[*Proceedings of the National Symposium on Machine Translation*, UCLA February 1960]

Session 6: SYNTAX

SYNTACTIC RETRIEVAL

Paul L. Garvin[1]
Ramo-Wooldridge Laboratories and Wayne State University

Let me state briefly that in my opinion the major purpose of a syntax routine in machine translation is to recognize and appropriately record the boundaries and functions of the various components of the sentence. This syntactic information is not only essential for the efficient solution of the problem of word order for the output, but is equally indispensable for the proper recognition of the determiners for multiple-meaning choices.

It is further becoming increasingly apparent in the work in which I am participating that it is the design of the syntax routine which governs the over-all layout of a good machine translation program and lends it the unity without which it would remain a patchwork of individual subroutines and piecemeal instructions. The conception of syntax thus becomes important beyond the immediate objectives which the routine serves in the program.

In the present paper, I should like to set forth some of the linguistic basic assumptions underlying my own approach to syntax, and some of the design features of the syntax routines that have been and are being developed from it.

My conception of linguistic structure, insofar as it concerns syntax, is comparable to what has become known as the "immediate-constituent model", but with some significant differences. Where the immediate-constituent approach takes the maximum unit--the sentence --as its point of departure and considers its step-by-step breakdown into components of an increasingly lower order of complexity, I prefer to start out with the minimum unit--the morpheme in straight linguistic analysis, the typographical word in language-data processing--and consider its gradual fusion into units of increasingly higher orders of complexity, which I call fused units. A sentence is thus conceived of, not as a simple succession of linear components, but as

[1] Editor's note: Dr. Garvin was a member of the faculty of Georgetown University at the time of the Symposium.

a compound chain of fused units of different orders of complexity variously encapsulated in each other. Syntactic analysis, including the automatic analysis which an MT syntax routine must perform, then has as its objective the identification of this encapsulation of fused units by ascertaining their boundaries and functions.

The fused-unit approach is particularly well suited to language-data processing, since the minimum units--for this purpose the typographical words--constitute the primarily given sensing units from which the program computes the fused units and their interrelations. The methodological basis for this computation is what I have called the fulcrum approach to syntax.

The fulcrum approach is based on the conceptualization of fused units as exhibiting the separate properties of internal structure and external functioning respectively. Internal structure is here defined as the constituency of a fused unit in terms of units of a lower order; external functioning is defined in terms of the relations of a fused unit to units of the same order, together with which it enters into the makeup of units of a higher order.

The concept of the fulcrum itself stems from the consistent observation that the various components of a fused unit have differential grammatical information content: one of them, the fulcrum, may be expected to be more informative than the remaining components about the properties of the unit of which it forms part. By using the fulcrum of each unit as a point of departure, its identification as to internal structure (and hence boundaries) and external functioning can be achieved more accurately and completely. By tying together fused units of different orders through their fulcra, the syntax program can acquire the hierarchic organization and unity desirable for maximum flexibility.

Let me give an example.

The fulcrum of a main clause in Russian is its predicate. Why the predicate rather than the subject or a complement is chosen as the fulcrum becomes clear if one considers the relative amount of information that each of these three clause members gives about the other two and hence the clause as a whole.

If the predicate of a clause is known, the agreement characteristics of the predicate, such as number and in certain cases gender,

allow a reasonable prediction as to a permissible subject, and its government characteristics allow a reasonable prediction as to permissible complements. A predicate thus allows a reasonable prediction with regard to both remaining clause members.

A nominal block (that is, the MT analog of a nominal phrase), on the other hand, will at best allow a partial prediction as to one of the two remaining major clause members: if its agreement characteristics as to case unambiguously mark it as subject, then its agreement characteristics as to number and gender will allow the assumption of a predicate in the plural if the nominal block is in the plural; but will allow a predicate in either the singular or the plural if the nominal block is in the singular, since the latter may be one of a string of blocks which together may permit a predicate in either number. No further predictions are possible from knowing a nominal block by itself: if its case-agreement characteristics mark it as a non-subject, it still does not follow that it is a complement, since it may be governed by non-predictive material or not subject to government at all; its government characteristics will allow an extension of the block but will not yield further information about the remaining major clause members.

The identification of the fulcra of units of lower orders is by comparison more obvious: the fulcrum of a nominal block is the noun; the fulcrum of a prepositional block is the preposition, since it allows the prediction of the case-agreement characteristics of the nominal block governed by it, etc.

A syntactic retrieval routine based on the fulcrum approach will first identify the fulcrum of a given fused unit and then use it as the initial point from which to retrieve the boundary and function information required for the continued operation of the program. The identification of the fulcrum is made possible by incorporating the relevant information in the grammar code of the words stored in the dictionary.

Identification of the fulcrum presupposes a grammar code organized in terms of the potential syntactic functioning of the words rather than in terms of their morphological origin. This is particularly significant in this connection as regards the indication of word class membership.

Thus, all words that may unambiguously function as predicates are given the same word-class designation in the grammar code--that of predicatives. This includes not only finite verb forms but also unambiguous predicative adjectives and certain other words. Conversely, the different forms of words that are traditionally considered the same part of speech are assigned different word-class membership if they have different syntactic function. Thus, the various forms of a verb are coded for word class as follows: finite verb forms, as mentioned above, are coded as predicatives; infinitives and gerunds are coded as separate word classes; participles are coded as "governing modifiers" together with certain adjectives which have government properties similar to those of participles.

To find a fulcrum, the program will read the word-class field of the grammar code of each word that the lookup has brought forth. If the word is of a class that functions as the fulcrum of a fused unit of a particular type, this information serves as the signal to call the subroutine designed to identify the boundaries and potential function of the unit in question.

The dependence on the grammar code for the initial identification of fulcra implies that this initial search must be limited to one-word fulcra. Since it is impressionistically obvious that not all fused units will have one-word fulcra--particularly, units of a higher order can be expected to have fulcra that are themselves fused units--the program will have to include provisions for the recognition of the boundaries and functions of multiword fulcra based upon the prior identification of their components, beginning with the initial identification of relevant one-word fulcra. This in turn implies, and is closely related to, the over-all problem of the order in which the fulcra of the fused units of different orders are to be identified, so that the sequence in which the search for the various fused units is conducted leads to the correct recognition of their encapsulation.

Rather than attempting a consecutive left-to-right solution of this set of problems, the syntax routines conceived in terms of the fulcrum approach have attacked it by a consecutive series of passes at the sentence, each pass designed to identify fused units of a particular order and type. The advantage of this pass method over a single consecutive left-to-right search is, in my opinion, that, instead

of having to account for each of the many possibilities at each step of the left-to-right progression, every pass is limited to a particular syntactic retrieval operation and only information relevant to it has to be carried along during that particular search. With the proper sequencing of passes, the syntactic retrieval problems presented by each sentence can be solved in the order of their magnitude, rather than in the accidental order of their appearance in the text.

In a program based on the pass method, each individual pass is laid out in terms of the information available when the pass is initiated, and in terms of the objective that the pass is intended to accomplish. These two factors are closely related to each other, in that the output of a preceding pass becomes the input of the subsequent pass. The scope of each pass and the order of the various passes thus together present the most significant design problem of the program.

The linguistic considerations entering into this design problem stem from the differential relevance to the over-all structure of the sentence of the various orders of units and their relations. Viewed in terms of the ultimate aim of the program in regard to syntactic resolution, which is the capability for rearranging the order of the major sentence components (that is, subjects, predicates, and complements), the relations between these components become the focal point around which the remaining syntactic relations can be said to be centered.

When this is applied to the organization of the passes, it means that the main syntax pass--that is, the pass designed to identify the boundaries and functions of the major clause members of the main clause--becomes the pivot of the program. The remaining passes can be laid out in terms of the input requirements and expected output of this central pass. Preceding it will be preliminary passes designed to assign grammar codes to words which are not in the dictionary (a missing-word routine), and to aberrant typographical matter such as symbols and formulae, as well as passes designed to compute the information needed as input to the main syntax pass from the information available to the program through the grammar code. Following the main syntax pass will be clean-up passes, the function of which is to fill the gaps in syntactic information remaining after the main syntax pass has accomplished its objective.

Let me now discuss the function of the preliminary passes

required by the discrepancy between the information contained in the grammar code and the information necessary for the main syntax pass.

The grammar code furnishes three sets of indications: word-class membership, agreement characteristics, and government characteristics. As is well known, for each dictionary entry, some of this information will be unambiguous, some ambiguous, depending on the particular word forms involved.

Aside from accidental typographical homonyms (such as est' = "is" or "eat"), grammatical ambiguities relate to word-class membership and agreement characteristics; where ambiguities as to government characteristics were found, they were dependent on another grammatical function, that of word-class membership.

While the main syntax pass may tolerate agreement ambiguities (although it is not always the most efficient place in the program for their resolution), it can not admit word-class ambiguities in its input, since the fulcrum approach is based on the recognition of the fulcra by their word-class membership. One of the essential functions of the preliminary passes is thus the resolution of ambiguous word-class membership.

It is, furthermore, reasonable to expect that sentences will contain discontinuous fused units, that is, fused units interrupted by variously structured intervening elements. Unless such intervening structures are properly identified in prior elimination passes, the program will not be able to skip over them in the search for elements functionally relevant to the objectives of the later syntactic passes.

Finally, given the relative independence of the internal structure, and the external functioning of units, alluded to further above, a number of constructions can be expected within each sentence which by their internal structure resemble potential major clause members, but do not in effect have that external functioning.

An example of this are relative clauses: their internal structure resembles that of a main clause, and they contain similarly structured clause members, but their external functioning is that of inclusion In nominal blocks as modifying elements. Constructions such as these have to be identified by appropriate preliminary passes and their boundaries and functions recorded for inclusion in the main syntax.

Once the inventory of linguistic problems has thus been systematically formulated and related to the general characteristics of the syntactic retrieval program, the actual operational sequence of passes will have to be ascertained by programming experimentation. It depends not only on the grammatical ordering of the data but also, and primarily, on the input and output features of the various passes. In addition to linguistic necessity which dictates the handling of certain information by preliminary passes, considerations of programming convenience and efficiency may lead to an increase in the number of passes, or conversely, bring about the merger of several passes into one. The pass method provides the frame within which the problems can be isolated well enough to allow control, and be viewed in a sufficiently general perspective to allow coordination and flexibility.