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

Session 9: SEMANTIC RESOLUTION

A GENERAL-PURPOSE LANGUAGE TRANSLATION PROGRAM FOR THE IBM 650 COMPUTER

Ramon D. Faulk
University of Texas

I will discuss the application of redundancy-sorting techniques to the resolution of semantic problems. I must leave it to you to decide whether or not these techniques actually deal with semantic problems. I believe they do, even though this paper describes a manual of operation for a computer program.

I should perhaps explain that the term "redundancy sorting" refers to a battery of data-processing techniques that operate directly on the sequences of language to obtain information which is useful for the purpose of translation. Because these operations have been designed for application to sequences in general, the adequacy, or inadequacy of their results is independent of the particular language processed; for the same reason, the sequences processed need not be those of language, but might represent any sort of non-verbal behavior, such as chess playing, etc.

In applying these procedures to machine translation, one thinks of the sequences processed as a set of examples which represent, more or less adequately, the relative distribution of the various linguistic entities contained in the set. To this set of source-language examples there corresponds a second set each element of which is a target-language equivalent of one of the source-language examples.

In a typical situation, the input to the translation procedure will be a sequence constituted of source-language entities, but not contained in the set of examples. Upon the presentation of such an input, the procedure is to compare each source-language example with the expression to be translated, and to compute for each example a similarity rating located within a range whose maximum value indicates identity of the two sequences, and whose minimum value is zero. The result of each computation is then assigned to the target-language equivalent of the source example just processed, on the assumption that each target-language example resembles the desired translation as much as its source-language counterpart resembles the input.

The next step is to submit the target-language data to a reversed form of the comparison process, thus generating the hypothetical target-language expression. The statistics involved in this procedure are such that, if the source-target relation is reversed, and the translation just produced is presented as input to the comparison process, then the results of the comparison will approximate as closely as possible those obtained with the original input sequence.

Notice that the results of this second pass of the comparison process can now be used to regenerate (again, as closely as possible) the sequence originally presented for translation. At this point one sees that the resources of such a translation procedure can be greatly extended by applying a feedback principle to render the process sensitive to its own errors. This is accomplished by modifying the original similarity ratings so that those examples which produced discrepancies in the retranslated input are down-graded, while those having a desirable effect on the retranslation are up-graded. The cycle is then completed by using these modified similarity ratings to obtain a new translation of the original input.

The above cycle is repeated until the quality of the retranslation, as measured by the comparison process, ceases to improve, indicating that it has become stabilized at a constant value, or has begun a progressive deterioration. In either case, the most recent translation of the input is produced as the best solution which can be derived from the available data, thus terminating the process.

In describing the above process, I have spoken loosely and in general, wishing to present a clear picture of this approach to MT. Now I wish to discuss a general-purpose language translation program which I will refer to as 650-MT-1A. The numeral 650 in this designation indicates the type of computer for which the program was written, namely, the IBM 650. The MT-1A denotes that the present program is the first of a series of small-scale experimental translation programs. By referring to this program as a general-purpose MT program, I allude to the fact that procedures of the type used in 650-MT-1A are applicable for the purpose of translation of any two languages.

650-MT-1A and its immediate successors are intended primarily as research tools to provide experimental computer outputs. These will be useful in evaluating and improving the performance of

translation programs based on statistical decision procedures of the type described in this paper. It is expected that these exploratory investigations will lead to a general solution of the machine translation problem, in the form of an integrated system of computer programs which will be

(1) adaptable to the specific needs of a variety of translation problems,
(2) operationally independent of the languages involved in a particular application.

I should remark at this point that nothing is predicated upon the success or failure of these experiments so far as the major programming efforts of our group are concerned. Naturally, we hope to take advantage in the future, as we have in the past, of whatever may be learned from research of the type I am describing.

From the standpoint of statistical research in the simulation of intelligent behavior, such as that exhibited by the human translator, there is more to be learned from the failure than from the successful attempt to solve a given problem. Thus, for the purposes of linguistic research, it is of relatively little interest when 650-MT-1A translates a sentence correctly. On the other hand, when the machine, i. e. , the program, fails at one or more points to produce the desired translation, it may be because the distribution of one or more linguistic entities is not realistically represented in the example-sentences present in the data tables. (The same thing might be true where the distribution of entities was represented by grammatical rules and class labels.) Indeed, the problem of machine translation, whether by grammatical or statistical analysis, can be stated as one of formulating a rigorous concept of sequential distributions, and defining precisely what constitutes the representation of these in a sample taken from a population of sequences. It is with this formulation of the problem in mind that 650-MT-1A has been designed to produce a table of decreasingly probable alternatives for each decision made. By examining the tables which served as the basis for an inadequate translation, it is possible to "adjust" the contents of the data tables to represent more realistically the distribution of entities involved in an erroneous decision-sequence. By conducting carefully designed experiments of this type, it should be possible to answer, or at least to

gain some insight into, the question of what is essentially involved in the concept of sequential distributions and their representation in samples (i. e. , examples).

The comparison and term-selecting procedures employed in 650-MT-1A are based on a concept of absolute position as the carrier of order information, as evidenced by the fact that only corresponding x-th terms are matched in the comparison of two sentences. In other words, the translation procedure in this program implicitly makes the following assumption:

> That any entity which might normally occur at a given position as one of the terms of a sentence actually does so in one or more of the available data-sentences.

It is obvious that, aside from other considerations, such an assumption greatly increases the number of sentences needed by the machine to generalize successfully over the entire population from which the examples are taken. The justification for programming such a procedure lies in the fact that the relatively simple program required by the process provides a useful test of the redundancy-quotient concept as a criterion of selection in statistical decision processes. It is too early to make any positive conclusions concerning the validity of the redundancy quotient. However, results thus far obtained with 650-MT-1A fully justify the programming of 650-MT-1B, which will be a self-validating general-purpose translation program embodying in its analysis and synthesis procedures the concept of relative position as the carrier of order information. It is expected that this program will serve as the prototype for an IBM 709 version of the process. Though there seems to be every reason for optimism concerning the behavior of these programs, I will not speculate further on the subject.

In closing, I feel obliged to express my personal belief, which I think is shared by other members of our project, that in this statistical formulation of the machine translation problem are involved some very fundamental aspects, not only of logic, linguistics, and mathematics, but of the behavioral sciences as well, including those concerned with the simulation of intelligent behavior.

APPENDIX
Details of the Machine Program

1. General.

650-MT-1A is a program for translating sentences from any one language S into any other language T. No restrictions are placed on S and T; for example, both might be English. The program is designed for use in conjunction with SOAP-2, and uses the same input and output formats. (SOAP-2 is a utility program used in the assembly of 650 programs from symbolic operation codes.) Though inadequate for any kind of large-scale translation, the SOAP-2 formats offer two distinct advantages in the design of 650-MT-1A, i. e., using the SOAP-2 program simplifies the program design while providing a convenient input-output facility for the wide variety of languages to be processed by 650-MT-1A. The price of this simplicity and convenience is the limitation of linguistic entities to five characters, the number of letters which may be stored in one location in the 650. This means that lexical items of six or more letters must be abbreviated to five letters for processing by 650-MT-1A. Although this would constitute an unacceptable limitation for some purposes, it poses no serious problem for the research applications for which the program is intended. 650-MT-1A is designed for use with the usual complement of index registers and core storage units available to the 650.

2. Input-Output Facilities.

Linguistic data for use with 650-MT-1A are prepared in the form of the SOAP-2 pseudo-operation input card, as follows:

LOC	OP	DATA	INSTR
S0001	ALF	THE	(blank)
S0002	ALF	MAN	
S0003	ALF	SAW	
S0004	ALF	THE	
S0005	ALF	CAT	
S0006	ALF		
S0007	ALF	A	
S0008	ALF	CAT	

```
S0009       ALF       SAW
S0010       ALF       THE
S0011       ALF       MAN
S0012       ALF
S0013       ALF       HAS
(etc)       ALF       (etc)
....        ....      ....
....        ....      ....
S0499       ALF
S0500       ALF
T0001       ALF       DER
T0002       ALF       MANN
T0003       ALF       HAT
T0004       ALF       DIE
T0005       ALF       KATZE
T0006       ALF       GESHN
T0007       ALF
T0008       ALF       EINE
T0009       ALF       KATZE
T0010       ALF       HAT
T0011       ALF       DEN
T0012       ALF       MANN
T0013       ALF       GESHN
T0014       ALF
T0015       ALF       HAT
(etc)       ALF       (etc)
....        ....      ....
....        ....      ....
T0499       ALF
T0500       ALF
X0001       ALF       A
X0002       ALF       MAN
X0003       ALF       SAW
X0004       ALF       THE
X0005       ALF       CAT
X0006       ALF
(Transfer to TRAN = 1900)
```

```
X0001        ALF        DER
X0002        ALF        MANN
X0003        ALF        SAH
X0004        ALF        EINE
X0005        ALF        KATZE
X0006        ALF
(Transfer to TRAN  =  1900)
(etc)        ALF        (etc)
```

(a) Location Field: The location field of a data card may specify an address within any one of three regions, S, T, or X. A regional address preceded by S means that the information specified in the data-address field of the card will be entered in the S-table as source-language data. Similarly, regional addresses preceded by T will cause data to be entered in the T-table as target-language data. Data entered in the X-table will be processed for translation.

(b) Data-Address Field: The data-address field contains five alphabetic characters (all five of which may be the special character, BLANK). Each group of five characters is stored as a ten-digit number, and processed as one of the terms of a sentence. A sentence, as defined for this program, is the sequence of terms in a consecutive series of memory locations, the last one of which contains a terminal code. A term consisting of five BLANKS is stored as ten zeros, and is interpreted by 650-MT-1A as a terminal code. Every terminal code is interpreted as the last term of a sentence, and a sequence of two terminal codes is interpreted as the end of a table.

(c) Operation Code: As indicated, each data card contains the symbolic pseudo-operation code, ALF. When processed by SOAP-2, cards bearing this operation code are reproduced as single-item load cards which will enter the numerical equivalent of the original alphabetic data in the table locations specified in the original data cards. In general, with 650-MT-1A on the drum, linguistic data enter and leave the machine in the form of single-item load cards. Transferring control of the program to location READ (= 1500) will cause such data cards to be read until either the end-of-file condition, or a transfer card, is encountered.

(d) When loading S and T data cards, the following precautions must be observed:

(1) The number of terms entered in either table should not exceed 500.

(2) The number of sentences in either table should not exceed 99.

(3) Every sentence should end with a terminal code, except the last sentence in each table, which should end with two terminal codes (to mark the end-of-table).

(4) The last data card should be followed by a transfer card to location START (= 1950). This allows the program to compile a table of first-term addresses for use with reference to the data just entered in the S and T tables.

Coming out of the table-compiling routine mentioned above, control of the program is transferred to location READ for the purpose of bringing a sentence into the X-table for translation. This means that the first (or, next) card in the hopper must be an X-data card.

When loading X data for translation by 650-MT-1A, normal operation requires that the following precautions be observed:

(5) The number of terms in the sentence to be translated should not exceed 50.

(6) Data for translation must be entered one sentence at a time, according to the definition of sentence already given. If two or more sentences are entered in the X-table, only the first of these will be processed.

(7) The last card of each X-sentence must be followed by a transfer card to location TRAN (1900). This transfers control of the program to the analysis routine, thus initiating the translation of the sentence in the X-table.

Upon completion of a translation, the program returns to location READ for its next instruction. At this point, a new sentence may be read into the X-table, or new S and T data may be loaded to supplement or replace the contents of these two tables. If no cards are present in the hopper, it is possible to restart the program at location TRAN, thus repeating the translation of the sentence just processed. (See the discussion of "Output Modes", Section 8.)

3. Self-Switching, or Input Recognition Feature.

Suppose the S and T tables contain sentences in languages A and B respectively. Then the sentence read into the X-table

for translation may be in either of these two languages, provided the following condition is met:

(8) The terminal code is the only entity appearing in both of the S and T tables.

When condition (8) is satisfied, it is possible, for instance, to translate a sentence from German into English, and then process the resulting English sentence for translation into German. When the data in the S and T tables do not satisfy condition (8), the input sentence must be in language A, where A is a language represented by one or more of the sentences in the S-table.

Automatic recognition of the input language is accomplished roughly as follows. (In this discussion it should be remembered that the relation between the S and T tables is that the x-th sentence in the T-table is processed as a translation of the x-th sentence in the S-table.)

Functionally speaking, the translation of a sentence by 650-MT-1A is accomplished by an analysis procedure followed by a synthesis, or assembly, procedure. Strictly speaking, however, the analysis procedure is a statistical comparison process in which each sentence in the S-table is compared with the one to be translated. The result of each comparison is a pair of four-place sums K which are first stored in a K-table location associated with the x-th sentence, and then added into a counter (LANG) initially set to zero. The comparison process is such that when the last sentence has been compared, the accumulated sums of the similarity constants may both be equal to zero, or to some positive whole number. The only other possibility is that the Y-sum is larger than the X-sum. At the end of the comparison process, the contents of LANG are tested. The condition Y=X indicates that, with the possible exception of the terminal code, no sentence in the S-table has anything in common with the sentence to be translated, in which case control of the program is transferred to location OTHER (= 1513), containing the first instruction of a switching routine which exchanges the contents of the S and T tables, and reverses the regional references in the table of first-term addresses. When this has been accomplished, the comparison process is re-initiated, on the assumption that the language of the input sentence is now represented in the S-table. If this is not the case, the

program will continue reversing the input-output language relation until the PROGRAM STOP button is depressed. (A simple modification would automatically stop the program after two failures of the comparison process to yield positive results.)

When the results of the comparison, or analysis, procedure are found to be useful for the purpose of translation, control is transferred to a term-selecting procedure which operates on the results of the comparison process (stored in the K-table), and the synthesis, or assembly, of a translation is begun.

4. The LK-Table.

One interesting feature of 650-MT-1A is the LK-, or Locations-Constants-table. L designates the table of first-term addresses already mentioned, while K designates the table in which are stored the results of the analysis procedure. Information is entered in corresponding locations of these two tables. Location L0005, for example, contains the regional address of the (first term of the) fifth sentence in the S-table, and also of the corresponding sentence in the T-table. (These are contained in the data- and instruction-address fields of the L-location in question.) Similarly, location K0005 will contain the result of comparing the fifth sentence in the S-table with the one currently being processed for translation. As indicated above, this result is in the form of two four-place numbers; and these, like the L-table entries, are stored according to the instruction format used by the IBM 650.

Notice that by sorting the L-table according to the contents of the K-table, it is possible (in effect) to sort the sentence-pairs in the S and T tables into any desired order without actual rearrangement of the data in these tables. For example, the table of first-term addresses could be sorted so that sentences in the T-table would be processed in order of the decreasing value of the similarity constants in the K-table. In general, any desired type of sorting keys might be entered in the K-table, and the data sentences processed in groups, or in an order, determined by these keys, e. g. , all the sentences beginning with a given term, etc. Though such sorting heuristics are not used in the present program, they could be used to advantage in applications where time requirements made it important to process the smallest possible amount of data to obtain a translation.

5. Analysis Procedure.

Each sentence in the S-table is compared with the first sentence in the X-table, as follows.

(1) Corresponding x-th terms of the two sentences are compared.
   (a) If found identical, a one is added in the units position of the D-address field of location Kxxxx (initially set to zero.)
   (b) If found not identical, (2), below, is executed.

(2) The terms just compared are tested to determine whether either one was a terminal code.
   (a) If not, a term-comparing index is set to (x+1), and (1), above, is executed.
   (b) If either term was a terminal code, the contents of Kxxxx are tested for zero. If a non-zero quantity is found, a one is added in the units position of the I-address field of Kxxxx, and (3), below, is executed. If the contents of Kxxxx are found to be zeros, (3), below, is executed.

(3) The first term of the next sentence is tested for the terminal code.
   (a) If a non-zero first term is found, the program is set to compare the next sentence, and (1) above, is executed.
   (b) An all-zero first term signifies that the last sentence has been examined by the comparison process, and (4), below, is executed.

(4) Step (4) is the input-recognition test described in a previous section. Assuming the results of that test to indicate the desired input-output relation, control of the program is transferred to the synthesis, or term-selecting, routine.

6. Synthesis Procedure.

In the synthesis procedure, the terms of the translation are selected one by one, in the order of their occurrence in the finished sentence. The selection of each term requires two operations. First there is a word-frequency routine, in which a list of possible terms is compiled and a pair of "frequencies", F and F , is accumulated for each term. This is followed by the computation of a redundancy

quotient RQ for each entity in the possible-term list. The computation performed is expressed by the following formula, which involves the values of F_y and F_x for a given term and the accummulated sums N_y and N_x of F_y and F_X for all the terms listed.

$$RQ = \frac{F_y \quad N_x}{F_x \quad N_y}$$

The word-frequency routine may be described as follows.

(1) A term-selecting index is set initially to (x=l), and the WQ-table is cleared.

(2) The x-th term of a sentence in the T-table is entered, or found, in the W-table. The similarity constant associated with that sentence is added to the contents of location Qxxxx associated with the entity in question.

(3) The same similarity constant is added to the contents of location SIZES, initially set to zero.

(4) Step (3), above, is repeated for the x-th term of every T-sentence having a non-zero similarity constant, unless the particular sentence does not contain an x-th term.

The numerical results of the above operation, stored in the Q-table, now serve as input to the redundancy-quotient routine, which replaces each pair of "frequencies" in the Q-table with a value of RQ, as described above. At the end of the process, i. e. , when each term in the W-table has been assigned an RQ, the term having the greatest redundancy quotient is punched out as an X-data load card, and the term selected is tested for the terminal code.

(5) If the term just selected was not the terminal code, then the term-selecting index is set to (x+1) , and control of the program is returned to the word-frequency routine, (2), above.

(6) If the term just selected was the terminal code, control of the program is returned to location READ, thus initiating the translation of the next input sentence.

7. Output Modes.

Two output modes are available with 650-MT-1A, and are selected at the console by setting the SIGN switch of the console to "plus", or to "minus". The condition of this switch is tested immediately

following the selection and punch-out of a term. If the sign of the console is positive, the program proceeds to the end-of-translation test already described. Thus, with the SIGN switch set to the "+" position, the only output produced is a translation of the input sentence; as indicated above, this is in the form of a deck of single-item load cards, and may be presented immediately as input data for translation by the program.

If the sign of the console is set to the "-" position, control of the program is transferred to location TABLE, the first instruction of a routine which punches a deck of cards containing the contents of the WQ-table. Each card is punched with one of the possible terms in the W-table, along with the redundancy quotient computed for that term. These cards are produced in order of the decreasing value of the redundancy quotients in the Q-table. Thus, when the sign of the console is negative, 650-MT-1A produces, in addition to each regular output card, the table of alternate choices from which the term was selected, and these are sorted into the decreasing order of their probabilities (as measured by the redundancy quotient.) Following the above table-punching routine, the program executes the end-of-translation test described in section (8).

In the present program, the WQ-table(s) contain 100 locations each. For this reason, the following precaution must be observed.

(9) In the preparation of example sentences for the S and T tables, care must be taken that the number of possible x-th terms will not exceed 100 during any one term-selecting operation.

Normally, the small size of the S and T tables will prevent any difficulties which might arise as a result of condition (9) above. However, this would not necessarily be true where it was desired to modify the program for use with magnetic tape units.

Program-decks and further information concerning 650-MT-1A will be supplied on request by the University of Texas Machine Translation Project.