Strategies for Interactive Machine Translation: the experience and implications of the UMIST Japanese project *

P. J. Whitelock, M. McGee Wood, B. J. Chandler,
N. Holden, and H. J. Horsfall

Centre for Computational Linguistics
University of Manchester Institute of Science and Technology
P. O. Box 88, Manchester M60 1QD
United Kingdom

1. Introduction

At the Centre for Computational Linguistics, we are designing and implementing an English-to-Japanese interactive machine translation system. The project is funded jointly by the Alvey Directorate and International Computers Limited (ICL). The prototype system runs on the ICL PERQ, though much of the development work has been done on a VAX 11/750. It is implemented in Prolog, in the interests of rapid prototyping, but intended for optimization. The informing principles are those of modern complex-feature-based linguistic theories, in particular Lexical-Functional Grammar (Bresnan (ed.) 1982, Kaplan and Bresnan 1982), and Generalized Phrase Structure Grammar (Gazdar et al. 1985).

For development purposes we are using an existing corpus of 10,000 words of continuous prose from the PERQ's graphics documentation; in the long term, the system will be extended for use by technical writers in fields other than software. At the time of writing, we have well-developed system development software, user interface, and grammar and dictionary handling utilities. The English analysis grammar handles most of the syntactic structures of the corpus, and we have a range of formats for output of linguistic representations and Japanese text. A transfer grammar for English-Japanese has been prototyped, but is not not yet fully adequate to handle all constructions in the corpus; a facility for dictionary entry in kanji is incorporated. The aspect of the system we will focus on in the present paper is its interactive nature, discussing the range of different types of interaction which are provided or permitted for different types of user.

2. The relationship between human and machine

The complexity of the translation task, and the diversity of the knowledge types involved, makes the implementation of an MT system a research problem in knowledge engineering. In order for intermediate results to be of practical value, it is necessary to integrate human expertise into the machine translation process. After this input, the machine's knowledge must be complete, adequate for carrying out all later stages in the translation chain. Three strategies are recognized for this involvement, post-editing, pre-editing, and interactive translation (see further Whitelock (ed.) 1985).

Of these three strategies, post-edited translation is obviously the safest; the human has the final say, and so can correct any translation errors made by the machine. However, the post-editor must be expert in the source language (at least for the text-type), target language, and subject matter, i.e. a competent translator in his/her own right. The majority of current machine translation systems are of this type - it has proved to be cost effective - but post-editing is both repetitive and totally indispensable. Current interactive systems, too, typically require a high degree of bilingual competence on the human's part.

The pre-editing option assumes input which has been either drafted or edited to use only a restricted sub-language. Despite its demonstrated utility (e.g. Meteo (Chevalier et al. 1978), Xerox Systran (Ruffino 1982)), it is often viewed as a poor alternative, conveying connotations of severe restrictions on input. We believe that the notion of pre-editing should be reexamined in the light of Kay's remark that machine translation is in principle possible only between formal languages (fn 1). Any MT system will produce intelligible output from only a proper subset of texts in what is claimed to be the source language. Translation of a natural language in its entirety is an illusory goal. If the particular subset which can be automatically translated is not formalised independently of its instantiation as a computer system, the entire output must be checked after translation, in conjunction with the source text. We believe that it is preferable to define explicitly the subset of the source language that the machine can translate. Moreover, we believe that the drafting of texts in such a restricted language is an ideal candidate for automation.

We have therefore designed a system to effect monolingual interactive pre-editing for pre-edited translation, questioning the human user about the source text, not about its translation (cf. Johnson & Whitelock 1985). An interactive system of this type should be able to accept an adequate subset of grammatically well-formed input, querying the user to resolve ambiguities or indeterminacies until a representation is reached which is sufficiently detailed and precise to guarantee acceptable translation to a given language (fn 2). A monolingual writer can thus produce finished target language text wihout further human intervention. As an incidental benefit, s/he can be expected to learn from experience which sentence patterns will not be accepted and which will give many ambiguities, and avoid them. Indeed it could be argued that such a system is valuable even monolingually, as an 'intelligent' style checker, with translation capability as an incidental benefit.

Thus we do not concur with Slocum's (1985) assessment of the potential of interaction. Slocum characterizes interactive machine translation as "Human Assisted Machine Translation" as distinct from 'full' MT, on the grounds that the exploitation of the bilingual expertise of a human is interleaved with that of the machine. It appears that the machine is not carrying out the task of translation automatically.

However, it seems more appropriate to consider the complete chain of processes from source language text composition to target language text completion. From this perspective, any type of machine translation is human assisted. It is important to assess the quantity and character of human intervention as well as its position in the system. A post-edited system resigns itself from the start to inadequacy, building in the requirement for (more or less) radical human revision of its output, so that it might better be called pre-translation than translation proper; while many current pre-editing systems, although offering fully automatic production of target language text from source language text, require a human contribution in the pre-input stage, controlling and restricting that source text, which qualitatively far exceeds the demands of on-line interaction (fn 3).

Moreover, the future of machine translation, and natural language processing in general, seems certain to lie with systems based on AI techniques such as the use of inference for ambiguity resolution. A primary consideration will be to facilitate the transfer of expertise from human to machine, by means of modular programming, knowledge engineering techniques, and, ultimately, machine learning. Interactive system design may well be the type most readily extended to incorporate such techniques as they are developed; the forms of interaction implemented for the present human user can be progressively delegated to a virtual or machine "user". Thus despite the admitted limitations of interactive translation per se, systems including some sort of interaction offer both the most efficient use of current resources and the most convincing basis and model for research aimed at greatly improving translation quality.

3. System development tools

With the aim of producing a tool for continuing research as well as a system of practical utility, we have conceived our translation system itself and the system development tools as an integrated entity. Recognising a variety of types of 'user', from end-user to system designer, our development system is organized as a tree of facilities, where different types of user are allowed or offered access at different levels. Facilities supported include:

1) using the system for writing, editing and translating,
2) writing the grammars,
3) developing the grammatical theory and the translation chain,
4) designing the system itself.

All tasks are carried out by traversing the tree under the guidance of menus. The menu system is designed to allow non-programmers (fn 4) to specify both the conversion of linguistic data to menus, and the interpretation of menu choices. This provides the organisation necessary to control:

a) different views of the data by different users, and

b) different processing/compilation of data according to its type.

Corresponding to the four types of task given above, we recognise four (idealised) types of user:

1) the end user,
2) the high-level linguist (grammar writer),
3) the low-level linguist (grammar designer),
4) the system designer.

They have access rights as follows:

1) The end user will be a monolingual (English) technical writer, with expert knowledge of the technical field and its terminology, but no knowledge of the target language. (For development purposes we are working with extant texts chosen to be typical of their kind; but the intention is to provide the writer with a tool for the initial composition of such texts. The end user will thus be able to respond more flexibly to the system, and make better use of its facilities, than we can ourselves do at the moment; although we intend it to produce its own documentation.) The facilities available to the end user will include:

a) standard monolingual text/document processing facilities,

b) on-line monolingual dictionary update, into "supplementary" dictionary files for later bilingual completion and full incorporation by some lower-level user,

c) tree-structured document organisation, with an associated dictionary structure, that handles terminology, including proper (e.g. procedure) names, at different levels of generality. This is important from both monolingual and bilingual perspectives. Monolingually, it provides a basis for indexing, document retrieval, glossary production, spelling checks etc. Bilingually, in terms of translation into Japanese, these distinctions map well onto orthographic conventions: general vocabulary is represented in kanji (ideographic) script, Japanese technical vocabulary from foreign languages in kana (Japanese syllabic) script, and proper nouns such as procedure names are simply carried over in Roman (alphabetic) script.

2) The second level of user, the high-level linguist, is responsible for writing the rules which compute well-formed sets of feature specifications (e.g. F-structures) from other sets, for source language analysis, transfer, and target language synthesis. A variety of rule types could be provided for these purposes. The system as implemented supports the following:

a) dictionary entries, which specify idiosyncratic information of all kinds. These define a mapping between lexemes and (underdetermined) sets of feature specifications, for analysis, and between pairs of sets of feature specifications, for transfer.

b) Context-free rules, augmented in the manner of LFG or PATR II (Shieber 1984), that is, regular expressions over strings of sets of feature specifications (i.e. lexical entries) that define a mapping to hierarchical (dependency) representations.

c) recursively structured sets of features, permitting grouping of features along various dimensions, e.g. noun/verb, syntactic/semantic, lexical/phrasal, etc.

d) feature co-occurrence restrictions (FCRs) in the manner of GPSG. These can be used

to define sets of features as mutually exclusive, and to specify that certain feature values necessarily follow from the presence or values of other features.

e) default values of lexical feature specifications.

f) rules which determine (surface) relational labels on the basis of feature values. By recognising these as a distinct rule type, we make a substantive claim equivalent to the formal claim embodied in the typical LFG analysis of prepositional phrases - that the value of a prepositional feature becomes the relation of the (F-structure of the) PP to (that of) its head.

g) subcat rules, which relate subcategorisation features from the lexicon to function-argument (deep to surface) mappings. Such features may be specified in a lexical entry itself, or filled in by FCR. Function-argument mappings are merely pairs of atoms, that is, each constituent is mapped independently of its sisters. This allows us to define efficient control strategies over these rules for use in either direction - analysis or synthesis. The control strategy, in combination with the uniqueness condition on f-structures, embodies the function-argument biuniqueness principle of Bresnan (1982,p163), and realises the coherence condition. Completeness is specified independently, or, in the case of Japanese, not at all.

The task of the high-level linguist is to define particular instances of rules of these types, for the purpose of describing texts in a particular language and the relationship between texts in a particular pair of languages.

3) The low-level linguist (the grammar designer) is responsible for defining the formalism, or metatheory, in which the high-level linguist will work (cf. Shieber 1985). This includes:

a) defining the nature and number of levels of text representation during the translation process. This includes deciding between phrase structure and dependency representations, the partition between the lexical and syntactic components in analysis, and the partition between analysis and transfer.

b) defining the rule types to reflect these partitions and specify the mappings between levels.

c) definition of the declarative semantics of the operators by which the rules of various types are applied, and thence the definition of the compilation of linguistic information.

4) The system designer is concerned with both the highest and lowest levels of the system: on the one hand, with the most general overall considerations of system architecture, including the points and types of interaction; on the other, with ensuring that the metalinguistic decisions made by the grammar designer have a tractable computational realisation (procedural semantics) and can be integrated with interaction.

For each level of user, there is a different, appropriate correspondence of privilege and responsibility; each should be able to use his/her specialist competence (monolingual technical writing, the writing of grammars, the design of grammatical theory, and the design of knowledge-based systems) with the full benefit of all facilities defined at lower levels, without concern for the particular details of either lower-level implementation or higher-level application.

Further corresponding to these four classes of user, the CCL Japanese system incorporates interactive routines of variable 'user-friendliness' appropriate to the presumed competence of a user at that level.

The linguist, for example, can enter the grammar to revise or expand it. The menu of the "Grammar Development System" at which the high level linguist is rooted provides the option

Edit a file of linguistic data.

This takes one to a nested menu which provides the alternatives:

1. lexicon
2. grammar
3. feature system

Selecting "2" takes one to a further menu controlling the editing of the augmented phrase structure rules, from which one can choose the appropriate rule, e.g.

1. sentence
2. noun_phrase
3. verb_phrase

...

M. remove nonterminal category
N. add new nonterminal category

Controlling access to linguistic information by means of menu ensures that the updated files are appropriately recompiled into the form used by the program. For instance, when the grammar writer updates the definition of feature sets and/or FCRs, a compilation process is initiated. This expands the tree structure into an extensional definition of the procedure which adds a feature specification to an f-structure. This realises f-structure (graph) unification as Prolog (term) unification, greatly enhancing the efficiency of parsing.

An example of variable interaction strategies for different types of user is provided by the dictionary creation and update procedures. The dictionary can be modified either en bloc for systematic large-scale creation or expansion, typically by the high-level linguist; or on-line, during analysis, by an end-user.

The linguist will enter dictionary creation by taking first the top level menu option "edit a file of linguistic data", then the option "lexicon". The end user, when s/he enters a word for which there is no existing dictionary entry, is offered the menu options "proper noun", "misspelling", "enter dictionary creation".

The dictionary creation process is driven by the tree-structured menu system, rooted in the choice of grammatical category; and here again different procedures are available for different predicted levels of use. It is presumed that closed class words such as determiners, quantifiers, and conjunctions will only be added by the linguist; therefore, when such classes are selected, the user sees lexical entries exactly as interpreted during translation.

For open class words, on the other hand, where update by the end-user is the norm, interactive routines are provided. In these cases the user never sees the program form. Questions are asked enabling the system to determine the appropriate inflectional paradigm: syntactic category, mass (i.e. without a plural form) or count, etc. Plausible surface forms are then generated by a morphological component, and presented to the user for confirmation or correction. The same component is used during morphological analysis. Thus if the user confirms the machine-generated form, the machine will be able to analyse that form when it appears in a text, and need not store it.

The syntactic and semantic feature specifications for new dictionary entries are also built up interactively. Where reasonable defaults can be established, these are presented to the user for confirmation or override. Verbs, for example, are assumed to be transitive unless the user exercises the option to specify some other valency pattern; nouns are assumed to be countable. Where a value is less predictable, the user is simply asked to provide it: does a given noun denote a physical object, a software object, or an abstraction? Verbs are described within a modified Vendler/Dowty-type classification (Vendler 1967, Dowty 1979, Steedman & Moens 1986); the user is asked to specify the appropriate value from a set including state, activity, achievement, and accomplishment.

The menu interpreter creates and stores from this input a dictionary entry in a neutral format. Subsequently, this is compiled to a program form entry. The new entries thus created are not added immediately to the master files, but are held in supplementary files, where they are available to the system, but also clearly isolated for the high-level linguist, who will eventually add translation equivalents (of which the end-user is presumed to be ignorant) and incorporate the completed entries into the master dictionary(s).

The creation of an intermediate neutral-form dictionary offers a facility for global revision of the program form of the complete dictionaries. The neutral form and program form are related by FCRs which embody generalisations about the syntactic behaviour of various lexical features. For instance, the count/mass feature on noun entries is related to two features in the program form specifying the values for number agreement and coocurrence with an article. The low-level linguist need only change such facts and recompile the relevant neutral form files to generate a new program-form dictionary.

Currently, the dictionary creation menu system must be written by hand. We are experimenting with the possibility of constructing it automatically from the feature system.

4. Interactive disambiguation

The two principal considerations relevant to interactive disambiguation are at what point it should take place, and what form it should take. We will discuss these in order.

1) Where should disambiguation take place?

One answer to the question of when to interactwith a user is: at different points according to the type of ambiguity. We believe that this is not the correct answer, but that all ambiguity resolution should be deferred to transfer. A distinction between types of ambiguity according to their point of origin does not help in their resolution. Nor is it possible to draw a sharp dividing line between 'spurious' and 'real' ambiguities. Rather, we derive a characterisation of ambiguity types from the types of knowledge needed to resolve them.

Ambiguities resolvable by syntactic knowledge, such as the ambiguities in major syntactic category so common in English, seem to present little problem. MT systems whose output is intended for post-editing often include a 'homograph resolution' phase devoted to this type of ambiguity resolution. Though largely successful, such an approach is obviated by the use of the simplest phrase structure grammars. Conversely, explicit homograph resolution seems unavoidable when the system must not, under any circumstances, reject input as ill-formed and untranslatable. (fn 5)

Ambiguities resolvable by consideration of the sub-categorisation/valency/case patterns of lexical items include both lexical and attachment ambiguities, but not all cases of either. For instance:

"... provides the interface to the system"

appears to require domain specific knowledge; and

"...is achieved by calling theprocedure X"

requires fairly sophisticated knowledge concerning the relative likelihood of achieving a state by giving an entity a particular name or giving it control.

The question of how an ambiguity is resolved is thus almost independent of how it arises. Even word-class ambiguities occasionally persist through a syntactic analysis and require knowledge of the discourse for resolution, e.g.

"loading instructions will start processing"

A view of organising resolution so that each stage of processing resolves the ambiguities introduced by the previous stage is thus naive.

In terms of interaction, questioning the user too early in the translation chain will be unacceptable. The user must appear to the disambiguator as just another knowledge source, the last to be exploited.

If ambiguities are to remain unresolved through several stages of processing, compact representations of multiple readings of texts seem essential. The chart (Kay, 1973, Martin et al., 1981), and similar devices such as Tomita's (1985) 'shared packed forest' are important contributions to the solution of this problem.

The approach to ambiguity typical in linguistic theory is to allow the grammar to induce it, and to treat the question of its compact representation as a matter for parsing, irrelevant to declarative description. We are investigating the alternative notion that linguistic description itself should explicitly underdetermine representation. An approach to syntactic underdetermination is that of Marcus (1985). We have not yet examined its applicability to the current task. The sparse semantics of Aronoff (1980, see also Miller (1978), Wood (1985)) is a theory of underdetermination of lexical entries. Such lexical entries become more fully determined in context. One might say that they become ambiguated and resolved simultaneously. (In fact, depending on the use which is to be made of the results of analysis, they may never become determined.) This approach is applicable even to ambiguities in major syntactic category. Then parsing can be considered as ambiguating them to the extent licensed by the phrase structure rules and textual context.

Such ideas are particularly interesting in the context of our system. The GPSG-like feature system is a means for describing the redundancies in the lexicon; the LFG-like notion of multi-level linguistic description (see Kaplan 1985) offers the possibility of utilising such descriptions to 'expand' the lexicon in several stages. In our system, ambiguities in subcategorisation behaviour (including that between past and passive participles) do not exist during surface syntactic parsing. We think of such an item as being a single morph-syntactic entity, but a pair of 'semantic' ones.

This is important in translation, since the indeterminacies of the source language may or may not become ambiguities in translation. For example, the state-event ambivalence of passive participles in many languages must be recognised and resolved in transfer to Japanese (though not in going from French to English, for example). For a linguistic description of the source language not to include this ambivalence would render it incomplete; for analysis to treat it, and many similar cases, as genuine ambiguities would be problematic computationally.

We believe that the invocation or application of monolingual (source) knowledge by a bilingual component is an attractive approach to this problem. A good human translator infers from the source text what is needed for translation, and a machine system should exhibit this same, goal-driven, behaviour. In the same way that a human translator, in the event of being unable to resolve an ambiguity **whose resolution is crucial for translation**, might contact the original author, so the machine's knowledge must be organised to allow the same fail-safe interaction.

2) What form should interaction take?

We can recognise a variety of forms that interaction could take, on a scale of increasing independence from linguistic analyses, as follows:

a) Presentation of alternatives as their linguistic representations (e.g. trees). Though unsuitable for a linguistically-naive end-user, this type of interaction is important during system development. While the high-level linguist is experimenting with different ways in which the system's knowledge can be deployed for automatic ambiguity resolution, s/he must be able to inspect a detailed representation of those cases where the current strategy and/or knowledge are inadequate. In addition, this form of presentation is a prerequisite to defining more user-friendly forms.

b) String template presentations. At its simplest, for, say, attachment ambiguities, this involves presenting the ambiguously attached string with a slot filled by the different heads. This was the approach adopted in Kay's (1973) MIND system, e.g.

"They filled the tank with gas"
This means:
1. Filled with gas
2. Tank with gas

A more sophisticated template structure is used by Tomita (1985). This involves the use of what Hudson (1985) has termed 'natural metalanguage', e.g.

"I saw a man in the park"
1. The action [saw] takes place [in the park]
2. [a man] is [in the park]

We believe that this is an important direction to explore, though it may lead to obvious absurdities, depending on the sophistication with which templates are defined and chosen, e.g.

"This module provides the interface to the system"
1. The action [provides] takes place [to the system]
2. [The interface] is [to the system]

c) Disambiguating paraphrase. This can be illustrated for the same sentence as above:

1. The interface to the system is provided by this module.
2. This module provides the system with an interface.

The success of this approach depends crucially on finding the set of relational rules (in the example, passive and benefactive) that will generate disambiguating paraphrases. We hope that our approach to subcategorisation, with independent function-argument mappings for each constituent, will make this possible, but it is too early to say.

5. Implications and conclusions

Our practical experience of implementing the CCL Japanese translation system, some aspects of which we have described above, suggests a number of points of more general interest and significance for machine translation and indeed for any computational treatment of natural language. We will end our paper with an indication of a few of these wider implications and conclusions.

It becomes increasingly clear with time and experience that machine translation and linguistic theory can be of great mutual benefit. Early attempts at MT, which made little or no use of any sort of linguistic theory, can now be seen as, in principle, dead ends, despite the commercial success of some (cf. Hutchins 1982). We can now draw on a valuable body of research, particularly on syntax and semantics, which allows us to handle language with a depth and sophistication, and thus a chance of adequate translation, far greater than was possible for the early pioneers.

Conversely, as well as putting a practical task on a sound footing, the grounding of MT in linguistic theory provides an ideal test-bed for evaluation and development of such a theory. Even monolingual computational implementation imposes valuable demands of explicitness and accuracy, and can have a significant influence on the organization, content, and notation of a theory (cf. the evolution of GPSG, as described in Gazdar et al. 1985); translation requires a comprehensive and tractable account of (at least) two languages and of the mappings between them. It is also easy to evaluate one's results for both thoroughness of coverage and quality of output. In other words, the practice of machine translation should be seen as a resource for research in linguistic theory.

Most importantly, we would like to re-affirm our espousal of interactive system design. It should be uncontentious that the principal problem in the design of any knowledge-based system is the transfer of expertise from human to machine.

Thus we have designed our system with three particular aims:

1) to provide a means of expressing easily and naturally the forms of knowledge we currently understand;

2) to recognise and cope with the limits of currently formalised knowledge, by interaction;

3) to provide an environment within which we can experiment with redefining those limits.

Footnotes

* The work on which this paper is based is supported by International Computers Limited (ICL) and by the UK Science and Engineering Research Council under the Alvey program for research in Intelligent Knowledge Based Systems. The first-named author is the project director; the two first-named authors are the principal paper-writers, and as such can be held responsible for any errors, unclarities, or misrepresentations which may have escaped the

careful and valuable scrutiny of the others. We are grateful for the support and comments of our colleagues in CCL, especially to Rod Johnson.

1. Remark during discussion of King (1984). This viewpoint, of course, begs the question of the possibility of human translation, as Pierre Isabelle pointed out in a formal language closely resembling English.

2. We have glossed over the matter of transfer ambiguities. We assume that these can be expressed in source language terms for interactive disambiguation. In this respect, the machine is in a preferable position to that of a human translator, who must in most cases determine the resolution of such ambiguities from the text alone. Were our assumption incorrect, it would throw serious doubt on the possibility of human translation.

3. We might even suggest that there is something positively perverse about the assumption that a machine which can in isolation convert raw to half-baked (post-edited) or pre-cooked to finished text (pre-edited) is in some way superior to one which can carry out the entire process asked of it, recognizing the points at which it needs help and taking appropriate action.

4. It should be pointed out here, however, that we do not see programming as inherently radically distinct from other types of formal description. This is particularly true for a high-level "programming language" like Prolog.

5. This is a good example of how 'have a go at anything' systems may be hindered from incorporating linguistically motivated techniques.

References:

Aronoff, M. 1980. "Contextuals". In T. Hoekstra, H. van der Hulst, & M. Moortgat. **Lexical Grammar**. Foris, Dordrecht.

Bresnan, J. (ed.) 1982. **The Mental Representation of Grammatical Relations**. MIT Press, Cambridge, Mass.

Chevalier, M., J. Dansereau, & G. Poulin. 1978. "Taum-Meteo: Description du systeme". Universite de Montreal.

Dowty, D.R. 1979. **Word Meaning in Montague Grammar**. Reidel, Dordrecht.

Gazdar, G. 1985. "Linguistic Applications of Default Inheritance Mechanisms". In Whitelock et al, eds.

Gazdar, G., E. Klein, G. Pullum & I. Sag. 1985. **Generalized Phrase Structure Grammar**. Blackwells, Oxford.

Hudson, R. 1985. "What is a grammatical relation?" Linguistics Association of Great Britain, Salford, Spring 1985.

Hutchins, W. J. 1982. "The Evolution of Machine Translation Systems". In Lawson, ed.

Johnson, R., & P. J. Whitelock. 1985. "Machine Translation as an Expert Task". Proceedings of the Conference on Theoretical and Methodological Issues in Machine Translaton of Natural Languages, Colgate University, August 1985.

Kaplan, R. 1985. "Three seductions of Computational Psycholinguistics". In Whitelock et al, eds.

Kaplan, R. & J. Bresnan. 1982. "Lexical-Functional Grammar: A formal system for grammatical representation". In Bresnan, ed.

Kay, M. 1973. "The MIND system". In R. Rustin (ed.) **Natural Language Processing**. Algorithmics, New York.

King, M. (chair) 1984. "When is the Next Alpac Report Due?". Panel Discussion. Coling 84.

Lawson, V. (ed.) 1982. **Practical Experience of Machine Translation**. North-Holland, Amsterdam.

Marcus, M. 1985. "Description Theory". In Whitelock et al, eds.

Martin, W.A., K.W. Church and R.S. Patil. 1981. "Preliminary Analysis of a Breadth-First Parsing Algorithm: Theoretical and Experimental Results". MIT/LCS/TR-261.

Miller, G.A. 1978. "Semantic Relations among Words". In M. Halle, J. Bresnan & G.A. Miller (eds.) **Linguistic Theory and Psychological Reality**. MIT Press, Cambridge, Mass.

Ruffino, J. R. 1982. "Coping with Machine Translation". In Lawson, ed.

Shieber, S. 1984. "Design of a Computer Language for Linguistic Information". Coling 84.

Shieber, S. 1985. "Separating Linguistic Analyses from Linguistic Theories". In Whitelock et al, eds.

Slocum, J. 1985. "A Survey of Machine Translation: Its History, Current Status, and Future Prospects". Computational Linguistics 11.1:1-17.

Steedman, M.J. and M. Moens. 1986. "A Computational Account of Temporal Reference". Conference on Temporal Logic and its Applications, Univ. of Leeds, January 1986.

Tomita, M. 1985. "An Efficient Context-Free Parsing Algorithm for Natural Languages and its Application". PhD thesis, Carnegie-Mellon Univ., Pittsburgh, Pa.

Vendler, Z. 1967. **Linguistics in Philosophy**. Cornell University Press, Ithaca, New York.

Whitelock, P.J. (ed.) 1985. "An in-depth study of Linguistic and Computational techniques in MT system design". CCL Internal Report 85/1, CCL, UMIST.

Whitelock, P.J., H.L. Somers, R.L. Johnson, P.A. Bennett, and M. McGee Wood (eds.) 1986. **Alvey/ICL Workshop on Linguistic Theory and Computer Applications (UMIST, September 1985): Transcripts of presentations and discussions**. CCL Internal Report 86/2, CCL, UMIST.

Wood, M. McGee. 1985. "The Semantics and Dynamics of Derivation". In G.A.J. Hoppenbrouwers, P.A.M. Seuren, & A.J.M.M. Weijters (eds.) **Meaning and the Lexicon**. Foris, Dordrecht.