[From: *Kommunikationsforschung und Phonetik* (Hamburg: Buske Verlag, 1974]

Peter Toma

Computer Translation: In its own Right

The technical feasibility of mechanical translation between natural languages was discussed as early as 1946, when, during a series of discussions, Warren Weaver suggested that all languages might contain detectable basic elements, and A. D. Booth proposed that any digital computing machine with sufficient storage could perform a dictionary translation.[1]

With the subsequent development of computers, the mid-Fifties saw the possibilities of machine translation in demonstrations of several small scale automatic translation systems. The first large scale demonstration was staged at the Pentagon on June 6, 1959. For the first time, a large text (100,000 words) was translated by computer, as the SERNA system processed Russian into English.[2]

The Pentagon demonstration, and the SERNA system itself, were significant in that they pointed up the need for a new generation of computers. (The SERNA system ran on a second generation computer.) In addition, during the development of this large-scale system capable of processing texts of 100,000 words, it had become glaringly evident that linguistic study had barely touched a good many language analysis problems, and had not even approached others. Thus, neither computers nor linguists were fully prepared for machine translation. This situation led, in several instances, to mistranslations whose inaccuracies proved humorous enough to live on as jokes among workers in this field.

For example, in August 1962, John Kouwenhoven cited, in Harper's, two attempts at English to Russian machine translation, one of which translated the aphoristic "out of sight, out of mind" as "Invisible Idiot". His other, more familiar example, was the translation of "The spirit is willing, but the flesh is weak" as "The liquor is holding out all right, but the meat has spoiled".

In 1963, the preliminary Automatic Language Processing Advisory Committee was instituted and commissioned to report on the status of machine translation. Three years later, it finally published its findings in what has come to be known as the ALPAC report.[3] The researchers had investigated the systems which grew out of the shortcomings of the Fifties; their report was permeated with negativism and concluded, by postulating that post-editing could not be allowed in their puristic definition of machine translation, that "there has been no machine translation of general scientific text, and none is in immediate prospect" (ALPAC report, page 19). "Immediate prospect" was not so meticulously defined; nonetheless, this damning conclusion sufficed to prevent, for quite some time, any further expenditure of U. S. Government funds on machine translation. Ironically, the interim between ALPAC's conception and the report's delivery also saw the birth of third generation computers.

Germany, apparently unaffected by the ALPAC report, continued to support research and development in MT. Its universities and science foundation (DFG) gave the author the opportunity to act upon his conviction that the time had come for the development and implementation of at least one generalized, large-scale computer translation system which could be used for translating between various different languages. The SYSTRAN system grew out of this belief.

The SYSTRAN concept recognizes the fact that translation is an almost infinitely complex task and that the accomplishment of this task on the computer requires an extensive system of programs. The development and final implementation of just such a system has required years of concentrated effort. Each routine, subprogram and major program had to be tested and debugged individually, yet their "meshing" or coordinated operation had to be ensured as well. The final result is a Russian-English machine translation system, for example, which is comprised of approximately 100,000 instructions functioning together like the minute components of a precision instrument. Figure 1 shows the major components of the SYSTRAN system, indicating the sequence of operation by arrows.

From a linguistic viewpoint, the most important group of programs is that which analyzes each source language sentence and determines the syntactic and semantic relationships linking the individual constituents (and groups of constituents) of the sentence together. This group of programs constitutes SYSTRAN'S recognition grammar, the parameters of which are dictated by the particular source language to be analyzed. whether the language is Russian, German or Chinese, the analysis component determines the function of each word in the sentence (e.g., object or preposition, relative clause marker, tense or aspect verb marker, etc.) and records this information using computer codes. That is, the program puts specific information in specific bytes by setting bits on or off by setting word pointers. Word pointers are set by establishing byte relationships, whereby a particular byte in the analysis area reserved for one word is supplied with the input sequence number of another word to which it is syntac-

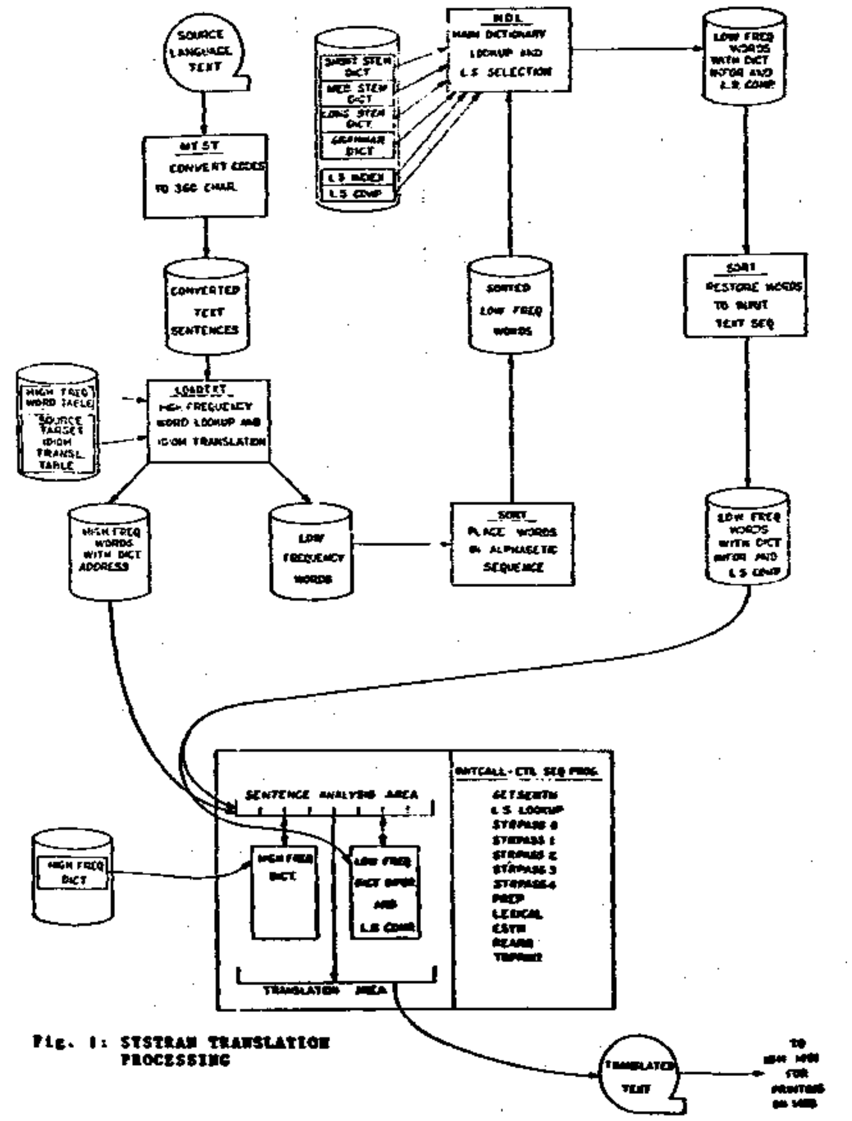
SOURCE LANGUAGE TEXT
MTST
CONVERT CODES TO 360 CHAR.
CONVERTED TEXT SENTENCES
HIGH FREQ WORD TABLE
SOURCE TARGET IDIOM TRANSL. TABLE
LOADTXT
HIGH FREQUENCY WORD LOOKUP AND IDIOM TRANSLATION
HIGH FREQ WORDS WITH DICT ADDRESS
LOW FREQUENCY WORDS
SORT
PLACE WORDS IN ALPHABETIC SEQUENCE
SORTED LOW FREQ WORDS
SHORT STEM DICT
MED STEM DICT
LONG STEM DICT
GRAMMAR DICT
L.S INDEX
L.S COMP
MDL
MAIN DICTIONARY LOOKUP AND L.S SELECTION
LOW FREQ WORDS WITH DICT INFOR AND L.S. COMP
SORT
RESTORE WORDS TO INPUT TEXT SEQ
LOW FREQ WORDS WITH DICT INFOR AND L.S COMP
SENTENCE ANALYSIS AREA
HIGH FREQ DICT
HIGH FREQ DICT
LOW FREQ DICT INFOR. AND L.S COMP
TRANSLATION AREA
ONTCALL - CTL SEQ PROG
GETSENTN
L.S LOOKUP
STRPASS 0
STRPASS 1
STRPASS 2
STRPASS 3
STRPASS 4
PREP
LEXICAL
CSTR
REARR
TRPRNT
TRANSLATED TEXT
TO IBM 1401 FOR PRINTING ON 1403

Fig. 1: SYSTRAN TRANSLATION
PROCESSING

tically connected. Conversely, the other word's analysis area will have a reciprocal byte set to point back to this word. In other words, the other word will carry this word's input sentence sequence number in a particular byte.

Regardless of the language being processed, SYSTRAN uses the sane byte relationships to express the same syntactic relationship. Thus, either a Chinese, Russian, or German Verb will point to its object in its byte 18 and its object will point back in the object's byte 28. Main verbs are linked to infinitive complement verbs by a 21-31 relationship, headwords of relative clauses are pointed to in byte 52 of either the word which functions as "which" or "who" in German and Russian or by the subject or verb of the Chinese relative clause (depending upon whether the headword serves a subject or object function in the relative clause). Subjects and predicates within each clause point to each other in a 108-111 byte relationship. In this way, the relationships of each word and each clause are expressed by the syntactic analysis programs, until the functions of all parts of each sentence in a text are accounted for. In addition, the type of clause is recorded (first main clause, second main clause, adverbial clause, conditional clause, concessive clause, etc.).

In developing computer systems to recognize, analyze, and translate Russian, Chinese and German, it became increasingly clear that transformational grammar's theory of language universals is indeed valid. The most obvious verification of this is the fact that the same types of syntactic relationships, such as those mentioned above, can be recognized in each of these languages. It is to reflect the fact that the same types of relationships exist in these different languages, that the same types of codes are used by the computer to express each type of relationship, regardless the source language.

The analysis algorithms which determine the presence of these relationships depend on contextual information (gleaned by scanning over the extent or the sentence or within individual clause boundaries). In order to recognize the elements which occur within the clause or sentence, the initial analysis programs look at the dictionary information which was attached to each word in the sentence during dictionary look-up or during the homograph resolution pass(es). A dictionary entry may consist of a stem or a full form (stem plus inflectional morph), or it may be made up of two, three, or even eight discrete lexemes which function as a grammatical unit. The latter type of entry is called a compound entry. A compound's constituents may be discontinuous; that is, its elements may be interrupted by essential or non-essential variables. The compound itself may function as any part of speech or may even be homographic.

Each dictionary entry has a set of codes either attached directly to it or accessible by an address attached to the entry. Its codes indicate its parts of speech, its proclivity for particular types of government, its influence on the translation of particular prepositions, its semantic properties, whether or not it is homographic, etc. All of this information is transferred to the text word during the particular dictionary look-up, low frequency look-up, or compound look-up.

Compound look-up recognizes the string which comprises the compound and replaces the entire string with a compound character, to which all dictionary information pertinent to the unit per se is attached, replacing the dictionary codes for each separate component. Interrupted compounds depend upon the existence of specific syntactic ties between a compound's components. Therefore, their look-up occurs separately, after the operation of sufficient syntactic analysis routines has established whether or not the appropriate syntactic links are present.

A computer dictionary is built up item by item, as human coders painstakingly fill in appropriate columns in an attempt to "tell" the computer as much as possible about each individual entry. The coder cannot rely just on his own training in both source and target language; he consults dictionaries and grammar books, special subject field references, abbreviations dictionaries and atlases. Still, he won't, most probably, be able to code everything pertinent to an entry until it has been entered in the dictionary and actually used during production runs. Only when it is processed during the translation of many different texts will the coder get the "feedback" he needs to fully code the word.

Because of these efforts, programs designed for grammatical analysis of sentences and of sub-sentence units have access to such information as each word's general function (e.g., noun, preposition, interrogative particle, verb, etc.), its specific form (genitive noun, third person plural past tense verb, present participle, literal number, ordinal number, etc.), its government requirements (e.g., "this noun exhibits a strong proclivity for adnominal genitives", or "this verb may take an instrumental object in addition to an accusative object", etc.), its semantic category (measure, motion, chemical process, physical process, etc.), and so on.

In designing programs to recognize syntactic relationships, the linguist/programmer soon finds that school grammars are hopelessly insufficient. Since the programs or routine must be designed by a human speaker-hearer (or, more appropriately in this case, a writer-reader) to recognize the form of a message conceived by another speaker-hearer, the program inevitably becomes a means of enabling a machine to replicate, to some extent, or to approximate human recognition-behavior. In this respect, Chomsky's arguments using performances as data (including the data provided by introspection[4]) become extremely important to the Russian-English linguist/programmer, who must ask himself, for example, "How do I know that this is a CEM ... TEX construction and that, here, TEM does not mean 'these' or anything like that?" Once he starts examining this problem, he realizes that this initial question leads to fifteen or twenty questions which break the problem down into logical steps. His program will consist of the proper questions about the construction; alternative answers will lead to separate subroutines within the program. In addition to establishing that a true CEM ... TEM construction (this approximates the English structure "The more, the merrier") has been encountered, the program must set certain markers indicating the relationship between the CEM clause or phrase and the TEM clause or phrase, establish relationships within clauses, and set the whole structure up for translation into an equivalent English structure. That is, the program establishes codes which will, seconds later, be acted upon by the general rearrangement and English synthesis program.

New turns in linguistic thought, theory and speculation can, at times, prove strikingly useful in linguistic programming (thereby probably becoming distressingly practical in the theoretical linguist's eyes). For example, a rather awesome impediment to the automatic translation of Chinese could have been presented by such characters as BAA and SHYY, which the Chinese speaker may insist have no English verb "meaning" (i.e., translation equivalent). Using theories of underlying structures ("depth grammar" vs. "surface grammar" or the Humboldtian "inner form" vs. "outer form"), however, BAA and SHYY can be seen to function as semi-abstract verbs whose closest English equivalent is "cause". That is, the presence of BAA and SHYY in a particular structure results in a sentence which can be predicted to mean "something/someone causes something to happen to something/someone". Thus, a reasonable linguistic analysis makes it possible to assign a dictionary meaning for this character and to program rules which will recognize its syntactic function.

Tremendous improvements in computer hardware have been equally important in making large-scale machine translation systems possible. Third generation computers are "characterized by miniaturized circuits, the integration of hardware with software (programming and operating aids), and an orientation to data communication and the handling of more than one operation simultaneously."[5] Speeds are faster and prices are lower. (Cf. Grosch's law: "the relative performance of computer systems varies according to the square of their prices".[6]) And, of course, computer memories are larger — much larger — allowing sequences of large sets of programs to operate without the interruption of writing the results of one series' sentence analysis out on file to be called in for the next set of programs to work on.

Both linguistics and computer science have now advanced far enough to establish that machine translation is a feasible approach to answering the translation requirements for the bulk of scientific and technical literature, and SYSTRAN is now being used to process newspaper texts as well. A good deal of further research is now necessary to make computer translation less an interdisciplinary challenge and more an established field in its own right.

References and Notes

1. William N. Locke and A. Donald Booth, Machine Translation of Languages. M.I.T.: 1955.

2. Peter Toma. SERNA System. Georgetown University: 1959.

3. Automatic Language Processing Advisory Committee. Language and Machines. Washington. D.C.. 1966.

4. Noam Chomsky, Aspects of the Theory of Syntax. M.I.T.: 1965. page 193.

5. Gordon B. Davis. Computer Data Processing. New York: 1969, page 64.

6. Ibid.