[*Proceedings of the National Symposium on Machine Translation*, UCLA February 1960]

Session 7: THE DICTIONARY

SEGMENTATION

Sydney M. Lamb
University of California, Berkeley

Introduction

Before a text can be translated it must be cut up, or segmented, into units small enough for the machine to handle efficiently. The two most frequently used types of units for this purpose have been the sentence and the word (where by word is meant a sequence of graphemes which can occur between spaces, or the representation of such a sequence on the morphemic level). Even where sentences are used as the units of translation, however, either words or parts of words are treated as units for the dictionary lookup and for the process of analyzing each sentence. This paper is not concerned with the pros and cons of sentence-by-sentence translation, but with the optimal size of smaller units into which the text is segmented, whether as the items to be translated or as preliminary units for sentence-by-sentence translation.

We must first distinguish between analytical segmentation and operational segmentation. By analytical segmentation is meant the determination, in the course of linguistic analysis, of what the units ought to be in a maximally efficient translation system. Operational segmentation refers to the process to be used by the machine in cutting actual text into units selected as a result of analytical segmentation.

In most previous considerations of the segmentation problem, the idea of cutting up words has been thought desirable, if at all, merely from the point of view of programming efficiency for the look-up process, and even for this purpose only because of the limited size of existing rapid-access storage media. However, let us consider the problem of analytical segmentation apart from computers and their properties. Let us, in other words, determine the optimal units in the abstract, on the assumption that whatever we decide is most efficient from the linguistic point of view will turn out also to be most efficient on the machine. This assumption can be regarded as permissible simply because of the great flexibility which computers are known to possess. Since they can do practically anything

that involves manipulation of information, we may safely assume that programming techniques can be found which will enable the machine to do whatever is determined in the abstract to be most efficient with regard to segmentation.

Number of Dictionary Entries

Considering our problem in the abstract, then, we may first state as a self-evident fact that, other things being equal, a smaller dictionary is more efficient than a larger one if both cover the same amount of material. To consider what effect degree of segmentation has on dictionary size, let us take any classes A and B whose members occur with each other. Let m be the number of members of A, and n the number of members of B. If there are no restrictions on the occurrence of members of A with members of B, the number of combinations is mn. If the members of A and B are treated as separate units, the number of dictionary entries required is only $m+n$, whereas failure to carry out segmentation would require mn entries. As long as m and n are both greater than 1 and at least one of them is greater than 2, mn is greater than $m+n$. If either quantity is very large, say several thousand, the difference is overwhelming.

Lest we arrive at a rash conclusion, however, we must refine the calculation. If the foregoing principle were the only one involved, we would be led to carry out segmentation clear to the ultimate constituents, namely the morphemes. But to do so is not economical because of the large number of allomorphs (e. g., "streng" in "strength") and allosemes (e.g., "dear" in "dearth") or portmanteau semes (e. g., "understand") which are encountered if segmentation is carried into the inner layers. If the target representation of a composite source form cannot efficiently be treated as the sum of the representations of the constituents of that form, with regard to both content and expression, it is more economical to treat the composite form as a unit, i. e., to leave it unsegmented. To return to our calculation, then, we may let x represent the number of combinations of members of our classes A and B which would be left unsegmented for the sake of avoiding extra allomorphs and allosemes. Thus $m+n+x$ would be the number of entries needed if all the combinations of members of A and B were segmented except the ones involving complications.

In addition, we must reckon with the possibility that not all possible combinations of members of A and B occur. This situation is generally true of derivational constructions. Let y be the number of combinations which do not occur. Then mn-y is the number of entries needed if combinations of members of our two classes are not segmented. Therefore, if mn-y>m+n+x, then segmentation is desirable; while if mn-y<m+n+x, it is more economical not to segment.

It is not worth the time required to attempt an exact calculation of these quantities. Instead, it is possible to get a rough idea fairly easily. In the first place, it will doubtless be conceded that segmentation should be carried out at least to the point at which words are separated from each other. Such a policy is doubly desirable since spaces on the graphemic level, which are easy to locate, can then be taken as morph boundaries.

Let us consider the possibility of segmenting inflectional affixes from stems, in a language like Russian. Let A be the class of noun stems, and B the class of case suffixes. In terms of the formulation given above, the size of y must be regarded as negligible, and the size of x is also very small, since alternations among allosemes of the case suffixes are generally conditioned not by the preceding stem but by other words in the sentence. (Note that we are considering the suffixes on the morphemic level, not the graphemic. The selection of the morpheme represented by the graphemic form of a case suffix often does depend on the noun stem.) Thus, since A has a very large number of members, the quantity mn-y exceeds that of m+n+x by several hundred percent, and there is no doubt about the desirability of segmentation in this case.

For inner-layer derivational constructions, on the other hand, the quantities x and y are generally rather large in comparison to mn and, except in the case of compounds, one of the quantities m and n represents a very small class of affixes. In fact, derivational affixes should usually be considered one at a time, so that one of the factors m or n is 2 (since the affix either may or may not occur). The economy of segmenting for such constructions is therefore limited.

Volume of Instructions

We have considered above the effect of segmentation only on the number of dictionary entries. Another measure of efficiency is the quantity of instructions (for determining proper target representations) in dictionary entries. The advantage of segmentation in this connection is present for constructions in which one or both members have allosemes that are conditioned not by the partners but by other units lying outside the immediate construction. Examples would be inflected noun forms of Russian, for which the choice of the correct allosernes of both the stem and the case suffix is conditioned by material outside the noun form itself. To fail to segment such forms would mean that the instructions for choosing the correct target representations of the noun stems would be repeated in the dictionary entry for each inflected form of those nouns, while the instructions for the case suffixes would be repeated for each of the thousands of noun stems with which each case suffix can occur.

The volume of the instructions required can be measured in terms of the type of formula given above for estimating the number of dictionary entries, but in this case the difference is even more striking. If we again take m and n as the numbers of items in the two classes involved, and let i be the volume of the set of instructions for one stem or case suffix (assuming for simplicity that they all require the same amount of instructions), then mi+ ni is the volume of instructions if we do segment, whereas if we do not, the figure is 2mni . In other words, the difference in economy is twice as great as that calculated in terms of number of dictionary entries alone. The reason for this is simply that each dictionary entry for the unsegmented forms must contain instructions not only for the stem but also for the suffix.

Of course, part of this multiplicity of instructions is avoidable under circumstances in which cross-references can be used instead of duplicate instructions. Such circumstances would be present when one of the classes consists of a sufficiently small number of members, occurring with sufficient frequency, to warrant keeping the instructions for them separate from the individual entries. This would be feasible for our example involving case forms of Russian nouns. The instructions for translating case suffixes could be

stored independently from the dictionary proper, and they would all be present in core storage during the stage of translation. If this technique is used, then the unsegmented version of the dictionary would be longer than the segmented one only by the amount of space necessary for the cross-references, multiplied, of course, by the number of all the extra entries.

It should be kept in mind that these considerations of efficiency are made without reference to any particular computer. Even if one had a computer with 100, 000 words of core storage, it would be more efficient to segment words than not to do so, since a small number of entries having smaller volume on the average are conducive to faster lookup speed regardless of the amount of storage capacity available.

Neologisms

There is, however, another type of consideration which is perhaps more important than either the amount of time or the amount of space used. This is the matter of neologisms. The inventory of words in a language is constantly changing. New words are freely, even unconsciously, formed from parts of words which have the capability of occurring in new combinations. In English, combinations of adjectives with the suffix "-ness" are almost as freely formed as combinations of adjectives with following nouns. The fact that a space occurs between the constituents for the latter type of construction but not for the first is irrelevant. The space, being a very obvious grapheme on a page, has tended to be very misleading and its grammatical significance has been highly overrated. It is, in short, a mistake to suppose that the units out of which a text is constructed are the same as the units which occur between spaces.

If productive prefixes and suffixes are entered in the dictionary as independent items, then their occurrence in new formations will cause no difficulty. The importance of this principle may be illustrated by a few examples from English which I have run across, without making any specific effort, during the past few weeks. Advertisers tell us that there is a certain vegetable juice which "outflavors" other juices, and that a certain type of peanut butter is the "peanuttiest. " (All underlining in the following quotes is mine.) From C. B.S. News: "... the many-sidedness of the Khrushchev personality. " From a letter to the Editor of the San Francisco Chronicle, complaining

about another writer's use of the word thusly: "But you should have added a warning against his ever being seen in public using the adverbial abomination of thusly. Can one appear over (or as he would say, overly) strict in the condemnation of the incorrect, ill-sounding and pseudo-jocular pomposity of the superadverbialized adverb? Neverly." Or, in the cartoon, as Lucy said to Schroeder: "I'm sure I can help you publicitywise with Beethoven's birthday. After all, this is a really big thing. We must do whatever is best Beethovenwise!" Other recently formed words now coming into common use in their fields are "containerization", "microcircuit", and "microminiaturization".

Operational Segmentation

If we may take it now as established that segmentation of inflectional affixes and productive derivational affixes should be carried out and that at least some compounds should also be separated into their constituents in languages in which compounding is a productive process, the units resulting from such segmentation may-be called "lexes" (the basic units of the lexicon). The terms "prefix", "base", and "suffix" may be used for different types of lexes or their corresponding lexemes (i.e., morphemic representations of lexes). In considering what is the most efficient way to carry out operational segmentation on the machine, we must make use of two additional terms. The heading of a dictionary entry is an instance (or coded representation) of the lex for which the entry exists; it serves to identify the entry. A word being looked up, or ready to be looked up, may be called the vestigand (based on the gerundive of Latin vestigare = "to track, trace out; to search after, seek out; to inquire into, investigate"; hence, "that which is to be traced out, searched after, investigated"). A vestigand will coincide with some heading in the special case in which it is not segmented.

There are four approaches to operational segmentation, corresponding to the four sets of choices possible for two pairs of alternatives. First, we can, as it were, start looking at a vestigand at the left, going into it a certain distance and making a cut; or, we can start at the right and move backwards. The second area of choice may be described as follows: As we are going into the word (from either the beginning or the end), we may either make a cut as soon as a letter sequence is encountered which coincides with a heading,

or we may follow a policy of cutting only when the longest heading contained within the vestigand is found. For this second pair of choices, the first alternative will often lead to a false or a quasi-segmentation since cuts will often be made for letter sequences which happen to coincide with some heading, even though they do not represent it. With regard to the first choice, starting from the left is more effective for most languages since most languages have more suffixes than prefixes, especially among the productive affixes. Greater diversity, providing greater discrimination, is therefore available at the left end. The most effective operational segmentation, then, will be that which works so that the longest heading contained within the vestigand, beginning at the left, is taken as the first lex, the longest heading contained in the remainder, if any, as the next, and so forth. However, it is necessary, even if this procedure is used, to check the tentative segmentation arrived at, to make sure that the provisional lexes can actually occur with each other in the order found. Such checking is necessary because it can happen that the correct cut must be made such that the first lex of a vestigand or remainder is actually shorter than the longest one contained in it. For example, the Russian form pozvoljat should be segmented to give pozvol plus -jat, even though the longest heading is pozvolja and the the remainder, -t, is a suffix. Segmentation checking, by the use of segmentation codes, can be used to reveal that the suffix -t cannot occur with the stem pozvolja, so the machine can select the next longest heading, this time coming up with the correct segmentation.

This method of operational segmentation makes it possible for the machine to make its cuts in exactly the same places in which they would be made by a structural linguist. A means of programming the system for a computer is explained in another paper.[1] The procedure is so efficient that the amount of time required for operational segmentation is insignificant in comparison with the saving of time made possible by the small size of the dictionary. Using an IBM 704 with 32, 000 words of core storage, the estimated lookup time for a dictionary accommodating a vocabulary of up to 500, 000 words is only 8 milliseconds per word (i. e. , 125 words per second).

[1] Sydney M. Lamb and William H. Jacobsen, Jr. , "A High-Speed Large-Capacity Dictionary System", (in press).

Of this time, only about 1 millisecond per word is taken up by segmentation. Naturally, even more dramatic savings of time are realizable by using the system on faster computers such as the 7090 and the Transac S-2000.