender and/or number from the two masculine singular nouns of the example. Therefore, the gender and number of the entire noun phrase must be decided either at the time of the production of the initial word in the noun phrase or at the syntactic level, "noun phrase". For reasons to be discussed later, this decision is made at the higher level, as illustrated below. When the noun phrase constituent (NPHR) is replaced by a more specific constituent, "Common Noun Phrase" (COMN), a choice is immediately made as to the gender and number of the phrase to be generated. This choice is noted in the diagram by the letters after the slash mark. The "f" and "s" are called subscripts: they serve to define certain characteristics of the constituent, characteristics which must be transferred to all lower constituents within the same phrase.

(a) Tree Structure:

```
                        ┌── DETM/f,s - DEFART/f,s ---------la
NPHR - COMN/f,s ────────┤                ┌ADJA/f,s ---------petite
                        └─NOUNAD/f,s ────┤
                                         └NOUN/f,s ---------maison
```

(b) The Pertinent Rules:

```
x = NPHR
NPHR = COMN/f,s
     = COMN/f,p
     = COMN/m,s
     = COMN/m,p
COMN/f,s = DETM/f,s + NOUNAD/f,s
etc.
```

As these rules are written, each COMN constituent must be expanded separately into its immediate constituents, determiner plus "noun-plus-adjective". In the actual COMIT program, only one expansion rule is necessary; the transfer of subscripts is handled automatically without the need of mentioning them specifically.[9]

(c) Explanation:

The gender and number of the noun phrase is decided at the level COMN, thus determining the gender and number of each of the components of this particular syntactic unit. Of course, there may be a noun phrase within a noun phrase, such as <u>la</u> <u>jeune</u> <u>fille</u> within <u>le</u> <u>livre</u> <u>de</u> <u>la</u> <u>jeune</u> <u>fille</u>, which is independent as to gender and number. Such constructions are accounted for in the program, where each noun phrase is treated independently even in instances where the constituent may be discontinuous.

Another inadequacy of the sample grammar is that the rules are not sufficiently flexible. We would have a highly restricted grammar if all sentences generated by it were composed only and necessarily of a subject and predicate, in that order. Therefore, instead of writing rules which permit only one kind of expansion for each constituent (for example, SENT = SUBJ + PRED), I write rules of the following type:

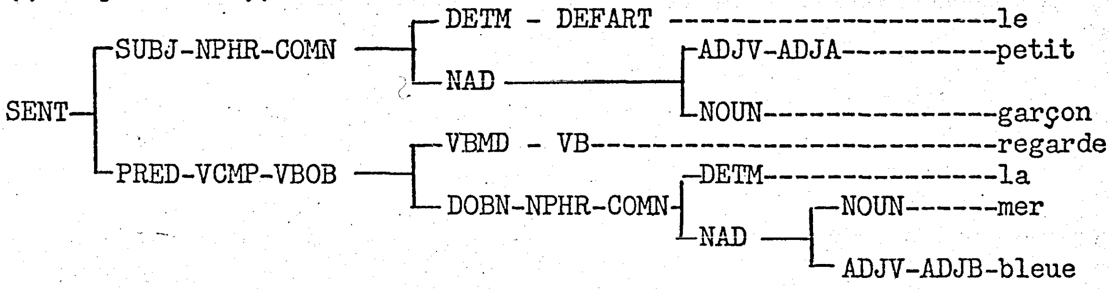
```
SENT = SUBJ + PRED
     = PRED + SUBJ
     = PRED
```

The possibility of choosing between the first two expansions allows for inversion; the third choice allows for the imperative and other sentence types which do not contain an expressed subject.

Similar but much more complex improvements are required in the expansion of the predicate to allow for various types of complements and to allow for various orderings of the elements of each complement type. In order to discuss some of these alterations in the grammar, I shall leave the example above and proceed to a discussion of some of the sentences studied in the development of the grammar giving their tree structures and some typical rules to exemplify the diverse methods of expanding syntactic forms.

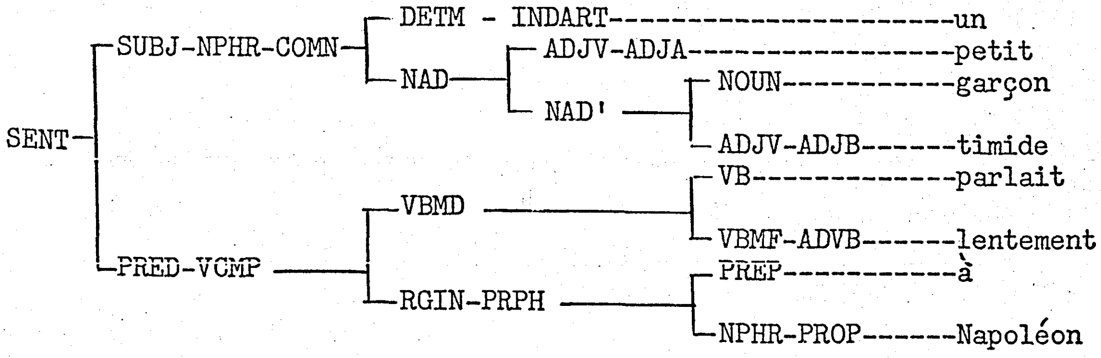
Until now we have seen only noun phrases composed of determiner plus noun. In the following sentence, I have added adjectives, one preceding and one following the noun:

((Example No. 2))



In both instances, the adjectives belong under the node, "common noun" (COMN), which determines the gender and number of each word within its range. Where we originally had simply determiner plus noun, we now have determiner plus "noun-with-its-modifiers" (NAD). Of course, NAD can simply be replaced by NOUN with no modifiers, or it can be expanded still further, as in the next sentence, to ADJV + NAD, permitting the final structure, DETM + ADJV + NOUN + ADJV:

((Example No. 3))



This sentence also illustrates a further addition to the verb phrase. Instead of just VB, we have a constituent VBMD which may be expanded to verb plus its modifiers. In the example above, the verb-modifier function is assumed by an adverb, but it could equally well be assumed by a prepositional phrase, e.g., <u>devant le tribunal</u>. Another addition seen here is that of expanding the noun phrase into a proper noun as well as into a common noun.

Let's take a moment to consider the effect of such additions on the set of rules. In the initial example, there was only one rule that allowed two choices, a vocabulary rule:

NOUN = garçon
     = stylo

This meant that mechanical application of the set of rules would have given these four sentences:

Le garçon me donne le stylo.
Le garçon me donne le garçon.
Le stylo me donne le garçon.
Le stylo me donne le stylo.

The only variation is in the noun chosen to serve as subject or object, not very interesting syntactically, and a bit bizarre semantically. (Please note that the grammar, even in its later form, does not contain any device for assuring that the sentences generated will be "logical" or will "make sense". Vocabulary items are chosen at random, the only restrictions being that they must belong to the syntactic category required for the particular point in the sentence and must conform to the rules of government and agreement.)

When we write rules to account for the additions in sentences 2 and 3, we find that there are now <u>grammar</u> rules, not only vocabulary rules, which contain more than one possible expansion:

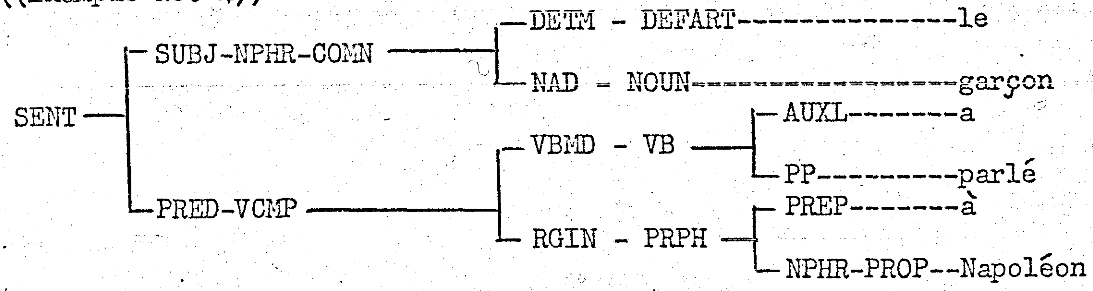
```
NAD = NOUN                    VCMP = VBOB + RGIN
    = ADJV + NOUN                  = RGIP + VBOB
    = NOUN + ADJV                  = VBMD + RGIN
    = ADJV + NAD'

NPHR = COMN                   VBMD = VB
     = PROP                        = VB + VBMF
```

Each possible expansion of a given constituent is called a subrule of the expansion rule for that constituent. In the complete program, the number of subrules for each grammar rule is determined by the number of distinct sets of components of which the particular syntactic structure may be composed and by the number of different possible arrangements of each of these sets of components. For example, I mentioned above the expansion of SENT into SUBJ + PRED, or PRED + SUBJ, or PRED. The first two expansions involve the same immediate constituents, but in reversed order: therefore, separate subrules are required. It is also clear that not all of the subrules of a given rule may correctly be chosen at a given point in the sentence, due to the limitations developed earlier in the sentence. For this reason, rules and subrules must have names so that any restrictions that are developed may be clearly and simply stated. These procedures will be discussed in the next chapter. At this point, I am principally interested in showing how the various syntactic structures were added and what devices had to be included in the grammar to account for them.

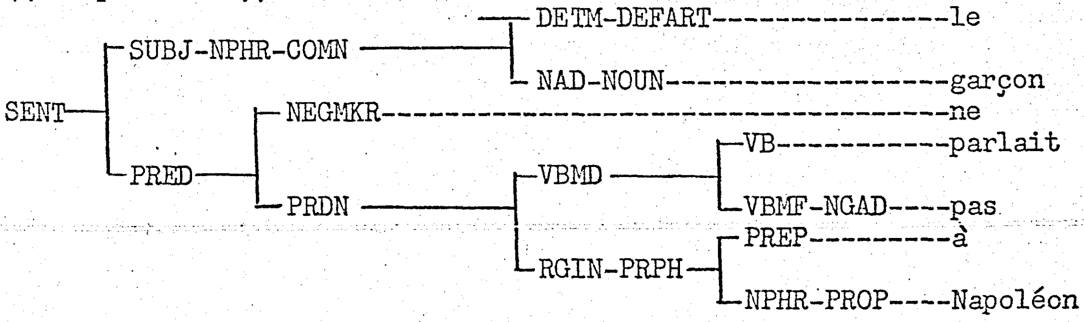
The following sentence illustrates the addition of the analytic (compound) form of the verb:

((Example No. 4))



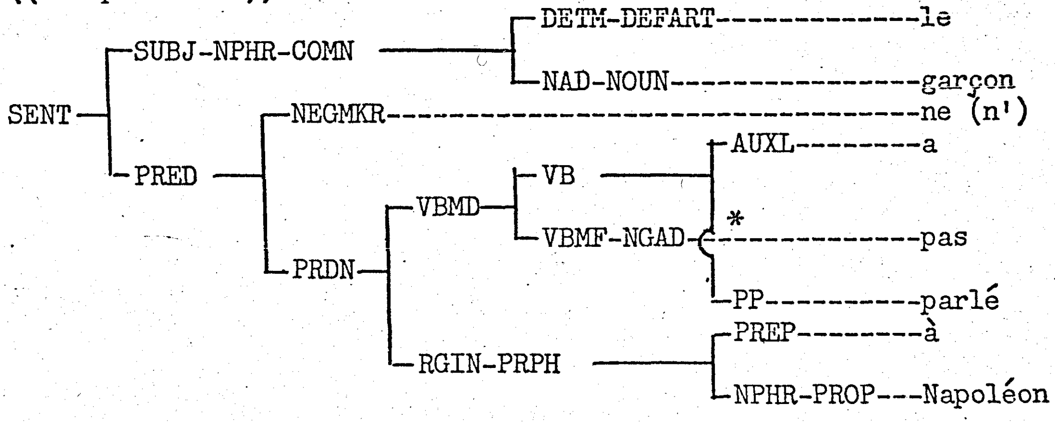
The verb, rather than being replaced by a synthetic form (parlait), can now be replaced by, or expanded into, an analytic form (a parlé). If we consider next the following negative sentence, we shall see that this variation in the expansion of the verb must be carefully recorded or predetermined in order to maintain the correct order of the syntactic elements of the predicate.

((Example No. 5))



The preceding sentence can be handled by existing rules (that is, by rules developed in this sample presentation of the grammar) but the following one, containing both a negative and an analytic verb form, presents difficulties which cannot be handled by the present grammar:

((Example No. 6))



If we did not foresee the structure of the sentence in Example No. 6, the rules with subrules for the expansion of VB would be:

VB	= SYNT	(e.g., parlait)
	= AUXL + PP	(e.g., a parlé)

The second subrule is correct for the following sentence: Le garçon n'a parlé qu'à Napoléon. However, it is not correct for sentence 6. It would have generated: * Le garçon n'a parlé pas à Napoléon. In order to avoid such ungrammatical positioning of the negative particle, I added a new type of expansion, one which permits the generation and handling of discontinuous constituents. Thus, to the possible expansions of VB given above, we add the following:

= AUXL + ... + PP (e.g., a pas parlé)

The " ... " indicates a constituent (specifically the high level constituent which at the time of expansion immediately follows PP) which will separate the immediate constituents, AUXL and PP, producing a discontinuous constituent structure. Of course, once each of these sub-

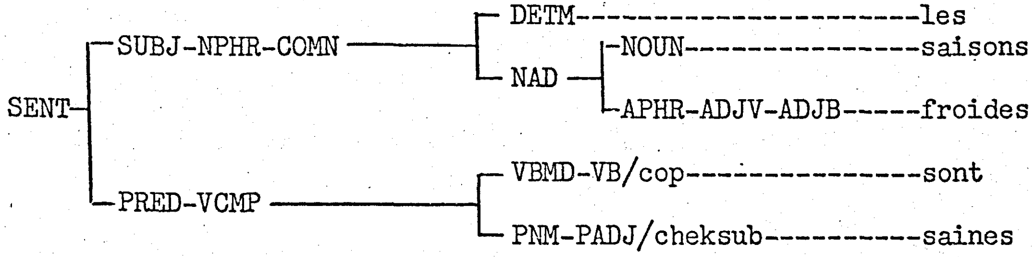
* The semi-circle crossing over the NGAD constituent indicates the discontinuous structure of AUXL . . . PP.

rules is applied, there are restrictions on the syntactic elements and classes of words that can be generated later in the sentence, and on the order of these units.

The remaining sentences and phrases illustrate structures requiring rules of the same general form as those given above, and I shall present their tree structures with brief comments since the reader now has a notion of how rules are formulated and of the types needed for each set of expansions. The program itself, Appendix V, provides the complete and detailed set of rules which evolved from the rules written for each of the following syntactic structures.

In order to add the predicate nominative complement, it is necessary to provide rules not only to expand VCMP to VBMD + PNM, but also to be sure that the verb is copulative and that the predicate adjective will be of the same gender and number as the subject. Therefore, when VCMP is expanded into VBMD + PNM, as in the sentence below, the program automatically keeps a record of the gender and number of the subject(s) and also indicates that the verb must be copulative.

((Example No. 7))



An adjective modifying one or more nouns in a "manifold substantive" follows the same rules for agreement as the predicate adjective. An example of the manifold substantive is given in the following phrase:

((Example No. 8))

```
                                   ┌DETM-DEFART------------la
                      ┌ NPHR-COMN──┤
                      │            └NOUN------------------mer
        ┌NPHR-COMN ───┤            ┌CONJ-------------------et
        │             └ MORNPH ────┤            ┌DETM-DFRT--le
NPHRGP ─┤                          └NPHR-COMN───┤
        │                                       └NOUN--------ciel
        └AADJ-ADJV-ADJB-------------------------------------bleus
```

NPHR', in the following sentence, illustrates the type of noun phrase that does not have a determiner as one of its immediate constituents:

((Example No. 9))

```
                 ┌DETM-INDART------------------------------une
                 │     ┌NOUN--------------------------------table
NPHR-COMN ───────┤     │               ┌PREP------------------de
                 └NAD ─┤               │
                       └APHR-PRPH──────┤             ┌NOUN--------bois
                                       └NPHR'-NAD'───┤
                                                     └ADJV-ADJB--dur
```

More important, the phrase in example 9 provides an example of syntactic ambiguity. Compare the following phrase:

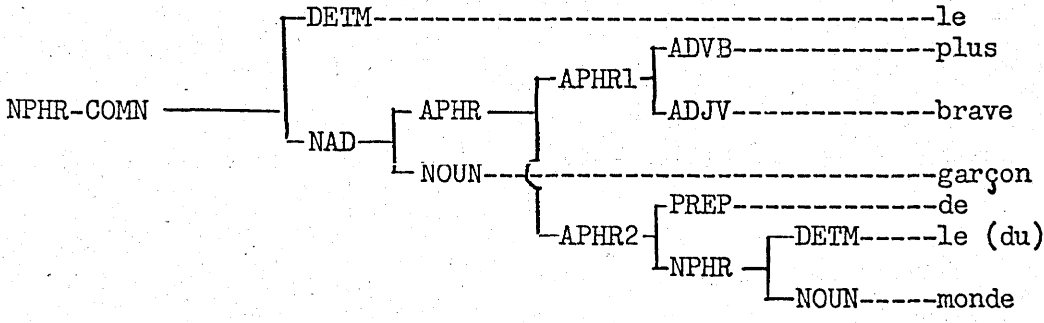
((Example No. 10))

```
                  ┌DETM-INDART------------------------------une
                  │     ┌NOUN--------------------------------table
NPHR ─────────────┤     │                 ┌PREP-------------de
                  └NAD ─┤     ┌APHR-PRPH──┤
                        └APHR'┤           └NPHR'-NAD-NOUN--bois
                              └APHR-ADJV-ADJB-------------carrée
```

As is indicated by the phrase structure diagrams and by the gender agreement, the adjective dur modifies bois in No. 9, while the adjective carrée modifies table in No. 10. However, if no diagram were provided and both nouns were of the same gender and number, it would not be possible to determine, by syntactic criteria, which noun was modified by the adjective. The generative grammar will produce both types of phrases, clearly noting the syntactic structure of each one. I do not suggest that this rather simple task in a generative grammar resolves the problem of syntactic ambiguity in recognition grammars. However, the study of these structures and of the rules for producing them may be helpful.

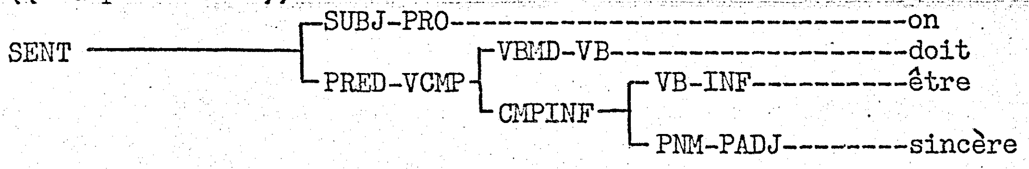
In Example No. 11, the structure of the common noun phrase (COMN) indicates that the determiner, le, is the determiner for the entire noun phrase, not just for the noun. Although this is not an extraordinary statement, it is important to note that decisions of this sort must be made to assure consistent application of the rules for showing the structure of noun phrases.

((Example No. 11))



The addition of a new type of complement, the complementary infinitive clause, required me to include a number of new restrictions on the finite verb and on the rest of the predicate, as is seen in the following example:

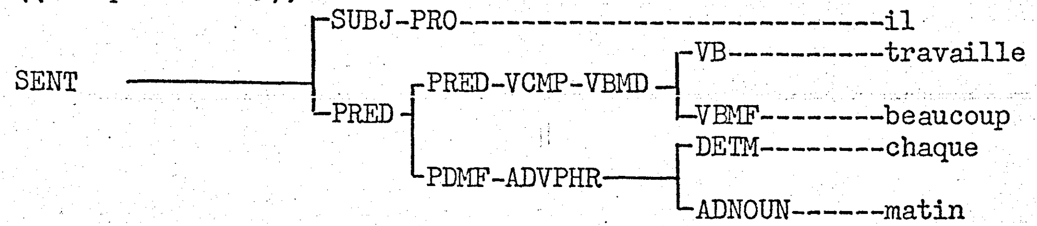
((Example No. 12))



The finite verb (*doit*) must belong to the category that can take complementary infinitives and, once that verb is chosen, the complementary infinitive must be preceded by the correct particle, *à* or *de*, or, as in this sentence, zero. Furthermore, in case the infinitive chosen is the copulative, the gender and number of the subject (or other referent: *Il nous a dit d'être sincères.*) must be recorded for the purpose of effecting the proper agreement in the predicate adjective.

Previously, the only adverbial modifier that could be generated was one which modified the verb directly. Now, we indicate a predicate modifier, which is shown to modify the predicate as a whole:

((Example No. 13))

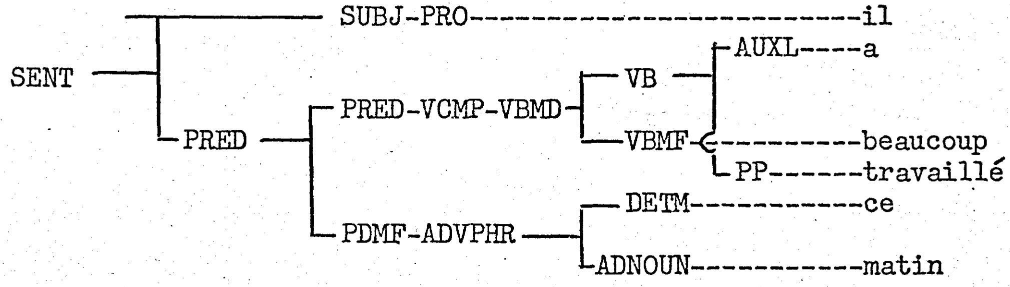


The predicate (PRED) is optionally expanded into PRED + PDMF before the expansion of the predicate into verb-plus-complement. This means that I am arbitrarily preventing the generation of a predicate modifier in the position between a verb and its complement. For example, in the sentence, *Il aime beaucoup sa femme*, the adverb *beaucoup* would necessarily be generated as a verb-modifier, never as a predicate-modifier.

A more detailed discussion of adverbial modifiers in various syntactic positions is given in Chapter five.

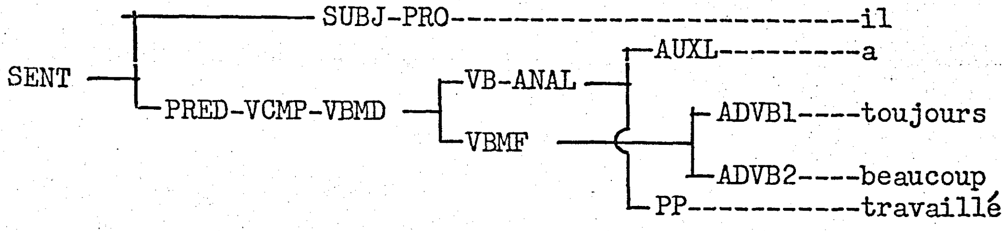
No new structure is added in the following sentence:

((Example No. 14))



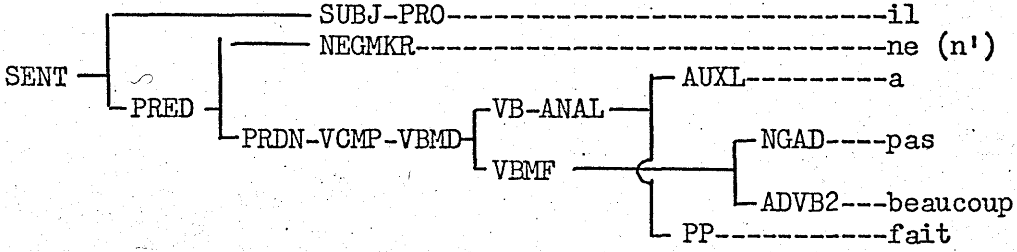
The sentence above illustrates the discontinuous structure of a ... travaillé, separated by the adverb beaucoup, and this, in turn, indicates why I decided to generate additional adverbs by introducing predicate modifiers as well as by adding verb modifiers in conjunction. The following sentence includes an example of verb modifiers in conjunction:

((Example No. 15))



Had sentence 15 been generated with a predicate modifier instead of verb modifiers in conjunction, it would have been: Il a toujours travaillé beaucoup or Il a beaucoup travaillé toujours. These are perhaps less acceptable stylistically than the first version (that is, the sentence in Example No. 15), but they are nevertheless grammatically correct. There are undoubtedly restrictions on the types of adverbial modifiers and classes of adverbs that can be generated in conjunction and as predicate modifiers, but the present grammar includes only the restrictions illustrated in the following sentence:

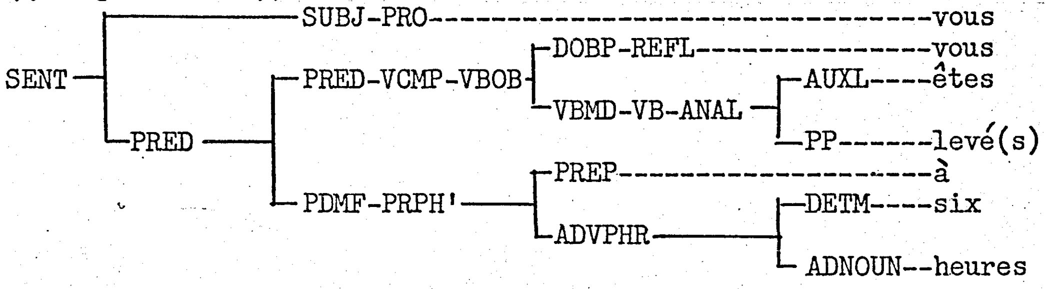
((Example No. 16))



Clearly, the negative particle pas, generated as a verb modifier, will restrict any other verb modifiers conjoined to it in the same structure.

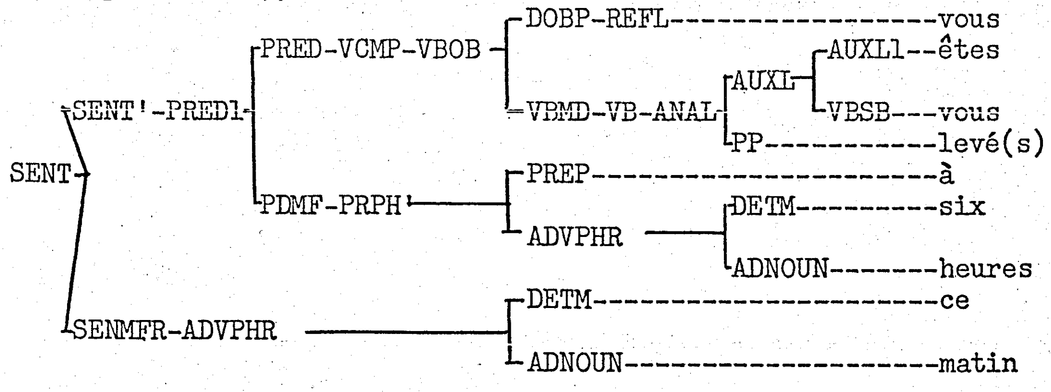
One of the common types of verb-plus-complement structures is the verb with its direct object. In the next sentence, the object is a personal pronoun, and further, a reflexive personal pronoun. The rules, therefore, must place the pronoun in its correct position before the verb, generate a reflexive form of the same person and number as the subject, and keep a record of the gender and number of the pronoun, to be used during the generation of the past participle. Since the referent of vous may be either singular or plural, the program chooses optionally between these two possibilities.

((Example No. 17))



Two new structures are added in Example No. 18: the sentence modifier, which is added to the program with no difficulty, before the sentence is expanded to subject plus predicate, and the VBSB, which requires some discussion:

((Example No. 18))



VBSB is an arbitrary abbreviation for a somewhat less arbitrary syntactic structure, which I have labeled "verb-subject". It is the personal pronoun subject form that is found immediately after the verb in certain interrogative sentences (as well as in other sentences where inversion takes place). In the above sentence, this pronoun is the only form functioning as subject; in other sentences, such as *Le petit garçon a-t-il frappé le chien?*, the "verb-subject" (*il*) is redundant in its function as subject since the noun phrase (le petit garçon) assumes the subject function. For this reason, the VBSB is not generated from the node SUBJ (that is, not as the form generated by the syntactic unit called "Subject") but is generated from the node VB-SYNT or AUXL (that is, as an immediate constituent with the finite form of the verb). This procedure is consistent with the treatment of interrogative words, discussed in detail in Chapter five.

The following example illustrates the use of the "verb-subject" structure in a sentence with an expressed noun subject, and also gives an example of the complement VBOB expanded into a verb plus a noun phrase functioning as the direct object.

((Example No. 19))

```
                   ┌SUBJ-NPHR-PROP----------------------------Jean
                   │                          ┌SYNT---------a
SENT ──────────────┤               ┌VBMD-VB ──┤                   (-t-)
                   │               │          └VBSB---------il
                   └PRED-VCMP-VBOB-┤          ┌-DETM--------un
                                   └DOBN-NPHR-┤
                                              └NOUN---------stylo
```

In sentence 20, I have introduced an indirect object pronoun (RGIP) which is generated as an immediate constituent with the VCMP constituent:

((Example No. 20))

```
      ┌──────────── SUBJ-NPHR-PROP------------------------------Jean
      │     ┌NEGMKR---------------------------------------------ne
SENT -┤     │         ┌RGIP----------------------------------------vous
      │     │         │           ┌DOBP-------------------------le (l')
      └PRED ┤         │           │                     ┌AUXL-a
            └PRDN ────┤           │              ┌AUXL  ┤              (-t-)
                      └VCMP-VBOB -┤      ┌VB ────┤      └VBSB-il
                                  └VBMD -┤       │
                                         └VBMF-NGAD-----------pas
                                                 │
                                                 └PP----------donné
```

Once the indirect object pronoun is generated (*vous*), any subsequent direct object pronoun must be in the third person. The rules for handling object pronoun word order and for generating permissible combinations of direct and indirect object pronouns are discussed in Chapter five.

Interrogative words, such as the INTADV here,

((Example No. 21))

```
                           ┌-INTADV--------------------------------où
SENT───────────────────────┤          ┌-PRED-VCMP-VBMD-VB------------va
                           └-SENT ────┤
                                      └-SUBJ-NPHR-PROP---------------Jean
```

are generated before the sentence is further expanded to SUBJ + PRED or PRED + SUBJ, because they affect the basic word order of the sentence.

The rules necessary to produce the types of sentences illustrated by examples 11 to 21 are, of course, much more complex than those given in the earlier, simple examples. In this chapter, I intended only to show how the grammar evolved in order to illustrate the necessity of a complete framework in which to place each new set of rules. Most of the particular problems in syntax that are exemplified in the sentences in this chapter are discussed in detail in Chapter five. In the following chapter, Chapter three, I shall give a complete explanation of the types of rules used in the grammar, describing in detail how the program operates. This should provide the necessary background for the explanations in Chapter five.

Description of the Programmed Grammar

In the Introduction I stated that the most effective way of testing a generative grammar is to program it for a computer and allow the computer to print out sentences at random. If the sentences generated are correct, then the grammar is "good", that is, its rules are consistent and complete, at least for the set of structures involved in these sentences. In the discussion of the development of the grammar, in Chapter two, I indicated that the structure of the grammar is oriented toward its application to a computer program. It is the computer program, therefore, that I shall describe in this chapter, explaining more completely the types of rules that I have used and also presenting the organization of the program into its various parts, or "routines".

The program is divided into two principal parts: the "grammar of specifiers" and the "grammar of sentences". Each sentence (and each clause within each sentence) must be operated on by each of these parts. The grammar of specifiers, described in detail in Chapter four, specifies certain basic characteristics of each clause, and, according to these characteristics, determines the overall word order of the clause. The grammar of sentences constitutes the body of the program. It contains the rules which actually expand constituents and generate words. The choice of rules and subrules in this part is partially determined by the operations that take place in the grammar of specifiers and partially determined by operations that take place within the word-generating routine itself. Wherever the previous operations have not determined a choice, the computer chooses at random.

This division of the grammar into two major sections is original, at least in its organization, [10] and is an important aspect of this generative grammar of French. It is one of the factors that cause this grammar to differ from the concept of a "finite state grammar" as described in Chomsky's study of grammars. Instead of starting the generative process immediately, and producing one word after another, from left to right, I first construct a model of the general form of the sentence and make decisions about certain aspects of the sentence. This means that when the grammar of sentences actually begins to operate, to expand the syntactic constituents, many decisions as to which steps must be taken have already been made within the grammar of specifiers. The machine is not permitted to "run headlong" into a sentence, producing structures and words as it pleases, limited only by the factors which are developed as each word is generated.

It is not until the grammar of sentences begins to operate, therefore, that the "left-to-right" procedure takes place. Up until this moment, within the grammar of specifiers, no expansions have taken place. The first operation of the grammar of sentences is that of expanding the initial constituent (e.g., SENTENCE or CLAUSE) into its immediate constituents (e.g., SUBJ + PRED). The next operation takes the first (leftmost) resulting constituent and expands it, and so on. Each constituent, once it has been expanded, has served its purpose and is therefore put aside for subsequent print-out. The resulting constituents of any expansion take their position in front of (to the left of) all constituents which remain to be expanded. When the point is reached in the process where a syntactic symbol is replaced by a terminal symbol (a word) rather than being expanded into one or two new constituents, the word is set aside for print-out and control passes to the next constituent to the right.

The following diagram illustrates the process I have just described. The first column corresponds to the place where constituents and words that have already been operated on are set aside for subsequent print-out. Column two contains, at each line, the constituent that the program is about to act upon. The line below any given line contains, in column two, the result of the expansion or replacement of the constituent directly above it in column two. If the operation was an expansion, rather than the generation of a word, then column three will contain the second element, if any, of the expansion. This most recently generated constituent will be leftmost in column three, which contains all the constituents remaining to be expanded.

	1.	2.	3.
Line No.	Printed out	Symbol to be operated on	Symbols resulting from expansions
1		SENT	
2	SENT	SUBJ	PRED
3	SUBJ	NPHR	PRED
4	NPHR	DETM	NAD + PRED
5	DETM	le	NAD + PRED
6	le	NAD	PRED
7	NAD	ADJ	NOUN + PRED
8	ADJ	grand	NOUN + PRED

Line	1.	2.	3.
9	grand	NOUN	PRED
10	NOUN	roi	PRED
11	roi	PRED	
12	PRED	NEGMKR	PRDN
13	NEGMKR	ne	PRDN
14	ne	PRDN	
15	PRDN	VCMP	
16	VCMP	VBOB	
17	VBOB	VBMD	DOBN
18	VBMD	VB	MFR + DOBN
19	VB	AUX	Q + PP + MFR + DOBN
20	AUX	a	Q + PP + MFR + DOBN
21	a	Q	PP + MFR + DOBN
22	(Q not printed out)	MFR	PP + DOBN
23	MFR	NGAD	PP + DOBN
24	NGAD	pas	PP + DOBN
25	pas	PP	DOBN
26	PP	mangé	DOBN
27	mangé	DOBN	
28	DOBN	NPHR	
29	NPHR	DETM	NAD
30	DETM	la	NAD
31	la	NAD	
32	NAD	NOUN	
33	NOUN	table	
34	table		

As examples of the operations illustrated by the diagram and of the purpose of each column, consider the following: At line 1, column 2 contains the symbol SENT. No other constituent has yet been operated on, so there is nothing in columns 1 or 3. At line 2, SENT has completed its function and is placed in the print-out column, column 1. The two constituents which resulted from the expansion are SUBJ and PRED. SUBJ, the leftmost one, is in column 2, ready to be operated on; PRED is held in column 3.

Later, at line 5, le has just been generated by DETM and is in column 2. Since it does not cause any further generation or expansion it simply passes over to column 1, as shown at line 6. The next constituent (NAD) from column 3 takes its place in column 2. At line 7, when NAD is expanded to ADJ + NOUN, NAD passes to column 1, ADJ is placed in column 2, and NOUN is placed in column 3, in front of PRED.

Further down, when VB is operated on at column 18, it is expanded into AUX + PP, but with a dummy constituent, Q, in between them, because the construction is to be discontinuous. When Q is in column 2 at line 21, it does not cause an expansion, but rather operates on the next two constituents (the first two on the left in column 3), inverting them. Thus MFR, which is to be the negative pas, is generated before the PP, mangé.

The diagram above (pages 31 and 32) provides examples of each of the basic operations in a generative grammar: expansion, generation and manipulation. I shall describe now the types of rules in the programmed grammar that control each of these basic operations.

Expansion refers to the operation of replacing a given syntactic constituent (NPHR) by one or two other syntactic constituents (by PROP or by DETM + NAD). In the following sample rule,

```
NPHR  A    $1  =  PROP                     PROP
      B        =  DETM + NAD               DETM
```

NPHR is the name of the rule. It serves as an "address" in the program, permitting the calculator to find it when necessary. In this sample rule, NPHR has two subrules, A and B, meaning that the constituent NPHR (represented by $1 in the rule) may be expanded in one of two ways. If subrule A is chosen, the rule expands NPHR to PROP (the "=" sign represents the words "is replaced by") and the computer is then given the address PROP, meaning that it should go to rule "PROP" where it will be given directions as to how to expand the constituent PROP.

(Of course, in the actual program, there are more subrules and there are other operations indicated, such as an operation which places the expanded constituent NPHR on a "shelf", equivalent in the program to the step of moving NPHR over to column 1 in the diagram above.)

By generation, I mean specifically the act of producing a terminal symbol, a word. The type of rule that executes this operation is similar to expansion rules, except that after the constituent being operated on (e.g., NOUN) has been replaced by the generated word (e.g., *garçon*), the machine is not then directed to a rule named GARCON. It may be directed to a routine which will handle any morphophonematic changes required in the word (e.g., an adjective form, *petit*, will pass through a routine to see if an *e* and/or an *s* should be added to the base form) or, if no such changes are possible, then the calculator will be told to take the next constituent to the right and operate on it.

Manipulation operations are executed by rules like "Q". The computer is directed to find the constituents under consideration and then to rearrange them as indicated. Only structures that have not yet been operated on may be involved in such manipulations: all structures that have been operated on and have (according to the diagram above) passed into column 1 must remain in the order in which they were generated.

The preceding description illustrates the basic function of the grammar rules: the expansion of constituents and the generation of words. The rules must fulfill other functions as well in order to produce correct sentences. They must provide the facility for choosing only those expansions of a particular constituent which are compatible with other choices previously made in the generation of the sentence. They must record certain choices as they are made (whether at random or according to some restriction) so that subsequent constituents which may be dependent on these will be generated correctly. These and other facilities of the program are explained in the following paragraphs.

Most grammar rules consist of a number of subrules. This means that the same constituent may be expanded and treated in a number of different ways. During the generation of a given sentence, when a rule is to be executed, the choice of the particular subrule is usually restricted to some extent. For example, if the VCMP has been expanded into VBOB, and VBOB into VBMD + DOBN (Direct object, noun phrase), then when we are about to expand VB, we must necessarily choose the subrule that will lead to the generation of a transitive verb. The way that this restriction is shown and effected is the following:

In the rule that produces VBOB, a subscript is added to the symbol VBOB which is carried along automatically (without the need of repeating it) during the expansion of VBOB into its various constituents, until the rule VB is reached. At this point, just prior to choosing the category of verb, this subscript is used to activate the "dispatcher"[11] which tells the machine that it must choose the type of verb designated by the subscript. In case there is no subscript, the machine is free to choose at random among the various subrules. If the subscript designates two or more subrules as being legitimate choices, then the machine is free to choose at random among these subrules. Since we may execute the same rule more than once in a given sentence, and require different choices at each execution of the rule, the system followed is this: Every rule that has subrules is preceded by a rule which, among other things, tells the machine first to cancel all previous restrictions, and then sets it with the subscript, if any, that is found on the constituent to be expanded. Thus, for example, if a relative clause with an intransitive verb situation is generated before the verb in the main clause (which is to be transitive), the verb of the relative clause will be decided by the subscript attached to its symbol (to which the program did not transfer most of the subscripts of the main, initial symbol), while the main verb will be determined by the subscript found on the principal symbol.

The facility for transferring subscripts is used also for maintaining agreement, for example, in the gender and number of all the members of the same noun phrase. At the point at which the noun phrase is generated, the choice of gender and number is made, the necessary subscripts are added to the symbol NPHR or COMN, and these subscripts are carried throughout the expansion of the noun phrase, activating the dispatcher as necessary before the generation of the definite article or adjective and the choice of type of noun.

After a symbol has been expanded, the resulting constituents may be treated in a number of ways depending on the stage of development that they have reached. A completed constituent (a "word") will be examined to see if any morphophonematic alterations may be required. If there are none, it will be "shelved", that is, set aside to be printed out later (or, in the earlier diagram, placed in column 1). If there are some alterations possibly required, the word will be considered, in its context, by the morphophonematic routine, described in Chapter six. A constituent that still requires expansion, if it is first (leftmost in the COMIT workspace, equivalent to column 2 in the diagram), will be used to send the computer to its next operation, that is, to the rule that will continue to expand this constituent. If there are other constituents (not in first position) still to be expanded, they are placed on another shelf from which they will be taken, one by one, in order, when the first constituent has been fully expanded. (This shelf corresponds to column 3 in the diagram above.) An operational symbol will usually be used immediately to set in motion the operation with which it is concerned. For example, Q simply fills the space between two constituents that have just been expanded and which must be discontinuous. In the same rule that generates the Q, the Q is replaced by the next constituent on the shelf that corresponds to column 3 of the diagram. Then the newly positioned constituent (just removed from the shelf) and the second member of the discontinuous structure are replaced on that same shelf, in their new order.

When the symbol PRIN is reached there are no other constituents remaining to be expanded. PRIN then activates a routine which takes the entire contents of the shelf on which the words were placed and prints them out, with a sequential number preceding the sentence. This same number is then compared with a counter to see if the required number of sentences have been generated. If yes, the running of the program ceases. If no, the number is increased by one and another sentence is generated.

The Grammar of Specifiers

The Grammar of Specifiers is a routine which is executed prior to the Grammar of Sentences each time a clause (independent or dependent) is to be generated. As each rule in this routine is executed, specific decisions are made about the nature of the clause, and appropriate subscripts are added to the symbol that represents the clause in the program. These subscripts are later used to delimit choices in rules within the Grammar of Sentences, rules which cannot be chosen completely at random because of the decisions made in the Grammar of Specifiers. They are transferred, when appropriate, to the constituent members of this clause as it is expanded.

The decisions regarding the basic characteristics of the clause are made before the symbol for it is expanded, rather than at various points during the actual generation of the clause, because they affect the word order to such an extent and in such a way, that postponement of the decisions would result in placing extreme limits on the possible sentences to be generated, preventing the generation of certain types altogether. This appears to be analogous to the way in which the human speaker formulates his sentence, that is, 1) he decides upon certain aspects of the sentence; 2) then he chooses the sentence-type (word order); 3) finally, he utters the specific words. The aspects of the sentence that I have included in the Grammar of Specifiers are:

1) The choice among the interrogative, declarative and imperative types of sentences.

2) How to signal the interrogative, and, when necessary, the choice of the other syntactic function of the interrogative word.

3) The choice between the affirmative and negative, and, if a negative sentence is to be generated, the choice of the other syntactic function(s) of the negative particle(s).

4) The choice between the active or passive voice, and, if passive, whether or not the agent will be expressed.

I first decided to construct and study a "grammar of specifiers" principally to help me in programming the grammar. In the earliest versions of the grammar-program, I found that many of the errors in the sentences being generated could have been avoided if certain general, high-level decisions had been made earlier in the program. My procedure in constructing the Grammar of Specifiers was to add one unit at a time as the need for it appeared in the print-out of sentences. I have, in fact, considered a number of other decisions that could be made in the specifier routine which would simplify the programming of the main grammar. However, I have included in this version of the grammar only those decisions that I consider basic and necessary parts of a Grammar of Specifiers, specifically those that cannot be restricted to a single constituent and that radically affect word order. My reason for not including the other aspects in the Grammar of Specifiers is that, although I believe that the most important criterium to use in judging the value of a given section of the generative grammar is whether or not it functions well, I am also anxious to consider the apparent analogy between the Grammar of Specifiers in my generative grammar and some similar component or activity in the mind of the human speaker. I did not feel that the aspects I decided against including in the Grammar of Specifiers were likely to be considered in advance by the human speaker, despite the fact that, for the computer program, it was definitely much simpler to consider them before the actual generation of the sentence.

The importance of the Grammar of Specifiers can be seen in its application to mechanical translation research as well as to research in general syntax. It is of particular interest in mechanical translation because it suggests a means of defining, developing and manipulating "specifiers" in the translation process. In describing the Grammar of Specifiers routine earlier I explained that, as decisions were made in the routine about certain aspects of the sentence to be generated, appropriate subscripts were added to the symbol representing the sentence constituent. The subscripts are the specifiers of the sentence. In a translation process such as the one described in the introductory chapter (pages 3 and 4), the specifiers of the sentence to be generated would be provided by the "translation step". The analysis of the original sentence (in the input language) would have produced the set of specifiers of the sentence. Then, rather than translating at the word level, or even at the level of syntactic structures, we would translate at the more abstract level, the specifier level. The resulting translation would be an equivalent set of specifiers in the output language, which would control the operation of the generative grammar, producing a sentence in the output language.

Taking just one aspect, the interrogative, of a simple sentence, I shall try to illustrate the steps described above. I am to translate the English sentence, "Is John sick?" into French. The first component analyzes the English sentence, producing a set of specifiers. One of these specifiers notes that the sentence is interrogative. (Others note that "John" is the subject, that the tense is present, that the verb is copulative, and so on.) After translation, the French set of specifiers still includes the fact that the sentence is interrogative, but it is also noted (because there is a noun subject and because of the absence of any interrogative adverb, particularly of the class to which *où* belongs) that inversion of the subject and verb is not possible. The generative grammar is permitted to choose between two word order types: using *est-ce que* plus normal word order, or using a pleonastic subject pronoun (called VBSB in the preceding chapter). Thus the output sentence could be either, *Est-ce que Jean est malade*? or, *Jean est-il malade*? That is, the specifiers do not necessarily specify particular structures and forms that must be chosen, but rather they specify the *limits* set on the grammar to produce a sentence that is syntactically equivalent to the sentence in the input language. I feel that the study of the use of subscripts in the Grammar of Specifiers in my program will help me to develop a more concise notion of the nature of "specifiers" in the mechanical translation scheme I have just outlined.

One of the ways the Grammar of Specifiers may be applied to research in general syntax would be to utilize the fact that it can control choices in the grammar of sentences in order to effect a limited study of a particular type of structure. For example, if I wished to study faire causal constructions and others which influence word order, it would be simpler both to add the structure at the grammar of specifiers level and also to test the new structure in actual runs of the program, rather than to add or alter many rules at various points in the main program. Ideally, the rules of the Grammar of Sentences should be complete and general enough to produce all combinations of structures. The Grammar of Specifiers must control the execution of the rules in the Grammar of Sentences in order to prevent the generation of ungrammatical combinations.

Description of the Grammar of Specifiers in the Program

The Grammar of Specifiers is a complete, organized set of rules that is used over and over again. It is called a routine in the programmed grammar. The following verbal description of the Grammar of Specifiers is intended to explain generally what the routine does and, to some extent, how the operations are executed. I believe that it illustrates sufficiently the function of this important routine in the complete program. However, I have also provided (in Appendices II-C and IV) a complete and more detailed description of the entire routine, by means of a flowchart and a step-by-step diagram.

In the Grammar of Specifiers routine, the first decision, which the machine is free to make at random, is "What basic type of sentence will be generated?" If it is to be declarative or interrogative, a choice is made about the voice (active or passive) and then, after further decisions are made about the interrogative sentence, the computer program goes on to decide if the sentence will be affirmative or negative. If, at the first step, it was decided that the sentence would be imperative, the computer immediately decides whether or not it will be an affirmative imperative sentence. If affirmative, there are no further decisions to be made and control passes to the Grammar of Sentences. If negative, the symbol for this negative imperative sentence is then operated on by the same set of rules regarding negative choices as operate on the symbols for declarative and interrogative sentences.

For both declarative and interrogative sentence types, if the sentence is to be affirmative, no further questions are asked and control passes to the Grammar of Sentences. This step was taken separately for the imperative type sentences because of the unique word order rules for the affirmative imperative. For all three basic sentence types, if the sentence is to be negative, a set of decisions must be made about the syntactic function of the negative particle(s) before control can pass out of the Grammar of Specifiers.

For interrogative sentences, the questions and subsequent restrictions on rules as indicated by the subscripts are concerned with the form of the interrogative (type of word or expression, inversion, or use of est-ce que), and the function of any interrogative word or expression. Following these decisions, the computer must choose the basic word order of the sentence.

As each of the decisions outlined above is made, a subscript is added to the symbol that is destined to be expanded into a string of syntactic structures and finally into a string of words, forming a sentence. The symbol will carry the subscripts and transfer them, as required, to each and every constituent that is generated by the grammar of sentences during the expansion of the sentence or clause under consideration. Then, whenever a rule is to be executed that is affected by one of these subscripts, the computer is directed to choose the subrule indicated by the subscript.

The following simple example illustrates how the decisions made in the Grammar of Specifiers affect operations in the Grammar of Sentences. In the Grammar of Specifiers, the machine chooses to generate an affirmative imperative sentence. Later, in the Grammar of Sentences, when the PREDICATE constituent is to be expanded, the computer is limited (by the subscripts transferred to that constituent from the original SENTENCE constituent) in its choice of subrules. It cannot choose to expand PREDICATE into INDIRECT-OBJECT-PRONOUN plus VERB-WITH-ITS-COMPLEMENTS in that order. Furthermore, when VERB-WITH-ITS-COMPLEMENTS is to be expanded, the computer is again limited to choices which do not include the generation of a pronoun direct object in front of the verb. The verb, when it is expanded, cannot be analytic, and so on. Each of these restrictions is noted as a subscript and when and if the pertinent constituent is to be expanded, the calculator refers to the subscript before executing the expansion.

In the discussion of individual problems in syntax in the following chapter, I shall have reason to refer to the Grammar of Specifiers frequently. The reader will thus have more examples of how the subscripts added to a symbol restrict specific choices in the expansion and generation steps. More important, I believe that the examples in the following chapter will help support the validity of the use of a Grammar of Specifiers.

Some Problems in the Syntax of French

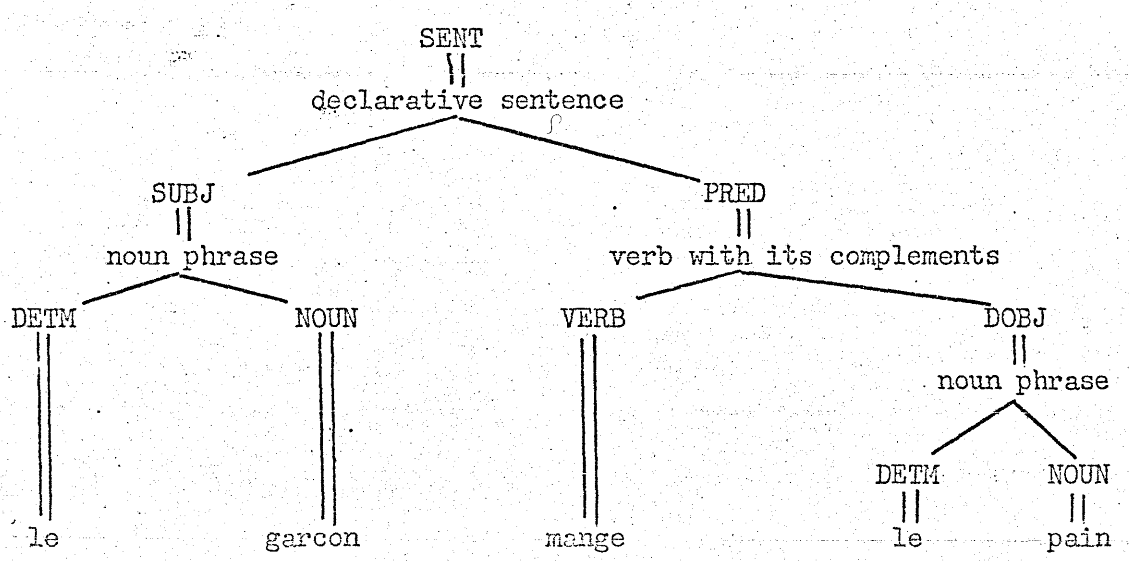
In the present chapter, I shall discuss specific questions in French syntax. The fact that I studied each of these problems in the framework of a left-to-right generative grammar naturally influenced my approach to them. The major advantage in studying each problem from the same point of view is that the resulting solutions to each one are more likely to be consistent with each other. The obvious disadvantage is that, as long as most syntactic structures seem to fit into the particular grammatical framework, the researcher may fail to recognize certain data, certain problems that cannot be solved in the given framework. In the following paragraphs, as I discuss the point of view I have taken and the effect of this point of view, I hope to demonstrate that I have been able to capitalize on the advantage and eliminate the disadvantage just mentioned.

The generative grammar described herein is a left-to-right grammar. As I mentioned earlier, the left-to-right hypothesis in general states that grammatical sentences of a natural language can be generated by a finite state grammar, producing structures and words from beginning to end, from left to right, without backstepping, that is, without returning to any word or structure to alter its form, and without changing the order of any structure once it has been generated. In addition to these general characteristics of a left-to-right generative grammar, my grammar of French contains two specific constraints: 1) Each constituent may be expanded into no more than two immediate constituents. 2) Discontinuous constituents may be separated by only one constituent. (Note that "one constituent" may be a syntactic structure composed of more than one word.)

Obviously, the presence of the general and specific constraints mentioned above affects the solutions to many problems in syntax. I have tried throughout the individual sections of this chapter to state explicitly where my approach is not consistent with the facts (as generally interpreted) and to discuss the new points of view adopted. However, perhaps the more general statement in the following paragraphs will indicate to the reader how the constraints inherent in my left-to-right grammar affect any given step in the grammar.

By limiting expansion to only two immediate constituents, I am not presenting a case for the binary structure of sentences. I simply have found that it is simpler to write a generative grammar with this constraint and that such a constraint does not prevent me from generating any type of structure. According to my system, a given constituent can be either expanded into its two immediate constituents (NOUN-PHRASE into DETERMINER plus NOUN-WITH-ITS-MODIFIERS) or replaced by another constituent (SUBJECT by NOUN-PHRASE).

Furthermore, the replacement-expansion operation involves more than the simple expansion of a given syntactic element into its immediate constituent(s). Ideally, and this is the way the grammar was originally conceived, each step involves either the statement of the syntactic function of a given form or the description of the form that a given syntactic function takes. Thus the function SENTENCE is replaced by the form "declarative sentence", which in turn is replaced by (or expanded into) the functions SUBJECT plus PREDICATE. The following diagram is a syntactic tree depicting the form/function distinction: (FUNCTIONS are capitalized; "forms" are in lower case letters.)



The present version of the grammar does not explicitly indicate the form/function distinction in every expansion. However, the distinction is implicit throughout the grammar, as I maintained the distinction always in the development of each syntactic structure. The reason that I did not always show this distinction in the final routine of the program for each structure is practical: By eliminating explicit notation of the distinction wherever the distinction was not required for the operation of the program, I was able to save space and keep the programmed grammar at a manageable size.

Returning to the matter of limiting expansions to only two immediate constituents, let us suppose I chose to describe the VERB + DIRECT-OBJECT + INDIRECT-OBJECT structure as being ternary (Fig. 1 below), rather than binary (Fig. 2 below). Adhering to a ternary system, I could undoubtedly generate the structure composed of the elements listed above in all of its possible forms and internal orderings (*le lui donner*, *lui donner le stylo*, *nous le donner*, etc.), in some cases even more easily than by following a binary system. However, I would not consistently show the relationship of the indirect object to the verb-plus-object or the verb-plus-complement. I would have to generate the indirect object separately (by a different rule) for the instance of verb (without object or other complement) plus indirect object, such as in *parler à Jean*. Similar considerations arose at every other point where a ternary structure appeared to be as correct as a binary one. Binary structure permits me to use a larger number of general rules, and keeps to a minimum the types of rules that generate duplicate constituents that differ only in minor ways.

Fig. 1

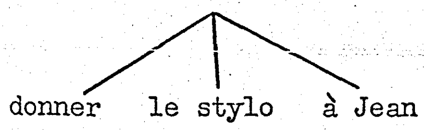
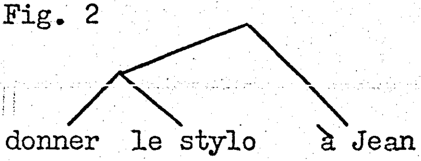
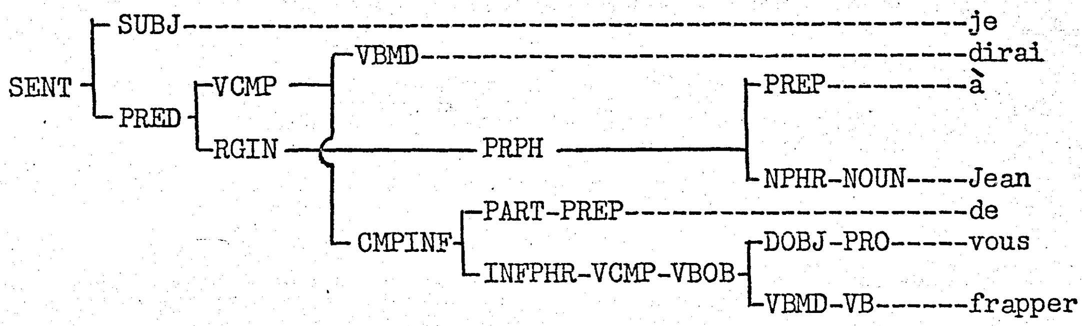


Fig. 2



As I noted above, when I say that discontinuous constituents may be separated by only one constituent, I do not mean that the words which make up the discontinuous constituents can be separated by only one word. Consider the following example:



The VCMP is composed of two major constituents, VBMD + CMPINF. They are separated by one constituent, the indirect object (RGIN), which happens to be composed of two words. VBMD could be further expanded into VB + VMFR and give, for example, Je dirai tout de suite à Jean de vous frapper, which would separate the words (dirai and frapper) still more, but would not alter the situation at the higher level of syntactic constituents.

It was the constraint of not allowing "double jumps" (jumping over two syntactic structures of the same level) that first caused me to look at the negative from a different point of view. The final decision on how to handle the negative, as I'll explain later in this chapter, was based on a number of other considerations which supported the somewhat unique way that I finally decided to parse the negative in French.

From these general remarks on the approach that I have taken, I shall pass now to a discussion of my treatment, within the framework of a left-to-right grammar, of particular problems in French syntax.

Negative and Interrogative

The interrogative and negative aspects of sentences present similar problems but they differ in various individual characteristics. In both cases, the basic problem is the presence of one or more words fulfilling two separate functions: the negative or interrogative and some other one. However, the ways in which this dual function characteristic affects the generation of the sentence are different in each case.

The fact that a sentence will contain a negative particle forces us to make certain decisions in advance of generating the sentence, as is explained in Chapter three, but the fact that a word is carrying the negative function does not affect the position of that word: Its position is determined by its other syntactic function. (Ne carries only the negative function and therefore is not included here.)

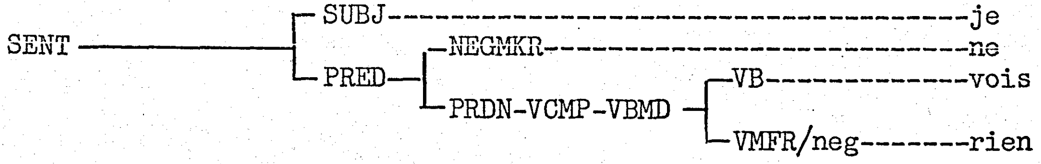
The interrogative, on the other hand, forces us not only to make decisions of a general nature before generation of the sentence, but also to make decisions of a particular nature early in the generation of the sentence. This is because the presence of the interrogative function can cause a word to take a position in the sentence that is not the one it would ordinarily take according to its other syntactic function.

The Negative

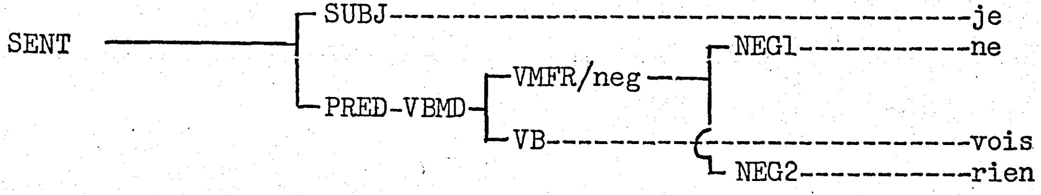
In teaching French to American students, I have often made the remark that a double negative is permitted in French, by contrast with English. I shall probably continue to do so in class, but I shall explain later, to interested students, exactly what I mean by that. Most important. I do not mean that the two negatives are *ne* and some other word. A true double negative in French is *Vous n'avez jamais rien fait*: (literally) "You have never done nothing", where *jamais* and *rien* constitute the double negative. *Ne* is simply a negative marker, with so little importance that its omission in modern colloquial French in no way affects the transmission of the message that one means "no" rather than "yes".

For this reason, *ne* is parsed in the grammar I have written as a negative marker, a sort of appendage at the front end of the predicate. It is generated at the time the predicate is about to be expanded, rather than as a constituent member of a negative expansion. Thus:

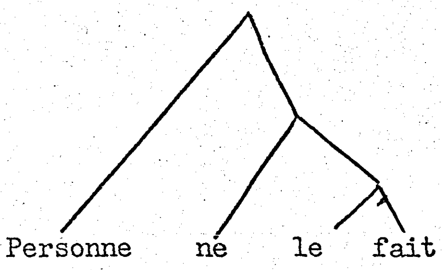
(This structure)



(Not this one)



This treatment of *ne* is consistent with the view that *ne* should be considered as a prefix to the verb, along with the object pronouns and *y* and *en*. If, for example, I parsed *personne* and *ne* as immediate constituents in the sentence, *Personne ne le fait*, I would very clearly be implying that *ne* was not a part of the predicate, which I believe it is, though a weak and unimportant part. To show the relationships properly, I diagram the sentence this way:



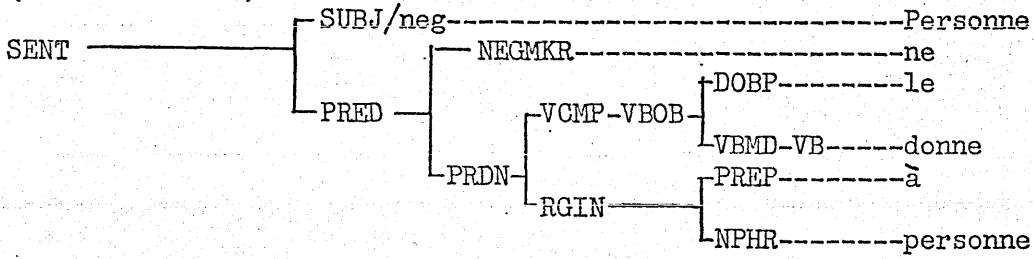
A difficulty arose in the programming of the grammar as a result of this way of parsing *ne*. Obviously, if I did not generate *ne* as a member of some negative group, *ne* . . . *pas*, there was the danger, in the program, of generating sentences with other negative words but no *ne*, and vice versa. This problem was resolved as soon as I started making use of the Grammar of Specifiers. If, in the Grammar of Specifiers, a decision is made to generate a negative clause, then provision is made to generate the negative marker in its proper place and to generate at least one other negative.

It is also in the Grammar of Specifiers that I handle the problem of the dual function of negative words. I have written the program so that first a decision is made as to whether or not *pas* will be generated. If so, no other negatives are permitted (I have not yet written rules for the *pas que* construction, but I am sure they will be compatible with the present framework). If not, one negative is chosen to be generated definitely, and the generation of other negatives, except *pas*,is made optional. For example, it may be decided to generate a negative subject (the specific choice between *rien* and *personne* remains optional; only the fact that the subject will carry the negative aspect is important, not the particular word used) which may, but need not, be followed by a negative object (direct or indirect) or a negative adverb. Possible combinations include:

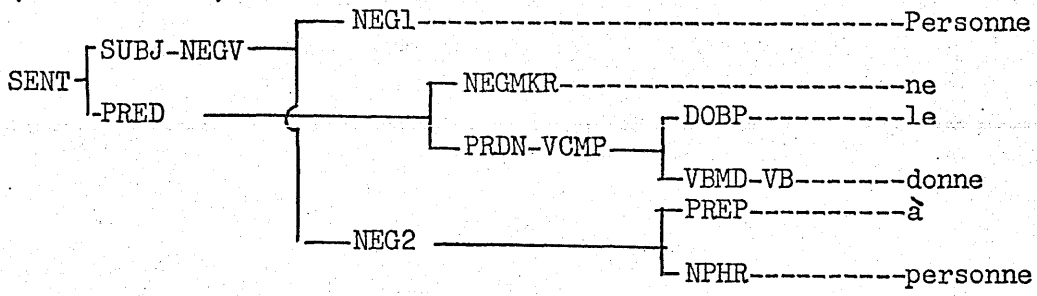
Personne ne le fait.
Personne ne le fait jamais.
Personne ne le donne à personne.
Personne ne m'a rien donné.

The other negatives, of course, are generated according to their proper syntactic functions, not as members of a negative group:

(This structure)



(Not this one)



It seems reasonable to generate the negatives according to their syntactic function since the position of a negative word in a given sentence is decided by its syntactic function, not by the fact that it is negative. The only apparent exception to this is <u>rien</u>. Although I have spoken of it as a substantive (functioning apparently as either subject or object), it does not take the position of an ordinary substantive object in sentences containing an analytic verb form:

(*rien* as object) Je n'ai *rien* donné à Jean.

(normal noun phrase) J'ai donné *le stylo* à Jean.

(pronoun object) Je *l*'ai donné à Jean.

The first of the sentences above could be presented in evidence for treating *rien* as an adverb. However, one of the earlier sentences produced by the grammar was:

* Elles ne la (direct object pronoun) ont rien donné.

In this sentence the incompatibility of *la* and *rien*, both as direct object, is quite obvious. We may want to consider *rien* as an adverb due to its position but it is evidently an "adverb" that places restrictions on other classes of words in the sentence, namely those that may function as the direct object. One solution is to insert a zero form among the direct object forms to provide for the object *structure*, without producing a final form, thus allowing *rien* to be generated in the position of an adverb while fulfilling the function of direct object.

Combinations such as *jamais plus* and *plus rien* may be generated by the grammar but I have not explicitly stated how I parse them, that is, as a string of adverbs, each modifying the verb, or as an adverb modified by another adverb. The way I have written the program permits the generation of groups of words such as these (also lists of adjectives, etc.) without restrictions, and I intend to leave the program this way until I have done further research on the problem of what combinations may be produced.

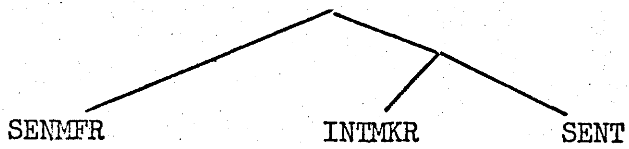
I said above that one negative word is chosen to be generated definitely and others remain optional (I gave the example of personne as the subject). This does not imply that I feel that the one chosen (the choice is optional) is "the" negative word nor that any others that may be generated are in any way less negative. It is simply necessary to be certain, once a negative sentence has been decided upon, to generate at least one negative particle other than ne. As the program is written now, the sentence, Personne ne le fait jamais, can be generated either after a negative subject (personne) has been made obligatory in the Grammar of Specifiers, with jamais appearing later, optionally, during the generation of the sentence, or vice versa. Neither resulting sentence is ambiguous as to "meaning" or grammatical relationship because the function of each word is explicitly noted in each case. The only possible ambiguity would be in the determination of which word (jamais or personne) carries the negative function, but I do not believe that either one does so to the exclusion of the other. Rather, the negative is pervasive and, with certain limitations, may appear in a number of different words and syntactic positions in the same sentence.

The Interrogative

Excluding intonation patterns, the interrogative is signalled in French by inverted word order, or est-ce que, or an interrogative word or expression, or by some combination of these. Because inverted word order is subject to a number of restrictions and because the ways in which the signalling devices just mentioned may be combined are limited, it is necessary to make a number of decisions about an interrogative sentence before it can be generated. For example, if we decide to provide for inverted word order, we must prevent the generation of _est-ce que_. A more complicated example will illustrate better why this decision is made in the Grammar of Specifiers. _Où va Jean_ is correct, that is, following the interrogative adverb _où_, inversion of the noun subject and the verb is permitted. However, both *_Où est Jean allé_ and *_Où est allé Jean_ are incorrect. The rule does not hold when the verb form is analytic. In such a situation, it is best to decide before generation of the sentence that, if the interrogative aspect is to be signalled by an adverb of the class that permits noun subject/verb inversion, then either 1) the verb will be synthetic and inversion of verb and noun subject will be provided for, or 2) the verb will _not_ be synthetic and inversion will _not_ be permitted.

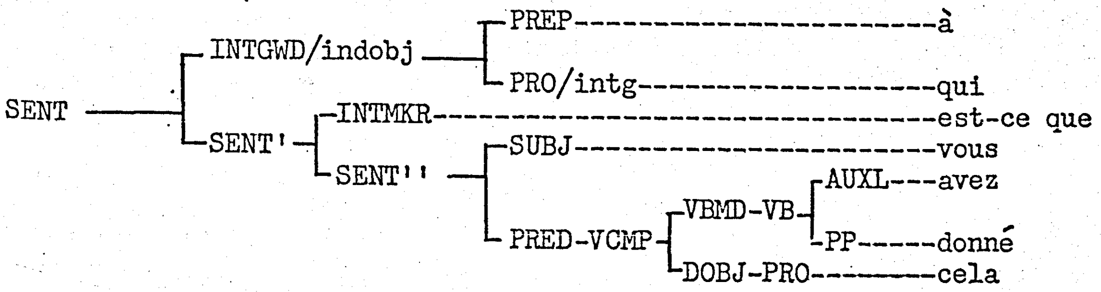
Aside from the rules governing the ways in which the interrogative signalling devices (interrogative words, _est-ce que_, or inversion) may be combined, there are rules determining the syntactic function to be assumed by the interrogative words or expressions, and, depending upon this function, determining the restrictions to be placed on other words or structures in the sentence. The interrogative pronouns, by their position and their form fulfill the interrogative "aspect" and at the same time assume the syntactic function of subject or object (direct, indirect, or object of a preposition). As subject, their position is normal; as object, any one of the three types, it is unique and it places limitations on the structures to be generated later. For example, if we choose to generate *qui* as the object, it takes initial position (not normal for the object) and, of course, the sentence must have a transitive verb. But the general rule, which I want to keep as generally applicable as possible, and which provides for the generation of transitive verbs, also provides for an object to be generated, in its proper place. Therefore, when the object function is chosen for *qui*, it is necessary to provide for the cancellation of the subsequent object form. This is done by inserting a zero form among the choices of objects, which is a generally applicable operation, because the same situation arises in relative clauses and in sentences containing negative objects.

An additional peculiarity of the interrogative words is the position and form that the marker *est-ce que* assumes when it is combined with them. Ordinarily, *est-ce que* takes initial position and thereby makes an otherwise declarative sentence interrogative. When combined with an interrogative word, instead of being initial in the sentence, *est-ce que* follows the interrogative word. The resulting structure would be simple if the interrogative word were always something that might be parsed as a sentence modifier giving,

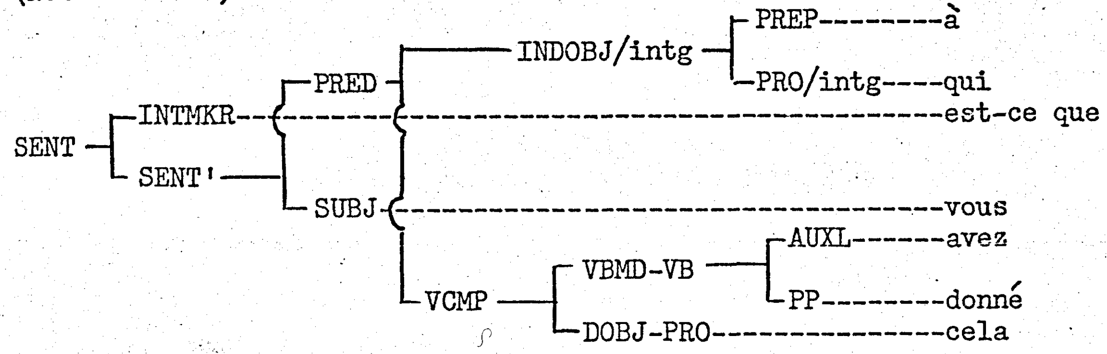


as, for example, in *Quand est-ce que vous faites cela?* But when the interrogative word is a constituent member of the sentence, such as the indirect object of the verb in, *A qui est-ce que vous avez donné cela?*, the *est-ce que* form definitely seems to disturb the structure of the sentence by separating the indirect object from the predicate, of which it is (by its function) an integral part, and from the subject. However, the interrogative marker belongs there, and since its position is peculiar, the rules governing its generation are decided upon in the Grammar of Specifiers. The interrogative words then, contrary to the negative words, are generated according to their function as interrogative markers rather than according to their syntactic function. The reason for this is that the interrogative "aspect" function is what determines their position in the sentence, just as the syntactic function determines the position of the negative words in a sentence. I have attempted to resolve all such problems consistently on this criterium: the constituent in question is generated according to that characteristic of it which determines its place in the sentence. For example, the sentence above is parsed as follows:

(This structure)



(Not this one)



It is interesting also that the form of the interrogative marker, est-ce que, varies in two cases: qu'est-ce qui (only subject form for inanimate referents) and qui est-ce qui (an alternate form for the subject function for animate referents). Since these are the only two instances, I have programmed the grammar so that each one is generated as a complete constituent with the proper form of qui (qu' or qui) at the time the animate or inanimate subject is generated.

The Relative Clause

The words which introduce relative clauses are similar to the negatives and interrogatives in that they assume two functions. Indeed, they are very similar to the interrogative pronouns and adverbs because their position in the sentence (and therefore, in the programmed grammar, the way they are generated) is dependent upon their function as relative clause markers. Here, as with the interrogative, it is necessary to make use of the zero notion to account for the syntactic functions that are assumed by the relative word. For example, when que, the object form of the relative pronoun, is generated, the object constituent within the relative clause must be generated as a zero constituent. The relative dont does not require use of a zero constituent, but does set definite restrictions on the word order of the remainder of the clause. These restrictions are clearly stated in a number of standard texts, [12] and they were included in the programmed grammar, as subscripts on the constituent RELSENT.

A characteristic which distinguishes the relatives from the interrogatives is the fact that they carry over the person and number (for verb agreement), and the gender and number (for agreement of participles and/or adjectives) of their antecedents. At the moment, this grammar takes the most recently generated noun of the previous phrase as the antecedent of the relative pronoun, and the subscripts of the pertinent characteristics are transferred from the antecedent to the relative pronoun. A major effect of the transfer of subscripts is that the subject-verb agreement, past participle agreement with preceding direct object or with subject, etc., are controlled by these transferred subscripts. The handling of subscript transfer requires a few extra rules in the routine for the relative clause, but most of the rules are controlled through the Grammar of Specifiers so that the relative sentence can then be generated using the same general rules as are used for other clauses.

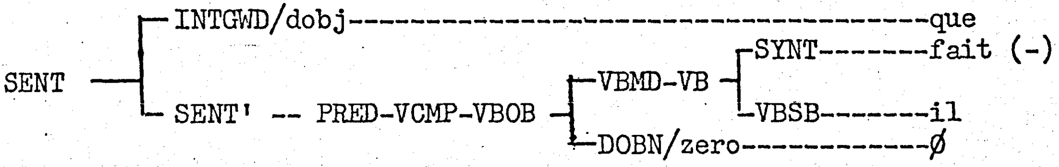
The Zero Constituent

In the past three sections I have mentioned the application of the "zero notion" and described, in each instance, why and how it was used in those particular circumstances. Perhaps a more general statement should be given here to sum up and to clarify the procedure.

In the generative grammar of French under consideration, decisions are made at each syntactic level as to which of the possible combinations of constituents will be generated. Syntactic level, as I explained in Chapter two, is roughly equivalent to a node in a tree structure, which is equivalent, in the program, to a rule which expands a given constituent into its two immediate constituents. For example, at the level VCMP, there is the possibility of expanding VCMP to VBMD + PNM (Predicate Nominative), or VBMD + DOBN (Direct Object, Noun), or a number of other combinations (See rules F0800 to F0830 in Appendix V).

At the time that one of these combinations is chosen rather than the others, subscripts are placed on the member constituents of the expansion. These subscripts restrict the way that the constituents may be further expanded according to their function as members of the combination just generated. For example, the verb involved in the expansion into VBMD + DOBN must necessarily be transitive.

Continuing with the same example, if an object has already been generated, as in the case of a relative pronoun object or an interrogative pronoun object, it will still be necessary to produce a transitive verb. I could, of course, add another choice to the rule VCMP, which would produce just VBMD, without an object constituent, but would at the same time place the required subscripts on VBMD. But it is not the VCMP function that the relative or interrogative pronoun has assumed: it is the object function, and so I feel it is more correct to avoid the creation of an extra subrule in VCMP and to place a zero choice in the rule for the expansion of the DOBN, as I have done in the following diagram:



The computer is directed to the subrule that creates a zero form only when the object function has already been assumed by a previously generated word. The same procedure is also used for subject functions.

Adverbs

In its present form, the programmed grammar will generate adverbial forms in four different parts of the sentence, that is, they are generated as the result of the expansion of the following syntactic constituents: Sentence Modifiers (at the beginning or at the end of the sentence), Predicate Modifiers (at the end of the predicate), and Verb Modifiers (at the end of the verb, including the option of taking a position either directly after the auxiliary or directly after the past participle in analytic tenses). One common type missing in the grammar is the adverbial form modifying an adjective. I omitted this type to save space; it can be added without difficulty.

The reason for categorizing the adverbials into the above-mentioned groups is the following. Adverbs can modify more than one type of construction, as is expressed in the usual grammar text definition: " . . . which modify verbs, adjectives, or other adverbs". [13] However, the texts do not go on to clarify the notion of adverbial modification, and particularly they do not discuss in detail the way that adverbs are related syntactically to the verbs that they modify. I have set up the above categories to reflect these various relationships.

The fact is that the same adverb (same form) can be found, in the same or similar context, in more than one position in the sentence, while still apparently modifying the verb. However, it has been noted that, in such situations, the adverb may have a different "meaning" (from the point of view of the emphasis implied or of the scope of modification) in each of these positions. I have chosen to generate the adverbs from these constituents (syntactic levels or nodes): Sentence Modifier, Predicate Modifier, and Verb Modifier, so as to be faithful to the facts and generate adverbs in each possible position, and also to provide material (i.e., sentences with their syntactic structures indicated) for further study of the problem of different meanings dependent upon different positions. This is one of the ways in which I hope to use the programmed grammar later as a research tool.

Direct and Indirect Object Personal Pronouns

The ordering of the personal pronouns when functioning as direct and indirect objects of the verb is always very difficult for students of French to handle. However, this difficulty, unlike many others met in the language, is not due to a lack of clear rules or to exceptions to the rules. It is simply difficult to apply the rules rapidly and with ease. It takes a long time for their use to become habitual. This is one of the reasons why students find repetition drills particularly useful in learning to use the object pronouns. The computer does not require repetition drills: it will always follow the rules once they are stated clearly and consistently.

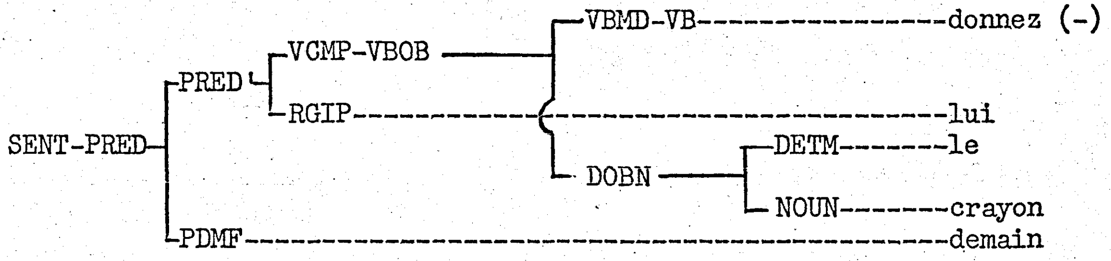
In my program, therefore, I was forced to state the rules for object pronoun order in the simplest and yet most detailed, complete way possible. I have decided to discuss this part of the program because it provides an interesting and complete example of how rules stated in English are translated into COMIT rules for the generative grammar.

Let's review the rules. There are two possible orderings of the personal pronouns. The first one is followed in every situation except the affirmative imperative (i.e., the negative imperative also follows the first, "normal" order). The pronouns are placed immediately in front of the verb (y and/or en, in that order, may intervene) and the first and second person (direct or indirect objects) precede the third person direct objects, which precede the third person indirect objects. Reflexive pronouns, direct or indirect, precede all others. The table below illustrates the order of pronoun objects:

SUBJECT	(ne)	me te (se) nous vous (se)	le la les	lui leur	y	en	VERB

The second ordering system is followed when the verb is in the affirmative imperative mood. The pronouns follow the verb directly (in the orthography, they are connected to it by hyphens) and direct object forms precede indirect object forms. Reflexive pronouns (direct or indirect) are not found here in combination with other personal pronouns. (**Lavez-vous-les* must be expressed *Lavez-vous les mains*.) *Y* and *en* are in final position. There are some morphophonematic alterations in certain of the pronouns in particular well-defined contexts, but I have not yet accounted for these changes in the program.

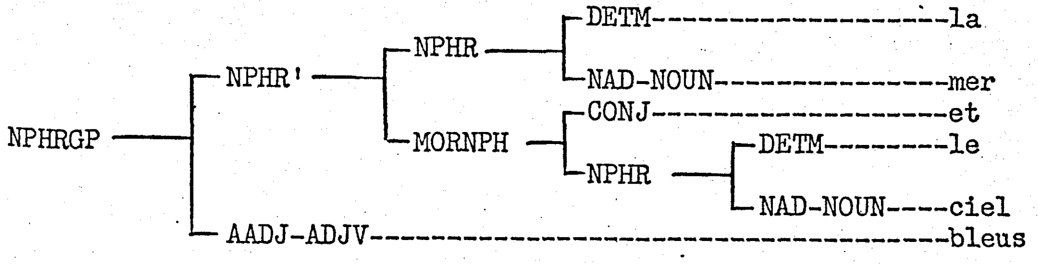
The program handles the word order for the affirmative imperative quite easily. One subrule in the PRED expansion rule expands PRED into VCMP + RGIP (Indirect Object, Pronoun). (The subrule is chosen under the control of a subscript added to the original clause symbol during the execution of rules in the Grammar of Specifiers.) If this rule is chosen (it is also possible to generate *no* indirect object form or a *noun phrase* indirect object), then the choice of subrules for expanding VBOB, if any, is limited (this time by a subscript added to VCMP at the time of the expansion of PRED to VCMP + RGIP) to one of two choices: VBMD + DOBP, or VBMD + Q + DOBN. The Q, as explained in Chapter three, will be replaced then by RGIP, effecting the discontinuous constituent structure of the verb and its direct object noun phrase, as in *Donnez-lui le crayon*. VBMD must be replaced by VB, with no modifiers. The fact that no modifiers can be generated from this syntactic level (VBMD) provides another example of why it is convenient to generate equivalent modifiers from the syntactic level, PRED (PRED + PDMF), as in the following sentence:



A similar, but much more complicated set of rules, involving additions of subscripts as particular choices are made in the expansion of the predicate, is applied in the program to assure adherence to the pronoun object order rules for all other types of sentences. I shall not burden the reader with the details of these rules. They are, of course, accessible to the interested reader in the listing of the program in Appendix V.

The Manifold Substantive

One type of structure which presented a difficulty in the programming of the grammar is the type illustrated by the following nominal phrase:



As can be seen in the diagram, this structure is not particularly complicated. There are, however, two interesting aspects and each requires a careful statement of the rules governing the permissible forms to be generated.

First, the form of the adjective which modifies the entire "noun phrase group", or manifold substantive, is determined by the gender of each of the nouns (the number, of course, is plural), following the same rules as apply to a predicate adjective modifying a compound subject. The routine I have written makes the choice of the possible combinations of nouns to be generated with each form of the adjective at the highest syntactic level, that is, at the node NPHRGP. The first choice is between generating all feminine nouns or a group including at least one masculine noun. Then, to permit optional generation of the various combinations available in the latter case, I have set up an optional choice between these situations: either all masculine nouns or any number of masculine nouns (but at least one) combined with any number of feminine nouns, in any order.

The second interesting aspect is the set of limitations that must be placed on the noun phrases that are members of the noun phrase group. This is one of the points that I intend to study, using the program as a tool. At present, for example, I do not permit the generation of any adjectives modifying a particular noun within the noun phrase group (the grammar will not generate _la mer et le joli ciel bleus_) and the only type of determiner permitted is the definite article. These restrictions are excessive, as are many of the restrictions in the grammar, but I prefer to keep all of the structures well-defined until I am ready to isolate one or more structures for a particular study. It is simple to remove restrictions, one by one, and to check the sentences produced after removal of a given restriction to determine whether or not the restriction should be maintained.

The Morphophonematic Routine

In the preceding chapters I have been principally concerned with the treatment and expansion of syntactic structures and have mentioned the generation of terminal structures, words, only when it was necessary for the explanation of some syntactic structure. However, I did state earlier that one reason why my generative grammar differs from other finite state grammars is that, in the generation of final words, it is in fact necessary sometimes to return and alter a word that has already been generated. In the present chapter, I give a detailed explanation of the routine in the program which accounts for all the changes necessary in the final shape of each word or group of words in the sentence.

The types of changes effected by the morphophonematic routine include elision, contraction and insertion of consonantal sounds to avoid cacaphony. (It should be noted that the routine is concerned with orthographic changes which *reflect* morphophonematic operations. The routine would be quite different for the spoken language.) The changes are made as each word is generated, not after the generation of the entire sentence, with the necessary exception that, whenever the change is dependent upon the form of the following word, no action is taken until that following word has also been generated. In one instance, that of contraction of the definite article with *de* and *à*, it is necessary to wait until a third word has been generated before making a potential contraction (*à* + *le* = *au*, but not when followed by a word whose first phoneme is a vowel, e.g., *à* *l'ami*).

The general operation of the routine is the following. After the generation of each terminal symbol (a word), control is directed to the rule "W" ("Shall we shelve this word and go on to the expansion of the next constituent?") which was initially set at "Y" ("Yes"). In most instances, of course, the answer is "yes" because there is most often no need to change the shape of the word at all. However, when a word is generated which may potentially be changed itself, depending upon the shape of the following word, or cause a change in the following word, rule "W" is reset at "N" ("No") and a set of questions are then asked about the structure of the word(s) involved in order to determine what changes, if any, should be made or if it will be necessary to wait for the generation of the next word. Rule "W" is reset at "Y" after either 1) the required change has been made and no further potential changes are under consideration, or 2) it has been determined that, in consideration of the shape of the following word, the change should not be made.

The particular situations that the program can handle at present are:

1) Elision of the final vowel of monosyllables (*que*, *me*, *se*, etc.; also *quoique*, *lorsque*, etc.) preceding a word whose initial phoneme is a vowel. The special case of *si*, the conjunction, which elides the "i" only when followed by another "i", is also handled.

2) Contraction of *à* and *de* with the definite article, *le* and *les*, and with *lequel*, *lesquels* and *lesquelles*.

3) Insertion of a consonant: ce to cet, ma to mon, sa to son, ta to ton.

I have not yet included the rules for adding a "t" between certain verb forms and a following subject pronoun (e.g., parle-t-il), nor have I provided for optional use of "l'" in such contexts as si on. These and any other additions to the routine that become necessary as the vocabulary items in the grammar are increased can be made without any difficulty.

Each lexical item in the grammar that may be involved in the morphophonematic routine contains one or more of the following subscripts, depending upon the shape of the word:

/E - Means that the word contains a final vowel that is subject to elision.

/C - Means that the word is subject to contraction with another word (following or preceding).

/S - Means that the word is subject to a special change (includes si, the demonstrative adjective ce, and the feminine singular forms of the possessive adjective).

/V - Means that the word has a vowel phoneme initially.

/P - Means that the word may have an influence on the preceding word.

/F - Means that the word may be affected by the following word. (Every time a word with an "F" subscript is generated, rule "W" is set at "N".)

The operations in the morphophonematic routine, therefore, consist of comparing the subscripts on adjacent words and making the necessary changes, or noting that no change need be made, or noting that it will be necessary to wait until the next constituent has been generated and then returning to the major routine. If the reader wishes to follow the step-by-step procedurc followed in the program, he may refer to Appendix V, Rules P0000 to P0310.

Conclusion

To summarize the major points discussed in the preceding pages, I shall discuss in this chapter the practical and theoretical importance of the grammar, both for mechanical translation and for linguistic research. In the introductory chapter, I listed three purposes for the thesis. Two of them are directly related to mechanical translation: utilization of the grammar as the final component of a complete mechanical translation program, and development of material for writing an analytic grammar (recognition routine) of French. The third purpose was to develop the grammar as a tool for research in syntax. In discussing the importance of the thesis, I shall also indicate how each of these purposes was fulfilled.

The practical importance of the completed grammar, in regards to mechanical translation, is that it does constitute a program which, with the proper operational rules, could be used as the final component of a translation scheme. The proof of this lies both in the fact that it has produced grammatically correct, structurally varied sentences and also in the flowcharts and rules themselves which, written in the COMIT notation, are accessible to researchers in the field of mechanical translation. Another practical result of my work is the print-out of sentences generated by the program, with structural descriptions. These sentences constitute data which may be used by research workers for many different purposes.

The principal theoretical importance of the thesis, for the field of mechanical translation, is that it provides evidence, by the existence of an effective grammar, that at least some of the grammatical sentences of French can be generated by a modified finite state grammar, operating from left to right. Although this does not constitute conclusive proof of the left-to-right hypothesis (since the grammar is not complete enough to produce, potentially, all and only the sentences of French) it does prove that left-to-right generative grammars can handle many varied structures. The theoretical importance of the left-to-right hypothesis (and therefore of any evidence in support of the hypothesis) is that it proposes a common framework in which to consider language structure, which is of vital importance in all basic research in mechanical translation.

I should like to emphasize one point in regards to the program as evidence in support of the left-to-right hypothesis. Although it is true that the grammar is not complete, it does provide a complete framework. At no time since I completed programming the framework of the grammar early in 1962 have I found it necessary to alter the framework (that is, the general order of rules in the program, which corresponds to my ordering of the syntactic hierarchy) in order to correct errors in the sentences generated. All grammatical errors detected thus far have been of a nature that could be corrected without making drastic changes in the program. If it were necessary to revise the grammar constantly at the higher syntactic levels (i.e., PRED, VCMP, RLCL, etc.) I would be less certain of the validity of the left-to-right hypothesis.

As for linguistic research, the grammar under consideration is of practical importance because it is a tool for the study of syntactic structures, just as the various electronic devices and their practical applications are tools for the study of the phoneme and particular features of the spoken language. The sentences already generated may serve as source material, or the program, with appropriate alterations, could be used to produce sentences illustrating specific structures under investigation. I should hope also that the detailed explanations in the preceding chapters would indicate to researchers how the facilities of COMIT and electronic computers could be used to help them in their linguistic research.

As evidence of the theoretical importance of the generative grammar in linguistics, I submit the development of the Grammar of Specifiers (Chapter three) and the study of the problem of word order, as exemplified frequently in Chapter five. I shall discuss each of these separately, although it is true that the use of the Grammar of Specifiers helped me to study and solve the word order problems that I considered in writing the left-to-right grammar.

The addition of a Grammar of Specifiers greatly simplified the grammar as a whole. I do not contend that simplicity by itself is a sufficient criterium by which to judge the theoretical importance of a component of a grammar. Two other factors attest to the importance of the Grammar of Specifiers. One is that the "specifiers" are principally concerned with the attitude of the speaker, that is, with an aspect of the sentence that may be denoted by some specific structure in the sentence but which is determined before the speaker utters the sentence. It would seem, for example, that the speaker determines he will utter a command before he starts the sentence, since the choice of an imperative has a critical effect on the word order. The same is true for the interrogative and negative. The other factor is that "specifiers" are of a somewhat abstract nature: their <u>effect</u> on word order and on the choice of particular words, the most concrete level, is quite clear, but the specifiers themselves are formed at the highest syntactic level, at the node SENTENCE. This means not only that they can be easily manipulated but also that they can be more easily compared with the specifiers of other languages: it is simpler to discuss the concept, "interrogative", in Language A and Language B than it is to discuss a particular interrogative in Language A as it compares with a particular interrogative in Language B.

Despite its importance, word order has always been one of the more poorly defined problems in syntax. By following the left-to-right system of generation, and by rigorously limiting myself to binary expansions (dividing each syntactic structure into its two immediate constituents) and permitting discontinuous constituents to be separated by only one syntactic constituent, I have arbitrarily defined the word order rules for my generative grammar. It is consistent with this system, therefore, to generate certain interrogative words, relative pronouns, and the pleonastic pronoun subject from a node in the phrase structure tree which functions primarily as an "interrogative marker" or "relative clause marker" or "verb-subject type pronoun" rather than from the node which functions as the marker of the Subject, Object, Adverb, Indirect Object, etc. Although it is not a discovery to observe that word order is affected by the introduction of an interrogative, etc., the way I have organized the rules pertaining to these structures in the grammar is novel. In my generative grammar, rather than describing the changes in word order as exceptions to a general rule, I have argued that the position of a given word is _regularly_ dependent on either its syntactic function or its secondary function, and that the function that will prevail in any given sentence must be decided before the sentence is generated.

In conclusion, the fundamental importance of any type of machine-oriented grammar is that the very nature of the tool the linguist is using - the computer - forces him to state his rules clearly and keeps him aware of the fact that his rules will be applied rigorously. It is possible for the linguist-programmer to write _ad hoc_ rules to solve particular problems, but such rules will prove themselves inapplicable as soon as the program is expanded. Thus, a complete, all-inclusive concept of language structure, such as the left-to-right hypothesis, is necessary for the efficient operation of a generative grammar.

Notes

1. Generative grammars have been described by:
Chomsky, Noam, *Syntactic Structures* (The Hague, 1957)
Yngve, Victor H., "A Model and an Hypothesis for Language Structure,"
Technical Report 369, M.I.T. (Cambridge, 1960).

2. Wells, R.S., "Immediate Constituents," *Language* 23.81-117 (1947).

3. Yngve, Victor H., "A Model and an Hypothesis."

4. The COMIT system was a joint project of the Mechanical Translation Group of the Research Laboratory of Electronics and the Computation Center at M.I.T. There exists a "Programmers' Reference Manual," published by the MIT Press (Cambridge, 1962).

5. The translation scheme I have described is similar to the one discussed in: Yngve, V.H., "A Framework for Mechanical Translation," *Mechanical Translation* 4.59-65 (1957)

6. Chomsky, *Syntactic Structures*;
Tesniere, Lucien, *Elēments de Syntaxe Structurale* (Paris, 1959).

7. Chomsky, Chap. 3.

8. In particular:
de Boer, Cornelis, *Syntaxe du Français Moderne* (Leiden, 1947);
Togeby, Knud, *Structure immanente de la langue française*, Travaux du Cercle Linguistique de Copenhague, Vol. VI (Copenhague, 1951);
Wartburg, W. and Zumthor, P., *Prēcis de Syntaxe du Français Contemporain* (Berne, 1947).

9. COMIT, "Programmers' Reference Manual," pp 22-25.

10. A similar concept, but with reference to transformational grammars, is held by E.S. Klima of M.I.T.

11. COMIT, p 6.

12. Fraser, W. and Squair, F. and Parker, C., *French Composition and Reference Grammar* (Boston, 1942), pp 441,2.

13. This, of course, is not a direct quotation. My sources are the many grammar texts I have used both as a student and teacher in French and English.

Bibliography

Boer, Cornelis de, *Syntaxe du Français Moderne* (Leiden, 1947).

Brunot, Ferdinand, *Précis de Grammaire Historique de la Langue Française* (Paris, 1899).

Chomsky, Noam, *Syntactic Structures* (The Hague, 1957).

Ewert, Alfred, *The French Language* (Longon, 1933).

Fraser, Squair and Parker, *French Composition and Reference Grammar* (Boston, 1942).

Grevisse, *Le Bon Usage: Grammaire Française avec des Remarques sur la Langue Française d'Aujourd'hui* (Paris, 1961).

Tesniere, Lucien, *Eléments de Syntaxe Structurale* (Paris, 1959).

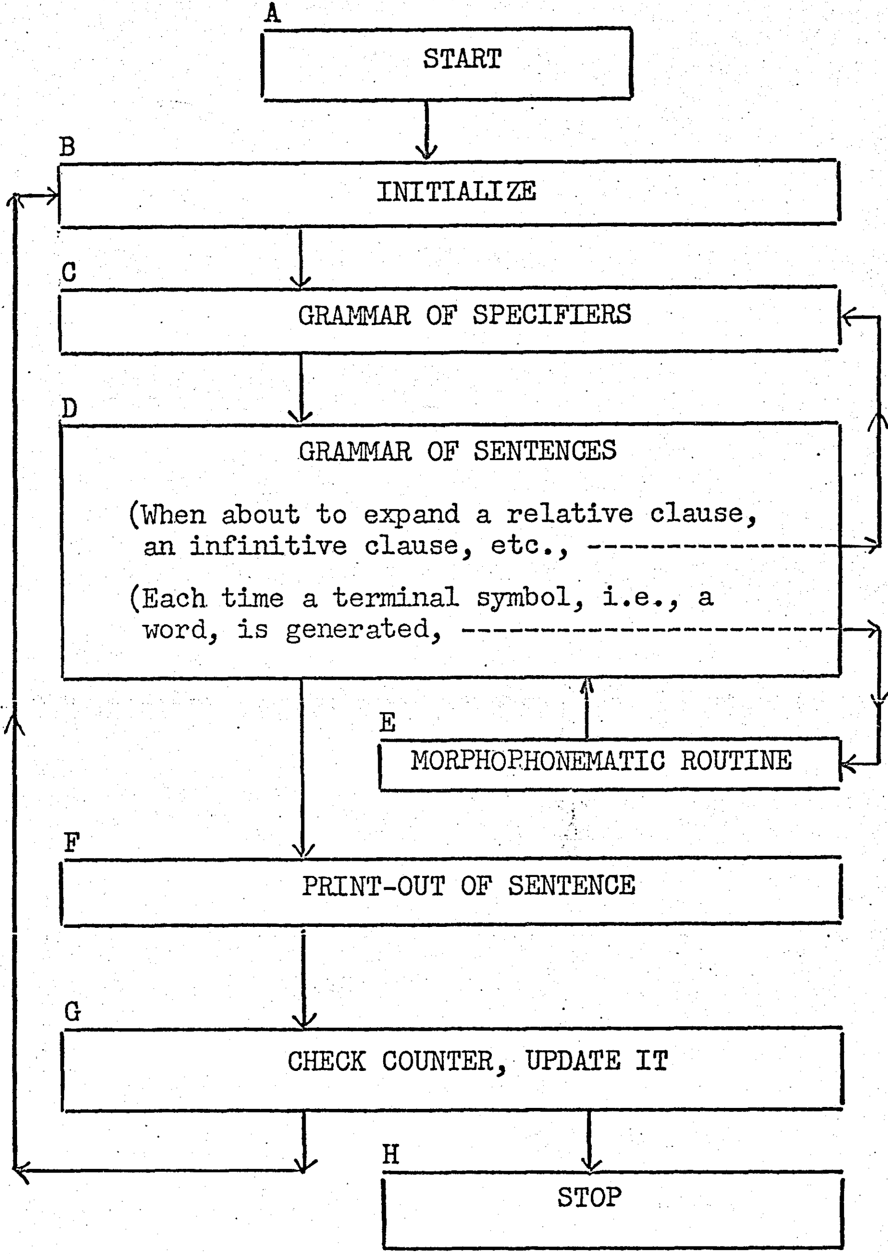
Togeby, Knud, *Structure immanente de la langue française*, Travaux du Cercle Linguistique de Copenhague, Vol. VI (Copenhague, 1951).

Wartburg, W. and Zumthor, P., *Précis de Syntaxe du Français Contemporain* (Berne, 1947).

Wells, R.S., "Immediate Constituents," *Language* 23.81-117 (1947).

Yngve, V.H., "A Model and an Hypothesis for Language Structure," Technical Report 369, M.I.T. (Cambridge, 1960).

The "Big Picture" of the Program *



* Appendix II gives a more detailed explanation of the operations that take place within each box in the "big picture".

II - A. START

1. Print out, on line, the title and code number of the program.

2. Print out, off line, the time the program is starting.

3. Initialize the counter (Shelf No. 2) at 1, and place (001) on Shelf No. 4, ready to be printed out at the head of the first sentence generated.

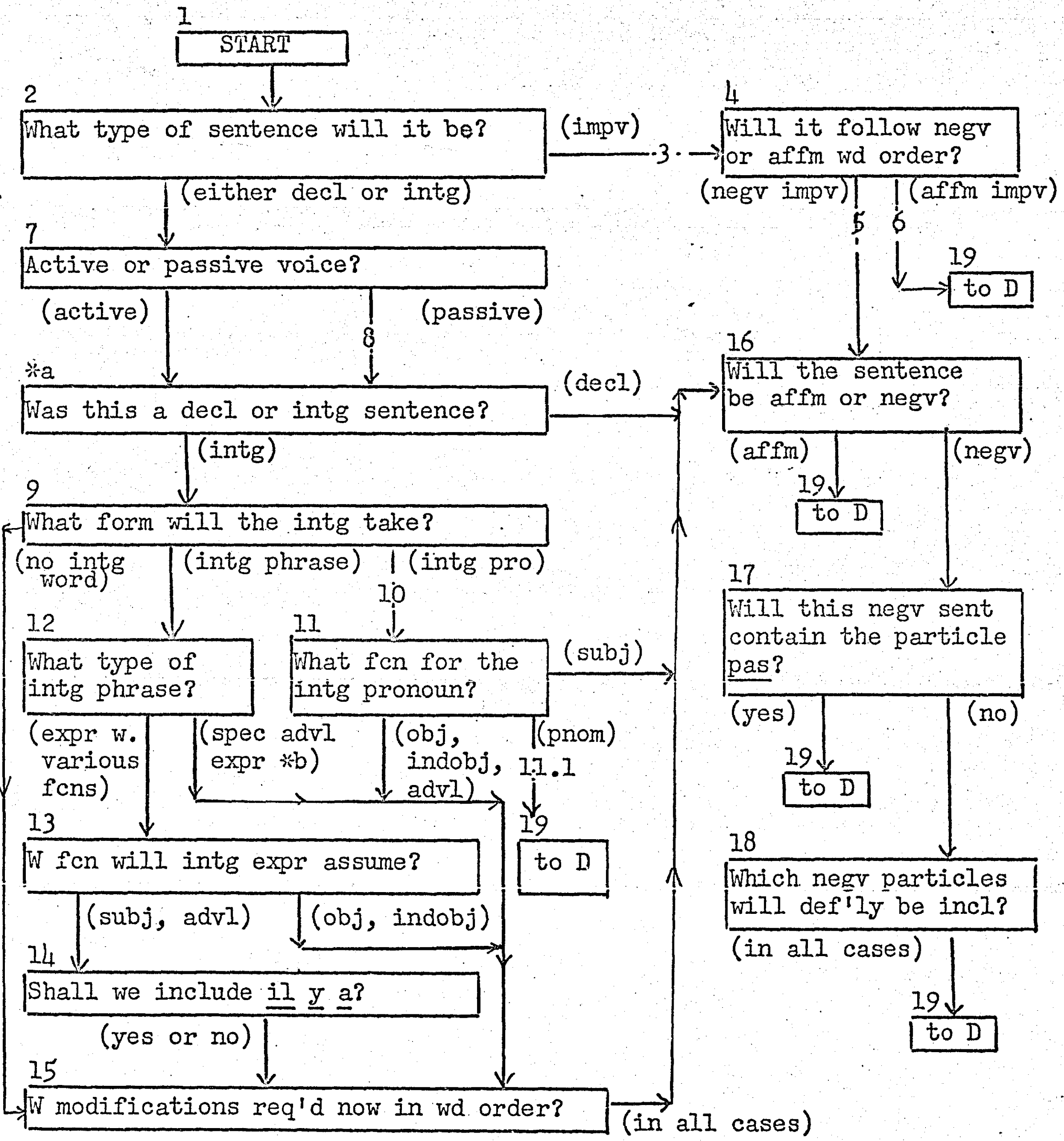
II - B. INITIALIZE

1. Set rule W at Y, that is, assume that any terminal symbol generated will not require any morphophonematic alteration. (It is reset at N only when such a word is generated: see Appendix II - E and/or Chapter five.)

2. Set rule UTT at A, that is, provide for a full sentence, with end punctuation and print-out of the sentence.

3. Add any special restrictions that may be desired for a particular run of the program, for example, when studying the interrogative, I can arrange to generate only interrogative sentences.

II - C. GRAMMAR OF SPECIFIERS

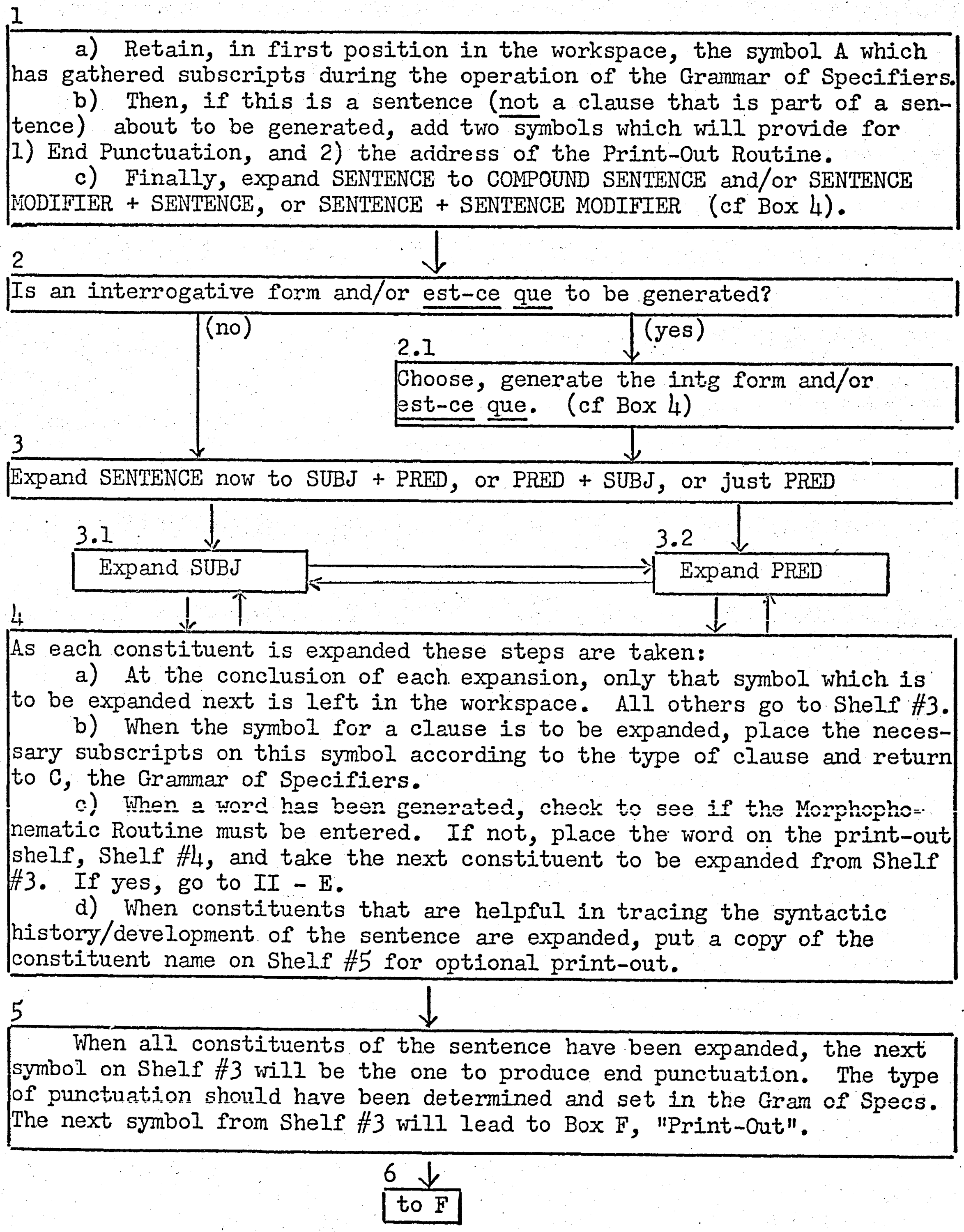


*a - In the program this is not a separate step, therefore no reference number is given for the step-by-step description.

*b - That is, the type that causes inverted word order.

NB - The numbers at each box correspond to the numbers in the step-by-step description in Appendix IV.

II - D. GRAMMAR OF SENTENCES



1

a) Retain, in first position in the workspace, the symbol A which has gathered subscripts during the operation of the Grammar of Specifiers.

b) Then, if this is a sentence (not a clause that is part of a sentence) about to be generated, add two symbols which will provide for 1) End Punctuation, and 2) the address of the Print-Out Routine.

c) Finally, expand SENTENCE to COMPOUND SENTENCE and/or SENTENCE MODIFIER + SENTENCE, or SENTENCE + SENTENCE MODIFIER (cf Box 4).

2

Is an interrogative form and/or est-ce que to be generated?

(no)

(yes)

2.1

Choose, generate the intg form and/or est-ce que. (cf Box 4)

3

Expand SENTENCE now to SUBJ + PRED, or PRED + SUBJ, or just PRED

3.1

Expand SUBJ

3.2

Expand PRED

4

As each constituent is expanded these steps are taken:

a) At the conclusion of each expansion, only that symbol which is to be expanded next is left in the workspace. All others go to Shelf #3.

b) When the symbol for a clause is to be expanded, place the necessary subscripts on this symbol according to the type of clause and return to C, the Grammar of Specifiers.

c) When a word has been generated, check to see if the Morphophonematic Routine must be entered. If not, place the word on the print-out shelf, Shelf #4, and take the next constituent to be expanded from Shelf #3. If yes, go to II - E.

d) When constituents that are helpful in tracing the syntactic history/development of the sentence are expanded, put a copy of the constituent name on Shelf #5 for optional print-out.

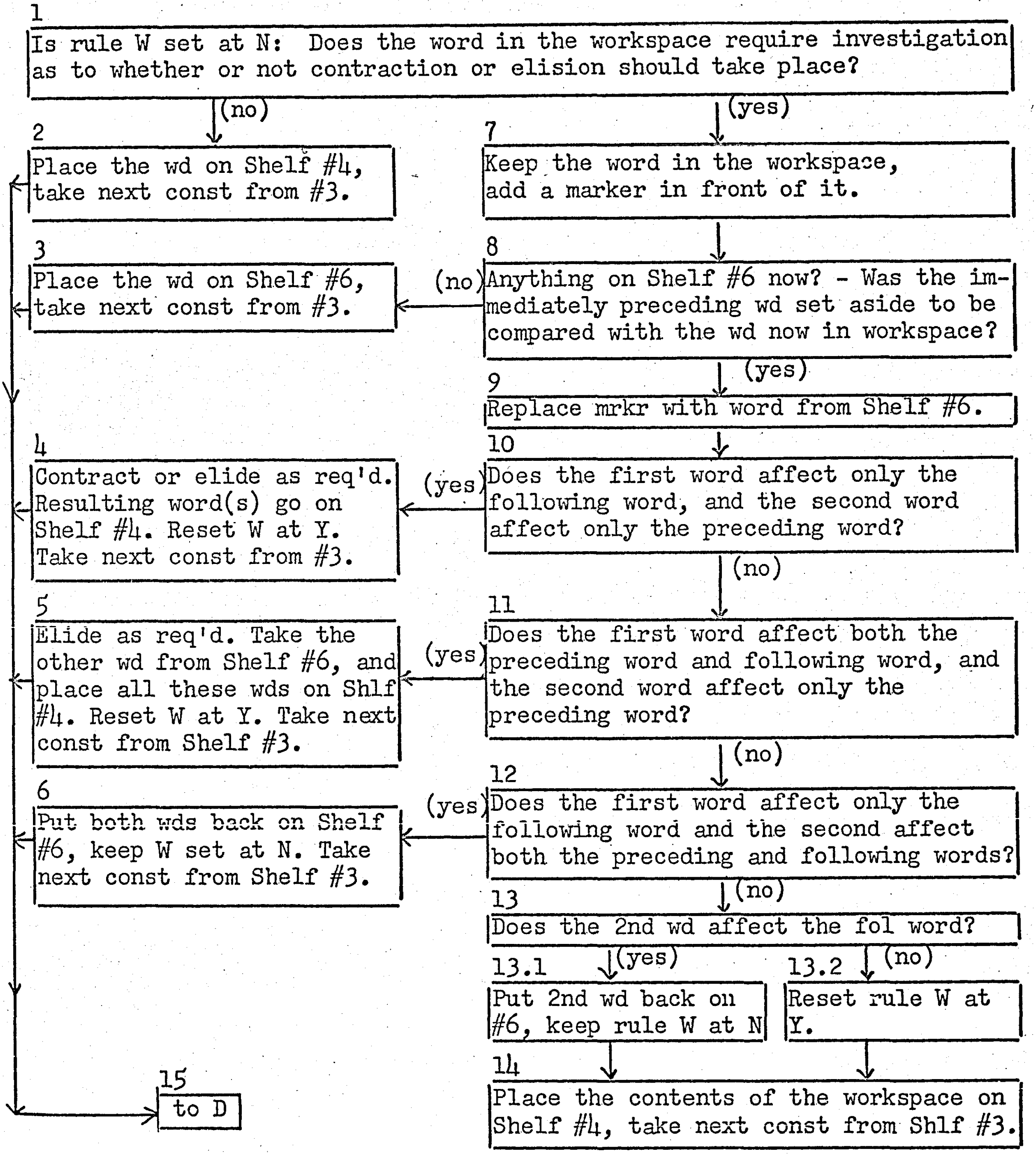
5

When all constituents of the sentence have been expanded, the next symbol on Shelf #3 will be the one to produce end punctuation. The type of punctuation should have been determined and set in the Gram of Specs. The next symbol from Shelf #3 will lead to Box F, "Print-Out".

6

to F

II - E. MORPHOPHONEMATIC ROUTINE


1
Is rule W set at N: Does the word in the workspace require investigation as to whether or not contraction or elision should take place?
(no)
(yes)
2
Place the wd on Shelf #4, take next const from #3.
7
Keep the word in the workspace, add a marker in front of it.
3
Place the wd on Shelf #6, take next const from #3.
(no)
8
Anything on Shelf #6 now? - Was the immediately preceding wd set aside to be compared with the wd now in workspace?
(yes)
9
Replace mrkr with word from Shelf #6.
4
Contract or elide as req'd. Resulting word(s) go on Shelf #4. Reset W at Y. Take next const from #3.
(yes)
10
Does the first word affect only the following word, and the second word affect only the preceding word?
(no)
5
Elide as req'd. Take the other wd from Shelf #6, and place all these wds on Shlf #4. Reset W at Y. Take next const from Shelf #3.
(yes)
11
Does the first word affect both the preceding word and following word, and the second word affect only the preceding word?
(no)
6
Put both wds back on Shelf #6, keep W set at N. Take next const from Shelf #3.
(yes)
12
Does the first word affect only the following word and the second affect both the preceding and following words?
(no)
13
Does the 2nd wd affect the fol word?
13.1
(yes)
Put 2nd wd back on #6, keep rule W at N
13.2
(no)
Reset rule W at Y.
14
Place the contents of the workspace on Shelf #4, take next const from Shlf #3.
15
to D

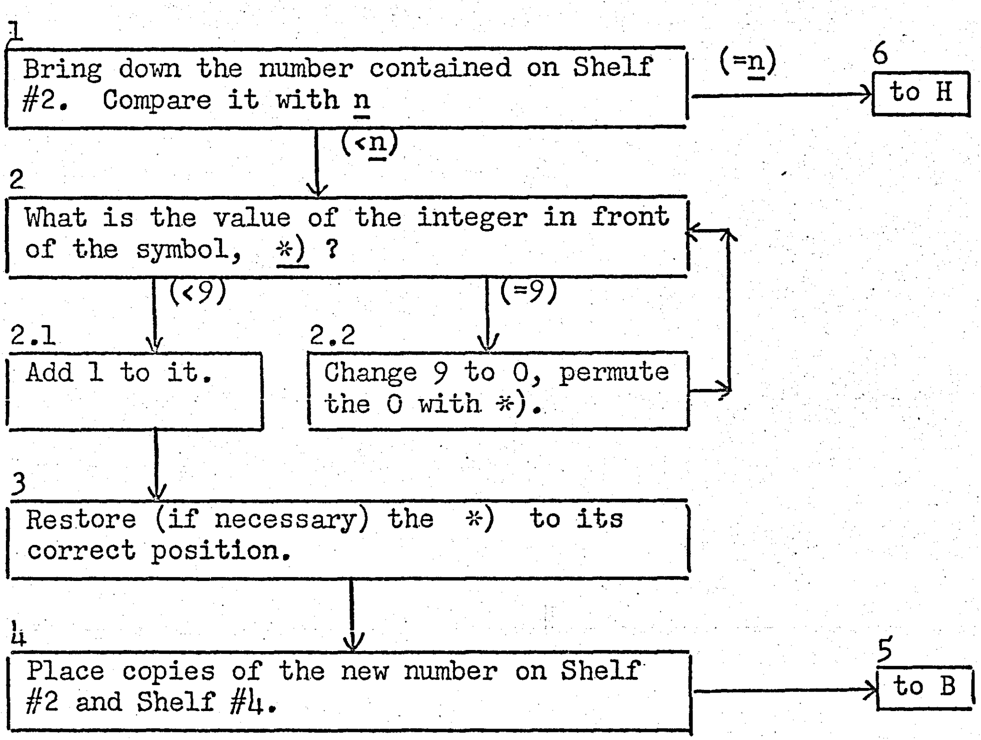
Shelf #3 is the shelf which contains the const's still to be expanded.
Shelf #4 contains the words ready for print-out.
Shelf #6 contains the words subject to elision or contraction after comparison with words still to be generated.

II - F. PRINT-OUT OF SENTENCE

1. Place the contents of Shelf #4 in the workspace and print them out in Format A, that is, as words with normal spacing between each word.

2. Optionally, place the contents of Shelf #5 (the one containing trace items - node names) in the workspace and print them out also in Format A.

II - G. CHECK COUNTER, UPDATE IT


1
Bring down the number contained on Shelf #2. Compare it with n
(=n)
6
to H
(<n)
2
What is the value of the integer in front of the symbol, *) ?
(<9)
(=9)
2.1
Add 1 to it.
2.2
Change 9 to 0, permute the 0 with *).
3
Restore (if necessary) the *) to its correct position.
4
Place copies of the new number on Shelf #2 and Shelf #4.
5
to B

II - H. STOP

1. Print out the time that this run of the program is stopping.

2. Stop the run by use of an END card.

III - GRAMMAR OF SENTENCES (Explanation)

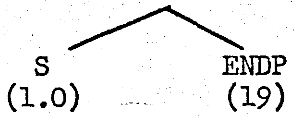
The following diagrams are not the flowcharts from which the program was written. They have been devised specifically for the purpose of explaining the operation of the major part of the program (that is, just the Grammar of Sentences) and do not necessarily include many details that would be important for the programmer.

Each set of diagrams illustrates the possible combinations of expansions of the constituent whose name appears in the upper left corner of the diagram. Each set is numbered for easy reference.

The constituent members of each expansion may be further expanded; the number below each constituent name will direct the reader to the set of expansions for that constituent. Of course, a given constituent in the actual program may be limited to a specific subset of these expansions according to the previous history of the sentence, including decisions made in the Grammar of Specifiers. Some of these limitations are indicated but not all of them could be included without introducing many cumbersome details. Such details, however, can be found in the program, Appendix V.

Diagrams of Expansions | Rule Numbers

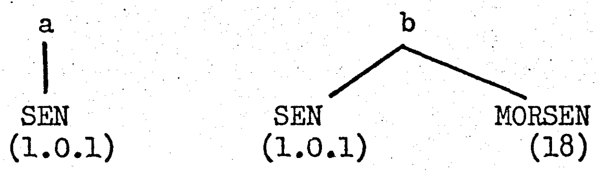
0. UTTerance



(An optional print-out rule, D0030, will give a list of the subscripts developed in the Grammar of Specifiers)

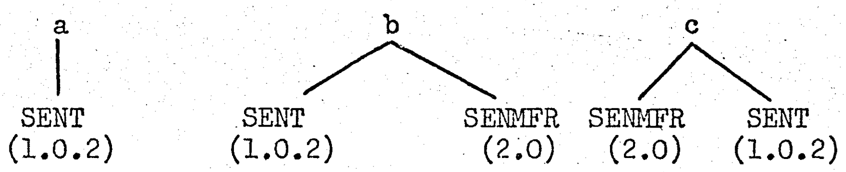
D0000 to D0022

1.0 S (Choose between simple and compound sentence)

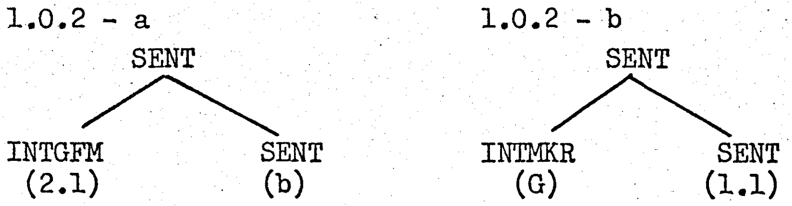


D0040 to D0062

1.0.1 SEN (Optional addition of a sentence modifier)



D0080 to D0104

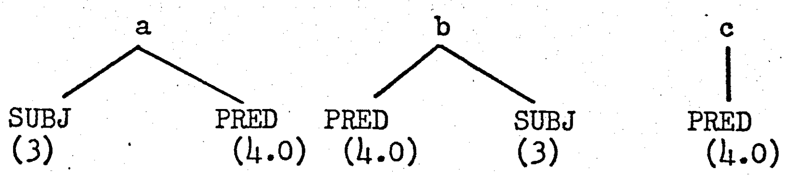


(At this point, after the generation of any interrogative forms, the gender, person and number of the subject is chosen and appropriate subscripts added to SENT)

D0120 to D0182

(D0200 to D0262)

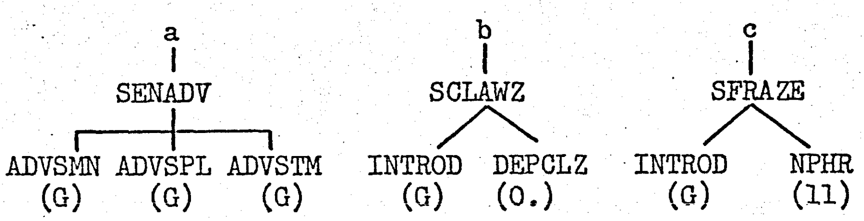
1.1 SENTence



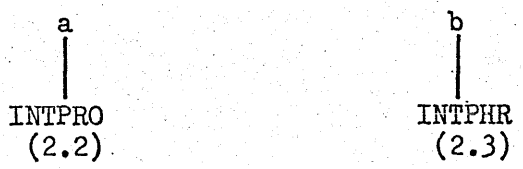
D0280 to D0308

Diagrams of Expansions | Rule Numbers

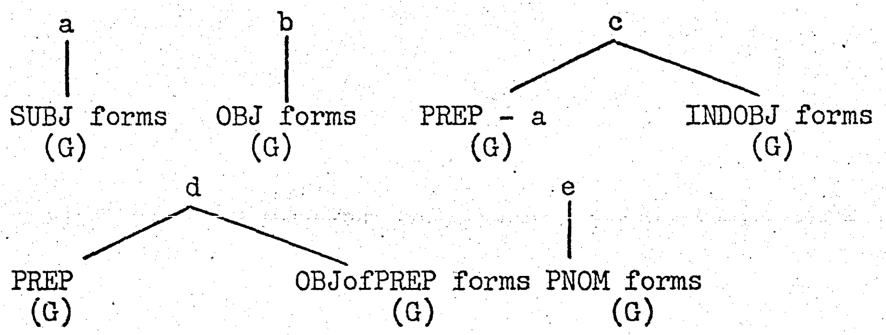
2.0 SENMFR (Sentence Modifier) — D0320 to D0600



2.1 INTGFM (Interrogative Forms) — D0602 to D0634



2.2 INTPRO (Interrogative Pronouns) — D0650 to D0784

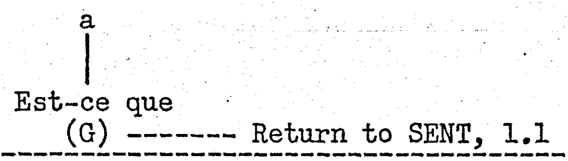


2.3 INTPHR (Interrogative Phrases) — D0900

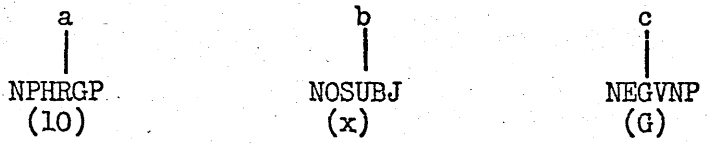
Not programmed. Prints out a remark.

(Note: After 2.2 and 2.3, go on to 2.4, INTMKR)

2.4 INTMKR (Interrogative Marker) — D0160 to D0182



3.0 SUBJ (Subject) — E0000 to E0024



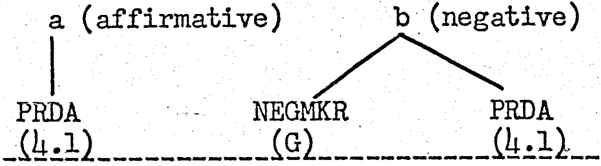
(Before proceeding to NPHRGP from SUBJ, a check is made of the gender, number and person settings made earlier) — (E0030 to E0042)

Diagrams of Expansions

Rule Numbers

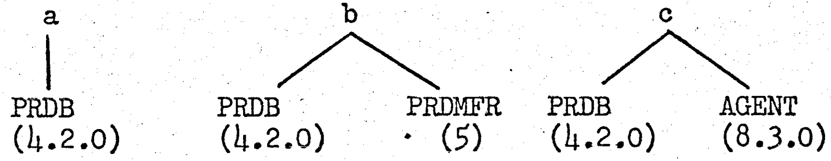
4.0 PREDicate

F0500 to F0542



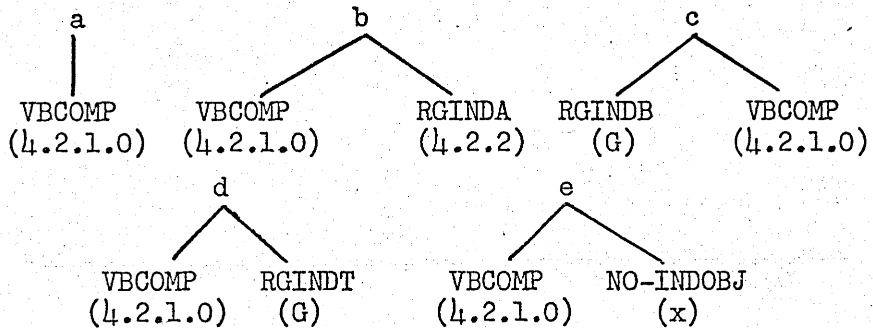
4.1 PRDA (Predicate Modifier or Agent added optionally)

F0580 to F0604



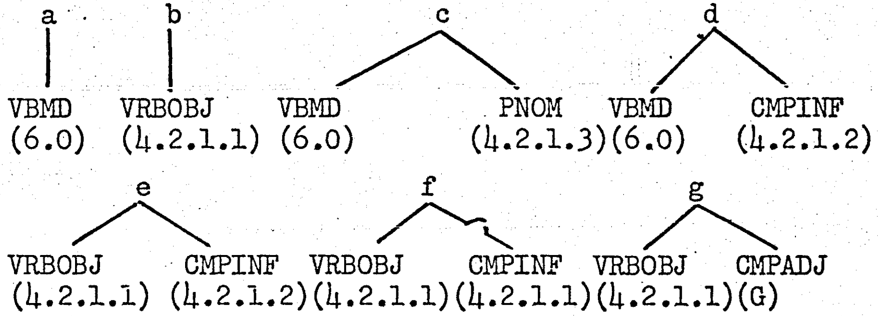
4.2.0 PRDB (Choice of various VCMP and Indobj combinations)

F0620 to F0670



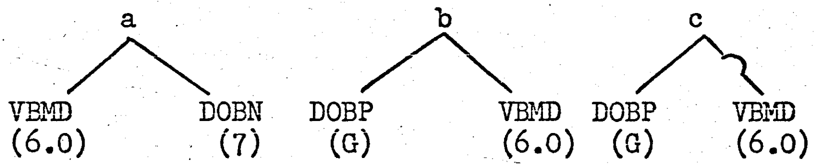
4.2.1.0 VBCOMP (Verb with its Complements)

F0700 to F0830



4.2.1.1 VRBOBJ (Verb with its objects - direct)

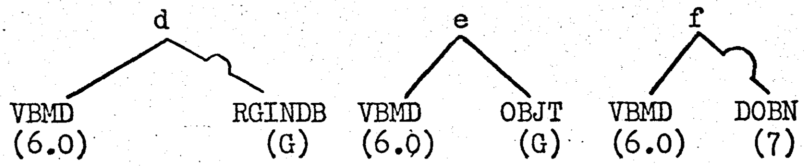
F1000 to F1025



d,e,f on the following page.

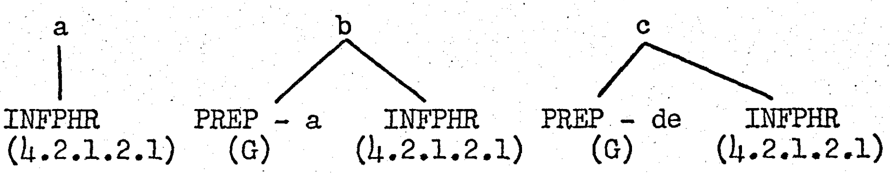
Diagrams of Expansions | Rule Numbers

4.2.1.1 (continued)



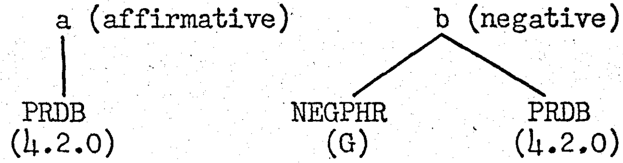
4.2.1.2.0 CMPINF (Complementary Infinitive Construction)

I0000 to I0054



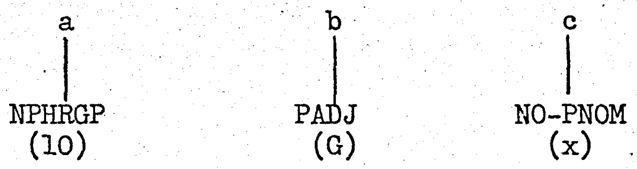
4.2.1.2.1 INFPHR (Infinitive Phrase)

I0100 to I0154



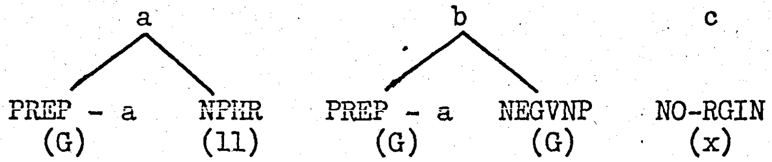
4.2.1.3 PNOM (Predicate Nominative Construction)

H0000 to H1002



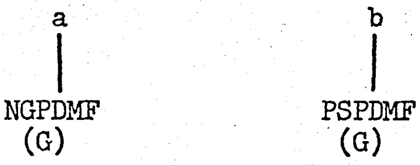
4.2.2 RGINDA (Indirect Object, Noun Phrase)

H4000 to H4056



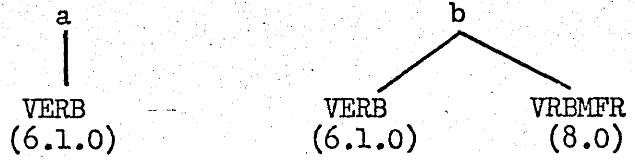
5. PRDMFR (Predicate Modifier)

F1380 to F1482



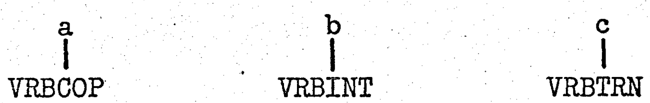
Diagrams of Expansions | Rule Numbers

6.0 VBMD (Verb plus its modifiers construction)



F1100 to F1205

6.1.0 VERB

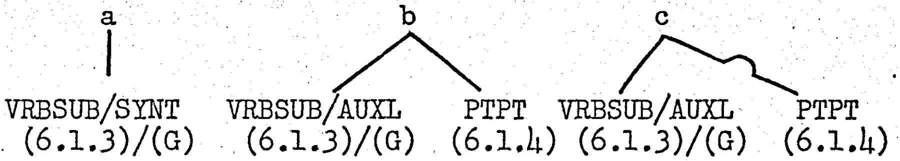


Choose the specific verb, generate the subscripts to be carried through the following routines.

(6.1.1) (6.1.1) (6.1.2.0)

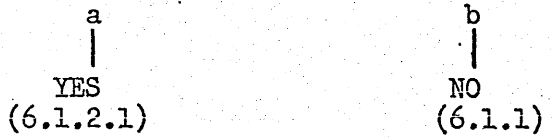
G0000 to G0004
for a:
G0050
for b:
G0100 to G0202 and L0400 to L0504
for c:
G0250 to G0402 and L0000 to L0304

6.1.1 VBFORM (Form of the verb - analytic or synthetic)



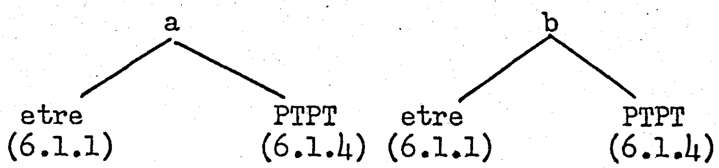
G0700 to G0704

6.1.2.0 PASSIVE? (Passive or Active Voice)



G0450 to G0502

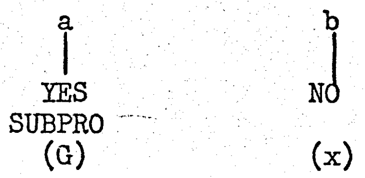
6.1.2.1 FIXPSV (Check for discontinuous constituent)



G0600 to G0602

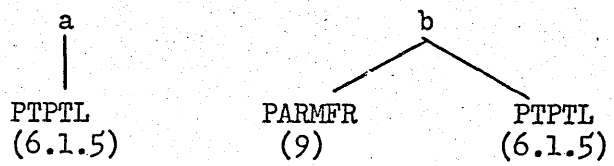
Diagrams of Expansions	Rule Numbers

6.1.3 VRBSUB (Check if pleonastic pronoun required)



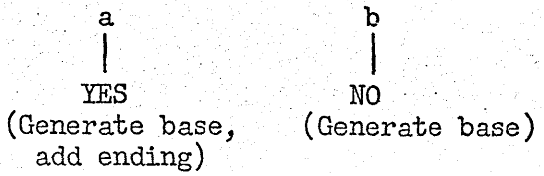
G0900 to G0902

6.1.4 PTPT (Optional addition of participial modifier)



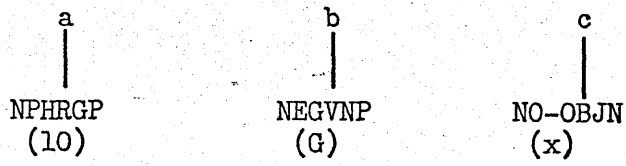
G1000 to G2002

6.1.5 PTPTL (Check for agreement requirements)



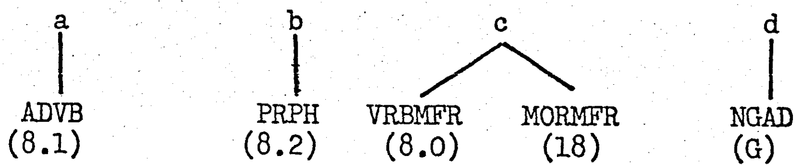
G2200 to G4102

7. DOBN (Direct Object, Noun Phrase)



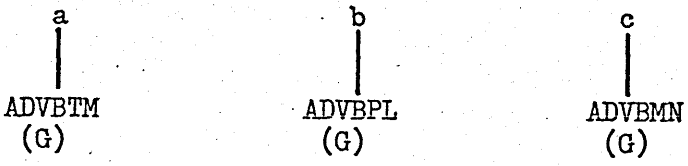
H3000 to H3084

8.0 VRBMFR (Verb Modifier)



F1250 to F1354

8.1 ADVB (Choice of type of adverb)

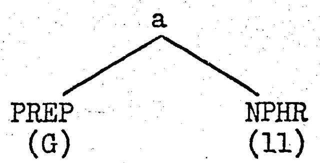


F2000 to F2202

Diagrams of Expansions | Rule Numbers

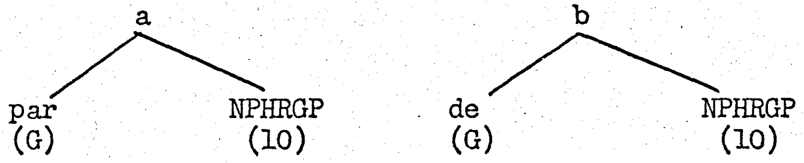
8.2 PRPH (Prepositional Phrase)

F0000 to F0048



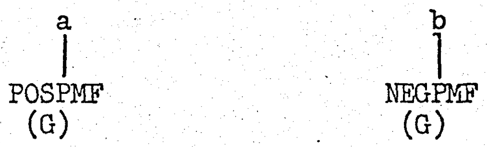
8.3 AGENT-FORM (Form of the agent in passive const.)

F3000 to F3022



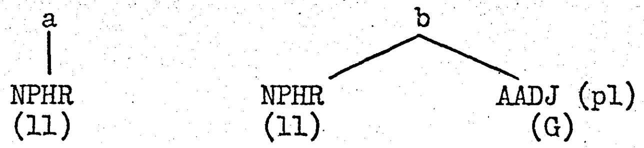
9. PARMFR (Participial Modifier)

G2020 to G2152



10. NPHRGP (Noun Phrase Group)

E0100 to E0162

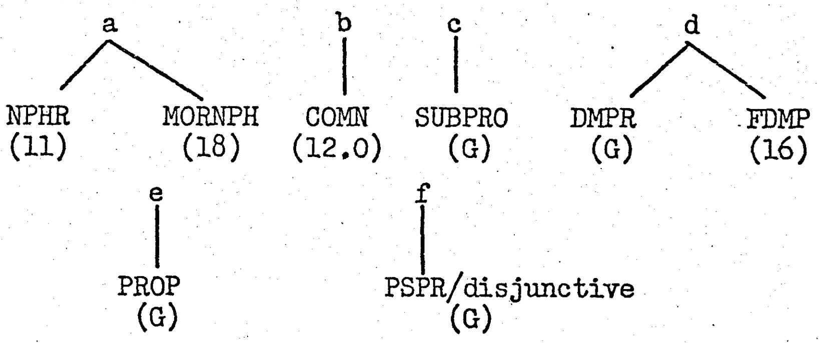


(If gender, number and person have not already been set - e.g., from SUBJ - they are set now before going to NPHR

(E0180 to E0188)

11. NPHR (Noun Phrase)

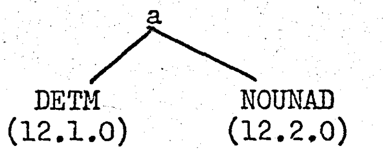
E0200 to E0234



Diagrams of Expansions | Rule Numbers

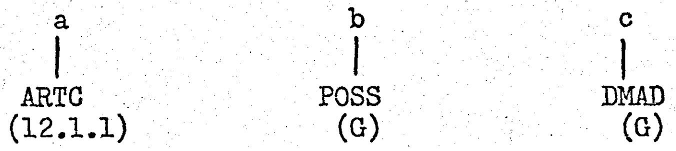
12.0 COMN (Common Noun Phrase)

E1270 to E1290



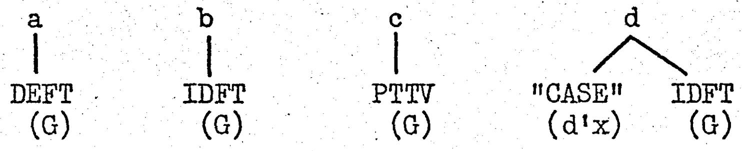
12.1.0 DETM (Determiners)

E2500 to E2554



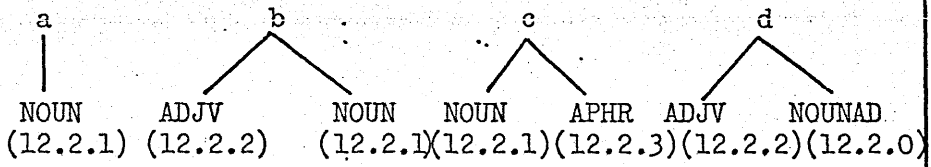
12.1.1 ARTC (Articles)

E2580 to E2780



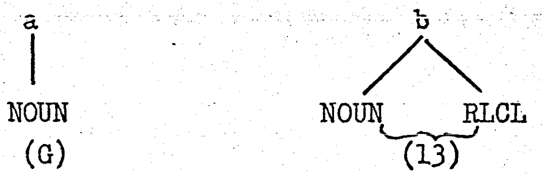
12.2.0 NOUNAD (Noun with its modifiers)

E1300 to E1362



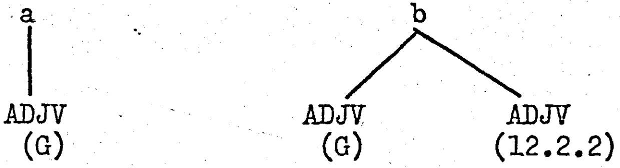
12.2.1 NOUN

E1700 to E2022



12.2.2 ADJV (Adjectives)

E1480 to E1680

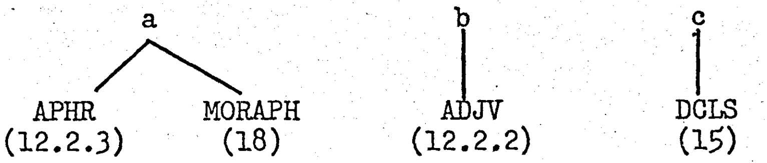


Diagrams of Expansions

Rule Numbers

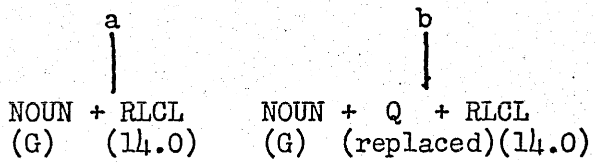
12.2.3 APHR (Adjectival Phrases)

E1440 to E1444



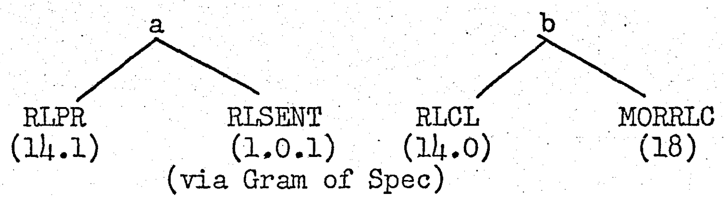
13 NOUN + RLCL (Works on both together for proper arrangement of potential discontinuous constituents.)

K0000 to K0052



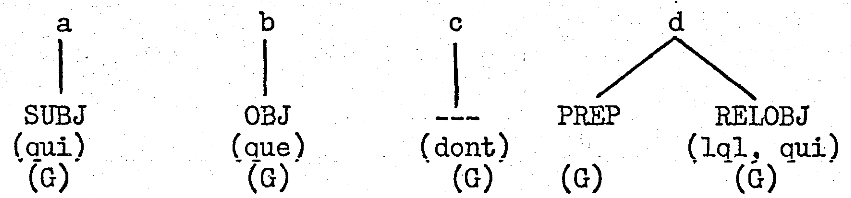
14.0 RLCL (Relative Clause)

K0060 to K0102



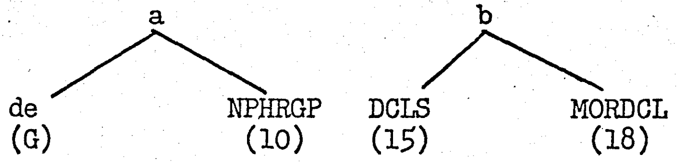
14.1 RLPR (Relative Pronouns)

K0200 to K0402



15 DCLS (de plus Noun Phrase)

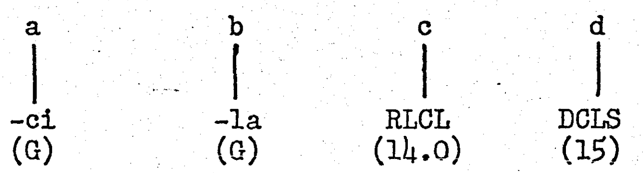
E0260 to E0282



Diagrams of Expansions	Rule Numbers

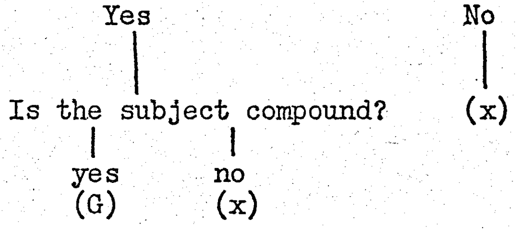
16 FDMP (Completion of demonstrative pronoun phrase)

E1250 to E1258



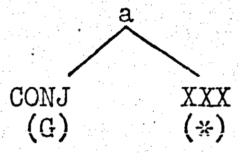
17 APPOSV (Has an "appositive" situation developed?)

J0000 to J0106



18 MORXXX (Additional members of compound constructions)

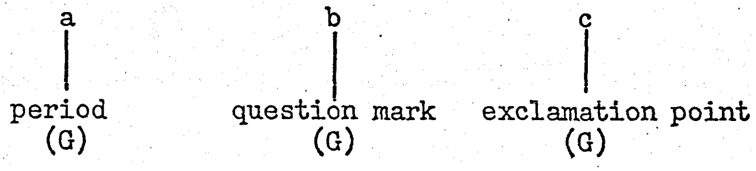
I0600 to I1000



```
- If XXX = SEN, then go to 1.0
         = NPH,            11
         = APH,            12.2.3
         = RLC,            14.0
         = DCL,            15
         = MFR,            8.0
```

19 ENDP (End punctuation)

I5000 to I5054



(G) represents the operation of generating a complete final form.

(x) means that there is no operation. Control passes automatically to the next constituent to be developed.

IV - GRAMMAR OF SPECIFIERS

1. Initialization (B9990)

Clear dispatcher of all entries. Reset dispatcher with subscripts on the symbol now in the workspace (symbol for sentence, relative clause, etc.). ((GO TO 2))

2. What basic type of sentence shall we generate? (C0000)

a. Declarative: Set subscripts for indicative mood, period as end punctuation; any conjoined independent clause must also be declarative. ((GO TO 4))

b. Interrogative: Set subscripts for indicative mood, question mark as end punctuation; any conjoined independent clause must also be interrogative. ((GO TO 4))

c. Imperative: Set subscripts to produce PRED only, for imperative mood, synthetic verb form, exclamation point as end punctuation, person-number at either 2nd singular or plural or 1st plural; any conjoined independent clause must be imperative. ((GO TO 3))

3. Will this imperative sentence be negative or affirmative? (C0020)

a. Affirmative: Set subscripts for affirmative predicate, restrict PRDB and VCMP, prohibit generation of a verb modifier, set a "switch" showing this is an affirmative imperative clause. ((GO TO 15))

b. Negative: Set subscripts for negative predicate, prohibit generation of a negative subject. ((GO TO 12))

4. Will the sentence be active or passive? (C0030)

a. Active:
1) and it will be declarative (noted at 2) - ((GO TO 12))
2) and it will be interrogative (noted at 2) - ((GO TO 6))

b. Passive: Set subscripts to restrict VCMP expansion to either VBMD or VBMD + CMPINF (no VBOB permitted); set ISPV and PSV "switches" to show this is a passive sentence, thereby assuring generation of a transitive verb and correct ordering of the elements of the verb phrase (particularly in case of imbedded negatives). ((GO TO 5))

5. Will the agent be expressed or not? (C0040)

a. Yes: Set PRDA to generate an agent.

b. No: No operation required (PRDA is normally set at no agent).

* - for both a. and b.: if sentence is to be decl, GO TO 12.
if sentence is to be intg, GO TO 6.

6. What form will the interrogative take? (C0050)

a. Interrogative pronoun: Set subscripts to assure entry into interrogative word routine in the Grammar of Sentences (before expanding SENT into SUBJ + PRED), also to assure choice of an intg pro. ((TO 7))

b. Interrogative phrase: Set subscripts as for a., except final choice here is set for an interrogative expression. ((GO TO 8))

c. No interrogative word: Est-ce que still permitted. No operation. ((GO TO 11))

7. What function will the interrogative pronoun form assume? (C0080)

a. Subject: Set subscripts to produce subject form of lequel or qui, to generate PRED only, and to prohibit generation of a negative subject. ((GO TO 12))

b. Object: Set subscripts to produce object form of lequel or qui, to prohibit generation of a negative object, to choose the zero form of the direct object in usual position, to prohibit choice of word order which permits SUBJ + PRED plus an inserted pleonastic pronoun subject form (that is, not Que le prof. dit-il?, but Que dit le prof.?), to assure that VCMP and VBOB go through usual VB + OBJ routines. ((GO TO 11))

c. Indirect Object: Set subscripts to produce indirect object forms of lequel or qui, to prohibit generation of a negative particle functioning as indirect object, to account for usual indirect object routines in the predicate. ((GO TO 11))

d. Adverbial: Set subscripts to produce adverbial constructions with qui or lequel. No other restrictions. ((GO TO 11))

e. Predicate Nominative: Set subscripts to produce predicate nominative forms of lequel or qui, to account for the predicate nominative routines in the predicate, generating a zero form in the usual position. ((GO TO 7.1))

7.1 Which of the two following word order types is to be used for the predicate nominative construction? (C0130)

a. Expand SENT to PRED only, but provide for the VRBSUB pleonastic subject pronoun (here, of course, not pleonastic). ((GO TO 15))

b. Inverted (PRED + SUBJ) order. ((GO TO 15))

8. What type of interrogative phrase will it be? (C0170)

a. Adverbial of type 1, that is, which permits inversion of the subject noun and predicate: Set subscripts to assure choice of a type 1 interrogative adverb, to leave word order choice free. ((GO TO 11))

b. Adverbial of type 2, that is, which does not permit inversion of subject noun and predicate: Set subscripts to assure choice of a type 2 interrogative adverb. ((GO TO 11))

c. An expression with various possible functions. ((GO TO 9))

9. What function will the interrogative expression assume? (C0200)

a. Subject: (Subscripts are set here as they were in 7 for each function.) ((GO TO 10))

b. Object ((GO TO 11))

c. Indirect Object ((GO TO 11))

d. Adverbial ((GO TO 10))

10. Shall we insert il y a? (C0230)

No operations. Still to be programmed. ((GO TO 11))

11. What modifications in word order are now required and/or permitted, in consideration of any interrogative constructions that are to be generated? (C0250)

* Note: The rule is normally set to prohibit the choice of subrules d. and e. They are only possible under certain conditions which may be developed in the preceding operations.

11. (continued)

a. Normal word order plus est-ce que: Set subscripts to assure generation of est-ce que and to assure expansion of SENT to SUBJ + PRED. ((GO TO 12))

b. Inversion of verb with pronoun subject. No noun subject expressed: Set subscripts to assure generation of a "pleonastic" subject pronoun, to expand SENT to PRED only, and to prohibit generation of a negative particle functioning as the subject. ((GO TO 12))

c. Inversion of verb with pronoun subject. Noun subject is expressed: Set subscripts to assure generation of a pleonastic subject pronoun, to expand SENT to SUBJ + PRED and to prohibit generation of a negative particle functioning as the subject. ((GO TO 12))

d. Inversion of verb with noun subject: Set subscripts to expand SENT to PRED + SUBJ, to assure generation of a synthetic verb form, to prohibit generation of a negative particle functioning as the subject. ((GO TO 12))

e. Leave word order unchanged. ((GO TO 12))

12. Will the sentence be affirmative or negative? (C0300)

a. Affirmative: No operations. ((GO TO 15))

b. Negative: Set subscripts to assure generation of a negative particle ne at the head of the PRED. ((GO TO 13))

13. Will this negative sentence contain the particle pas? (C0360)

a. Yes: Set subscripts to assure generation of a verb modifier (the modifier must be negative, specifically pas), to assure that the verb form, if analytic, will be discontinuous and, if a passive construction is formed, it also must be discontinuous. ((GO TO 15))

b. No: Set subscripts to assure that, if a negative verb modifier is generated, it will not be pas and to permit optional generation of other negative particles (initially prohibited). ((GO TO 14))

14. Which of the possible negative particles shall we definitely choose to be generated? (C0390) We have decided to make the sentence negative, and have provided for a ne, but have chosen not to generate a pas. We must therefore be sure to generate at least one other negative particle. Regardless of the one we choose here, the choice of additional negative particles remains optional.

a. Subject: Set subscripts to assure generation of a negative subject. ((GO TO 15))

b. Object: Set subscripts to assure generation of a negative object, via restricted expansion of VCMP and VBOB. ((GO TO 15))

c. Indirect Object: Set subscripts to assure generation of a negative indirect object, via limited expansion of PRED. ((GO TO 15))

d. Verb Modifier: Set subscripts to assure generation of a negative verb modifier. ((GO TO 15))

e. Predicate Modifier: Set subscripts to assure generation of a negative predicate modifier. ((GO TO 15))

f. Participle Modifier: Set subscripts to assure generation of a negative participle modifier and to assure that the verb form will be analytic and non-discontinuous. (The "participle modifier" was added to the program for research purposes only. I do not consider it an integral part of the program-grammar.) ((GO TO 15))

15. End of the Grammar of Specifiers. Go on to the Grammar of Sentences.

V. The Program in COMIT Notation

The following pages contain a complete listing of all the rules of the programmed grammar, in the COMIT notation. I have included this appendix in the thesis principally because the program constitutes the major portion of the work I have done and I feel that the program in its present state should be recorded now, before I return to the task of expanding and improving it.

It is very difficult "read" a program, even when the programming language is a completely simple one, but after studying the COMIT Reference Manual and the preceding Appendices (I - IV), the interested reader should be able to find and consider any detail of the program which may interest him. The identification numbers in the extreme right hand column were used as references in the text of the thesis and in Appendices III and IV.

```
       COM  DINNEEN  L-TO-R  4 SEP 62                                  *
* $=-DINNEEN-PROGRAM-B                                            *        A0050
* $=-M*1*1*9*0-+A+1+*. //*RSL2,*WAL 1 2 3 4                       *        A0060
* $=-START-AT-+A+*. //*RAL2,*WAM1 2 3                             *        A0080
* $=*(+*0+*0+*1+*)                                                *        A0090
AA $=1+1+A/UTT A //W Y,*Q4 1,*Q2 2                                ZNSN     A0100
I $=M+Z //*A2 1                                                   *        B0000
* *(+*1+*0+*0+*)=0                                                  STOP   B0010
ADD *0+*)=*1+2                                                    PC       B0020
* *1+*)=*2+2                                                      PC       B0030
* *2+*)=*3+2                                                      PC       B0040
* *3+*)=*4+2                                                      PC       B0050
* *4+*)=*5+2                                                      PC       B0060
* *5+*)=*6+2                                                      PC       B0070
* *6+*)=*7+2                                                      PC       B0080
* *7+*)=*8+2                                                      PC       B0090
* *8+*)=*9+2                                                      PC       B0100
* *9+*)=2+*0                                                      ADD      B0110
PC *)+$+Z=2+1                                                     AA       B0120
ZNSN $1=1/-ZNSN //*D-,*D1,W Y                                          A   B9990
A A $1=1/ZZ,MOOD IN,ENDP A,A A                                    C        C0000
 B =1/ZY,MOOD IN,ENDP B,A B                                       C        C0003
 C =1/STYP C,MOOD IM,VBFM SYN,ENDP C,PSNM B D E,A C               *        C0006
B A $1=1/PRDB -C E,VBOB -B C,VBMD A,ISMV Y,PRD AF                 UTT      C0020
 B =1/NGFN -S PTM,PRD NG                                               O   C0023
C A $1=1                                                          $        C0030
 B =1/VCMP A D,ISPV Y,PSV Y                                       *        C0033
D Y $1=1/PRDA C                                                   $        C0040
 N =1                                                             $        C0043
ZY $1=1/-ZY                                                       *        C0050
E A $1=1/QFM A,INTG Y                                                  *   C0060
 B =1/QFM B,INTG Y                                                     J.  C0063
 C =1                                                             M        C0066
G S $1=1/QQUX A B D,QLQL A,STYP C,NGFN -S                         *        C0080
 O =1/QQUX C E,QLQL A,NGFN -O,OBN C,WDRD -C,VCMP B E,VBOB A       M        C0083
 I =1/QQUX F G,QLQL B,NGFN -I,PRDB E                              M        C0086
 A =1/QQUX H,QLQL C                                               M        C0089
 P =1/ZUTT,QQUX A,QLQL A,PRDB A,VCMP C,PNM C                      H        C0092
R A $1=1/PSNM C,NUM S                                             *        C0110
```

```
 B =1/PSNM F,NUM P,QQUX -B D                                   *     C0112
GEN M $1=1/GEN M                                               N     C0114
 F =1/GEN F                                                    N     C0116
H A $1=1/STYP C,VBSB Y                                         *     C0130
 B =1/STYP B //HH C F                                          *     C0133
HH A $1=1/PSNM A,NUM S                                         *     C0150
 B =1/PSNM B,NUM S                                             *     C0152
 C =1/PSNM C,NUM S //HH                                        *     C0154
 D =1/PSNM D,NUM P                                             *     C0156
 E =1/PSNM E,NUM P                                             *     C0158
 F =1/PSNM F,NUM P //HH                                        *     C0160
GEN M $1=1/GEN M                                               $     C0162
 F =1/GEN F                                                    $     C0164
J A $1=1/QWD A,WDRD                                            M     C0170
 B =1/QWD B                                                    M     C0172
 C =1/QWD C                                                    *     C0175
K S $1=1/QXPR S,NGFN -S,STYP C,WDRD STAY,CMP ADVL,PSNM C       *     C0200
 O =1/QXPR O,NGFN -O,OBN C,WDRD -C,VCMP B E,VBOB A             M     C0202
 I =1/QXPR I,NGFN -I,PRDB E                                    M     C0204
 A =1/QXPR A,CMP NOML                                          *     C0206
L A ($1=1/STYP D,IDYM ILYA,2B FINSHD LTR)                      *     C0230
 B                                                             *     C0232
M $1 //WDRD -D STAY,*D1                                        *     C0250
WDRD A $1=1/QMKR Y,STYP A                                      N     C0270
 B =1/VBSB Y,STYP C,NGFN -S                                    N     C0272
 C =1/VBSB Y,STYP A,NGFN -S                                    N     C0275
 D =1/STYP B,VBFM SYN,NGFN -S                                  N     C0277
 STAY =1                                                       N     C0280
ZZ $1=1/-ZZ                                                    *     C0300
N AF $1=1/PRD AF                                               UTT   C0330
 NG =1/PRD NG                                                  *     C0332
O Y $1=1/VBMD B,VBMF D,NGAD A,VBFM -ANL,ARR B                  UTT   C0360
 N =1/NGAD -A,SUB A D,OBN,RGIN,VBMF,PDMF,PTMF                  *     C0362
NGFN S $1=1/SUB D,PSNM C                                       UTT   C0390
 O =1/OBN B,VCMP B E,VBOB A                                    UTT   C0392
 I =1/RGIN B,PRDB B                                            UTT   C0394
 VM =1/VBMD B,VBMF D                                           UTT   C0396
 PDM =1/PRDA B,PDMF A                                          UTT   C0398
 PTM =1/PP A,PMF B,VBFM ANL                                          UTT    C0400
(END OF PREPARATORY ROUTINE)
ZUTT $1=1/-ZUTT //UTT A,*D1                                        *        D0000
UTT A $1=1/-ENDP+A/ZEND,ENDP*1+A/PRIN //*S3 3 2,UTT B                   *   D0020
 B =1/-ENDP                                                        *        D0022
* $1=*.*0+-SPECIFIERS/$*1+1 //*WAM1,*WSM2                                *   D0030
ZSEN $1=1/-ZSEN //SEN,*D1                                          *        D0040
SEN A $1=1                                                         SNT      D0060
 B =1+A/ZMRS,SEN A,A*1 //*S3 2                                     SNT      D0062
ZSNT $1=1/-ZSNT                                                    *        D0080
SNT A $1=1                                                         *        D0100
 B =A+1/ZSNT //*S3 2                                               ZSNM     D0102
 C =1+A/ZSNM //*S3 2                                               *        D0104
ZNTG $1=1/-ZNTG //INTG N,QMKR N,*D1                                *        D0120
INTG Y                                                             ZQFM     D0140
 N                                                                 QMKR     D0142
ZQMK $1=1/-ZQMK                                                    *        D0160
QMKR Y $1=-EST*-CE-QUE/E,V,F,P+1/ZPSN //W N,*S3 2                  W        D0180
 N =1                                                              *        D0182
ZPSN $1=1/-ZPSN //PSNM,GEN,*D1                                     *        D0200
PSNM A $1=1/PSNM A,NUM S,NFR C                                     *        D0230
 B =1/PSNM B,NUM S,NFR C                                           *        D0232
 C =1/PSNM C,NUM S                                                 *        D0234
 D =1/PSNM D,NUM P,NFR A C F (IMPROVE)                             *        D0236
 E =1/PSNM E,NUM P,NFR A C F (IMPROVE)                             *        D0238
 F =1/PSNM F,NUM P                                                 *        D0240
GEN M $1=1/GEN M                                                   *        D0260
 F =1/GEN F                                                        *        D0262
ZSTP $1=1/-ZSTP //STYP A,*D1                                            *   D0280
STYP A $1=-SENT+1/-VCMP,-PRDB,-MOOD+1/ZPRD,-SUB,-NFR //*Q5 1,*S3 3 ZSUB     D0300
 B =-SENT+1/-SUB,-NFR+1/ZSUB,-VCMP,-PRDB,-MOOD //*Q5 1,*S3 3       ZPRD     D0304
 C =1/-SUB,-NFR                                                    ZPRD     D0308
ZSNM $1=-SNMFR+1 //*Q5 1                                                *   D0320
SNMF A                                                             *        D0370
 B                                                                 SCLZ     D0372
 C                                                                 SFRZ     D0374
SNAD A                                                             *        D0390
 B                                                                 ADSP     D0392
 C                                                                 ADST     D0394
ADSM A $1=-LENTEMENT                                               W        D0410
 B =-HEUREUSEMENT/P,E                                              W        D0412
ADSP A $1=-LA                                                      W        D0440
 B =-ICI/P,E                                                       W        D0442
ADST A $1=-MAINTENANT                                              W        D0480
 B =-RECEMMENT                                                     W        D0482
SCLZ $1=A+A/ZNSN,A A,UTT B,SEN A,SNT A //*S3 2                     *        D0500
INTR A $1=-QUAND                                                   W        D0530
 B =-LORSQUE                                                       W        D0532
SFRZ A $1=-DEPUIS+A/ZNFR //*S3 2                                   W        D0600
 B =-EN/P,E+A/ZPRA(,MORSUBSCRIPTS) //*S3 2                         W        D0602
ZQFM $1=1/-ZQFM                                                         *   D0630
QFM A                                                                   *   D0632
 B                                                                   ZQFR   D0634
QPRO A $1=1 //QQUX,*D1                                             *        D0650
 B =1+1/ZPSN //*S3 2,QLQL,*D1                                      QLQL     D0652
QQUX A $1=-QUI+1 //*Q4 1                                           ZSTP     D0670
 B =-QUI-EST*-CE-QUI+1 //*Q4 1                                     ZSTP     D0672
 C =-QUI+1 //*Q4 1                                                 QMKR     D0674
 D =-QU-EST*-CE-QUI+1 //*Q4 1                                      ZSTP     D0676
 E =-QUE/E,F+1/ZQMK //*S3 2,W N                                    W        D0678
 F =-INDOBJ+-A+-QUI+1 //*Q5 1,*Q4 2 3                              QMKR     D0680
 G =-INDOBJ+-A+-QUOI+1 //*Q5 1,*Q4 2 3                             QMKR     D0682
 H =-PRPH+A+A/ZQUX+1/ZQMK //*Q5 1,*S3 4 3                          P        D0684
ZQUX (MAY ADD OPNS LATER)                                               *   D0700
QUX A $1=-QUI                                                      W        D0720
 B =-QUOI                                                          W        D0722
QLQL A $1=1                                                        ZLQL     D0750
 B =-INDOBJ+-A/C,F+1 //*Q5 1,W N,*S6 2                             ZLQL     D0752
 C =-PRPH+A+1/ZLQL //*Q5 1,*S3 3                                        P   D0754
ZLQL $1=1/-ZLQL //WLQL,*D1                                              *   D0780
WLQL A $1=1+1/ZDCL,DCTP A //*S3 2                                     LQL   D0782
 B =1                                                                 LQL   D0784
ZQFR $1=-ZQFR-NOT-READY+1 //*Q4 1                                    ZPSN   D0900
ZSUB $1=1/-ZSUB //SUB A,*D1                                             *   E0000
SUB A $1=-SUBJ+1 //*Q5 1                                           *        E0020
 C =A //*N3 1                                                      $        E0022
 D =-SUBJ+1 //*Q5 1                                                NGNF     E0024
```

```
NUM S $1=1/NFR -A                                                     ZNFG   E0030
 P =1                                                                     *   E0032
BD A $1=1/NFR -A                                                          *   E0040
 B =1/NFR A+A/ZAPV //*S3 2                                                *   E0042
ZNFG $1=1/-ZNFG //NFGR -B,*D1                                       *         E0100
NFGR A $1=1                                                         CHUZ      E0120
 B =A/GEN*1 //GEN,*D1                                               *         E0122
GEN M $1=A+A/ZAD,GEN M,NUM P //*S3 2                                *         E0140
 F =A/NFR B,GEN F+A/ZNFR,NFR B,GEN F+A/ZAD,GEN F,NUM P //*S3 3 2    ZNFR      E0142
BM A $1=A/GEN M,NFR B+A/ZNFR,NFR B //*S3 2                          ZNFR      E0160
 B =A/NFR B+A/ZNFR,NFR B,GEN M //*S3 2                              ZNFR      E0162
CHUZ                                                                *         E0180
GEN M //GEN M                                                       *         E0182
 F //GEN F                                                          *         E0184
NUM S //NUM S                                                       *         E0186
 P //NUM P                                                          *         E0188
ZNFR $1=-NFR+1/-ZNFR //*Q5 1,NFR,PSNM,ILON,*D2                      *         E0200
NFR A $1=1/NFR,NFR -A C+1/ZMNF,NFR,NFR -A C //*S3 2                   ZNFR    E0220
 B =1                                                               ZCMN      E0224
 C =1                                                               ZSPR      E0226
 D =1+1/ZFDM //*S3 2                                                ZDMP      E0230
 E =1                                                               ZPRO      E0232
 F =1                                                               ZDSJ      E0234
ZDCL $1=-DCLS+1/-ZDCL //DCTP,DCLS,*D2,*Q5 1                         *         E0260
DCTP A $1=A/GEN*1,NUM P                                             *         E0270
 B =A //NUM,GEN                                                     *         E0275
 C =1                                                               *         E0278
DCLS A $1=-DE/F,E,C+1/ZNFG,NFR -A C,FDM -D //*S3 2,W N              W         E0280
 B =1+1/ZMDC,DCLS A //*S3 2,DCLS A                                  DCLS      E0282
ZSPR                                                                *         E0300
PSNM A $1=-JE/F,E //W N                                             W         E0320
 B =-TU                                                             W         E0322
 C =1                                                               *         E0324
 D =-NOUS                                                           W         E0326
 E =-VOUS                                                           W         E0328
 F =1                                                               THPL      E0330
GEN M $1=1                                                          *         E0340
 F =-ELLE/P,V                                                       W         E0342
```

```
ILON A $1=-IL/P,V                                               W          E0360
 B =-ON/P,V                                                     W          E0362
THPL                                                            *          E0380
GEN M $1=-ILS/P,V                                               W          E0400
 F =-ELLES/P,V                                                  W          E0402
ZDSJ $1=1/-ZDSJ //PSNM,*D1                                      *          E0420
PSNM A $1=-MOI                                                  W          E0440
 B =-TOI                                                        W          E0442
 C =1                                                           *          E0444
 D =-NOUS                                                       W          E0446
 E =-VOUS                                                       W          E0448
 F =1                                                           TPLA       E0450
GEN M $1=1                                                      *          E0480
 F =-ELLE/P,V                                                   W          E0482
ILON A $1=-LUI                                                  W          E0500
 B =-SOI                                                        W          E0502
TPLA                                                            *          E0520
GEN M $1=-EUX/P,V                                               W          E0540
 F =-ELLES/P,V                                                  W          E0542
ZPRO                                                            *          E0580
NUM S                                                           *          E0600
 P                                                              PRPL       E0602
GEN M                                                           PMS        E0640
 F                                                              PFS        E0642
PRPL                                                            *          E0680
GEN M $1=-LES/P,C+1/ZPMP //*S3 2                                W          E0700
 F =-LES/P,C+1/ZPFP //*S3 2                                     W          E0702
PMS A $1=-JEAN                                                  W          E0740
 B =-PARIS                                                      W          E0742
 C =-HENRI/P,E                                                  W          E0744
PFS A $1=-LOUISE                                                W          E0800
 B =-LA-FRANCE                                                  W          E0802
 C =-HENRIETTE/P,V                                              W          E0804
ZPMP A $1=-FRANCAIS                                             W          E0850
 B =-ETATS*-UNIS                                                W          E0852
 C =-EUROPEENS/P,V                                              W          E0854
ZPFP A $1=-FRANCAISES                                           W          E0900
 B =-NATIONS*-UNIES                                             W          E0902
 C =-AMERICAINES/P,V                                            W          E0904
```

```
ZDMP                                                        *         E1000
NUM S                                                       *         E1020
 P                                                          DMPL      E1022
GEN M $1=-CELUI                                             W         E1050
 F =-CELLE                                                  W         E1052
DMPL                                                        *         E1080
GEN M $1=-CEUX                                              W         E1200
 F =-CELLES                                                 W         E1202
ZFDM $1=1/-ZFDM //FDM,*D1                                   *         E1250
FDM A $1=*-CI                                               W         E1252
 B =*-LA                                                    W         E1254
 C =1                                                       ZRLC      E1256
 D =1/DCTP C                                                ZDCL      E1258
ZCMN $1=-CMN+1 //*Q5 1                                           *    E1270
(NOW SET GEN-NUM FOR WHOLE NP,IF NOT ALREADY SET)
GEN M $1=1/GEN M                                            *         E1275
 F =1/GEN F                                                 *         E1277
NUM S $1=1/NUM S                                            *         E1280
 P =1/NUM P                                                 *         E1282
* $1=1+1/ZNAD //*S3 2                                       ZDTM      E1290
ZNAD $1=1/-ZNAD //NAD,*D1                                   *         E1300
NAD A $1=1                                                  ZNM       E1350
 B =1+1/ZNM //*S3 2                                         ZADJ      E1354
 C =1/SPES B+1/ZAPH //*S3 2                                 ZNM       E1358
 D =1+1/ZNAD,NAD A C //*S3 2                                  ZADJ    E1362
ZAPH $1=1/-ZAPH //APH,*D1                                   *         E1400
APH A $1=1/APH -A+1/ZMAP,APH -A //*S3 2                     ZAPH      E1440
 B =A                                                       ADJB      E1442
 C =A/DCTP B                                                  ZDCL    E1444
ZADJ $1=1/-ZADJ //ADJ,*D1                                   *         E1480
ADJ A $1=A                                                  *         E1500
 B =A+1/ZADJ,ADJ A //*S3 2                                  *         E1504
ADJA A $1=FX+-PETIT                                         FX        E1520
 B =FX+-GRAND                                               FX        E1522
 C =FX+-JOLI                                                FX        E1524
ADJB A $1=FX/P,E+-INTERESSANT                               FX        E1550
 B =FX+-BLEU                                                FX        E1552
FX                                                          *         E1580
```

```
GEN M FX+$1=1+2                                *            E1600
 F =1+2+E //*K2 3                              *            E1602
NUM S FX+$1=1+2                                *            E1640
 P =1+2+S //*K2 3                              *            E1642
TR FX+$1=2/$*1                                 W            E1680
ZAD A                                            ADJA       E1690
 B                                               ADJB       E1692
ZNM $1=1/-ZNM //NM,*D1                         *            E1700
NM A $1=A                                      *            E1720
 B =A+1/ZRLC,NM A //*S3 2                      *            E1722
GEN M                                          *            E1740
 F                                                  MF      E1742
MM A $1=FX+-GARCON                             Y            E1760
 B =FX+-CRAYON                                 Y            E1762
 C =FX+-LIVRE                                  Y            E1764
 D =FX+-COMMENCEMENT                           Y            E1766
 E =FX+-JOURNA+L                               SP           E1768
 F =FX/P,V+-AMI                                Y            E1770
MF A $1=FX+-FILLE                              Y            E1800
 B =FX+-BONTE                                  Y            E1802
 C =FX/P,V+-HISTOIRE                           Y            E1804
 D =FX+-FIN                                    Y            E1806
Y                                              *            E1900
NUM S FX+$1=2/$*1                              W            E1950
 P =1+2+S //*K2 3                              TR           E1952
SP                                             *            E2000
NUM S FX+$1+$1=1+2+3 //*K2 3                   TR           E2020
 P =1+2+UX //*K2 3                             TR           E2022
ZDTM $1 //DET,ART,PSNM,PTV A,ART -D,*D1                *    E2500
DET A                                          *            E2550
 B                                             POS          E2552
 C                                             DMD          E2554
ART A                                          *            E2580
 B                                             IDF          E2582
 C                                             PTV          E2584
 D                                             CAS          E2586
NUM S $1=1                                     *            E2600
 P =-LES/P,C                                   W            E2602
```

```
GEN M $1=-LE/P,C,F,E //W N              W      E2620
 F =-LA/F,E //W N                       W      E2622
ZIDF $1=1/-ZIDF //NUM,*D1               *      E2640
IDF                                     *      E2660
NUM S $1=1                              *      E2680
 P =-DES                                W      E2682
GEN M $1=-UN/P,V                        W      E2700
 F =-UNE/P,V                            W      E2702
PTV A $1=1                              *      E2720
 B =-DE/F,E //W N                       W      E2722
NUM S $1=1                              *      E2740
 P =-DES                                W      E2742
GEN M $1=-DU                            W      E2760
 F =-DE-LA                              W      E2762
CAS $1=-D+A/ZIDF,NUM S //*S3 2          W      E2780
POS                                     *      E2800
PSNM A $1=-M                            *      E2820
 B =-T                                  *      E2822
 C =-S                                  *      E2824
 D =-N                                  PSB    E2826
 E =-V                                  PSB    E2828
 F =-LEUR                               PSC    E2830
NUM S $1=1                              *      E2860
 P =1+ES //*K1 2                        W      E2862
GEN M $1=1+ON //*K1 2                   W      E2880
 F =FX/S,F+1+A //*K2 3                  TR     E2882
PSB                                     *      E2900
NUM S $1=1+OTRE //*K1 2                 W      E2920
 P =1+OS //*K1 2                        W      E2922
PSC                                     *      E2940
NUM S $1=1                              W      E2960
 P =1+S //*K1 2                         W      E2962
DMD                                     *      E2980
NUM S $1=1                              *      E3000
 P =-CES                                W      E3002
GEN M $1=-CE/S,F                        W      E3020
 F =-CETTE                              W      E3022
ZPRF $1=1/-ZPRF                         *      F0000
```

```
PRF $1=-PREPH+A+A/ZNFR,NFR -A C //*Q5 1,*S3 3             *     F0020
P A $1=-AVEC/P,V                                    W           F0040
 D =-SUR                                            W           F0046
 E =-DANS                                           W           F0048
LQL                                                 *           F0100
GEN M                                               *           F0120
 F                                                  FL          F0122
NUM S $1=-LEQUEL/P,C                                W           F0140
 P =-LESQUELS/P,C                                   W           F0142
FL                                                  *           F0180
NUM S $1=-LAQUELLE                                  W           F0200
 P =-LESQUELLES/P,C                                 W           F0202
ZPRD $1=1/-ZPRD //PRD AF,*D1                              *     F0500
PRD AF $1=-AFPRD+1 //*Q5 1                                *     F0540
 NG =-NGPRD+-NE/F,E+1/ZPRA //*Q5 1,*S3 3,W N              W     F0542
ZPRA $1=1/-ZPRA //PRDA,PRDA -C,*D1                  *           F0580
PRDA A $1=1                                         *           F0600
 B =1+A/ZPDM,PDMF*1 //*S3 2                         *           F0602
 C =1+A/ZAGN,AGNT*1 //*S3 2,PRDA -C                 PRDA        F0604
ZPRB $1=1/-ZPRB //PRDB,TPIN,ISMV N,TPRI,*D1         *           F0620
PRDB A $1=1/VCMP -F,VBOB -C D F                     *           F0650
 B =1/VCMP A B F,VBOB A B,VB -A,VBD C,VBE B               -     F0655
+A/RGIN*1,ZRGN //*S3 2                                    *     F0656
 C =1                                               TPIN        F0660
 D =1                                               TPRI        F0665
 E =1/VCMP A B D E,VBOB A B,VB -A,VBD C             *           F0670
ZVCM $1=1/-ZVCM //VCMP,ISPV N,*D1                         *     F0700
VCMP A $1=1                                         ISPV        F0800
 B =1/VB C                                          *           F0805
 C =1/VB A+1/ZPNM //*S3 2                           ZVBM        F0810
 D =1/TKNF Y,RFNF Y,VB B+1/ZCMF //*S3 2             ISPV        F0815
 E =1/TKNF Y,RFNF Y,VB C+1/ZCMF //*S3 2             *           F0820
 F =1/TKNF Y,RFNF Y,VB B+Q+1/ZCMF //*N3 2,*S3 3 2         *     F0825
 G =1/VBOB B,VB C+1/ZCAD //*S3 2                    *           F0830
ZVBO $1=1/-ZVBO //VBOB,TPOB N,*D1                   *           F0900
VBOB A $1=1+A/ZOBN,OBN*1 //*S3 2                    *           F1000
 B =1                                               TPOB        F1005
 C =A/OBP C D G+Q+1/ZVBM //*N3 2,*S3 3 2            ZOBP        F1010
 D =1(1/VBMD A+Q+1/ZRGP //*N3 2,*S3 3 2) //VBOB -D      VBOB       F1015
 E =1                                                   TPBJ       F1020
 F =1+Q+A/ZOBN //*N3 2,*S3 3 2                          *          F1025
ZVBM $1=1/-ZVBM //VBMD,VB,*D1                                *     F1100
VBMD A $1=1                                             VB         F1200
 B =1+A/ZVBF,VBMF*1,NGAD*1 //*S3 2                      VB         F1205
ZVBF $1=-VBMFR+1/-ZVBF //VBMF -D,NGAD,*Q5 1,*D2                *   F1250
VBMF A $1=1                                             ADVB       F1300
 B =1                                                   PRF        F1302
  C =1/VBMF -C D+1/ZMMF,VBMF - C D //*S3 2                ZVBF     F1304
 D =1                                                   *          F1306
NGAD A $1=-PAS                                          W          F1350
 B =-JAMAIS                                             W          F1352
 C =-GUERE                                              W          F1354
ZPDM $1=-PRDMF+1 //*Q5 1,PDMF -A,*D2                           *   F1380
PDMF A                                                  *          F1400
 B                                                      PSPD       F1402
NGPD A $1=-JAMAIS                                       W          F1450
 B =-PLUS                                               W          F1452
PSPD A $1=-TOUJOURS                                     W          F1480
 B =-SOUVENT                                            W          F1482
ADVB A                                                  *          F2000
 B                                                      ADPL       F2002
 C                                                      ADMN       F2004
ADTM A $1=-MAINTENANT                                   W          F2020
 B =-BIENTOT                                            W          F2022
ADPL A $1=-ICI/P,V                                      W          F2100
 B =-LA                                                 W          F2102
ADMN A $1=-BEAUCOUP                                     W          F2200
 B =-LENTEMENT                                          W          F2202
ZAGN $1=-AGENT+1 //*Q5 1,AGNT A,*D2                            *   F3000
AGNT A $1=-PAR+A/ZNFG //*S3 2                           W          F3020
 B =-DE/F,E,C+A/ZNFG //*S3 2,W N                        W          F3022
VB A $1=-VBCP+1 //*Q5 1                                 *          G0000
 B =-VBIN+1 //*Q5 1                                     VBIN       G0002
 C =-VBTR+1 //*Q5 1                                     VBTR       G0004
VBCP $1=1/V ETRE,AX AV //VBFM -QANL,*D1                 VBFM       G0050
VBIN $1 //TKNF N,VBE,VBF,VBFM -QANL,*D1                 *          G0100
```

```
TKNF Y                                                            VBE      G0200
 N                                                                VBF      G0202
VBTR $1 //VBIS NR,TKNF N,VBA,VBB,VBC,VBD,RFNF N,*D1               *        G0250
VBIS R $1=1/AX ET                                                 RFNF     G0300
 NR =1/AX AV                                                      *        G0302
TKNF Y                                                            VBC      G0350
 N                                                                VBD      G0352
RFNF Y                                                            VBA      G0400
 N                                                                VBB      G0402
ZPSV $1=1/-ZPSV //PSV N,*D1                                       *        G0450
PSV Y $1 //ARR A,VBFM -QANL,*D1                                   *        G0500
 N //VBFM -QANL,*D1                                               VBFM     G0502
ARR A $1=1/V ETRE+1/ZPP //*S3 2                                   *        G0600
 B =1/V ETRE+Q+1/ZPP //*N3 2,*S3 3 2                              *        G0602
VBFM SYN $1=A/VBSB*1+1/ZSYN //*S3 2,VBSB N,*D1                    *        G0700
 ANL =A/VBSB*1+1/ZAUX+1/ZPP //*S3 3 2,VBSB N,*D1                  *        G0702
 QANL =A/VBSB*1+1/ZAUX+Q+1/ZPP //*N3 3,*S3 4 3 2,VBSB N,*D1       *        G0704
VBSB Y $1=Q+1/ZSPR //*N3 1,*S3 2                                  $        G0900
 N =Q //*N3 1                                                     $        G0902
ZPP $1=1/-ZPP //AGRM N,PMF -B,*D1                                 *        G1000
(NU AGRM RUTIN GOS HERE)
PP A $1=A+A/ZPPL //*S3 2                                          *        G2000
 B =A                                                             ZPPL     G2002
PMF A                                                             *        G2020
 B                                                                NEGM     G2022
POSM A $1=-BEAUCOUP                                               W        G2100
 B =-TOUT                                                         W        G2102
NEGM A $1=-JAMAIS                                                 W        G2150
 B =-RIEN                                                         W        G2152
ZPPL                                                              *        G2200
V ETRE $1=-ETE                                                    W        G3000
 ALLER =-ALLE                                                     *        G3002
 VENIR =-VENU                                                     *        G3004
 MANGER =-MANGE                                                   *        G3006
 TROUVER =-TROUVE                                                 *        G3008
 DONNER =-DONNE                                                   *        G3010
 LAVER =-LAVE                                                     *        G3012
 LEVER =-LEVE                                                     *        G3014
 COMMENCER =-COMMENCE                                             *        G3016
 PERSUADER =-PERSUADE                              W          G3018
 PARLER =-PARLE                                    W          G3020
 EFFORCER =-EFFORCE                                *          G3022
 APPELER =-APPELE                                  *          G3024
AGRM Y                                                  *     G4000
 N                                                 W          G4002
GEN M $1=1                                         *          G4050
 F =1+E //*K1 2                                    *          G4052
NUM S $1=1                                         W          G4100
 P =1+S //*K1 2                                    W          G4102
ZSYN $1 //MOOD IN,PSNM,*D1                         *          G5000
V ETRE $1=FX/CG ET+A                               *          G5050
 ALLER =FX/CG AL+A                                 *          G5052
 VENIR =FX/CG VE+A                                 *          G5054
 MANGER =FX/CG ER+-MANG                            *          G5056
 TROUVER =FX/CG ER+-TROUV                          *          G5058
 DONNER =FX/CG ER+-DONN                            *          G5060
 LAVER =FX/CG ER+-LAV                              *          G5062
 LEVER =FX/CG ER+-LEV                              *          G5064
 COMMENCER =FX/CG ER+-COMMENC                      *          G5066
 PERSUADER =FX/CG ER+-PERSUAD                      *          G5068
 PARLER =FX/CG ER+-PARL                            *          G5070
 EFFORCER =FX/CG ER,P,V+-EFFORC                    *          G5072
 APPELER =FX/CG ER,P,V+-APPEL                      *          G5074
MOOD IN  //TNS A                                   *          G6100
 SO                                                SO         G6102
 IM                                                IM         G6104
 IF                                                IF         G6106
TNS A                                              PRS        G6200
 B (MUST DEVP)                                     EROR       G6202
 C                                                 EROR       G6204
 D                                                 EROR       G6206
 E                                                 EROR       G6208
SO                                                 *          G6300
TNS A (MUST DEVP)                                  EROR       G6400
 B                                                 EROR       G6402
 C                                                 EROR       G6404
 D                                                 EROR       G6406
 E
PRS $1 //*D1
CG ER $1+$1=1+2
 ET =A
 AL =A
 VE =A
PSNM A FX+$1=1+2+E //*K2 3
 B =1+2+ES //*K2 3
 C =1+2+E //*K2 3
 D =1+2+ONS //*K2 3
 E =1+2+EZ //*K2 3
 F =1+2+ENT //*K2 3
EE
PSNM A $1=-SUIS
 B =-ES/P,V
 C =-EST/P,V
 D =-SOMMES
 E =-ETES/P,V
 F =-SONT
EF
PSNM A $1=-VAIS
 B =-VAS
 C =-VA
 D =-ALLONS/P,V
 E =-ALLEZ/P,V
 F =-VONT
EG
PSNM A $1=-VIENS
 B =-VIENS
 C =-VIENT
 D =-VENONS
 E =-VENEZ
 F =-VIENNENT
ZAUX $1 //*D1
AX ET
 AV
AV
PSNM A $1=-AI/P,V
 B =-AS/P,V
```

```
EROR      G6408
*         G6450
*         G6500
EE        G6502
EF        G6504
EG        G6506
TR        G6550
TR        G6552
TR        G6554
TR        G6556
TR        G6558
TR        G6560
*         G6600
W         G6602
W         G6604
W         G6606
W         G6608
W         G6610
W         G6612
*         G6620
W         G6622
W         G6624
W         G6626
W         G6628
W         G6630
W         G6632
*         G6640
W         G6642
W         G6644
W         G6646
W         G6648
W         G6650
W         G6652
*         G7000
EE        G7050
AV        G7052
*         G7100
W         G7102
W         G7104
```

```
 C =-A/P,V                                                       W        G7106
 D =-AVONS/P,V                                                   W        G7108
 E =-AVEZ/P,V                                                    W        G7110
 F =-ONT/P,V                                                     W        G7112
IF $1 //*D1                                                      *        G8000
CG ER FX+$1=1+2+ER //*K2 3                                       TR       G8050
 ET =-ETRE/P,V                                                   W        G8052
 AL =-ALLER/P,V                                                  W        G8054
 VE =-VENIR                                                      W        G8056
EROR $=-VBFX+-+1+-VBFX //*Q4 1 2 3                                    W   G9999
ZPNM $1=1/-ZPNM //PNM,*D1                                        *        H0000
PNM A $1=1                                                       ZNFG     H0050
 B =1                                                            *        H0052
 C =Q //*N3 1                                                    $        H0054
PADJ A                                                           ADJA     H1000
 B                                                               ADJB     H1002
NGNF A $1=-PERSONNE                                              W        H2000
 B =-RIEN                                                        W        H2002
ZOBN $1=-DOBN+1/-ZOBN //OBN A,*D2,*Q5 1                          *        H3000
OBN A $1=1/NFR -C F                                              ZNFG     H3080
 B =1                                                            NGNF     H3082
 C =Q //*N3 1                                                    $        H3084
ZRGN $1=-RGIN+A/RGIN*1 //RGIN A,*D2,*Q5 1                        *        H4000
RGIN A $1=-A/F,C+1/ZNFR,NFR -A C //*S3 2,W N                     W        H4050
 B =-A+A/NGNF //*S3 2                                            W        H4052
 C =Q //*N3 1                                                    $        H4056
TPIN A $1=A+1/VCMP A B D,VBOB A B,VBF C,VB -A,VBD C,ZVCM //*S3 2 ZRGP     H5000
 B =A/PSNM*1+1/VCMP B,VBOB A B,VBIS R,TPOB N,VB -A,ZVCM //*S3 2  ZREF     H5002
TPRI A $1=1                                                      *        H5050
 B =1/VBOB F,VBIS R,VB -A,VBD C+A/PSNM*1,ZREF //*S3 2,VCMP B     VCMP     H5052
ISMV Y $1=1/VBOB F,VB -A,VBD C,VBF C+A/ZRGP                      -        H5100
 //VCMP,VCMP A B D F,*S3 2                                       VCMP     H5101
 N =1/VBOB C,VBD C,VBF C+A/ZRGP,RGIP C F //VCMP B,*S3 2          VCMP     H5103
ISPV Y $1=1/VB C                                                 ZVBM     H5200
 N =1/VB -C                                                      ZVBM     H5202
TPBJ N $1=1+A/ZOBP //*S3 2                                       ZVBM     H5300
 R =1/VBIS R+A/ZREF,PSNM*1 //*S3 2                               ZVBM     H5302
TPOB N $1=A/OBP*1+1/ZVBM //*S3 2                                 ZOBP     H5400
```

```
 R =A/PSNM*1+1/ZVBM,VBIS R //*S3 2                               ZREF      H5402
ZCMF $1=1/-ZCMF //CMPF,TYPP,*D1                                  *         I0000
CMPF DE $1=-DE/F,E+A/ZTYP //*S3 2,W N                            W         I0050
 A =-A+A/ZTYP //*S3 2                                            W         I0052
 X =A                                                            *         I0054
ZTYP                                                             *         I0100
TYPP AF $1=A/MOOD IF,VBFM SYN,PRDB -D E,VCMP -D E F G            ZPRB      I0110
 NG =-NE+A/ZNEG+A/ZPRB,MOOD IF,VBFM SYN,                         -         I0115
PRDB -D E ,VCMP -D E F G  //*S3 3 2                              W         I0117
ZNEG A $1=-PAS                                                   W         I0150
 B =-JAMAIS                                                      W         I0152
 C =-PLUS                                                        W         I0154
ZCAD A                                                           ADJA      I0200
 B                                                               ADJB      I0202
ZREF $1=-REFPR+1 //PSNM,*D2,*Q5 1                                    *     I0500
PSNM A $1=-ME/F,E //W N                                          W         I0550
 B =-TE/F,E //W N                                                W         I0552
 C =-SE/F,E //W N                                                W         I0554
 D =-NOUS                                                        W         I0556
 E =-VOUS                                                        W         I0558
 F =-SE/F,E //W N                                                W         I0560
ZMAP $1=-MAPH+A+1/-ZMAP,ZAPH //*Q5 1,*S3 3                       CNJ       I0600
ZMNF $1=-MNFR+A+1/-ZMNF,ZNFR //*Q5 1,*S3 3                       CNJ       I0650
ZMDC $1=-MDCL+A+1/-ZMDC,ZDCL //*Q5 1,*S3 3                          CNJ    I0700
ZMRL $1=-MRLC+A+1/-ZMRL,ZRLC,RL A //*Q5 1,*S3 3                  CNJ       I0750
ZMRS $1=-MRSN+A+1/-ZMRS,ZNSN //*Q5 1,*S3 3                          CNJ    I0800
ZMMF $1=-MMFR+A+1/-ZMMF,ZVBF //*Q5 1,*S3 3                          CNJ    I0850
CNJ $1=-ET                                                       W         I1000
ZEND $1 //*D1                                                    *         I5000
ENDP A $1=-PERIOD                                                W         I5050
 B =-QUESTION-MARK                                               W         I5052
 C =-EXCLAMATION-POINT                                           W         I5054
ZAPV $1=1/-ZAPV //DUBL N,*D1                                     *         J0000
DUBL Y $1=1                                                      *         J0050
 N =Q //*N3 1                                                    $         J0052
PKPR A $1=-NOUS                                                  W         J0100
 B =-VOUS                                                        W         J0102
 C =-ILS/P,V                                                     W         J0104
 D =-ELLES/P,V                                                  W           J0106
PRIN $ //*A4 1                                                       *      J0500
ERAS A/PRIN=0                                                     ERAS      J0502
* $=*.*0+1 //*WAM1 2                                                 *      J0504
* $=*.*0+-NODE-NAMES+-+-+1 //*A5 5,*WAM1 2 3 4  5                    I      J0508
ZRLC $1=-RELCL+1 //SPES A,RL,RLCL,*D2,*Q5 1                     *           K0000
SPES A $1=A/NM*1,GEN*1,NUM*1,PSNM*1,AN*1                        RL          K0050
 B =Q+A/ZRL,NM*1,GEN*1,NUM*1,PSNM*1,AN*1 //*N3 1,*S3 2          $           K0052
ZRL $1=1/-ZRL                                                   *           K0060
RL A $1=1                                                       *           K0100
 B =1+1/ZMRL //*S3 2                                            *           K0102
RLCL A $1=-QUI+1/ZZ,NGFN -S,SEN A,STYP C //*S3 2                W           K0200
 B =-QUE/F,E+A/ZZ,NM*1,SEN A,STYP A,PRDB -E,                    -           K0206
VCMP B,VBOB A,OBN C //*S3 2,W N                                 W           K0208
 C =-DONT+A/ZZ,NM*1,SEN A,STYP A,VCMP B C //*S3 2                    W      K0212
 D =A+1/ZRLO+A/ZZ,SEN A,STYP A,NM*1 //*S3 3 2                   P           K0215
 E =-A/F,C+1/ZRLO+A/ZZ,NGFN -I,STYP A,NM*1,                     -           K0220
SEN A,PRDB B,RGIN C //*S3 3 2,W N                               W           K0222
ZRLO $1=1/-ZRLO //AN,*D1                                        *           K0300
AN Y $1=-QUI                                                    W           K0400
 N =1/WLQL B                                                      ZLQL      K0402
VBA $1=1/V EFFORCER(CMPF DE)                                    VBFM        L0000
VBB $1=1/V LAVER                                                VBFM        L0100
VBC A $1=1/V APPELER (CMPF A)                                     ZPSV      L0200
VBD A $1=1/V MANGER                                             ZPSV        L0300
 B =1/V TROUVER                                                 ZPSV        L0302
 C =1/V DONNER                                                  ZPSV        L0304
VBE A $1=1/V COMMENCER,AX AV (CMPF A)                             VBFM      L0400
 B =1/V PERSUADER,AX AV (CMPF DE)                                 VBFM      L0402
VBF A $1=1/V ALLER,AX ET                                        VBFM        L0500
 B =1/V VENIR,AX ET                                               VBFM      L0502
 C =1/V PARLER,AX AV                                            VBFM        L0504
ZRGP $1=-RGIP+A/RGIP*1 //RGIP,*D2,*Q5 1                         *           L1000
RGIP A $1=-ME/F,E //W N                                         W           L1100
 B =-TE/F,E //W N                                               W           L1102
 C =-LUI                                                        W           L1104
 D =-NOUS                                                       W           L1106
 E =-VOUS                                                       W           L1108
 F =-LEUR                                                      W         L1110
ZOBP $1=-DOBP+A/OBP*1 //OBP,*D2,*Q5 1                          *         L2000
OBP A $1=-ME/F,E //W N                                         W         L2100
 B =-TE/F,E //W N                                              W         L2102
 C =-LE/F,E //W N                                              W         L2104
 D =-LA/F,E //W N                                              W         L2106
 E =-NOUS                                                      W         L2108
 F =-VOUS                                                      W         L2110
 G =-LES                                                       W         L2112
ZGND A $1=-MENTIR                                              W         L4000
 B =-TOMBER                                                    W         L4002
IM $1 //CG,*D1                                                       *   M5000
CG ER $1+$1=1+2                                                      *   M5010
 ET =1+2                                                            MB   M5012
 AL =A                                                              EF   M5014
 VE =A                                                              EG   M5016
PSNM A $1+$1=-ERA                                                 EROR   M5042
 B =1+2+E //*K2 3                                                   TR   M5044
 C =-ERC                                                          EROR   M5046
 D =1+2+ONS //*K2 3                                                 TR   M5048
 E =1+2+EZ //*K2 3                                                  TR   M5050
 F =-ERF                                                          EROR   M5052
MB                                                                   *   M5060
PSNM A $1+$1=-ETA                                                 EROR   M5062
 B =-SOIS                                                            W   M5064
 C =-ETC                                                          EROR   M5066
 D =-SOYONS                                                          W   M5068
 E =-SOYEZ                                                           W   M5070
 F =-ETF                                                          EROR   M5072
NXT $1 //*Q4 1,*N3 1                                           $         N0000
W Y $1=1 //*Q4 1,*N3 1                                         $         P0000
 N =*V+*Z+1 //*N6 1                                                  *   P0002
* $1+*Z=1                                                          CON   P0005
* *Z+$1=2 //*S6 1,*N3 1                                              $   P0015
CON $1/F,C+$1/P,C,E,F //*S6 1 2,*N3 1                                $   P0018
* $1/F,C+$1/P,C                                                Q         P0020
* $1/F,E+$1/P,V                                                S         P0025
* $=*V+*Z+1 //*N6 1                                                  *   P0030
* $1+*Z=1                                                                   Q   P0035
* *Z=0                                                                      T   P0040
Q -A=0                                                                 U        P0050
* -DE+-LE=-DU                                                          VV       P0060
* -DE+-LES=-DES                                                        VV       P0070
* -DE+-LEQUEL=-DUQUEL                                                  VV       P0080
* -DE+-LESQUELS=-DESQUELS                                              VV       P0090
* -DE+-LESQUELLES=-DESQUELLES                                          VV       P0100
U -LE=-AU                                                              VV       P0110
* -LES=-AUX                                                            VV       P0120
* -LEQUEL=-AUQUEL                                                      VV       P0130
* -LESQUELS=-AUXQUELS                                                  VV       P0140
* -LESQUELLES=-AUXQUELLES                                              VV       P0150
T $1+$1/F //*Q4 1,*S6 2,*N3 1                                          $        P0160
VV $ //*Q4 1,*N3 1,W Y                                                 $        P0170
* $=0                                                                       I   P0176
S -SI                                                                  X        P0180
* $1/S                                                                 Z        P0190
* $1=1+*M //*E1                                                        *        P0200
* $+$1+*M=1 //*K1                                                      *        P0210
TT $=*Z+1 //*N6 1                                                           *   P0220
* $ //*Q4 1,*N3 1,W Y                                                  $        P0240
X -SI+-IL=-S-IL                                                        TT       P0250
* -SI+-ILS=-S-ILS                                                      TT       P0260
*                                                                      TT       P0270
Z -CE=-CET                                                             TT       P0280
* -SA=-SON                                                             TT       P0290
* -MA=-MON                                                                 TT   P0300
* -TA=-TON                                                             TT       P0310
STOP $=-STOP-AT-+A+*. //*RAL2,*WAM1 2 3                                     *   Z9000
END

                                                                            720

                                                              TOTAL         720*
```

VI. SAMPLE RESULTS

The following sentences are the actual results of a computer "run" on September 4, 1962. Under the direction of the deck of IBM cards that constitute the programmed generative grammar, the IBM 7090 Computer at the Massachusetts Institute of Technology produced these sentences.. Each sentence below is a copy of the sentence with the same sequential number that was printed out by the off-line printer. The full print-out also includes the set of syntactic structures for each sentence as well as other supplementary material which is of interest to me for further research.

As the reader will observe, there are errors that remain to be corrected. I consider the framework of the grammar to be complete, but I have not stopped adding structures and vocabulary items. As I do this, both grammatical and programming errors occur. I have regularly been able to correct such errors on subsequent runs of the program. Most of the errors in sentences not included here were due to a lack of restrictions on coordination, one of the most difficult problems in generative grammars and one which I intend to study further.

It should be noted that the printer prints only in capital letters and that few punctuation marks are available.

(001) ELLES N APPELENT PERSONNE DE VOUS DONNER SUR LES AMERICAINES LORSQUE VOUS NE LE DONNEZ PAS A DES GRANDES HISTOIRES INTERESSANTES SOUVENT PERIOD

(005) SOYEZ VOUS TOUJOURS EXCLAMATION POINT

(008) MAINTENANT NOUS NE DONNONS PAS CEUX-LA TOUJOURS QUAND IL NE M APPELE PAS NE PLUS DONNER VOTRE CRAYON DONT TU NE LE LEUR DONNES PLUS ET DONT JE SUIS BLEUE SOUVENT A CETTE HISTOIRE BLEUE QUE TU DONNES A CELLE-CI SOUVENT ET A LAQUELLE TU NE VIENS PAS ET JE SUIS APPELE LA DE NE PAS MANGER LENTEMENT LA PERIOD

(012) VOUS AVEZ ETE DONNE BIENTOT ET DANS SOI PERIOD

(014) NE PARLE PAS SOUVENT EXCLAMATION POINT

(016) NE LUI SOMMES NOUS PAS APPELE DE NE PAS NOUS LAVER DES JOLIES JOLIES FINS SOUVENT DEPUIS LES FRANCAISES QUESTION MARK

(020) VOUS N ETES PAS BLEUS ET PERSONNE N A BEAUCOUP DONNE AVEC HENRI HENRI SOUVENT LORSQUE VOS GRANDES GRANDES FILLES ET VOUS M AVEZ BEAUCOUP DONNE DANS ELLES TOUJOURS PERIOD

(024) CELLES-CI ET NOUS NE TE SOMMES PAS APPELE A NE JAMAIS ME PARLER PAR CELLES DE CES GRANDES HISTOIRES ET ILS ONT ETE APPELE NE PAS SE LAVER BIENTOT LES EUROPEENS TOUJOURS PAR LES ETATS-UNIS PERIOD

(026) NE TE DONNE A PERSONNE SOUVENT ET HEUREUSEMENT NE LE DONNE PAS INTERESSANT EXCLAMATION POINT

(029) EN NOUS LAVE (error has been corrected, should be LAVANT) LES LES FRANCAISES TOUJOURS NE PERSUADEZ D ETRE SUR MOI ET SUR VOS PETITES FILLES JAMAIS RECEMMENT EXCLAMATION POINT

(031) DEPUIS NOUS NE VOUS LE LAVEZ PAS TOUJOURS LORSQUE CELUI-LA VOUS ME DONNE LA EXCLAMATION POINT

(035) NE PARLONS GUERE MAINTENANT EXCLAMATION POINT

(043) VOUS ET CELLES-LA LES LEUR AVEZ DONNE LENTEMENT PERIOD

(047) NE PARLONS PAS EXCLAMATION POINT

(050) NE L APPELONS PAS DE NE PAS ALLER LENTEMENT A DES CRAYONS BLEUS ET DE MES FINS BLEUES AUXQUELLES CES JOURNAUX ET LES FRANCAIS DONNEZ LES EUROPEENS SOUVENT QUESTION MARK

(053) ELLES NE SONT PAS BEAUCOUP APPELE A NE PLUS ETRE SUR SES PETITES GRANDES BONTES QUE TU MANGES EXCLAMATION POINT

(056) SOIS SOUVENT QUAND DES GRANDES PETITES HISTOIRES ET VOUS LES LEUR DONNEZ ET NE NOUS LAVONS RIEN JAMAIS EXCLAMATION POINT

(062) LENTEMENT CES PETITES GRANDES FINS INTERESSANTES NE VIENNENT PAS A CELLES DE CELLES-LA ET DES NATIONS-UNIES TOUJOURS ET JE LE LEUR AI BEAUCOUP DONNE ICI SOUVENT PERIOD

(063) CELLE D ELLE NE DONNE PAS LA FRANCE TOUJOURS PERIOD

(064) ICI ICI VOUS N ETES PAS APPELE A NOUS DONNER AUX AMERICAINES PAR MOI DEPUIS LES FILLES BLEUES ET LES NATIONS-UNIES PERIOD

(066) ZQFR NOT READY CELLES AVEC QUI TU NE PARLES PAS QUESTION MARK

(The interrogative phrase routine (ZQFR) is not yet included in the program but it is provided for in the framework of the grammar.)

(069) ZQFR NOT READY SA PETITE PETITE FIN NE LA LEUR DONNE PAS SOUVENT QUESTION MARK

(073) RECEMMENT LORSQUE TU N ES PAS TOUJOURS JE NE ME TE LAVE GUERE PERIOD

(076) ON ME LES A BEAUCOUP DONNE SOUVENT MAINTENANT ET HENRIETTE S EST LENTEMENT LAVE HENRIETTE TOUJOURS PERIOD