Magic Spell

Spelling checker programs are available for many word processors, and style and grammar checkers are also starting to appear

Computer designers are still a long way off creating machines with the ability to generate and manipulate natural languages, such as English. One of the intended applications for the fifth generation of computers, which should appear in the 1990's, is machine translation between, say, English and Japanese. Machine translation facilities already exist for relatively simple prose such as government reports and proceedings, though the draft produced by the mainframe computer invariably has to be corrected and polished by hand. Stories of errors abound: the quotation 'The spirit is willing but the flesh is weak' is said to have been translated from English to Russian and back again by two different programs, with the final result of 'The wine is agreeable, but the meat is spoiled'!

Such apocryphal stories illustrate a very important point — the difficulties encountered when a computer is processing data without understanding what it means. A problem often posed to students of computer science is to consider how a computer could distinguish between the meanings of the following two sentences:

TIME FLIES LIKE AN ARROW
FRUIT FLIES LIKE A BANANA

The construction of both sentences appears identical, but in the first instance FLIES is a verb, whilst in the second it forms part of a noun phrase. The only reason why we can tell them apart is through experience. It is possible to simulate experience on a computer, given enough memory, but this comes under the field of artificial intelligence, and research in this area is not very advanced. What we are really talking about here is the difference between 'syntax' and 'semantics'. Syntax, meaning the rules concerning the construction processes used in a language, is a fairly easy subject for computers to get to grips with (as all home programmers who have encountered SYNTAX ERROR? messages know). Semantics, however, concerns the meaning which those phrases and constructs convey.

In the 1950's, Noam Chomsky developed the basis for contemporary theory about human languages and the rules of grammar, and although he was not directly involved with the computing sciences, his theories are directly pertinent, both to machine translation and to the writing of interpreters and compilers for programming languages.

One of the by-products of his research has been the creation of various software tools to assist in the writing of text. In addition to word processing packages, which assist in the creation, editing and printing of text, there are programs to proof-read documents for spelling and typing mistakes, and even to check on the grammar and style of writing. Though none of the contemporary products contain anything outstanding in the way of artificial intelligence, it is instructive to look at their operation — both in terms of the way they are presented to the user and how they are internally programmed.

All spelling checker programs make use of a dictionary held on disk, which typically stores between 25,000 and 50,000 words. If you intend buying such a product, incidentally, check that the dictionary has been compiled for English use — many originate in the USA and use American spelling. Most packages will allow you to add items to the dictionary, such as unusual jargon terms that you may use, or the names of companies and products that you wish to have checked.

A problem arises, however, in finding adequate memory space for a full dictionary. You will remember that one eight-bit byte can hold a single alphanumeric character using the ASCII code. So, even allowing an optimistic average of just five characters per word, a 30,000 word dictionary would require 150 Kbytes of storage, which is very much larger than most single disk drives for home computers. Fortunately, this kind of data can be quite easily compressed, by using two techniques.

First, if we assume that our dictionary need only contain lower case letters (a routine in the spelling program will handle the conversions), and numeric digits and some punctuation symbols will not be needed, then these can be removed by the program. Subsequently, we could construct our entire dictionary using a maximum of 32 different characters, instead of the full ASCII range of 128 (or 256 if you include graphics symbols). We could therefore reduce the storage requirement of each character from eight to five bits. The word 'computer', for example, could be stored in a total of 40 bits, or five bytes. The first five bits of the first byte would specify the letter 'c', and the next three bits, plus the first two of the second byte, would specify 'o', and so on.

The second technique employed within spelling checkers is called 'tokenising'. This works on the premise that certain combinations of characters appear so frequently that they could be represented as, perhaps, a single byte. This would be 'flagged' in some way to indicate that it was a token for a group of characters and not a single character. Your home computer almost certainly uses tokenising in BASIC — each keyword, like PRINT or NEXT, is stored in RAM as a single byte to save space.