INVESTIGATING THE POSSIBILITY OF A MICROPROCESSOR-BASED
MACHINE TRANSLATION SYSTEM

Harold L. Somers

Centre for Computational Linguistics
University of Manchester Institute of Science and Technology
PO Box 88, Manchester M60 1QD, England

ABSTRACT

This paper describes an on-going research
project being carried out by staff and students at
the Centre for Computational Linguistics to
examine the feasibility of Machine Translation
(MT) in a microprocessor environment. The system
incorporates as far as possible features of large-
scale MT systems that have proved desirable or
effective: it is multilingual, algorithms and
data are strictly separated, and the system is
highly modular. Problems of terminological
polysemy and syntactic complexity are reduced via
the notions of controlled vocabulary and
restricted syntax. Given these constraints, it
seems feasible to achieve translation via an
'interlingua', avoiding any language-pair oriented
'transfer' stage. The paper concentrates on a
description of the separate modules in the
translation process as they are currently
envisaged, and details some of the problems
specific to the microprocessor-based approach to
MT that have so far come to light.

I. BACKGROUND AND OVERVIEW

This paper describes preliminary research in
the design of Bede, a limited-syntax controlled-
vocabulary Machine Translation system to run on a
microprocessor, translating between English,
French, German and Dutch. Our experimental corpus
is a car-radio manual. Bede (named after the 7th
Century English linguist) is essentially a
research project: we are not immediately concerned
with commercial applications, though such are
clearly possible if the research proves fruitful.
Work on Bede at this stage though is primarily
experimental. The aim at the moment is to
investigate the extent to which a microprocessor-
based MT system of advanced design is possible,
and the limitations that have to be imposed in
order to achieve a working system. This paper
describes the overall system design specification
to which we are currently working.

In the basic design of the system we attempt to
incorporate as much as possible features of large-
scale MT systems that have proved to be desirable
or effective. Thus, Bede is multilingual by
design, algorithms and linguistic data are
strictly separated, and the system is designed in
more or less independent modules.

The microprocessor environment means that
criteria of size are important: data structures

both dynamic (created by and manipulated during
the translation process) and static (dictionaries
and linguistic rule packages) are constrained to
be as economical in terms of storage space and
access procedures as possible. Limitations on in-
core and peripheral storage are important
considerations in the system design.

In large general purpose MT systems, it is
necessary to assume that failure to translate the
given input correctly is generally not due to
incorrectly formed input, but to insufficiently
elaborated translation algorithms. This is
particularly due to two problems: the lexical
problem of choice of appropriate translation
equivalents, and the strategic problem of
effective analysis of the wide range of syntactic
patterns found in natural language. The reduction
of these problems via the notions of controlled
vocabulary and restricted syntax seems
particularly appropriate in the microprocessor
environment, since the alternative of making a
system infinitely extendable is probably not
feasible.

Given these constraints, it seems feasible to
achieve translation via an Interlingua, in which
the canonical structures from the source language
are mapped directly onto those of the target
language(s), avoiding any language-pair oriented
'transfer' stage. Translation thus takes place in
two phases: analysis of source text and synthesis
of target text.

A. Incorporation of recent design principles

Modern MT system design can be characterised by
three principles that have proved to be desirable
and effective (Lehmann et al, 1980:1-3): each of
these is adhered to in the design of Bede.

Bede is multilingual by design: early MT
systems were designed with specific language-pairs
in mind, and translation algorithms were
elaborated on this basis. The main consequence of
this was that source language analysis was
effected within the perspective of the given
target language, and was therefore often of little
or no use on the addition into the system of a
further language (cf. King, 1981:12; King &
Perschke, 1982:28).

In Bede, there is a strict separation of
algorithms and linguistic data: early MT systems
were quite simply 'translation programs', and any

underlying linguistic theory which might have been present was inextricably bound up with the program itself. This clearly entailed the disadvantage that any modification of the system had to be done by a skilled programmer (cf. Johnson, 1980:140). Furthermore, the side-effects of apparently quite innocent modifications were often quite far-reaching and difficult to trace (see for example Bostad, 1982:130). Although this has only recently become an issue in MT (e.g. Vauquois, 1979:1.3; 1981:10), it has of course for a long time been standard practice in other areas of knowledge-based programming (Newell, 1973; Davis & King, 1977).

The third principle now current in MT and to be incorporated in Bede is that the translation process should be modular. This approach was a feature of the earliest 'second generation' systems (cf. Vauquois, 1975:33), and is characterised by the general notion that any complicated computational task is best tackled by dividing it up into smaller more or less independent sub-tasks which communicate only by means of a strictly defined interface protocol (Aho et al, 1974). This is typically achieved in the MT environment by a gross division of the translation process into analysis of source language and synthesis of target language, possibly with an intermediate transfer stage (see 1.D below), with these phases in turn sub-divided, for example into morphological, lexical and syntactico-semantic modules. This modularity may be reflected both in the linguistic organisation of the translation process and in the provision of software devices specifically tailored to the relevant sub-task (Vauquois, 1975:33). This is the case in Bede, where for each sub-task a grammar interpreter is provided which has the property of being no more powerful than necessary for the task in question. This contrasts with the approach taken in TAUM-Météo (TAUM, 1973), where a single general-purpose device (Colmerauer's (1970) 'O-Systems') is provided, with the associated disadvantage that for some 'simple' tasks the superfluous power of the device means that processes are seriously uneconomical. Bede incorporates five such 'grammar types' with associated individual formalisms and processors: these are described in detail in the second half of this paper.

B. The microprocessor environment

It is in the microprocessor basis that the principle interest in this system lies, and, as mentioned above, the main concern is the effects of the restrictions that the environment imposes. Development of the Bede prototype is presently taking place on Z80-based machines which provide 64k bytes of in-core memory and 720k bytes of peripheral store on two 5-1/4" double-sided double-density floppy disks. The intention is that any commercial version of Bede would run on more powerful processors with larger address space, since we feel that such machines will soon rival the popularity of the less powerful Z80's as the standard desk-top hardware. Programming so far has been in Pascal-M (Sorcim, 1979), a Pascal

dialect closely resembling UCSD Pascal, but we are conscious of the fact that both C (Kernighan & Ritchie, 1978) and BCPL (Richards & Whitby-Strevens, 1979) may be more suitable for some of the software elements, and do not rule out completing the prototype in a number of languages. This adds the burden of designing compatible data-structures and interfaces, and we are currently investigating the relative merits of these languages. Portability and efficiency seem to be in conflict here.

Microprocessor-based MT contrasts sharply with the mainframe-based activity, where the significance of problems of economy of storage and efficiency of programs has decreased in recent years. The possibility of introducing an element of human interaction with the system (cf. Kay, 1980; Melby, 1981) is also highlighted in this environment. Contrast systems like SYSTRAN (Toma, 1977) and GETA (Vauquois, 1975, 1979; Boitet & Nedobejkine, 1980) which work on the principle of large-scale processing in batch mode.

Our experience so far is that the economy and efficiency in data-structure design and in the elaboration of interactions between programs and data and between different modules is of paramount importance. While it is relatively evident that large-scale MT can be simulated in the microprocessor environment, the cost in real time is tremendous: entirely new design and implementation strategies seem to be called for. The ancient skills of the programmer that have become eroded by the generosity afforded by modern mainframe configurations become highly valued in this microprocessor application.

C. Controlled vocabulary and restricted syntax

The state of the art of language processing is such that the analysis of a significant range of syntactic patterns has been shown to be possible, and by means of a number of different approaches. Research in this area nowadays is concentrated on the treatment of more problematic constructions (e.g. Marcus, 1980). This observation has led us to believe that a degree of success in a small scale MT project can be achieved via the notion of restricting the complexity of acceptable input, so that only constructions that are sure to be correctly analysed are permitted. This notion of **restricted syntax** MT has been tried with some success in larger systems (cf. Elliston, 1979; Lawson, 1979:81f; Somers & McNaught, 1980:40), resulting both in more accurate translation, and in increased legibility from the human point of view. As Elliston points out, the development of strict guidelines for writers leads not only to the use of simpler constructions, but also to the avoidance of potentially ambiguous text. In either case, the benefits for MT are obvious. Less obvious however is the acceptability of such constraints; yet 'restricted syntax' need not imply 'baby talk', and a reasonably extensive range of constructions can be included.

Just as problems of syntactic analysis can be alleviated by imposing some degree of control over

the syntactic complexity of the input, so the corresponding problem of lexical disambiguation that large-scale MT systems are faced with can be eased by the notion of controlled vocabulary. A major problem for MT is the choice of appropriate translation equivalents at the lexical level, a choice often determined by a variety of factors at all linguistic levels (syntax, semantics, pragmatics). In the field of multilingual terminology, this problem has been tackled via the concept of terminological equivalence (Wüster, 1971): for a given concept in one language, a translation in another language is established, these being considered by definition to be in one-to-one correspondence. In the case of Bede, where the subject-matter of the texts to be translated is fixed, such an approach for the 'technical terms' in the corpus is clearly feasible; the notion is extended as far as possible to general vocabulary as well. For each concept a single term only is permitted, and although the resulting style may appear less mature (since the use of near synonyms for the sake of variety is not permitted), the problems described above are somewhat alleviated. Polysemy is not entirely avoidable, but if reduced to a bare minimum, and permitted only in specific and acknowledged circumstances, the problem becomes more easily manageable.

D. Interlingua

A significant dichotomy in MT is between the 'transfer' and 'interlingua' approaches. The former can be characterised by the use of bilingual transfer modules which convert the results of the analysis of the source language into a representation appropriate for a specific target language. This contrasts with the interlingua approach in which the result of analysis is passed directly to the appropriate synthesis module.

It is beyond the scope of the present paper to discuss in detail the relative merits of the two approaches (see Vauquois, 1975:142ff; Hutchins, 1978). We should however consider some of the major obstacles inherent in the interlingua approach.

The development of an Interlingua for various purposes (not only translation) has been the subject of philosophical debate for some years, and proposals for MT have included the use of formalized natural language (e.g. Mel'čuk, 1974; Andreev, 1967), artificial languages (like Esperanto), or various symbolic representations, whether linear (e.g. Bölting, 1961) or otherwise (e.g. Wilks, 1973). Most of these approaches are problematic however (for a thorough discussion of the interlingua approach to MT, see Otten & Pacak (1971) and Barnes (1983)). Nevertheless, some interlingua-based MT systems have been developed to a considerable degree: for example, the Grenoble team's first attempts at MT took this approach (Veillon, 1968), while the TITUS system still in use at the Institut Textile de France (Ducrot, 1972; Zingel, 1978) is claimed to be interlingua-based.

It seems that it can be assumed a priori that an entirely language-independent theoretical representation of a given text is for all practical purposes impossible. A more realistic target seems to be a representation in which significant syntactic differences between the languages in question are neutralized so that the best one can aim for is a languages-specific (sic) representation. This approach implies the definition of an Interlingua which takes advantage of anything the languages in the system have in common, while accomodating their idiosyncrasies. This means that for a system which involves several fairly closely related languages the interlingua approach is at least feasible, on the understanding that the introduction of a significantly different type of language may involve the complete redefinition of the Interlingua (Barnes, 1983). From the point of view of Bede, then, the common base of the languages involved can be used to great advantage. The notion of restricted syntax described above can be employed to filter out constructions that cause particular problems for the chosen Interlingua representation.

There remains however the problem of the representation of lexical items in the Interlingua. Theoretical approaches to this problem (e.g. Andreev, 1967) seem quite unsatisfactory. But the notion of controlled vocabulary seems to offer a solution. If a one-to-one equivalence of 'technical' terms can be achieved, this leaves only a relatively small area of vocabulary for which an interlingual representation must be devised. It seems reasonable, on a small scale, to treat general vocabulary in an analagous way to technical vocabulary, in particular treating lexical items in one language that are ambiguous with respect to any of the other languages as 'homographs'. Their 'disambiguation' must take place in Analysis as there is no biligual 'Transfer' phase, and Synthesis is purely deterministic. While this approach would be quite unsuitable for a large-scale general purpose MT system, in the present context – where the problem can be minimised – it seems to be a reasonable approach.

Our own model for the Bede Interlingua has not yet been finalised. We believe this to be an area for research and experimentation once the system software has been more fully developed. Our current hypothesis is that the Interlingua will take the form of a canonical representation of the text in which valency-boundness and (deep) case will play a significant role. Sentential features such as tense and aspect will be captured by a 'universal' system of values for the languages involved. This conception of an Interlingua clearly falls short of the language-independent pivot representation typically envisaged (cf. Boitet & Nedobejkine, 1980:2), but we hope to demonstrate that it is sufficient for the languages in our system, and that it could be adapted without significant difficulties to cater for the introduction of other (related) Western European languages. We feel that research in this area will, when the time comes, be a significant

and valuable by-product of the project as a whole.

II. DESCRIPTION OF THE SYSTEM DESIGN

In this second half of the paper we present a description of the translation process in Bede, as it is currently envisaged. The process is divided broadly into two parts, analysis and synthesis, the interface between the two being provided by the Interlingua. The analysis module uses a Chart-like structure (cf. Kaplan, 1973) and a series of grammars to produce from the source text the Interlingua tree structure which serves as input to synthesis, where it is rearranged into a valid surface structure for the target language. The 'translation unit' (TU) is taken to be the sentence, or equivalent (e.g. section heading, title, figure caption). Full details of the rule formalisms are given in Somers (1981).

A. String segmentation

The TU is first subjected to a two-stage string-segmentation and 'lemmatisation' analysis. In the first stage it is compared word by word with a 'stop-list' of frequently occurring words (mostly function words); words not found in the stop-list undergo string-segmentation analysis, again on a word by word basis. String-segmentation rules form a finite-state grammar of affix-stripping rules ('A-rules') which handle mostly inflectional morphology. The output is a Chart with labelled arcs indicating lexical unit (LU) and possible interpretation of the stripped affixes, this 'hypothesis' to be confirmed by dictionary look-up. By way of example, consider (1), a possible French rule, which takes any word ending in -issons (e.g. finissons or hérissons) and constructs an arc on the Chart recording the hypothesis that the word is an inflected form of an '-ir' verb (i.e. finir or *hérir).

(1) $V + "-ISSONS" \rightarrow V + "-IR"$
     [PERS=1 & NUM=PLUR & TENSE=PRES & MOOD=INDIC]

At the end of dictionary look-up, a temporary 'sentence dictionary' is created, consisting of copies of the dictionary entries for (only) those LUs found in the current TU. This is purely an efficiency measure. The sentence dictionary may of course include entries for homographs which will later be rejected.

B. Structural analysis

1. 'P-rules'

The chart then undergoes a two-stage structural analysis. In the first stage, context-sensitive augmented phrase-structure rules ('P-rules') work towards creating a single arc spanning the entire TU. Arcs are labelled with appropriate syntactic class and syntactico-semantic feature information and a trace of the lower arcs which have been subsumed from which the parse tree can be simply extracted. The trivial P-rule (2) is provided as an example.

(2) $<NUM(DET)=NUM(N)$ & $GDR(DET).INT.GDR(N) \neq \{ \}>$
     $DET + N \rightarrow NP$
     $<GDR(NP):=GDR(N)$ & $NUM(NP):=NUM(N)>$

P-rules consist of 'condition stipulations', a 'geometry', and 'assignment stipulations'. The nodes of the Chart are by default identified by the value of the associated variable CLASS, though it is also possible to refer to a node by a local variable name and test for or assign the value of CLASS in the stipulations. Our rule formalisms are quite deliberately designed to reflect the formalisms of traditional linguistics.

This formalism allows experimentation with a large number of different context-free parsing algorithms. We are in fact still experimenting in this area. For a similar investigation, though on a machine with significantly different time and space constraints, see Slocum (1981).

2. 'T-rules'

In the second stage of structural analysis, the tree structure implied by the labels and traces on these arcs is disjoined from the Chart and undergoes general tree-to-tree-transductions as described by 'T-rules', resulting in a single tree structure representing the canonical form of the TU.

The formalism for the T-rules is similar to that for the P-rules, except in the geometry part, where **tree structures** rather than **arc sequences** are defined. Consider the necessarily more complex (though still simplified) example (3), which regularises a simple English passive.

(3) $< LU(AUX)="BE"$ & $PART(V)=PASTPART$ &
         $LU(PREP)="BY"$ & $CASE(NP\{2\})=AGENT>$
         $S(NP\{1\} + AUX + V + NP\{2\}(PREP + S))$
         $\rightarrow S(NP\{2\}(S) + V + NP\{1\})$
         $<DSF(NP\{2\}):=DSUJ$ & $VOICE(V):=PASSV$ &
                 $DSF(NP\{1\}):=DOBJ >$

Notice the necessity to 'disambiguate' the two NPs via curly-bracketted disambiguators; the possibility of defining a partial geometry via the 'dummy' symbol ($); and how the AUX and PREP are eliminated in the resulting tree structure. Labellings for nodes are copied over by default unless specifically suppressed.

With source-language LUs replaced by unique multilingual-dictionary addresses, this canonical representation is the Interlingua which is passed for synthesis into the target language(s).

C. Synthesis

Assuming the analysis has been correctly performed, synthesis is a relatively straight-forward deterministic process. Synthesis commences with the application of further T-rules which assign new order and structure to the Interlingua as appropriate. The synthesis T-rules for a given language can be viewed as **analogues** of the T-rules that are used for analysis of that language, though it is unlikely that for synthesis

the analysis rules could be simply reversed.

Once the desired structure has been arrived at, the trees undergo a series of context-sensitive rules used to assign mainly syntactic features to the leaves ('L-rules'), for example for the purpose of assigning number and gender concord (etc.). The formalism for the L-rules is again similar to that for the P-rules and T-rules, the geometry part this time defining a single tree structure with no structural modification implied. A simple example for German is provided here (4).

(4)   <SF(NP)=SUBJ>
              NP(DET + N)
         <CASE(DET):=NOM & CASE(N):=NOM &
         NUM(DET):=NUM(NP) & GDR(DET):=GDR(N)>

The list of labelled leaves resulting from the application of L-rules is passed to morphological synthesis (the superior branches are no longer needed), where a finite-state grammar of morphographemic and affixation rules ('M-rules') is applied to produce the target string. The formalism for M-rules is much less complex than the A-rule formalism, the grammar being again straightforwardly deterministic. The only taxing requirement of the M-rule formalism (which, at the time of writing, has not been finalised) is that it must permit a wide variety of string manipulations to be described, and that it must define a transparent interface with the dictionary. A typical rule for French for example might consist of stipulations concerning information found both on the leaf in question and in the dictionary, as in (5).

(5) leaf info.: CLASS=V; TENSE=PRES; NUM=SING;
                 PERS=3; MOOD=INDIC
     dict. info.: CONJ(V)=IRREG
     assign: Affix "-T" to STEM1(V)

D. General comments on system design

The general modularity of the system will have been quite evident. A key factor, as mentioned above, is that each of these grammars is just powerful enough for the task required of it: thus no computing power is 'wasted' at any of the intermediate stages.

At each interface between grammars only a small part of the data structures used by the donating module is required by the receiving module. The 'unwanted' data structures are written to peripheral store to enable recovery of partial structures in the case of failure or mistranslation, though automatic backtracking to previous modules by the system as such is not envisaged as a major component.

The 'static' data used by the system consist of the different sets of linguistic rule packages, plus the dictionary. The system essentially has one large multilingual dictionary from which numerous software packages generate various subdictionaries as required either in the translation process itself, or for lexicographers

working on the system. Alphabetical or other structured language-specific listings can be produced, while of course dictionary updating and editing packages are also provided.

The system as a whole can be viewed as a collection of Production Systems (PSs) (Newell, 1973; Davis & King, 1977; see also Ashman (1982) on the use of PSs in MT) in the way that the rule packages (which, incidentally, as an efficiency measure, undergo separate syntax verification and 'compilation' into interpretable 'code') operate on the data structure. The system differs from the classical PS setup in distributing its static data over two databases: the rule packages and the dictionary. The combination of the rule packages and the dictionary, the software interfacing these, and the rule interpreter can however be considered as analgous to the rule interpreter of a classical PS.

III. CONCLUSION

As an experimental research project, Bede provides us with an extremely varied range of computational linguistics problems, ranging from the principally linguistic task of rule-writing, to the essentially computational work of software implementation, with lexicography and terminology playing their part along the way.

But we hope too that Bede is more than an academic exercise, and that we are making a significant contribution to applied computational linguistics research.

IV. ACKNOWLEDGMENTS

V. REFERENCES

Aho, Alfred V., John E. Hopcroft & Jeffrey D. Ullman. The design and analysis of computer algorithms. Reading, Mass.: Addison-Wesley, 1974.

Andreev, N.D. The intermediary language as the focal point of machine translation. In A.D. Booth (ed), Machine Translation, Amsterdam: North-Holland, 1967, 1-27.

Ashman, Beverley D. Production Systems and their application to Machine Translation: Transfer Report (CCL/UMIST Report No. 82/9). Manchester: Centre for Computational Linguistics, University of Manchester Institute of Science and Technology, 1982.

153

Barnes, Antonia M.N. An investigation into the syntactic structures of abstracts, and the definition of an 'interlingua' for their translation by machine. MSc thesis. Centre for Computational Linguistics, University of Manchester Institute of Science and Technology, 1983.

Boitet, C. & N. Nedobejkine. Russian-French at GETA: Outline of the method and detailed example (Rapport de Recherche No. 219). Grenoble: GETA, 1980.

Bölting, Rudolph. A double intermediate language for Machine Translation. In Allen Kent (ed), Information Retrieval and Machine Translation, Part 2 (Advances in Documentation and Library Science, Volume III), New York: Interscience, 1961, 1139-1144.

Bostad, Dale A. Quality control procedures in modification of the Air Force Russian-English MT system. In Veronica Lawson (ed), Practical Experience of Machine Translation, Amsterdam: North-Holland, 1982, 129-133.

Colmerauer, Alain. Les systèmes-Q: ou un formalisme pour analyser et synthétiser des phrases sur ordinateur. (Publication interne no. 43). Montréal: Projet de Traduction Automatique de l'Université de Montréal, 1970.

Davis, Randall & Jonathan King. An overview of Production Systems. In E.W. Elcock & D. Michie (eds), Machine Intelligence Volume 8: Machine representation of knowledge, New York: Halsted, 1977, 300-332.

Ducrot, J.M. Research for an automatic translation system for the diffusion of scientific and technical textile documentation in English speaking countries: Final report. Boulogne-Billancourt: Institut Textile de France, 1972.

Elliston, John S.G. Computer aided translation: a business viewpoint. In Barbara M. Snell (ed) - Translating and the computer, Amsterdam: North-Holland, 1979, 149-158.

Johnson, Rod. Contemporary perspectives in machine translation. In Suzanne Hanon & Vigge Hjørneger Pedersen (eds), Human translation machine translation (Noter og Kommentarer 39). Odense: Romansk Institut, Odense Universitet, 1980, 133-147.

Hutchins, W.J. Machine translation and machine aided translation. Journal of Documentation, 1978, 34:119-159.

Kaplan, Ronald M. A general syntactic processor. In Randall Rustin (ed), Natural Language Processing (Courant Computer Symposium 8), New York: Algorithmics Press, 1973, 193-241.

Kay, Martin. The proper place of men and machines in language translation (Report No. CSL-80-11). Palo Alto, CA: Xerox, 1980.

Kernighan, Brian W. & Dennis M. Ritchie. The C programming language. Englewood Cliffs, NJ: Prentice-Hall, 1978.

King, M. EUROTRA - a European system for machine translation. Lebende Sprachen 1981, 26:12-14.

King, M. & S. Perschke. EUROTRA and its objectives. Multilingua, 1982, 1:27-32.

Lawson, Veronica. Tigers and polar bears, or: translating and the computer. The Incorporated Linguist, 1979, 18:81-85.

Lehmann, Winfred P., Winfield S. Bennett. Jonathan Slocum, Howard Smith, Solveig M.V. Pfluger & Sandra A. Eveland. The METAL system (RADC-TR-80-374). Austin, TX: Linguistics Research Center, University of Texas, 1980.

Marcus, Mitchell P. A theory of syntactic recognition for natural language, Cambridge, MA: MIT Press, 1980.

Melby, Alan K. Translators and machines - can they cooperate? META, 1981, 26:23-34.

Mel'čuk, I.A. Grammatical meanings in interlinguas for automatic translation and the concept of grammatical meaning. In V. Ju. Rozencvejg (ed), Machine Translation and Applied Linguistics, Volume I, Frankfurt am Main: Athenaion Verlag, 1974, 95-113.

Newell, A. Production systems: Models of control structures. In William G. Chase (ed) - Visual information processing, New York: Academic Press, 1973, 463-526.

Otten, Michael & Milos G. Pacak. Intermediate languages for automatic language processing. In Julius T. Tou (ed), Software Engineering: COINS III, Volume 2, New York: Academic Press, 1971, 105-118.

Richards, Martin & Colin Whitby-Strevens. BCPL - the language and its compiler. Cambridge: Cambridge University Press, 1979.

Slocum, Jonathan. A practical comparison of parsing strategies for Machine Translation and other Natural Language Processing Purposes. PhD dissertation. University of Texas at Austin, 1981. [= Technical Report NL-41, Department of Computer Sciences, The University of Texas, Austin, TX.]

Somers, H.L. Bede - the CCL/UMIST Machine Translation system: Rule-writing formalism (3rd revision) (CCL/UMIST Report No. 81-5). Manchester: Centre for Computational Linguistics, University of Manchester Institute of Science and Technology, 1981.

Somers, H.L. & J. McNaught. The translator as a computer user. The Incorporated Linguist, 1980, 19:49-53.

Sorcim. Pascal/M user's reference manual. Walnut Creek, CA: Digital Marketing, 1979.

TAUM. Le système de traduction automatique de l'Université de Montréal (TAUM). META, 1973, 18:227-289.

Toma, P.P. SYSTRAM as a multilingual Machine Translation system. In Commission of the European Communities, Third European Congress on Information Systems and Networks: Overcoming the language barrier, Volume 1. München: Verlag Dokumentation, 1977, 569-581.

Vauquois, Bernard. La traduction automatique à Grenoble (Documents de Linguistique Quantitative 24), Paris: Dunod, 1975.

Vauquois, B. Aspects of mechanical translation in 1979 (Conference for Japan IBM Scientific Program). Grenoble: Groupe d'Etudes pour la Traduction Automatique, 1979.

Vauquois, Bernard. L'informatique au service de la traduction. META, 1981, 26:8-17.

Veillon, G. Description du langage pivot du système de traduction automatique du C.E.T.A. T.A. Informations, 1968, 1:8-17.

Wilks, Yorick. An Artificial Intelligence approach to Machine Translation. In Roger C. Schank & Kenneth Mark Colby (eds), Computer models of thought and language, San Francisco: Freeman, 1973, 114-151.

Wüster, Eugen. Begriffs- und Themaklassifik-ationen: Unterschiede in ihrem Wesen und in ihrer Anwendung. Nachrichten für Dokument-ation, 1971, 22:98-104.

Zingel, Hermann-Josef. Experiences with TITUS II. International Classification, 1978, 5:33-37.