[*Proceedings of the Ninth International Congress of Linguists*, Cambridge, Mass., August 27-31, 1962 (The Hague: Mouton, 1964)]

CONTENT RECOGNITION AND THE PRODUCTION OF SYNONYMOUS EXPRESSIONS

WAYNE TOSH

1.0 The Linguistics Research Center of The University of Texas is developing several models for machine translation. It is the purpose of this paper to explain in part certain aspects of the linguistic work now in progress and the direction of the work projected for the future.[1] The efforts of the linguistics, mathematics and programming groups of the Center are so directed as to develop the most general translation schemes possible. The computer programming group is developing a system of programs which allow the linguist to work independently of considerations for the computer. The mathematics group is concentrating on formal proofs of the working assumptions of the linguistics group. In this way we hope to anticipate and circumvent unduly expensive operational problems with the computer.[2]

2.0 To prepare a language for use in the translation system, we begin with the description of its phrase structure. We accomplish this by writing a grammar of replacement type rules of the general form $X \rightarrow Y$ where X is interpreted as a syntactic class name, the symbol $\rightarrow$ is read as "may replace/may be replaced by". Y represents either a sequence of alphabetic characters, one or more syntactic class names or a combination of both alphabetic characters and class names. Thus, we provide replacement rules of the two basic types: terminal and non-terminal. Terminal rules are those of the form $X \rightarrow a$, where the lower case *a* represents an alphabetic expression. The non-terminal rules are represented by rules such as the following

$$P \rightarrow Q^1$$
$$X \rightarrow Y^1 + Z^2$$
$$M \rightarrow N^2 + b + P^1$$

The upper case symbols denote variable, syntactic classes.

2.0.1 As in other systems utilizing replacement rules, the restriction on replac-

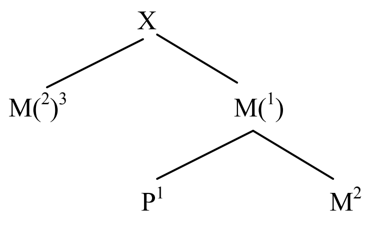
[1] Support for work on this paper was provided by the U.S. Army Signal Corps.

[2] For a more detailed accounting of all aspects of the work at the Linguistics Research Center see *Machine Language Translation Study,* quarterly progress report 8 and later, 1 February 1961 —, The University of Texas. Support for work in the various areas of investigation is provided by the National Science Foundation and the U.S. Army Signal Corps.

ing one of the right-hand variable elements of a rule with the right-hand element(s) of another rule is that the right-hand variable being replaced must be identical with the left-hand element naming the elements being substituted. In the phrase structure of the languages which we are analyzing, we require that an additional condition be met. Any rule to be inserted must be substituted at the lowest order superscript. For instance, suppose we are given the following set of rules:

$$X \rightarrow M^2 + M^1, \qquad M \rightarrow P^1 + M^2$$

The second rule must be substituted into the first rule at M^1. After the substitution has been made, a branching structure diagram would have the appearance below.



Each time a substitution is made the superscripts associated with the variable classes are re-evaluated in the following manner. The right-hand elements of the inserted rule retain their original superscript values, and the unfilled variables of the rule substituted into have their superscript values increased by $n - 1$, where n is the number of variables in the right-hand side of the inserted rule. The parentheses above show the original superscript values of the first rule. The numbers without parentheses denote the superscript values resulting from the substitution. If we were to make further substitutions into the above network, the next rule would have to be insertable at the variable element P^1.

2.0.2 The superscripts associated with the right-hand variable elements of non-terminal rules have another function as well as that of specifying the order of substitution. In the two rules $X \rightarrow Y^1 + Z^2$ and $M \rightarrow N^2 + b + P^1$, we find a like number of variable elements. Superscript 1 denotes the correspondence of semantic content for the respective variables Y and P. Superscript 2 likewise denotes a similar relation between the variables Z and N. In other words, the superscripts guarantee that the same semantic content found to fill one variable slot will be substituted into only those other variable slots which are appropriate correspondents.

2.1 Any two or more rules which stand in this semantically equivalent relation to one another are said to be in the same semantic equivalence class. We require that all rules in the same equivalence class be composed of the same number of variable elements. The variable elements of rules in the same equivalence class and bearing the same superscript do not necessarily have identical syntactic class names.

2.1.1 Since an equivalence class must contain only rules which have the same number of variable elements, we have thus specified a particular kind of transformation. We have established the relation which allows for the permutation of elements, whether the respective correspondents have identical class names or not, and for the correlation of semantic content among correspondents having the same class names, whether or not the corresponding elements are permuted.

2.1.2 Equivalence classes of non-terminal rules provide us with variety in the order of expressions. Equivalence classes of terminal rules provide variety of expression in that they allow us a choice of one or more morphs or "words".

2.1.3 As the terminal expressions are initially classified in the translation system, each terminal rule is placed in a unique equivalence class. Whenever two expressions, or more precisely, terminal rules are found to be mutually substitutable in some context, each rule is assigned to the other's equivalence class. For example, suppose the rules *A:* $N_x \rightarrow$ abridgement and *B:* $N_y \rightarrow$ abstract are found to substitute for one another in some context. The upper case symbols which are in italics denote the equivalence class name of each rule. If the rules are substitutable we assign the rule $N_y \rightarrow$ abstract to equivalence class *A,* as well as to *B,* and $N_x \rightarrow$ abridgement to equivalence class *B,* as well as to *A.* Class assignments made on this basis will not necessarily result in the two classes *A* and *B* containing the same list of member rules. A representative list of the rules in *A* might be as follows:

A:	$N_x \rightarrow$ abridgement	$N_b \rightarrow$ digest	$N_f \rightarrow$ contraction
	$N_y \rightarrow$ abstract	$N_c \rightarrow$ abbreviation	$N_g \rightarrow$ truncation
	$N_z \rightarrow$ condensation	$N_d \rightarrow$ curtailment	$N_h \rightarrow$ summarization
	$N_a \rightarrow$ brief	$N_e \rightarrow$ shortening	$N_i \rightarrow$ reduction

I.e., the rules substitutable for $N_x \rightarrow$ abridgement. Similarly, a representative list of member rules for equivalence class *B* might read:

B:	$N_y \rightarrow$ abstract	$N_b \rightarrow$ digest
	$N_x \rightarrow$ abridgment	$N_j \rightarrow$ essence
	$N_z \rightarrow$ condensation	$N_k \rightarrow$ distillation
	$N_a \rightarrow$ brief	$N_l \rightarrow$ extract

i.e., the rules substitutable for $N_y \rightarrow$ abstract. Every rule in the grammar will belong to at least one equivalence class.

2.1.4 Whenever a rule is found to apply in the analysis of an input expression, this fact is registered along with the information as to what other rules are members of the same equivalence class, i.e., are substitutable for the given rule. For output, then, we are able to call not only upon this rule but all the rules in the same equivalence class, assuming that we are synthesizing equivalent expressions in the same language as the input. Not all such rules should be considered as likely candidates for substitution. Therefore, we must provide some way of establishing a preference for selecting among the various alternatives.

2.2 We are able to provide a measure of synonymy between any two rules by considering the set of all equivalence classes in which either rule occurs. The synonymy S is given by

$$S = \frac{N(C_x \cap C_y)^3}{N(C_x \cup C_y)}$$

C_x is the set of all equivalence classes in which, say, the rule

$$N_x \rightarrow \text{abridgement}$$

occurs. C_y is the set of all equivalence classes in which the rule

$$N_y \rightarrow \text{abstract}$$

occurs. If we compare each of the above rules which are substitutable for

$$N_x \rightarrow \text{abridgement}$$

against this rule in the manner indicated, we arrive at numerical values for each of the rules such that $0 \leq S \leq 1$. Any rule compared with itself will give the value for identity, i.e., 1. Rules which stand in the relationship of being allomorphs will have the value 1. Rules which are not mutually substitutable will have the value 0 with respect to each other. All the rules which are in equivalence class *A* can thus be assigned a value between 0 and 1 which will allow us to establish a preference among the various synonyms for the expression *abridgement.*

3.0 None of the above computations are a part of the process of translation as such. Rather, such computation is carried out on the linguistic data as they are compiled for use in the system. The linguist is not himself concerned with making these computations. They are carried out by the compiling facilities of the program system. Once the computations are made, the values are available at all times for the translation process. The values are not absolute, since they depend on the size of the grammar and the partitioning of rules into equivalence classes. They are constant, however, within the grammar for any one language and at any one time.

4.0 In order to carry out the transfer of information from one language to another, we establish a table of interlingual equivalence classes which give the correspondences among semantically equivalent equivalence classes of the several languages in the system. This is to be accomplished in essentially the same manner in which we set up equivalence classes of rules. Let us consider again the rule

$$N_x \rightarrow \text{abridgement}$$

We shall denote its equivalence class as A_e to indicate that it is an English equivalence

[3] Cf. "Work in Mathematics", *Machine Language Translation Study* (= *Quarterly Progress Report* 9), p. 18.

class name. Suppose we find that the English expression may be translated into French by the rule

$$L_f: \; N_q \rightarrow \text{abrégé}$$

and into German by

$$M_g: \; N_s \rightarrow \text{Abkürzung}$$

We place each of the above equivalence classes A_e, L_f, M_q in a unique interlingual equivalence class 01, 02, 03, respectively. Since the above rules are considered translation substitutions for one another, we have the respective assignments:

$$\begin{array}{l} 01: A_e, \; L_f, \; M_q \\ 02: L_f, \; A_e, M_q \\ 03: M_q, \; A_e, L_f \end{array}$$

So far, the membership of the three interlingual equivalence classes appears to be identical. But if we continue building up the membership assignments as we did above, with rules in equivalence classes, we will arrive at similar results. Likewise, we may state a preferential ordering of one equivalence class compared to another equivalence class by considering the set of all interlingual equivalence classes to which the two belong. The computation of such "synonymy" values is similar to that for rules as described above.

5.0 Briefly, the translation process functions as follows. A certain set of rules is found to apply in analyzing the text of the input language. The equivalence classes to which each of the applicable rules belongs, respectively, are recorded. The interlingual equivalence classes to which each equivalence class belongs are likewise recorded. Assuming that we have selected the language into which we want to translate, we transfer on a one-one basis from the given equivalence classes through the interlingual classes to the equivalence classes of the target language. We pick the latter set of classes on the basis of highest synonymy values and in turn select rules from each of these equivalence classes on a similar basis. The resulting synthesis is a close translation of the original. We may refer to this kind of translation as a rule-for-rule translation.

6.0 The translation model described thus far consists only of the phrase structure descriptions of the various languages in the system. It takes into account only the overt, formal characteristics such as inflection, case and number agreement, etc. Furthermore, it provides only for transformations of the type which permute elements. It does not take into account transformations which delete or add elements. Nor does it provide for any restrictions such as semantic agreement, e.g., animate actor with animate verb.

7.0 We may provide for these additional requirements by expanding our model.

We shall refer to the grammar which describes the phrase structure of a language as the first order grammar. The second order grammar is a description of the transformational structures. The input for a first order grammar is the sequence of alphabetic characters, numerals, blank spaces and punctuation occurring in a language. The input for the second order grammar is the sequence of first order rules found to apply in the analysis of an alphabetic input. If we visualize the replacement rules of a phrase structure as lying in a plane, we may think of the transformation rules as extending into the third dimension.

7.1 In the original model, the equivalence class serves a double purpose in that it brings together rules which stand in the relation of being allomorphs as well as bringing such morphemic subsets of rules together as allosemes. In the extended model, the equivalence class is obviated, for the terminal rules of the second order grammar take over the load of allomorphic classification. The next stratum of rules up from the terminal rules may be interpreted as those second order rules which classify allosemes.

7.1.1 To illustrate, let us consider such first order rules as

$$\begin{aligned} AJ_x &\rightarrow \text{good} \\ AJ_y &\rightarrow \text{bett} \\ AJ_z &\rightarrow \text{be} \end{aligned}$$

which are used in the analysis of the expressions *good, better* and *best.* Each rule in the first order grammar will have assigned to it a unique and permanent reference number. I shall represent these numbers here as *m, n,* and *o* — these tags are not to be confused with equivalence class names above. Thus, as each of the above rules is coded into the first order grammar, it receives the respective tag:

$$\begin{aligned} m&: AJ_x \rightarrow \text{good} \\ n&: AJ_y \rightarrow \text{bett} \\ o&: AJ_z \rightarrow \text{be} \end{aligned}$$

In the second order grammar we now code each of the first order tags *m, n* and o as second order alphabetic expressions. We write the second order rules

$$\begin{aligned} \{\text{GOOD}\}_m &\rightarrow m \\ \{\text{GOOD}\}_m &\rightarrow n \\ \{\text{GOOD}\}_m &\rightarrow o \end{aligned}$$

where $\{\text{GOOD}\}_m$ denotes a second order terminal class which we may interpret as a morpheme class.

7.1.2 The next higher stratum in the branching structures of the second order grammar may be interpreted as a sememic classification of the second order terminal classes. For instance, we would find the second order rules

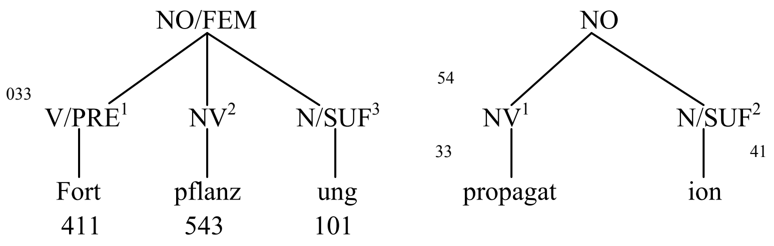
$$\{\text{GOOD}\}_s \rightarrow \{\text{GOOD}\}_m$$

where $\{GOOD\}_s$, denotes a sememe. In addition to the above rule, we might also expect

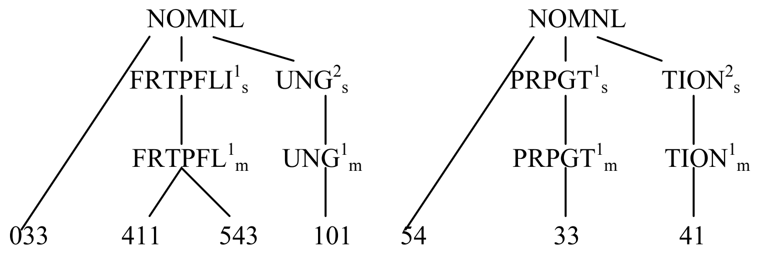
$$\{GOOD\}_s \rightarrow \{KIND\}_m$$
$$\{GOOD\}_s \rightarrow \{COMPETENT\}_m$$
$$\{GOOD\}_s \rightarrow \{BENEFICIAL\}_m$$

etc. Each of the above morphemes $\{KIND\}_m$,$\{COMPETENT\}_m$, $\{BENEFICIAL\}_m$, etc. will likewise appear in its own sememe along with its own list of substitutable morphemes. As earlier, we are able to arrange a set of morphemes in decreasing order of preference of substitution for a given morpheme. Instead of comparing two rules for their distribution over sets of equivalence classes as earlier, we now inspect their distribution over respective sets of sememic classes.

7.2 I have said that the two main functions of the second order grammar are to provide a semantic grammar and a transformation structure: In the interests of space I shall restrict consideration of the transformational problem to an example which relates an unlike number of elements in two nouns, one from German and one from English. The first order analyses for each noun are given below:



The numbers associated with each rule are the first order tags. If we read the tags in the order of the superscripts, the German second order alphabetic sequence is 033, 411, 543, 101. The English sequence is 54, 33, 41. We may perform a second order analysis on these sequences as below:



Note that the second order rules NOMNL → 033 + $FRTPFL^1_s$ + $UNG,^2_s$ and NOMNL → 54 + $PRPGT^1_s$ + $TION^2_s$ are composed of the same number of corresponding variable elements. In fact each German-English pair of rules which corresponds in the above illustration is composed of the same number of variables. We may, therefore, establish an interlingual correspondence between each such pair of rules for purposes of translation. The approach for relating transformationally all such dissimilar first order constructions is essentially the same.

8.0 As of the writing of this paper, extensive linguistic coding has been carried out only on the phrase structures of German and English — two languages about which we know a great deal, compared to many other languages. Even so, much remains to be done before we can say that we have, for machine translation purposes, a reasonably complete phrase structure description of the two languages. Recently, we have just begun compilation of synonymy lists which will provide us with the necessary information for coding the sememic structures of the second order grammar.

University of Texas

DISCUSSION

DETOLLENAERE:

It was fascinating for me to recognize the content recognition in the lecture, but 1 failed to see "the production of synonymous expressions", which is mentioned in the title of his paper.