[From: A. Kent & O.E.Taulbee (eds.) *Electronic Information handling*. Washington, DC: Spartan Books, 1965.]

Mechanical Resolution of Linguistic Problems

A. D. BOOTH

Dean of Engineering, University of Saskatchewan
and Profess or at Large, Autonetics, Western Reserve University

The optimum information retrieval system is one which I should like to call a symbiosis of man and machine. Men do some things very well that machines do very badly. One should not use machines for such purposes. So, if you expect a champion for the machine, you won't find him here. I ought to say that in the University of Saskatchewan and occasionally in the University of London I lecture on the use of computing machines on numerical analysis. I always preface my remarks by the statement that: "Machines are the last refuge of the inept," which ought to put them into perspective.

On the other hand, having bowed to Dr. Perlis on that subject, I should dispute him when he says that no progress has been made in machine translation. This, as a matter of fact, is quite untrue. Depending on the level at which you want to consider the translations, some progress has been made. There are quite decent programs for translating English into Russian. I suspect there are some programs in the United States for scientific translation of Russian into English, and there are certainly some programs, because I was concerned with part of the writing of them myself, for the translation of French into English. These work and, if you wanted to look at the output of a machine doing this sort of work, it would be rather doubtful whether you could distinguish the output from that produced by a human being. However, I suspect that Dr. Perlis' remarks were in the nature of being provocative and not supposed to be a statement of fact.

By way of an introductory remark, I want to tell a story. It has been remarked of academics that they are good for two hours of speechifying, although somebody else remarked in the same context, "That's what *they* think." I'll try not to take two hours, but anyway, let me tell you a little story. A few years ago I was invited to read a paper at a conference that was held in a place called Alpbach in the Tyrol. This conference had some highbrow title like "Language, the World, and its Philosophy." I looked at this with horror, but it provided me with a means of getting a free holiday to a rather nice place. I said I'd go. When I got there I was com-

pletely horrified. There was a collection of very long-haired professors, obviously of enormous erudition and of a mental caliber I couldn't compete with, and I was set down to open the proceedings. Of course I noticed this beforehand and had come prepared with a text constructed by one of our computers on the subject: "Cybernetics and the World." I had programmed the computer to do the sort of thing that Shannon did originally: produce a text by taking a word from random from some page in a book on the subject of cybernetics, then finding some other page on which the same word occurred and taking the next word on that page, then going to some other page selected at random, and so on. This way I constructed twelve minutes of fairly plausible text. At the meeting, I noticed the simultaneous translators making a fine go of this and they were nodding and the audience was sitting in the front row looking intelligent and saying "Mmm, mmm, very profound." At the end of this performance, I took the parliamentary utterances of various Ministers in the British Parliament for successive days of one week and took the second sentence of each pronouncement, irrespective of the Minister. And I finished up with this. It read very well and was a really high-powered speech. Then I turned to the president of this meeting and I said, "I am sure, sir, that you will appreciate the profundity of those remarks." I am afraid that this was a bit unfair because he turned to me, and in a very audible voice said, "Yes, that was a very fine account of the subject." At this point, of course, I did the sort of thing that all comics do—I turned to the audience and said, "Well, gentlemen, you will be interested to know that there was no meaning whatever in that twelve minutes of discourse." The front row of the audience rose and left like a black cloud; the remainder of the audience were rather young people, and when we came to get our groupings of young men for the classes which we were giving later on, I am delighted to say I got about 95 percent of them. The graybeards, I'm afraid, didn't get to first base.

Well, now to come to something more serious. I think I have entertained you for five minutes; let me now deal with the subject of mechanized linguistics.

I'm going to try to give you a view of the structure of this operation because there are some important things in it, whether Dr. Perlis' remarks have much justice or not. There are some important things we can do; there are some important ideas in this field, and it's worth describing them. You'll see that at many points I make contact with some of the remarks of Dr. Perlis on things like structure. First of all, a remark about the machines themselves. I am not one of those people who believe in building gadgets. You may almost paraphrase Wittgenstein and say that whatever can be done can be programmed on a computer. Therefore,

you shouldn't build a special machine. You ought to be quite sure what you want to do before you build a machine. The structure of computing machines as they exist at the moment really divides itself into two depending largely on the type of storage involved. This is rather important because whatever the future of computers is going to be, and this isn't by any means certain in some of our minds, present computers are, in a sense, unfortunate because many computers have adequate amounts of storage to contemplate attacking problems of language, but have this storage arranged in what I might call a hierarchial structure. The computers have a very small amount of very high-speed store, a rather larger amount of medium-storage sometimes, and quite often a great deal of very low-speed storage. On the other hand, there are the ultraexpensive computers, which have all of their storage on immediate access media. Now the way that you think of language in connection with a computing machine depends very largely on the structure of the machine with which you are concerned.

Actually, right at the very beginning of processing any data, whether linguistic or otherwise, derived from a list, involves deciding whether the statistics of the data are of paramount importance or whether the importance is secondary. Let me quote an example that makes this point. If you have a machine which is operating such a simple thing as a dictionary or look-up procedure there are many ways of using this, from the very simplest (which Dr. Perlis mentioned) in which you address the item of information by the code word of the unknown word, if you like. If you want to look up *et* in the dictionary, you find the code number of *et* (e.g. $e \equiv 05$, $t \equiv 20$, so that $et \equiv 0520$) and in the storage position having that code number, you find the translation *and* or whatever the equivalent is in the language you are concerned with translating it into. This type of storage is completely unworkable for very good reasons concerned with the structure of language. For example, if you take words of less than or equal to ten letters in English, it turns out the number of possible words is slightly over 10^{14}. The number of actual words in English is about 10^6. To those of you who are not clued up on these big numbers, this means that if you wrote down these words in a list, on average there would be about 10^8 blank spaces between each entry in your list of words. It would not be a good idea to have a store unit in this sort of way. This is an elementary example.

Consider next the dichotomy of storage in present machines, the fact that you can have hierarchial storage or immediate-access storage. For hierarchial stores, it turns out that probably the best way of proceeding is to consider the statistics of your word list and then to sort the input text into some order before presenting it to the computing machine. On the

other hand, with random-access storage the best argument suggests that you needn't concern yourself with these statistics, you just go straight to the list and, if you have an appropriate look-up procedure, whether this is by a method which involves a treelike structure, of the sort you heard about a moment ago, or whether it involves a simple partitioning of the list doesn't matter too much. Both of these methods are workable and reasonably efficient. But you do have to know quite a bit about the machine you are going to have available in the future before you start committing yourself to large amounts of work in this particular field. This is, if you like, a preliminary word of warning.

While having said this about language statistics, or data statistics, what sort of pieces of information do you want? There exists one very general law that applies to language particularly (it was discovered, in fact, originally as applying to language) but also applies to almost any list of information one can write down in some structurable order. This law is known as Zipf's law. I don't know why it's called Zipf's law because, although Zipf ennunciated it in the 1930s and made a great stir, it was first enunciated by a Frenchman called Estoup about 1919. This Estoup law states that for ordinary language, and for a lot of other things as well (numbers of entries in telephone directories under each name, for example), if you arrange your list of entries in terms of their rank—that is, the most frequent entry has rank 1, the second most frequent, rank 2, and so on—and if for each entry in this list you put down the frequency of occurrence of this word, then rank times frequency is constant. It's a very important law for look-up procedure analysis, and for mathematicians, too. Because whatever one may think to the contrary, mathematicians have not been completely oblivious to the need of considering the effects of structure on function. One of the situations you can analyze is this. If you want to operate a dictionary, would it be a good thing to plan the dictionary so that the most frequent word in the language is the first entry in the dictionary, the next most frequent word the second entry, and so on? The problem is then to determine, for this ordering, whether or not looking up words in a frequency-ordered dictionary is better than looking in a dictionary in monotonic increasing order of word magnitude expressed as a code number. It turns out that the answer is that this dictionary is unworkable; that the normal dictionary is better used with binary partitioning. However, one of the things mathematicians got interested in was wondering if there were any laws of occurrence of data for which frequency-structured dictionaries would be better than any other variety. It turns out rather interestingly that if the Zipf-Estoup law wasn't (rank $\times$ frequency = const.) but instead (rank^n $\times$ frequency = const.), $n > 2$, then it is more efficient to use a frequency-order list than it is an ordinary

dictionary. This is one of the sorts of information that any respectable person working in the field of language data processing ought to consider for himself before he starts. It's certainly no good going blindly to a computer, mechanizing some wonderful idea derived from hot air, and then wondering why your system is inefficient. You should investigate these efficiencies before you start. This is the basis of the remark I made earlier that the numerical calculation on computing machines is the last refuge of the inept. You can do quite a lot without using a machine, some of it purely mathematical.

We have thus decided that we must consider our computing machine and the lists to be used. That leads to discussion of what I might call the mechanics of linguistic statistics. You notice that the title of my talk (which incidentally I more or less approved because I would have hated to have been put down as talking about machine translation, in which I frankly don't believe) is "Mechanical Resolution of Linguistic Problems." It starts with the mechanical resolution of problems of linguistic statistics. Here again one begins with the problem of how to get the data into the machine. As far as I can see from the program, you're going to hear a number of ways in which data can be presented to the machine. The classical way is to present it on punched cards, and the classical way may be the best, but I doubt it. In the first place, a decent punched-card producing machine with a typewriter input costs a great, deal of money, so generally you have to rent it. So for this reason, although the punched card is not a bad way of putting machines in, it certainly isn't a very economical way.

The second direct form of input is by a punched paper tape. This is very attractive because modern electric typewriters can produce tape as a by-product, so that the typist does your letters and at the same time produces a machinable record on punched paper tape. Tape is also very important in that many books are produced by the monotype process, and the monotype rolls used in producing books, can in principle, at least, be read into a computing machine.

Incidentally, on the subject of tape and cards, I might remark that of tape doesn't involve great redundancy because you don't leave a large space between words. You put a space symbol and go on to the next word. On the punched card, you have the difficulty of deciding in advance the format of the information you are putting on, and this quite often leads to the undesirable situation in which you plan for words with a certain maximum length, although many words do not have the maximum length at all. In English they have average lengths of five letters, and you are quite likely to waste quite a bit of the surface of the card. (This doesn't bother the punched-card manufacturer!)

The two other forms of input which have merit are the direct character reader and the spoken word. Many workers, including the Russians, regard character readers as very important, and certainly they are for any language which does not use a Roman script. The Russians are working on Chinese characters. So far I haven't heard the results of this work, but in 1960 they had a prototype reader.

Finally—and this sounds something like a physics text—the spoken word is a quite good method of input to computers. You have all seen things like *Shoe Box* into which you can speak the digits 0 through 9 and out of which you can obtain a suitable digital input for a computing machine. Actually, spoken-word input is probably not too useful for normal data processing but is quite likely to be useful for cataloging and stockkeeping, operations of all sorts in the areas where one does keep stock, and this goes from libraries to stocks of shoes in a shoe factory.

So much for the basic mechanisms. Now for two of the tools of mechanical language data processing. Many people say, "Let's sit down with a classical conventional dictionary and a classical conventional grammar, start from scratch, and see if we can work out a program to do a machine translation." My own concept is that the method to be adopted should be quite different. Machines are useful, whatever one may say to the contrary, in symbiosis with men; and an ideal symbiosis of machine and a man is in producing the basic material on language for use, if you like, in making a dictionary or making a grammar. Our own machine translation work has been based from the beginning on the notion that we use the machine to help us get the data which we want. Specifically, I view machine translation as a highly structured operation. The structure is twofold—the structure of the words themselves and the structure of the grammar. Machine translation works in a hierarchial process, starting with a list of words represented, from the point of view of analysis, not by a conventional dictionary starting with the word "a" in English and ending with "zymurgy," but rather by a dictionary starting with the most frequent word and then the next most frequent word and so on. If you are working out the program for a machine, it's a good idea if the first time you demonstrate the machine it doesn't fall down on the simplest sentence merely because somebody started with an obscure portion of a complicated dictionary of a technological subject. You first must produce a frequency or ordered list of words. Of course, this has been done by people like Dewey, but it pays to do it again when dealing with scientific material, and you do it on the machine. Having produced a structured list of words we then get to work putting in the relevant data about these words using a human operator and starting with the most frequent word. You then know that at any stage you are likely to deal with quite a large amount of the material in the text. The same thing goes for the grammar.

I can tell you a story here. Years ago when we were beginning to translate French into English, I went to the Professor of French at our College in the University of London and asked him what was the most frequent difference between word order in French and English. First he disclaimed any knowledge of this; then he came up with something obscure, which I have never been able to find in any French text, and which I suspect was something deriving from his speciality, Medieval French. We did eventually get the answer to this one—the most frequent ordering difference between English and French is, in fact, the inversion of the order of nouns, adjectives, and adverbs, and the next most frequent is pronoun-verb structures. We derived these pieces of information by analyzing sentences on a computing machine, using a combination of the linguist and the computer to produce this statistical data. Thus our program started off from zero on the assumption that we could do word-for-word translation (which of course we can't) and then worked its way up through an increasing list of complications—for example, the noun-adjective-adverb situation, the pronoun situation—eventually ending up in what we call MT6, which was quite a potent program. In Saskatchewan at the present time, we are applying just these principles to the analysis of the combination of English-French. English is most interesting in a number of respects, chiefly because it is the most ungrammatical language in the world, which makes it rather attractive.

I think I've talked long enough, but let's say a word or so about machine translation. We've heard something about its limitations. What sort of things can machine translation do? At various levels, I would maintain—other people's opinions notwithstanding—that machine translation can be useful. For example, if you merely translate the scientific nouns and verbs in a text, with no attempt whatever to do anything about their relation to one another, the result is very useful indeed to a human scientist. Perhaps some of you don't believe this but the fact is that many scientists who do not have access to a translating machine—I suppose this means, at present, all scientists because there are no translating machines doing this sort of work—and who are not skilled linguists start off merely by looking at the text to find what they conceive to be technical words and then looking these up in a dictionary. Quite often they go no further than this and say "Well, obviously this paper is of no interest." At this level, even word-for-word translation, with no particular assistance with the grammar, is useful. A machine *can* do it; it does at least save the scientists from looking up words in a dictionary. Of course one can go considerably further than this. If you are prepared to specify your field of interest and your language, it doesn't take too long (using the machine-man combination for the rules and the word lists) to produce a rough machine translation. There are a lot of lacunae in this. The dis-

advantage of statistical ordering is that the machine does not deal with all of the words. It makes no claim to do this. It will deal with the hundred or thousand or ten-thousand most frequent words, but when the word is not one of the most frequent, the machine first makes a check that something isn't merely wrong with the works (which all good machines ought to be programmed to do), then says "Well, this word is not in my list of words," and outputs it in original form with a note to some human being to look it up in the dictionary or to consult a colleague. Machines are useful at this level.

I can't help remarking as a little *jeux d'esprit* that one of the amusing things that people sometimes talk about is to do literary equality translation on machines. There are some bogus characters who say that we can do literary translation on machines, and while this is completely false in the general sense we *can* do something—and this something is quite amusing for a reason which I'll try to explain. Supposing that we want to translate Shakespeare into Goethe. We first make a list of all of the sentences that Shakespeare ever wrote, which is a fairly trivial operation, machinewise. Next we get a human being to go through this list, just as in making a telegraphic abstract or any other indexing operation, putting alongside each sentence certain category numbers which indicate the area of human endeavor into which the sentence falls—for example, boy meets girl, or boy loves girl, boy falls in love with girl, girl jilts boy, and so on. Having done this, we do exactly the same thing for Goethe, and now have two lists of sentences, each of which has associated with it some category numbers which effectively tell what the sentence is about. When we present our Shakespearian corpus to the machine, it looks up the Shakespeare sentence in the list, finds the category number, and goes to the list of Goetherian utterances. It will probably find several Goethe utterances of the same sort so it flips a coin, or, machinewise, consults a table of random numbers, picks out the equivalent of Goethe, and says "This is what Goethe says about the situation Shakespeare has described." When finished, we have exactly what Goethe said about the Shakespearean situation.

We've actually tried this on a small scale and there's one most interesting consequence. In using machine analysis of word statistics and structural occurrences, we can usually detect whether or not an author has been faked. For example, we've recently done some work on the authenticity of certain Johnsonian fragments from newspapers, in which word statistics show quite clearly whether or not he was the author of a particular fragment. When we do this particular analysis on a text constructed in the manner just described—that is, taking the actual utterances of A about the situations described by B, the interesting thing is that the word statis-

tics in the utterances of A are now correct for A. You can no longer do literary detection on it. This is rather fascinating because it does give a means of rewriting a few sonnets of Shakespeare or a few new Shakespeare plays and getting away with it. The literary detectives won't be able to operate, because the words are what Shakespeare (or Goethe) actually wrote.

That is just an aside but it is one of the things which a study of the structure and statistics of language makes possible. It is in a real sense machine translation because we are creating an artifact. We can go even further and make the selections from Goethe rhyme in the proper way; the possibilities are endless.

Finally, I thought I ought to say something about recent work, such as that done by Bar Hillel and Chomsky, the two oracles of Israel. Bar Hillel has been described by various people as the leader of the destructive school of machine translation. He wants to knock you on the head. He goes around producing counterexamples as to why one cannot do machine translation. Quite frankly (being brought up to regard any problem as a source of irritation until I have solved it) I go around saying just how you can solve Bar Hillel's paradoxical problems, most of which are quite trivial. However, he has done some good work. One good thing he did was to upset the Wittgenstein hypothesis. I mentioned a paraphrase of this earlier; what Wittgenstein actually said was interesting, particularly for anyone concerned with information science. He said, "What can be said, can be said simply." Oh, that this were written on the hearts of authors!

Bar Hillel, being the devil's advocate, examined this hypothesis in the context of a rather restricted grammar and showed that the hypothesis was wrong. In fact there exist utterances of infinite complexity in any language in this artificial language group—and by extension, in all natural languages. These sentences are not reducible to any simpler form. Later, Shamir and Bar Hillel advanced the interesting hypothesis that there exists a reduction algorithm that can be applied to sentences in a certain restricted class of grammars in which there is hope that some subset of natural language may fall. Bar Hillel and Shamir showed that there exists an algorithm for the reduction of sentences to sentences of canonical form or of minimum complexity. A sentence may indeed be of infinite complexity, but, in this event, we can show that it can be reduced no further. If a sentence is just badly put together, the algorithm gives a formal means of reducing it to a sentence of minimum complexity. The importance here is that, by taking a number of documents, we can in principle reduce the contents to minimum complexity and form the union of this information for all documents. The effect is to produce an output which contains all of

the original material in the original documents but none of the redundant material.

I can't help concluding with a piece of statistical information derived from a survey I make of the computer engineering literature for 1960. I was doing this as a survey article for a British journal and the interesting result was that, in 1960, there were very approximately 10,000 pages of periodical publication in the field of computer hardware. The original material in this 10,000 pages could be described adequately in 40 pages. A plausible inference is that the exponential growth, or information explosion, is a figment of the imagination. The growth is much more nearly linear. The moral of this should, I think, be left to university presidents to unravel!