[From: Symposium on Modern Techniques of Language Translation,
Information Processing 1962: proceedings of IFIP Congress 62,
Munich 27 August – 1 September 1962 (Amsterdam: North-Holland, 1963)]

3. A PARSING PROCEDURE

M. KAY *(UK)*

A large family of strategies can be devised for a parsing procedure, in such a way that all immediate constituent structures allowed by a given grammar are developed quickly and easily. The one presented here has been chosen, because it enables the notion of *presupposition* to be fully exploited as a means of referring to the grammar. This works as follows: For each immediate-constituent rule of the form $A \rightarrow BC$, one of the elements on the right-hand side is chosen as presupposing the other. For example, we may say "*C* presupposes a preceding *B* to form *A*" and write

$$C|- B \mid A.$$

Alternatively, we may write

$$B| + C \mid A.$$

The plus and minus signs indicate whether the presupposed item comes to the left or the right. The choice of the presupposed item in a rule is made with a view to minimizing the greatest number of items presupposed by any one item.

The parsing procedure has the following salient characteristics:

1. A constitute consists of two and only two constituents.

2. All partial structures ending at a given word in the sentence are developed before the following word is examined.

In describing the procedure, we shall allow ourselves to say that a constitute *A* precedes another constitute *B* only if the first word of *B* immediately follows the last word of *A*.

The first word of the sentence is read into the machine, and its *n* grammar codes entered in the first *n* spaces on the main data list. The entries are annotated to show that they are constitutes beginning with the first word of the sentence. A counter is associated with each word read into the machine, indicating the point in the list where the first grammar code for that word is stored. These counters, together with the word numbers in the data list, enable the items which precede a given word to be located very readily.

Also associated with each word is a *prediction list.* The prediction list for the first word of a sentence is always empty, and the list for the second word is completed as soon as the first word has been read into the machine. If the first word, or, in general any new item, presupposes a following item, a note of the relevant grammatical details is made on the prediction list of the following word. We are now ready to read the second word. Its *m* grammar codes become the next *m* entries in the data list. A new pointer is set and predictions are made as before. Parsing proper now begins. Each grammar code for the new word is considered in turn to see if it fills any of the predictions in the list for that word, and if it presupposes any of the items which precede it in the sentence! If either of these conditions is met, a new constitute is formed and an appropriate entry made in the next available space in the data list. The new entry is marked with the word number of the first word which it includes, and also with references to the two entries in the list which represent its constituents, all having the new word as last member.

These are now taken in turn as potential right-hand constituents of new constitutes. The process continues in this manner until all the items currently on the data list have been considered together with all the items which precede them, as candidates for a new constituent. Only then is a new word brought into the store.

Consider the sentence: "We are parsing sentences". A very simplified grammar will enable us to develop two structures.

VERB	+ NOUN	PREDICATE
VERB	— NOUN	SENTENCE
PRES. PARTICIPLE	— AUXILIARY VERB	
PREDICATE	— NOUN	SENTENCE
ADJECTIVE	+ NOUN	NOUN

At the end of the procedure, the data list might appear as follows as in Table 1.

Table 1

No	WORD	GRAMMAR CODE	WORD NUMBER	FIRST CONSTIT.	SECOND CONSTIT.
1	WE	NOUN	1	0	0
2	ARE	AUXILIARY	2	0	0
		VERB	2	0	0
3		SENTENCE	1	1	3
5	PARSING	PRES. PART.	3	0	0
6		ADJECTIVE	3	0	0
7		NOUN	3	0	0
8		VERB	2	2	5
9		PREDICATE	2	3	7
10		SENTENCE	1	1	8
11		SENTENCE	1	1	9
12	SENTENCES	NOUN	4	0	0
13		NOUN	3	6	12
14		PREDICATE	2	8	12
15		PREDICATE	2	3	13
16		SENTENCE	1	1	14
17		SENTENCE	1	1	15

The results happen to be the last two entries on the list. The complete trees can be traced out using the last two columns.

DISCUSSION

R. M. NEEDHAM *(UK).* Since it is easy to devise many procedures, and individual ones present little novelty, how does one choose a good one?

M. KAY *(UK).* I wish I could give a good general answer to this question. Whatever reasons one may find are unlikely to be more than superficial. However the following points may be of interest.

1. Presupposition is only useful in a few procedures.
2. It is often difficult to identify the end of a sentence, since the period is not used only for this purpose. A procedure, such as the one described, which identifies everything up to a given point in the sentence may be useful, in that it can be discontinued only when a suitable result occurs together with appropriate punctuation. In other words, the length of the sentence need not be known initially.
3. It might be useful to discover alternative sentence structures in order of their increasing depth, as defined by Yngve. However, this can readily be computed in the process of working down the tree.

All these procedures involve nested cycles of instructions and it is worth considering if any one is particularly suited for use on a given machine.

4. THE PROGRAMMING OF GRAMMAR

H. SCHNELLE *(Germany)*

I shall describe a dynamic interpretation of a given grammar quite different from the interpretations currently in use. In the latter, the rules which constitute the grammar are interpreted as rewriting rules for given sequences of symbols. In contrast to this, it will be shown how syntactic structures, formulated in a context-free constituent-structure grammar, can be activated directly and automatically. Among the several advantages this programming system offers, I want to stress the fact that this procedure activates the grammar both for generation of sentences and for recognition of their structure.

From the programming point of view our system can be characterized as a generalized *bidirectional list processor* (which may contain recursion cycles). The list can be obtained from a constituent-structure grammar, e.g. in the Chomsky regular form[1]), by an automatic procedure. Since such a grammar can be viewed as a unidirectional list, each rule indicating the two "subsequent" list members (i.e. rules), in addition to the (operator-)relation between them (e.g. P and Q, in $N = P + Q$ or $N = P, Q$); and since it is essential for our program to find the way back in the list, we make the list bidirectional by indication of the "previous" list members and the alternatives in this direction as well. Moreover, places or programs which specify the choices at alternative rules should be made explicit. The following is an example of such a transformation.

The grammar for the (English) object clause is the following list of rules:

OBJ = POBJ, DOBJ
POBJ = *Prep* + DOBJ
DOBJ = *Art* + *Subst*

The corresponding bidirectional list is shown in Table 1.

Table 1

Number of rule	rule name	Number of position for activation signal: 1 2 3 4 5 6 7 8 9 10 11 12
1.	OBJ	-PRED - - - POBJ - - - DOBJ - - ∇OBJ- - -
2.	POBJ	-OBJ - - - PREP - - - DOBJ - - +
3.	DOBJ	-DOBJ - - - OBJ - - - POBJ - - ΔDOBJ - -
4.	DOBJ	-DOBJ - - - *Art* - - - *Subst*- - +

Obviously the 3rd rule has been added to indicate that DOBJ can be reached both from OBJ and from

POBJ ; the rule DOBJ mediates this relation. In the other rules, the first element indicates the previous rule (e.g. it is assumed that OBJ is reached from PRED) while the three subsequent elements, being a combination of operator and possibly rule name, indicate a specification address. The underlined names are terminal categories to be generated or recognized. The processing of the list is made by activation signals be placed on and transferred over the positions indicated by the hyphens. (Putting addresses into index registers would be equivalent to placing activation signals.) The hyphens to the left of the names are for signals in the "down" or "enter" direction, those to the right for "return" signals, the first one for "structure generation finished" resp. to "structure affirmatively recognized", the second for "structure negatively recognized". These return signals are set by the outcome of the writing or reading program for terminal categories. For the purpose of illustration we describe the process of 1. generation of a direct object (DOBJ) and 2. recognition of the same structure by giving the sequence of positions of the activation signal during the processes.

1. *Generation,* (Specification activation placed on 1.11: activation signal from PRED): 1.1, 1.7, 2.4 (Set spec. to 3.10 for reference of return signal), 3.1, 4.1, 4.4 (output of "article" finishing by:) 4.5, 4.7 (!), (output of "substantive" finishing by:) 4.8, 4.2, 3.2 (obtain spec.: 3.10), 3.5, 1.8, 1.2 (finishing signal to PRED).

2. *Recognition,* (Activation signal from PRED, spec. obtained: 1.10 and 1.11 i.e.: ask left structure and right structure) 1.1, 1.4 and 1.7, 2.1. and 3.4 (set spec. 3.10), 2.4 and 3.1, (2.4 activated the reading program "preposition present?" answer "no":) 2.6 and 4.1, 2.3 and 4.4, 1.6 and (4.4: "article?": "yes":) 4.5, (n.t, i.e. no transfer of 1.6, wait for second return signal) and 4.7, (n.t.:) 1.6 and (4.7 "substantive?": "yes":) 4.8, (n.t.:). 1.6 and 4. 2, (n.t.:) 1.6 and 3.2, (n.t.:) 1.6 and (spec. obtained 3.10:) 3.8, 1.6 and 1.8, 1.2 (and spec. set to 1.11:) structure DOBJ affirmatively recognized, "yes" signal return to PRED.

In a *translation* system the specification obtained by the recognition routine for the source language is transferred to a generation routine for the target language. In our extremely simple example, the transfer from English to German will be executed simply by identifying the address ∇ OBJ of the two systems, since the system for German will be the same with the addition of a morphological part, the description of which cannot be included here.

Recursion can be taken care of within the given framework in a very natural and simple manner. For each self-embedding (or nesting) recursion cycle there must be provided a potentially infinite (or growing) push-down store, whereas simple recursion (e.g. adjectival chains) can be programmed without additional storage facilities.

Unrestricted combinatorial grammars (and therefore also transformational grammars) can be programmed by a slight extension of the concept of specification, namely that the specification is communicated not only to the exterior but to other points *within* the system[2]).

REFERENCES

[1]) Chomsky, N.: *On Certain Formal Properties of Grammars.* Information and Control **2** (1959) 137-167.

[2]) Schnelle, H.: *Rules for the Programming of Grammar.* Mimeographed, Institut für Phonetik und Kommunikationsforschung — Universität Bonn.

5. EXHAUSTIVE PARSING PROCEDURES *

D. G. HAYS *(USA)*

The separability of parsing procedures from grammars is now well known. Equally well known is the fact that parsing procedures can be written with a variety of purposes: to find a single structure (if any) for each sentence, to find a few structures, or to find all structures allowed by the grammar. This note merely calls attention to three different plans for the design of *exhaustive* procedures, and examines one plan in fairly general terms. The grammar assumed here must be of the "context-free phrase-structure grammar type". This includes dependency grammars, predictive grammars, immediate-constituent grammars with two or more constituents per constitute, string-constituent grammars, etc. The three plans are here called *backtrack, parallel,* and *morsel* plans.

A backtrack plan is based on a decision tree. Parsing involves a equence of decisions; at each decision point: zero, one, or more alternative branches are open, and exactly one must be chosen. When a point with zero paths is reached, either the sentence has been parsed or the sequence of decisions already made cannot lead to a complete parsing. In the first case, the completed structure is output. In either case, the next branch (in arbitrary but definite order) at the latest decision point still containing an open branch is followed, leading to another dead end or another complete structure. The parsing is complete when the decision tree has been completely traced.

In a parallel plan, all branches of a decision tree are followed simultaneously. With present equipment, this can only mean that all branches at a given decision point are considered before passing to the next. For each branch, a separate result is stored.

In a morsel plan, fragments are put together in such an order that the *n*-th fragment to be attempted is necessarily constructable as some combination of a selection of the *n*-1 fragments already constructed.

Consider as examples, a backtrack plan with predictive grammar, a parallel plan with immediate constituents, and a morsel plan with dependencies.

* This note is based on work performed in close collaboration with Sheila Greibach and M. J. Kay at The RAND Corp. Space for citation of the literature is lacking, but the reader should consult the Proc. 1961 Inter. Conf. on Machine Translation of Languages, published by H. M. Stationery Office, London, for examples of parsing procedures hitherto proposed.

Backtrack, predictive: At each decision point a list of predictions is compared with the possible grammatical interpretations of a word. The word may satisfy a prediction in zero, one, or more ways. Choice of a branch changes the prediction list and leads to a new decision point. In order to backtrack, the procedure must store at each point the structure determined up to that point, including the list of predictions, and also a way of deciding which branch to take next.

Parallel, immediate-constituent: Over the first *n* words of the sentence, all possible structures are built, including those in which no constitutes are made. The (*n* + l)st word is tested as partner with the largest constitute ending at word *n* in each structure. If a new constitute is formed in this way, it must be tested with possible partners, etc.

Morsel, dependency: Each morsel is a subtree. The ordering variables (from major to minor) are span (number of words included), location, span of left-hand member, grammatical interpretation of first member, grammatical interpretation of second member, and direction of dependency.

The morsel plan can be designed in such a fashion that no unnecessary tests are made, no necessary test is made twice, and no portion of the structure of the sentence is stored in two places. Perhaps the same advantages can be obtained for the other plans, but the housekeeping seems simpler for the morsel plan. For a dependency grammar, it is only necessary: (a) to make connections between heads of subtrees, and to attach all following dependents to a node before any of its preceding dependents, and (b) to store in a list all subtrees alike with respect to length, location, and grammatical type, but differing in other respects, and to refer to the list rather than to the individual subtrees in all further constructions.

6. MACHINE TRANSLATION: The End of an Illusion

Y. BAR-HILLEL *(Israel)*

Thirty years after the first studies on the mechanization of translation between natural languages, sixteen years after the first serious discussions on the use of electronic digital computers for this purpose, ten years after the first conference on this topic, now that thousands of man-years and tens of millions of dollars have already been spent on machine translation research, it is not too early to admit defeat.

The failure to obtain high-quality fully-automatic translation of natural languages could not have been predicted *a priori,* and most arguments given by critics ten years ago were fallacious.

There are at least five prerequisites for high-quality human translation:

1. Mastery of the source language,
2. Mastery of the target language,
3. Good general background knowledge,
4. Expertness in the field,
5. Intelligence (or "know-how").

Ten years ago, the extent to which the last three factors were important for obtaining good translations was not sufficiently well known, nor the extent to which the fact that computers were lacking of them could be counterbalanced by a fuller utilization of redundancy in ordinary writing.

Experience has shown that these factors are of decisive importance and that there exists a definite limit to their replacement by operations available on computers.

Nor does it seems possible that computers, which in some way are able to learn, could overcome the above-mentioned handicap. There is no way in view, of imbuing computers with the *faculté de langage,* in the sense in which this term was used by the Swiss linguist Ferdinand de Saussure. This is characteristic of, and innate to, humans and enables them fully to master, at an early age and under appropriate conditions, any language spoken in their environment. Therefore, even, "learning machines" will not reach a level of human-like use of language.

Current methods of mechanizing the determination of syntactic structure all suffer from the inadequate model of grammar with which they are explicitly or implicitly working. It has been shown that (almost) all of them are variants of context-free phrase-structure grammars, and such grammars are far too clumsy for an adequate description of natural languages, if not theoretically inadequate. The mastery that can be achieved by a computer fed with instructions based on these models is therefore restricted. Better models, such as transformational grammars, or the grammars sketched by Dr. Schnelle in his talk, should increase this mastery, perhaps up to practical adequacy. There is, however, the prohibitive restriction that the mechanical determination procedure of syntactic structure will, for long and complex sentences, be extremely tedious, time consuming, and expensive, and will have as its output a large number of possible syntactic analyses. Though machine translation of natural languages has ended in failure, there remain the highly interesting tasks of further developing algebraic linguistics in general, and computational linguistics in particular, as well as of mechanizing translation between programming languages.

DISCUSSION

M. PIVOVONSKY *(USA).* I agree that high quality machine translation is not feasible, but I believe that translation of better quality than that achieved now, is feasible. The problems are: What is the upper limit of the quality attainable, and can this question be asked without any further experimentation?

A. KENT *(USA).* I, too, agree that high quality translation is not feasible. However, one of the chief purposes of translation is the reasonable communication of the contents of foreign language records to specialists, and this may be accomplished by providing the specialist with a low-quality mechanical translation. This

will allow identification of an interesting document from a collection of documents, and identification of interesting portions of the document for "precise" translation by humans.

Y. BAR-HILLEL *(Israel).* Whether low-quality machine translation is of any use, is for others to decide. I tend to believe, that in this particular game, humans will be able to beat a translation machine in all foreseeable time. Though I agree that the exact place of the upper limit of the quality attainable has not yet been determined, I think that we know enough to state unqualifiedly that this limit is too low for almost all practical uses.

H. P. GLOCKMANN *(Germany)* and J. VAN HORN *(USA).* What do you propose as a means of solving the tremendous translation problem now existing for the transmission of information, particularly of results of scientific and technical research, generated in various languages?

Y. BAR-HILLEL *(Israel*). I do not think that the translation of scientific material today poses a worse problem than in previous times. I dealt with this question on another occasion and do not want to repeat my arguments here.

A. CARACCIOLO *(Italy).* I should ask Pr. Bar-Hillel whether he thinks that human translation is possible, at least in general. Natural languages make it quite easy to formulate contradictory sentences. How should such a sentence be translated?

Y. BAR-HILLEL *(Israel).* The translation of a contradictory sentence should, of course, be a contradictory sentence in the target language. Under certain conditions the translator might wish to add a *sic*.

Human translation is possible, and many people make a living out of it. What you probably intended to ask was rather whether humans can provide "perfect" translation. I would however prefer not to draw this red herring of "perfect" translation into our discussion. The standard of comparison is, of course, just what one would ordinarily call high-quality translation.

HELEN BROWNSON *(USA).* The references to a time period of thirty years and to the expenditure of tens of millions of dollars are misleading. Although a conference was held ten years ago, the oldest research project in the field is only about six years old, and only a few million dollars have been spent on research. My point is that the field is still in its infancy, and the contention that we should admit defeat seems just as premature as claims that the problem is virtually solved. Do you have any *evidence* that we will not learn how to program machines to handle the complexities, as we learn more and more about these complexities?

Y. BAR-HILLEL *(Israel).* Though the references might mislead some people, they are still correct. Around 1933 the Russian, Smirnov-Troyanski, and the Frenchman, Artsouni, made some quite detailed proposals for mechanical devices for translation.

I had no intention of implying that research time and money were evenly distributed during the periods mentioned. Though it is surely a matter of judgment when to admit defeat in the pursuit of a certain aim, I believe that the time has arrived when the responsibility for making such a decision in the field of mechanical translation should no longer be postponed.

P. STONE *(USA).* The work of such psychologists as Piaget and Bruner has shown that the detailed strategies of human cognitive functioning can be revealed by quite simple experimental techniques. Have such techniques been employed to study the strategies used by humans in translation? It would seem that such a study of human functioning would be a basic initial step towards building later mechanical models.

Y. BAR-HILLEL *(Israel).* What Piaget and Bruner have shown is only that *some* detailed strategies of human cognitive functioning can be revealed by such techniques. To my knowledge, no such techniques have been employed to study the strategies used by human translators, though it was the first idea I had when I started working on mechanical translation eleven years ago. I would agree that a study of human translation might well be a necessary preliminary step towards the construction of mechanical models.

A. SESTIER *(France).* I suggest that nobody knows how to measure objectively and accurately the quality of any translation chosen at random. Failing which, everybody tries to have his own scale of evaluation, and our discussion has reflected primarily the disagreement of these scales. I suspect that only evaluations based on the satisfaction of users can be valid, but human and mechanical translation have almost disjoint sets of advantages and drawbacks, and this situation will undergo no substantial change in the foreseeable future. Therefore, as long as this situation lasts, their purposes must be different and the same scale of evaluation cannot apply to both.