Maxwell H. McKay

# THE ROLE OF THE COMPUTER IN LINGUISTIC RESEARCH

## The nature of a computer

There are many misconceptions abroad about computers and what they can do. Their designation as "electronic brains" has engendered in some people the belief that they are to be feared as some sinister power which may at any stage break away from man's control and go berserk or subjugate man to its devilish will.

The facts are that computers are not capable of doing things they are not told to do, but, on the contrary, have the regular habit of doing precisely what they are instructed to do. This may be quite annoying at times, since they give strict obedience to incorrect instructions as well as to correct ones right down to the last detail. Inconsistent or impossible instructions will cause an immediate reaction in the computer, but any instruction which is possible will be carried out in detail, whether or not it fulfils the intention of the programmer.

It is true that a computer may be programmed to make decisions, but only on the basis of carefully compiled criteria which must take into account every possible eventuality. A programmer may instruct the computer to organize a problem into steps and to use the results of one step to execute the next, but he must foresee all the possible outcomes of the first step in order to ensure that the succeeding steps will be meaningful.

Basically a computer's capability is limited to addition, subtraction, multiplication, division, comparison and rearrangement of data, and any combinations of these. However, it operates at a prodigious speed, performing hundreds of thousands of these operations in a second. It can also store an enormous amount of information in what is called its "memory", and any individual item of information stored in this way can be accessed in a minute fraction of a second. The speed of operation and storage capacity of electronic computers enable them both to perform a vast number of menial and monotonous tasks in a very short time and also to cope with extremely complicated procedures simply by breaking them into their basic steps and dealing with these basic steps at a fantastic pace, if necessary storing or "remembering" partial results while intermediate steps are being carried through.

Although the basic operations of which a computer is capable are normally thought of in terms of numbers, its operation is not restricted to numeric data. The letters of the alphabet and up to about a dozen special characters, including some punctuation marks, are coded for insertion into the machine in much the same way as numbers are, and the computer can be programmed to compare and rearrange these just as readily as can be done with numbers. Thus we find that many of the processes involved in linguistic analysis may be handled on a computer.

## Application to linguistics

Generally speaking, an electronic computer is capable of carrying out any process which may be specified in terms of a number of well-defined steps in which all decisions may be made on the basis of precise criteria. The computer can use the experience it gains as it processes the data if it is programmed to take note of the relevant facts and modify its modus operandi according to the nature of the facts. Otherwise it depends completely on the knowledge, experience and foresight of the programmer and has no intuition or "Sprachgefühl" of its own.

The most elementary type of operation which can be carried out of linguistic data is that of counting the number of occurrences of various letters or symbols in a piece of text. All that is involved in this is for a list of all the letters and symbols and then the text to be punched on cards or tape and a simple programme can be written which instructs the machine first to read the list and then to read the text one letter or symbol at a time, comparing each character with each of the characters on the list in turn until it finds a match. It then adds one to the counter for that particular character and proceeds to the next character of the text.

The above procedure will automatically give the number of sentences (specified by stops), the number of phrases or clauses defined by commas or stops, the number of words (defined by blanks) and the number of morphemes (defined by hyphens or blanks). If the text contains strings of successive blanks at the end of a line or paragraph, the machine must be programmed to treat these as one.

A slightly more complex programme is needed to count the number of occurrences of various digraphs, trigraphs and syllable types and then an additional complication is introduced if one wants a count of the number of occurrences of each word, morpheme or syllable in a given text. This last process requires all the words, morphemes or syllables to be stored as they are read and then, when all have been read they must be rearranged into alphabetical order. Comparison of each item with the one which precedes it will then enable the count to be made. A listing of all the words, once they have been rearranged into alphabetical order could form the basis for a dictionary to be prepared.

If a concordance is required, several more complications are introduced. It is possible to compile concordances of words, morphemes, syllables, phrases or any other combination, provided each is defined in the text by blanks, hyphens or other specified punctuation mark. The procedure in each case is essentially the same, so that the following outline for a word-concordance indicates the pattern which would be followed for the others. First each word has to be assigned a context before it is stored away and then after the alphabetizing process some means has to be found of reproducing the required amount of text before and after the given word. Another thing which needs to be taken into account is the order in which the various entries involving the same word should be arranged. Some details of a programme to compile a concordance are given in the appendix.

The procedures mentioned above are of value in analysing the grammar of a language and in the preparation of reading primers, in which it is important to know the comparative frequency of occurrence of various letters, symbols, digraphs, morphemes, words and combinations of words. Other uses to which a computer may be put in this connection are in checking a hypothetical rule of grammar by scanning a text for exceptions, examining the ordering of affixes, listing and counting part-of-speech sequences and tagmeme sequences, syntactic and phonemic analysis, classification of verbs, and reversal of a dictionary from one language to another. In the case of some of these, it may not be possible to specify all the necessary criteria to ensure a perfect analysis and some post-editing may be needed.

Comparisons between languages may be assisted readily with the aid of a computer. For example a simple programme was recently written to determine the percentage of cognates in all possible pairs of some 22 languages and dialects from the Sepik Hill district, taking 120 word-meanings into consideration. A table had previously been compiled in this case in which the presence or absence of cognates was indicated by means of numbers and all the computer had to do was to compare numbers. However the 3,000 odd comparisons which had to be made and the relevant calculations were completed in a few minutes by the computer, saving the linguist several days of tedious work.

The linguist may go further and ask the computer actually to identify probable cognates in groups of languages, but in doing this it must be remembered that precise criteria have to be laid down beforehand and the computer is not capable of using its intuition. The resulting decisions could obviously not be so refined as those made by a skilled linguist weighing up all the evidence in the individual cases and learning continually as he proceeds from word to word, leading to a change of mind in many cases. Of course, the linguist can enlist aid from the computer in first instance and then proceed to examine the results with a view to revising some of the decisions made on inadequate or unsatisfactory criteria, but it is difficult to hand the whole job over to a computer.

Finally, a few words about machine translation of languages. Much has been claimed about the potential of computers for the translation of languages and many man-hours and machine-hours have been spent in seeking to achieve the goal of fully automatic high quality translation. However, there are many difficulties in the way.

Let us first examine the most elementary of objectives, namely to achieve a word for word translation from one language to another. Such a translation would be very rough in the target language and would sometimes fail to convey any meaning at all without some reference to the original language by a reader who is familiar with it. Thus a certain amount of post-editing by a skilled linguist is required to complete the translation. Whether or not the linguist's time has been significantly saved by the rough machine translation depends on the passage concerned and, of course, on the skill of the linguist.

But it should be mentioned that a word for word translation by machine would have to list the various possible meanings of many words as it would not be capable of selecting the appropriate one from the context. For example, the word "pen" may be used with two meanings in English as indicated in the sentences:

1. "The pen is in the box".
2. "The box is in the pen".

In the first case "pen" is a writing instrument, while in the second an enclosure is meant. The reader's knowledge of the implications of the two meanings must be used to distinguish between the two, and the context may help in this, but there is no conceivable way of teaching a computer to distinguish between the two. There are many similar instances in English, and some other languages are much worse.

In order to achieve better than a word for word translation, a considerable amount of syntactic analysis of the source sentences is required in order to eliminate syntactical ambiguities. This would leave the post-editor only the task of elimindting the semantical ambiguities and of polishing up the style of the machine output. Much effort has been put into the problem of mechanical syntactic analysis of several different languages, but there are still many obstacles to be overcome in this direction. It is the removal of some of these obstacles which is the aim of most workers in machine translation, recognising that a perfect translation which needs no post-editing, is in general unattainable but seeking to minimize the amount of post-editing needed.

# APPENDIX

The Production of a Concordance by Computer

## Preliminaries

To give an idea of the sort of approach required to programme a linguistic process for a computer, a description is given of the significant features of a morpheme concordance programme which has been written for an IBM 1130 computer. It should be noted that this machine is rather small for this sort of programme, so that supplementary disk storage has to be used and a great deal of care has to be exercised to pack the maximum amount of information into the disk in order to use the computer to its fullest extent.

The actual size of text which can be processed depends on three main factors - the number of lines in the text, the average number of morphemes per line, and the number of characters considered desirable for alphabetization. The programmes which are described in this appendix have used for alphabetization purposes a string of twenty characters (including blanks) starting with the first letter of the relevant morpheme. This allows a text of up to about 1400 lines with about 17 morphemes per line to be processed. If there are fewer morphemes per line, or if only a word concordance is required, it would be possible to handle more lines. On the other hand, the number of lines would have to be reduced if there were more than about 17 morphemes per line. The alphabetization can be done in a straightforward way, listing all the occurrences of a particular morpheme in alphabetical order with respect to what follows the morpheme, or the order can be varied in different ways. For example, if desired one could write the programme in such a way that for a given morpheme which occurred in various positions in a word, a list would first be given of all the occurrences in which it stood alone as a separate word, then all the occurrences in which the morpheme was at the end of a word, then those in which it was at the beginning and finally those in which it was in an intermediate position. One could ask the computer to alphabetize forwards until a stop or a comma was encountered, and then to continue the alphabetization backwards from the beginning of the morpheme. These last two arrangements would require a little more complexity in the programme than the first straightforward procedure.

The complexity of the programme would also be affected to some extent by the format in which the text is punched onto cards or tape. The simplest programme would be one for which the test was punched continuously 75 characters to the line with just one blank after each word and words broken at the end of the line. However this is rather an unnatural way of presenting text and makes checking a bit difficult. The programmes to which this appendix refers were written to accept text punched in much the same way as it would appear in a typescript or a printed page, provided only that the morphemes were marked by means of hyphens and the line length did not exceed 75 characters. No hyphens should be used except to mark morphemes. At the end of each line (in positions 76 to 80) a five-figure number (not greater than 32765) is inserted to identify the line.

In numbering the lines, consecutive numbers should be used, except at the end of a section of the text or perhaps at the end of a paragraph, where the following section or paragraph has no significance as far as the usage of the words at the end of the given section or paragraph are concerned. The last line of the text should be followed by a line of 75 blanks followed by the number 32767, the largest integer which can be used in the computer. This is the signal for the computer to conclude the concordance.

## Reading and Processing the Text

The first instruction given to the computer is to "read" the characters which may occur in the text. This list must contain a blank, a hyphen, punctuation marks and other special characters as well as the alphabet and the numbers 0 to 9. The machine must have these stored in order to be able to "recognise" them when they crop up. This "recognition" can only be done by the computer by comparing each symbol "read" with each of the symbols on the list in turn until it finds a match. The comparison is made by subtracting machine code numbers and a zero answer indicates a match.

Next the computer is instructed to "read" the title and then the first two lines of the text. The title is printed out immediately, and the computer is instructed to examine the text in detail. At this stage, it is a simple matter to ask the computer to count the number of occurrences of each symbol and, after reading the whole text, to work out the percentage frequency of each letter of the alphabet. A count of the blanks will give the number of words and the number of hyphens added to this will give the number of morphemes. Of course, the computer has to be instructed to ignore surplus blanks at the end of a line and also to insert a blank at the end of the line if none has been left.

Each time the computer starts a new line or reads a blank or a hyphen, it records the next letter as the first letter of a morpheme and registers its context in the form of line number plus position in line. It then records the twenty characters which are to be used for alphabetization purposes and sends this record, with its context, to the disk to be stored, having checked the morpheme count and line count to ensure that the capacity of the disk is not being exceeded. In recording these twenty characters, the computer is instructed, if it comes to the end of a line, to ignore surplus blanks and insert a blank if necessary before going to the next line. It also checks that the next line and the line being considered have consecutive numbers. If this is not so, it fills up the rest of the twenty characters with blanks.

Having performed the above operation for every morpheme in the first line, the computer sends this line as it stands, with its number, to the disk, again checking that there is enough room for another line to be considered. Then another line is read, and the line which is already in the working core of the machine is processed in the same way as the previous one has been.

## Alphabetizing

When all the lines have been treated, the computer has in the disk two files of records. One of these contains one record of twenty characters for each morpheme in the text, and the other consists of the lines of the text in order.

At this stage, if it is required, a listing could be made of the frequencies of each symbol, the total number of symbols, words, morphemes and lines and the percentage frequency with which each letter occurred. For purposes of percentage frequencies, blanks, hyphens and other non-alphabetic symbols are excluded.

The next procedure is to sort the morpheme file into alphabetical order. As the working core will only hold a small fraction of the information stored in the disk, one has to bring the morpheme records back into the core one batch at a time, sort these and then return them to the disk. When this has been completed, the various batches have to be merged in such a way as to give the complete file in alphabetical order. This sorting is done by comparing each pair of records with one another, character by character, and re-ordering the pair if the comparison indicates that the second should precede the first. The alphabetical order will take account of blanks, hyphens and other non-alphabetic characters by allocating them a specific order before A for blanks, punctuation and special characters, and after Z for numbers, if any have been used.

## Printing the concordance

After the sorting process has been completed, the disk still contains two files, but now the morpheme file is alphabetized exactly as required. It remains now to print the concordance, and in order to do this, reference will have to be made to the text file in order to insert each morpheme into its context.

As the 1130 computer is capable of printing a line 120 characters long, one may take advantage of this and print out a context 110 characters long with the relevant morpheme starting in the centre, and then, at the end of each entry, the reference to the line number and position within the line.

In order to do this, the computer is instructed to access the entries in the alphabetized morpheme file one at a time in order. The context is noted, and the relevant line of the text is called from the other file. If the morpheme in question commenced to the left of the 56th position in its line, the previous line would also need to be called so as to obtain the full number of 55 characters required to be printed before the given morpheme. A check has to be made as to whether the previous line and the given line have consecutive numbers to make sure that no irrelevant material from a different text or paragraph is included.

If the numbers are not consecutive, sufficient blanks are inserted at the beginning to make up the required number of characters. On the other hand, if the numbers are consecutive, an adjustment has to be made to ensure that just one blank is inserted between the last symbol on the previous line and the first symbol on the context line.

If the relevant morpheme commenced to the right of the 54th position counting backwards from the last non-blank character in the line, the next line would have to be called from the file. Again, a check would have to be made that two lines are consecutively numbered and if not, the appropriate number of blanks must be added. If the lines are consecutively numbered, just one blank is inserted at the end of the context line, and then the appropriate number of characters from the next line are added.

Having sorted out a string of 110 characters, with the morpheme in question commencing in the 56th position, the context, consisting of line number and position in the line, is added at the end. This forms a complete entry for the concordance, but, before it is printed, a check is made as to whether the relevant morpheme is the same as that listed in the previous entry. If they are the same, one is added to the number which is keeping account of each occurrence of each individual morpheme, and the entry is printed. If the morphemes are different, the number of occurrences of the previous one is printed out, the counter is re-set to zero, and then the entry is printed.

## Concluding Remarks

It will be seen from the above that programming a computer to compile a concordance is quite a complicated procedure, especially when it is realized that some of the steps described in a few words actually take up.to about a dozen computer instructions to carry out. In fact, the entire programme involves about 400 instructions and would take a good programmer several days to write. However, once written it can be used for any text which is prepared in the appropriate format and produces in a matter of a few hours of computer time, a concordance which would take a linguist many months to compile.