Mechanical Translation by the Thesaurus Method Using Existing Machinery

A.F.PARKER-RHODES
and C. WORDLEY

The thesaurus method of machine translation is designed to provide a means of obtaining the best translation of a word in a given context by defining every word in terms of the context in which it can occur. These definitions can be represented by elements of a lattice. By operations on the elements representing the words of a text, it is possible in principle to derive a new set of elements (a) which together carry the same information as the input set; and (b) each of which corresponds to some word in the given target language. Work is proceeding on practical methods of programming such procedures using (a) punched-card equipment and (b) digital computers.

Most of the work done hitherto on machine translation has been based on the hope that both semantic and syntactic problems could be solved by a set of binary decisions. That is, it was hoped that the problem of finding the right rendering of a given word in a text into a word of a given target language could be solved by deciding between a fixed set of choices according to a fixed schema of questions relating to a manageable number of adjacent words or structures. Experience shows that whenever such methods have been tested adequately, that is on unselected material not itself used in the preparation of the dictionary entries which the procedure requires, only low-level translation has been achieved. The value of low-level translations of this type varies greatly with the nature of the target language. In a highly inflected language indifferent to word-order, such as Russian, this level of translation may be of some practical use, but with English as the target language it tends to be too great a strain on the reader to understand what is being said for it to be acceptable.

Accordingly, the work of the Cambridge Language Research Unit has been based on the attempt to construct an adequate theory by means of which this basic translation problem could be solved in a more fundamental manner than by inadequately generalizable choice-procedures. It has also been our belief that any theory really capable of solving the semantic problem would also be applicable or at least adaptable to the problem of syntax transformation as well. The thesaurus method, with which our name has come to be associated, is the outcome of our search for such a theory.[1]

When asked, "What is a thesaurus?," the simplest answer is to point to some literary thesaurus, such as the well-known Roget,[2] and say that a thesaurus is any system of classification of words of a language of this form and type. For those who are not familiar with such works, they may be described as inverted dictionaries; whereas in a dictionary one looks up a word and finds its meaning, defined typically by a list of its possible uses, in a thesaurus one starts from some indication of what one wants a word to mean, and from this indication the thesaurus leads one to a word or selection of words which will express the desired meaning. Essentially then a thesaurus, like a dictionary, is a means of defining words in terms of their possible use, but the arrangement of the two works is on opposite principles.

The mathematician will appreciate that a thesaurus thus defined may be regarded as a method of encoding the meanings of words. It is arranged under a number of "heads," each of which contains words which have in some sense a similar meaning. In practice, the heads can be regarded as contexts; all the words listed under one head are capable of occurring in the same context, or at least set of related contexts; since these contexts are essentially extra-linguistic, the system is in principle interlingual. The thesaurus is capable of distinguishing between words which occur under different selections of heads; it does not make any distinction between two words which are given under the same set of heads. It follows that the thesaurus definition of a word can be represented as a sequence of binary digits the same in number *as* the heads of the thesaurus, each being a 7 if the word occurs under the corresponding head and a *0* if it does not.

The number of these heads in Roget's Thesaurus is 1000. In other works of the kind (there are remarkably few in any language) comparable numbers are found. It would seem to follow that a useful (but not necessarily sufficient) distinction between the words of a language can be made by assigning to each word a symbol of 1000 bits. It is therefore possible to take as the first step of a translation procedure the replacement of each word (or part of a word, for the principle is equally applicable to affixes and other word-components) in the input text by one 1000-bit symbol, and to expect that its semantic properties will be fairly adequately represented thereby.

Presented on October 23, 1958, at the Society's Convention at Detroit by A. F. Parker-Rhodes (who read the paper) and C. Wordley, Cambridge Language Research Unit, 20 Millington Rd.,Cambridge,England.
(This paper was received on October 2, 1958.)

The Thesaurus as a Lattice

This device immediately suggests that a thesaurus might be regarded as a lattice; and moreover that it could be *explicitly* interlingual. The set of all possible 1000-bit symbols constitutes the Boolean lattice of order 2^{1000}, under the ordering relation that an element *a* includes an element *b* whenever every 1 in the symbol of *b* corresponds to a 1 in that of *a*. The elements of this lattice which correspond to actual words of a given language will not be all of these of course (the vast majority of possible symbols will not represent words in any language: remember that 2^{1000} is an exceedingly large number); but they may form a sublattice of it, and can in fact be made into a lattice by the addition of entries, all of which can be equated if not with single words at least with phrases, except for one element representing a word of *completely indeterminate meaning* and one representing a word of *null meaning;* these are the top and bottom elements of the whole lattice. The value of this construction is that if the thesaurus can be represented as a lattice, the methods of lattice algebra can be called upon to devise algorithms with which to handle the material.

As a simple example, consider the two English sentences, "He's been working on that problem for two months," and "He's been working on that site for two months." The word *working* carries a different meaning in the two cases, and if either were to be translated into some other language it would in general be necessary to know which of the two meanings was operative. A thesaurus will have, among many others, heads relating to *manual labor* and *intellectual labor;* the word *work* (and of course its inflected forms if these are to be separately listed) will itself occur under both. The fact that in the first sentence we have the word *problem* can be made use of to select the head *intellectual labor,* in such a way that in the symbol which we find for the translation of *working* there will be a 1 in the place given over to this head, but a 0 in the place assigned to *manual labor.* Likewise, the opposite arrangement would be produced by the same procedure on the second sentence, because there we have the word *site,* which will occur under the *manual labor* head but not under *intellectual* head.

Of course, this is an extremely simplified case; but the selection of the appropriate heads, once we have decided what words belong together and how we are to bring them into the calculation, is an extremely simple operation. If all the cases we come across can be reduced to a sequence of such operations, even if very many of them are required, then we shall have an essentially simple procedure for getting at the translation of each word in any input sentence. In particular, if this is how we are to handle the effect of context on the translation of words, we can handle any amount of context (i.e. any number of neighboring words) with equal facility, and moreover by performing the operation in appropriate stages we can represent in our procedure whatever details of the syntactic structure of the input text we find it necessary to carry over. There is, therefore, reason to expect that the method may be a very powerful one, if it can be conveniently mechanized.

The Problem of Syntax

It is convenient to treat the processing of the syntactic and semantic parts of the translation process separately, though they are intended ultimately to be integrated into a single process. We are still uncertain as to the best way of handling syntax, but the choice broadly lies between reducing the syntactic information recorded in our dictionaries to an absolute minimum, consistently with being able to ascertain, when necessary, the grammatical structure of any passage; and attempting to assimilate the grammatical structure to the pattern of thesaurus heads. That the latter may be possible is suggested to us by the fact that word-classes such as nouns or prepositions may be defined, at least approximately, by reference to the sequences of word-classes in which they can occur, that is, in a certain sense, by the contexts in which they occur, which is the same principle that the thesaurus uses to define semantic properties of words.[3] However, working out this idea in practice is difficult, and we have made more progress with the other method, of setting aside the minimum number of bits of information to determine syntactic behavior.

Investigation at the linguistic level has shown us that a very simple classification of word functions may be attainable which is both sufficient to infer the complete sentence structure from, and valid for all languages, at least with the help of comparatively few special rules regarding word order, etc. Roughly, we classify words first between noun-like and verb-like functions, and next between principal qualifiers and secondary qualifiers; these terms being defined with sufficient rigor to make the dictionary entry determinate and not subject to personal opinion. We also recognize not only a word's own function, but note whether its occurrence is confined to larger groups of definite function as well. Thus, a relative pronoun is not only a word of noun-like function, but it always belongs to a group (the relative clause) which has a qualifying function for another noun. We also require to use information regarding the place of a word in its group when this is available. Thus, we distinguish initial words (prepositions for instance), final words (like the English possessive *'s),* etc. To deal with the cases, numerous in some languages, where words show ambiguity of function even when this is reduced to such a simple scheme as we have described, we also recognize a series of indeterminate word-classes. These, together with the basic terms indicated above, make the syntax classification itself a lattice. This not only means that it can be treated in the same way as the semantic material (and ultimately, we hope, simultaneously with it), but also makes it possible to calculate, in a simple manner, the function or range of functions possible to any word group, when we know those of the separate words in it. The necessary operations are of the same kind as those used for manipulating the semantic thesaurus.

The procedure in applying this method is as follows. First, we use the word-function indications present in our dictionary readings for each word to work out the structure of each sentence. The rules for doing this are quite simple, provided that not too many ambiguous functions occur; when they do, it is necessary to eke out the general rules with rules based on word order and word combinations which are peculiar to each language. There is no reason to doubt that this will always be possible, but we don't yet know whether the procedure may not be too slow to be useful in some languages. The structure that we work out at this stage can be conveniently represented as a system of brackets. For instance, the English sentence "I thought you had seen me" has the bracket structure (I (thought (you ((had seen) me)))), or more briefly (A(B(C((DE)F)))). In practice we probably don't need the whole of the bracketing pattern; we certainly need to know at least the limits of clauses, because these provide, in most cases at least, limits of relevance of particular contexts, and are therefore needed to provide boundary conditions for the thesaurus operations. Once a structural group has been identified, the next step is to apply the thesaurus procedure to the words within the group, and thus get out specifications for their translation in terms of list of heads to which each word in the output text ought to belong. Meanwhile we also work out, from the function indications attached to each word on the group, what the function indication of the whole group should be, and (at the same time) provide a collective specification, being the list of all the heads that any of the words occur in, which we may be able to use to translate the group as a whole. This makes it possible to transform groups of words in the input text into single words (or at least single dictionary units) in the output, which in turn makes possible the translation of idiomatic phrases, and may have wider implications also. We are not yet able to say, however, how much of this method can be claimed as truly general.

The Use of Punched-Card Devices

The fundamental operation in using a thesaurus method of translation (irrespective of the detailed structure of the thesaurus) is that of selecting the heads common to two word-symbols (or symbols constructed by previous operation from the input text). This is an elementary Boolean operation which can be realized very readily by means of punched cards. If each head is represented by a place on a card, and if presence of a word in the head (represented by a 1 in binary notation), is represented by punching a hole in that place on the card representing the given word, then this fundamental operation is effected by merely superposing the two cards concerned, and treating the resultant pattern of holes as if it were punched on a single card. The actual handling of the cards is of course a time-consuming operation; but we shall nevertheless proceed with the exposition as if it were all to be done by hand, because this makes the procedures easier to visualize and to follow in the mind. In real life, various devices can be employed to speed up the operations, but these need not be considered here.

It must be emphasized that this is to use the punched-card equipment as a calculating device, even though the calculation is of a mathematically trivial character. It is a different kind of application of the apparatus from that in which punched cards are used as clerical aids, for example in actually compiling the dictionaries which we shall need. For the purpose of clerical aids punched-card methods will always have a place in machine translation; but as calculating devices their future is less clear.

We shall now turn to consider in more detail the sort of translating procedure we have been working on.

Our punched-card program "presupposes an unlimited time and an unlimited space."[4] This is mainly due to the inadequacy of the machines in existence to deal with the types of coding on the punched cards. The four types of machine needed for the program, as it is at present, are a hand punch, a duplicating punch, specially adapted to make "meets" and "joins," a sorter and a collator. (A "meet" is a card with all the holes which two or more cards have in common, and a "join" is one with all the holes two or more cards have on them.) It is thought that the soundest nucleus for the automatic project on punched cards would be

a twin-feed card comparator, and reproducing punch, together with a collator, rather than a sorter, for marshalling and selection.

A brief outline of the punched-card program with the machines used at various stages follows.

In the program as it: stands now there are three permanent dictionaries needed for the thesaurus method of translation. Two of these dictionaries are in the form of punched cards, while the other is in the form of an ordinary dictionary, i.e. alphabetically ordered and in book form. These three dictionaries are called (1) the "chunking reference dictionary," (2) the "input language dictionary" and (3) the output language "fan" dictionary.

The chunking reference dictionary consists of the words of the input language alphabetically, in the manner in which they are split up and the number of possible meanings they may have. The words of the input language are usually divided into stems and endings. The stems are further split when the chunks thus obtained have some semantic meaning. Each chunk in input language may have various meanings; these must be noted to enable the correct number of input cards to be generated. Then all the various possibilities will be drawn from the dictionary. This chunking of the input language will make the input language dictionary smaller, when it is completed, than it would be if it were to consist of input language words.

The input language dictionary consists of cards, one for each chunk in the input language. These cards have the following information on them: *semantic* heads, *syntactic* classification, supplementary information peculiar to the given language; and the *coded spelling* of the word. The semantic heads are taken from a special "compacted"' thesaurus, since Roget has 1000 heads whereas we have room for only 780. This semantic information takes up columns 1-78 on rows 0-9 of the card. The syntactic information is coded on the card on the second row down from the top of the card. The monolingual information on the chunk dictionary cards is the information pertaining to the declension, conjugation, gender and singular and plural.

The fan output dictionary has on each card the set of output words which belong to the heads punched on the card. These entries cover the total fan of uses that the output word can have in the target language. This information is coded in the same manner as the semantic information on the input language dictionary cards.

The input text is chunked clerically with the aid of the chunking dictionary. The chunks of the text are then numbered by reference to the paragraph, sentence and word. This numbering is called the text position indicator coded in 21 bits as a binary numeral. The chunks are also numbered in the sequence in which they occur in the text. The coded spelling for each chunk is then worked out. This spelling is in the form of a compressed binary coding of 20 bits. This coding is punched in columns 79 and 80 in rows 0-9. The above information is then punched onto a pack of cards, called the input pack; it will be in the order of the input text.

To enable the relevant cards to be drawn from the input language dictionary, the input pack must be ordered alphabetically. The ordering of the pack is carried out by a series of sorts operating on the coded spelling. Then by means of a single collation on the coded spelling, with the input cards in one of the collator feeds and the dictionary in the other feed, all the cards relevant to the input cards may be drawn out.

The dictionary cards are copied by the reproducing punch and the original dictionary cards returned to the dictionary. The dictionary cards and their equivalent input cards are then joined to form a pack of cards called the second dictionary pack; this is then restored to the original text order.

This second dictionary pack contains all the information needed to carry out a thesaurus translation procedure.

Some of the input chunk cards will have drawn out more than one card for some of the input text chunks, from the dictionary. The next step in the procedure is to remove the cards which do not apply to the piece of discourse under consideration. This is called the "pun-removal" procedure. It is intended to deal mainly with ambiguous endings. Intersections are made on the monolingual information on the cards. These intersections are intraword intersections, and will remove nearly all the puns. In some cases this procedure will not suffice; e.g. in Latin it cannot distinguish between the ablative and the nominative uses of the ending —*a*. Therefore when some chunks are still ambiguous, intraclause intersections must be made on the semantic entries on the chunk cards. The card of whose holes most survive these intersections is retained, and the others abandoned. If more than one card still remains for any chunk these cards are all abandoned, as it is better to lose too much information than to retain incorrect information.

The input text is then sorted into the clauses which make up the input text. Then a series of parallel operations are carried out on each of these clause packs.

The first of these operations is to find the frequency of the semantic and the syntactic heads in these clauses. This is carried out by a series of meets and joins. This operation is carried out on the adapted reproducing punch. When the heads that have occurred more than a certain number of times have been obtained for each clause, then a card for each clause is punched with the most frequent head holes on them. These cards are called the head grid cards.

The next stage in the procedure is to form word cards from the chunk cards. This is simply done by making meet cards for the chunks of each word. Then with the head grid card a further pack is generated, which is the meet of this card and each of the word cards.

The fan output dictionary is then sorted for each word card. This consists of sorting the fan dictionary *n* times, where *n* is the number of holes in the word card. The card chosen for each word card will be that card which has the nearest pattern of holes in it to the word card. This card will give the translation of the input word. These words are written on the fan cards chosen.

The sorting in the above procedure is the time-consuming process, and this could be shortened considerably if face-reading machines were available instead of the column-reading machines as they are at the present time. An example of this time factor is that the sorting of the second dictionary pack back into the text order would take a maximum 27,756 sorts.

Each stage in this procedure may be tested separately and perfected without running the complete procedure through the machines. The final version of the punched card procedure may easily be transferred on to a digital computer, as the complete set of punched-card machines constitutes such a computer.

Computer Programs

It will be realized that this procedure, carried out manually with actual cards for an actual text, is very laborious and slow. It is not intended that a practicable translation method be in this form. Its value is however twofold: on the one hand its practicality would be much enhanced by relatively minor advances in punched-card technology, and on the other hand by its very slowness and step-by-step quality it serves admirably as a preprogramming method for digital computers or any other calculating devices we may want to use. Naturally, when one comes to write actual digital computer programs for achieving the same results as are achieved by the punched-card program just described, there appears to be a radical reshaping of the whole procedure because the "housekeeping" operations required by the two methods are totally unlike. In the punched-card procedure these consist mainly in tedious copying operations which have no counterpart at all on a digital computer which, however, spends much of its time in counting items and recording and reading the resulting serial numbers.

A main object of our research has been to find out, and this is largely an empirical matter, what sort of variations in the procedure are the most profitable to

make, and what parts of it are most often the seat of failure. The punched-card method is especially valuable in that it makes easy the necessary breakdown of the procedure into portions which can in this way be separately examined. Digital computer programs are however a much more convenient way of testing completed procedures, because they make possible runs on realistically long passages. Moreover, they go through fast enough to be within sight of commercial targets of speed, so that we are justified in claiming that any procedure which stands up to such testing on a computer would be a reasonable basis from which to advance to a fully salable translation technique.

The one great barrier to straightforward conversion of the punched-card procedure we have described to a form acceptable on a digital computer is the fact that the cards assume, in effect, a word-length of up to 860 bits, which is far beyond the capacity of any existing computer to handle otherwise than by slow multilength methods. If it could not be overcome, this fact would shut us off from the desirable prospects of really fast procedures suggested above. Fortunately we have good hopes that this particular difficulty can be satisfactorily surmounted.

The manner in which we propose to do this is to use a much more economical encoding of the information than the simple procedure of assigning one bit in each computer word to one place on the card. That such economy is possible, without confounding the information (which for mechanical translation purposes is at best marginally admissible) depends on the statistical properties of the lattice by which the thesaurus is represented.[5] By means of a sampling technique applied to *Roget's Thesaurus,* we have found that the information contained in the index of the standard edition of that work can be represented by a lattice whose degree is about 37; that is to say, that this lattice can be contained in the Boolean lattice with 2^{37} elements, and its elements therefore represented by symbols of 37 bits each. A thesaurus made adequate for translation purposes by the addition of extra entries and special heads, as for instance to represent syntactic relations, would of course be larger, though its degree would be increased relatively slightly in comparison with the number of elements which would have to be added. Nevertheless, we judge it to be unlikely that one could not encode a workable thesaurus in 60 bits, and virtually certain that 100 bits would more than suffice for any foreseeable thesaurus to be used for mechanical translation. These figures, though somewhat in excess of what current commercial machines provide for, are not absurd as a target for the future (which 1000-bit words may be). Moreover, we can test all our programs, at the sacrifice of only a part of their quality, on existing machines providing for handling 40-bit words.

The actual procedure for encoding the thesaurus is somewhat elaborate, and is the subject of a paper soon to be published.[6] The operation is one requiring a computer, but being a thing done once for all one can in principle afford to give it an hour or so of machine time, which is what it is likely to require. The output of the procedure is to assign to each head of the thesaurus, in place of the serial number which originally identifies it, a 40-bit symbol which will represent it in the encoded form. The code sign for any word in the input or output language can then be formed from the corresponding punched card by taking the head symbols corresponding to every hole in the card and forming their Boolean join. Join and meet operations on the resulting symbols will give results the same as would be got from the same operations on the untreated 860-bit symbols; in this sense there is no confounding. It must be emphasized however that any new entry added to the thesaurus, and thus also to the input or output dictionaries, after the encoding has been completed, will introduce confounding, and on such a scale that the new entry will in fact be unusable. The coded thesaurus must therefore be regarded as a closed system, amendable only by going through the encoding routine all over again.

Another point in the procedure where a good deal of research has been required to obtain efficient programming on digital machines, is at the output stage, represented by the fan dictionary in the punched-card procedure. In a machine handling actual cards there are various methods of sorting which will quite quickly find the card or cards having a particular pattern of holes on them, and thus serve as sorting procedures on the output dictionary. This is especially easy if edge-punching can be used. But to find, in the long-term store of a digital computer, a given entry identified only by certain features of its content is liable to be very slow (at any rate on the time-scale to which programmers are accustomed). We have therefore given some thought to problems of programming this type of operation. The problem turns out to be again essentially one of coding. It appears that the minimum search time required depends very markedly on the shape of the lattice representing the thesaurus, and by good fortune the optimum shape turns out to be one in which the height of the lattice (length of longest chain) is small compared to the number of elements; this is in fact the shape that the thesaurus lattice happens to be. As a result of this, the time required for the output stage of the completed program is likely to be, if anything, less than that required for the initial dictionary look-up. We have not yet however tested this part of the procedure in actual machine runs.

We are still only just beginning our program, of testing variations *on* the basic procedure on digital computers, so we cannot yet say what the results will be. There is every prospect, however, that mechanical translation by the thesaurus method will be successfully accomplished. We can also say that, with certain qualifications, it should be feasible, using only machines of existing type. The main qualification is that existing memory devices are either too small or too slow to make a fully mechanized translation procedure an immediate commercial prospect. However, advances in this field are being made so rapidly that this reservation may at any moment cease to be valid.

A word of caution may however not be out of place in conclusion. Even though in principle mechanical translation using machines existing or soon-to-exist may be possible, it has never yet been actually realized. The demonstrations which are given from time to time have hitherto been essentially demonstrations of mechanical dictionary searching, helped out by simple reordering techniques. Results achieved on selected texts cannot be reliably reproduced on unselected (i.e. randomly chosen) texts in the chosen language, even within the same subject or style. True machine translation will not have been achieved till a readable and understandable translation of a genuinely unselected and unpre-edited text has been produced. And before this is done much theoretical and practical work remains to be accomplished.

References

1. M. Masterman and R. M. Needham, "The analogy between mechanical translation and library retrieval," Cambridge Language Research Unit.
2. *Roget's Thesaurus,* Longmans, Green & Co., London, 1936.
3. M. Masterman, A. F. Parker-Rhodes, K. I. B. S. Jones, R. M. Blackmore and C. Wordley, "Description of tests carried out on methods of extracting sentence lattices," Cambridge Language Research Unit, 1958.
4. J. M. Staniforth, "General schema of a thesauric translation program, using punched card techniques," Cambridge Language Research Unit, 1958.
5. A. F. Parker-Rhodes and R. M. Needliam, "Encoding Roget's Thesaurus," Cambridge Language Research Unit, 1958.
6. A. F. Parker-Rhodes and R. M. Needham, "Computation methods in lattice algebra (in press).

Discussion

Max Kosarin (Army Pictorial Center): What language or languages have you been experimenting with?

Mr. Parker-Rhodes: We plan to be able to cope with any of the major languages of the world. We started with Chinese as the simplest in structure; from that we went over to the opposite extreme and experimented with Latin (we would have preferred Russian but we haven't enough people available to do that -- that being one of the vagaries of the British educational system). We have also done a bit on Italian, and we have being prepared for us an Italian dictionary on which we propose to test a modified form of this procedure right now -- it should be starting this year.