[From: Internat.conf. *Methodology & Techniques of Machine Translation*,
British Computer Society, Cranfield, 13-15 February 1984]

SYNICS - A Tool for Translating Natural Language Syntax

M. Roberts & E.A. Edmonds

Human-Computer Interface Research Unit
Leicester Polytechnic

Of all those activities which fall within the embrace of IKBS research, work on systems for the automatic translation of natural language addresses some of the most challenging of problems, whilst the objective itself is among the most elusive of goals. Communication through natural language represents behaviour which is unquestionably rule-governed, yet the rules are many and their constraints diverse. They are also subject to change (witness, for example, the increasing intrusion of the 'split infinitive' into both written and spoken English). Thus the task of eliciting and formalising sufficient of these rules to enable translation by automata, whilst not impossible, is far from facile.

Certain aspects of the translation process appear obvious and determine some essential requirements for any automated system, e.g., reference to the relevant dictionaries and a cogniscence of the interaction between semantic and syntactic factors. Whilst such activity may be tacit for the human, it must be made explicit if a computer is to accomplish the same ends. Other features of the act of translation remain open to question, and contentious. One may question, for instance, whether it is either necessary or desirable to introduce an 'interlingua ' into the translation process. There is some evidence in the psychological literature (e.g., Tversky, 1969; Pylyshyn, 1973) to suggest that such a cognitive device may be employed at certain levels of processing, where the 'translation' is between visual and verbal representations of given concepts. However, demonstrations of perceptually related phenomena may not be pertinent, necessarily, to the translation of languages per se. Indeed the declared trend of certain research groups, already very experienced in the field of machine translation, is away from reliance upon any putatively universal machine representation of statements in natural language, and toward an alternative form of three-stage process (Durham, note 1.). According to this view the intermediate 'transfer module' should embody knowledge of both source and target languages. It thus constitutes an active component of the process rather than a passive representation of semantic intent. Nevertheless the principle of an 'interlingua' remains theoretically attractive and doubtless still has its adherents.

The emergence, within the past decade, of logic programming, has provided the impetus for some researchers to explore the utility of logic formalisms as means of representing natural language statements (e.g., Pereira and Warren, 1980). However, the techniques of predicate logic which have proved themselves in other IKBS domains (Gero, personal communication) have yet to be exploited with absolute success in the arena of natural language processing. In the opinion of Ritchie and Thompson (in press) attempts at simplistic association of the objects and relations of a logic model with those of the real world are naive.

It would seem inevitable that whatever computational shorthand one might seek to adopt, certain of the problems attending natural language translation will always remain to be faced. As an example, the logic statement $(\forall x)(\forall y)((M(x) \vee C(y)) \longrightarrow H(x,y))$ may reasonably be taken to represent the English sentence "All mice hate all cats." Given that

"x is a mouse": M(x) and "y is a cat": C(y). It would hold true also for a similar declaration in German, but an automatic translator would have to know that it was now dealing with "Katze and Mause" and that the relationship between them was one of "hassen".

These remarks should not be seen as an attempt merely to pour scorn on logic, since its potential value to systems for natural language understanding, and hence automatic translation, may well exceed its present worth. It could be the case that formal logic is an appropriate tool being, at present, inappropriately applied. Certainly it would seem reasonable to suppose that statements in first order predicate calculus, for example, should be more computationally tractable than the unabridged syntax of a natural language. From a purely pragmatic standpoint therefore, some form of intermediate representation, even if it should not constitute an 'interlingua' in the fullest sense, could still prove a useful gambit in the translation process. An analogy may be drawn with mathematics. Sometimes the solution of a given problem may be facilitated by the application of one particular technique, yet obscured by another, less appropriate method.

In computing also there are occasions when it becomes expedient to employ appropriate extant software tools rather than resort to a diversity of programming methods occasioned by the need to implement a particular strategy in different (programming) languages and/or between incompatible systems. In particular, if the generation of some preferred representation of English, or other syntax, were among ones primary objectives, then considerable time and effort would be saved by applying a 'package' capable of accomplishing this transformation efficiently. In SYNICS there exists just such a package.

SYNICS is a translator writing system in which the translation SYNtax and semantics is specified in a modified BNF notation. One program is used to develop and test the translation rules and then to generate a file containing a set of tables which represent the required translation. A set of subprograms may then be incorporated in a user's program to perform the translations as needed. This arrangement allows the user of the system to make whatever use he cares to of the generated strings. He is free to perform operations before or after translation.

The notation of SYNICS is based upon that of BNF, or context free notation and uses the concept of a syntax tree. BNF notation specifies how to generate strings in a language and if treated in certain ways can double as a means of specifying how strings in the language might be recognised. In a translator writing system such as SYNICS it is only necessary to specify how to recognise the source language. It thus becomes possible to extend the notation so as to include commands that direct the recognition process, even though they would not be readily meaningful with respect to the generation of strings. In SYNICS an if-then-else command is included. This has the effect of allowing some context to be taken into account. It can also be seen as a way of allowing sane bottom-up analysis to be introduced into what is basically a top-down system.

To exemplify the capability of SYNICS in the context of natural language translation the following discussion will concern the application of SYNICS to a single English sentence extracted, incidentally, from a letter written to those delegates offering a paper at this conference: "I am writing further to my earlier letter to you accepting your paper." A parse tree for this sentence is shown in figure 1.

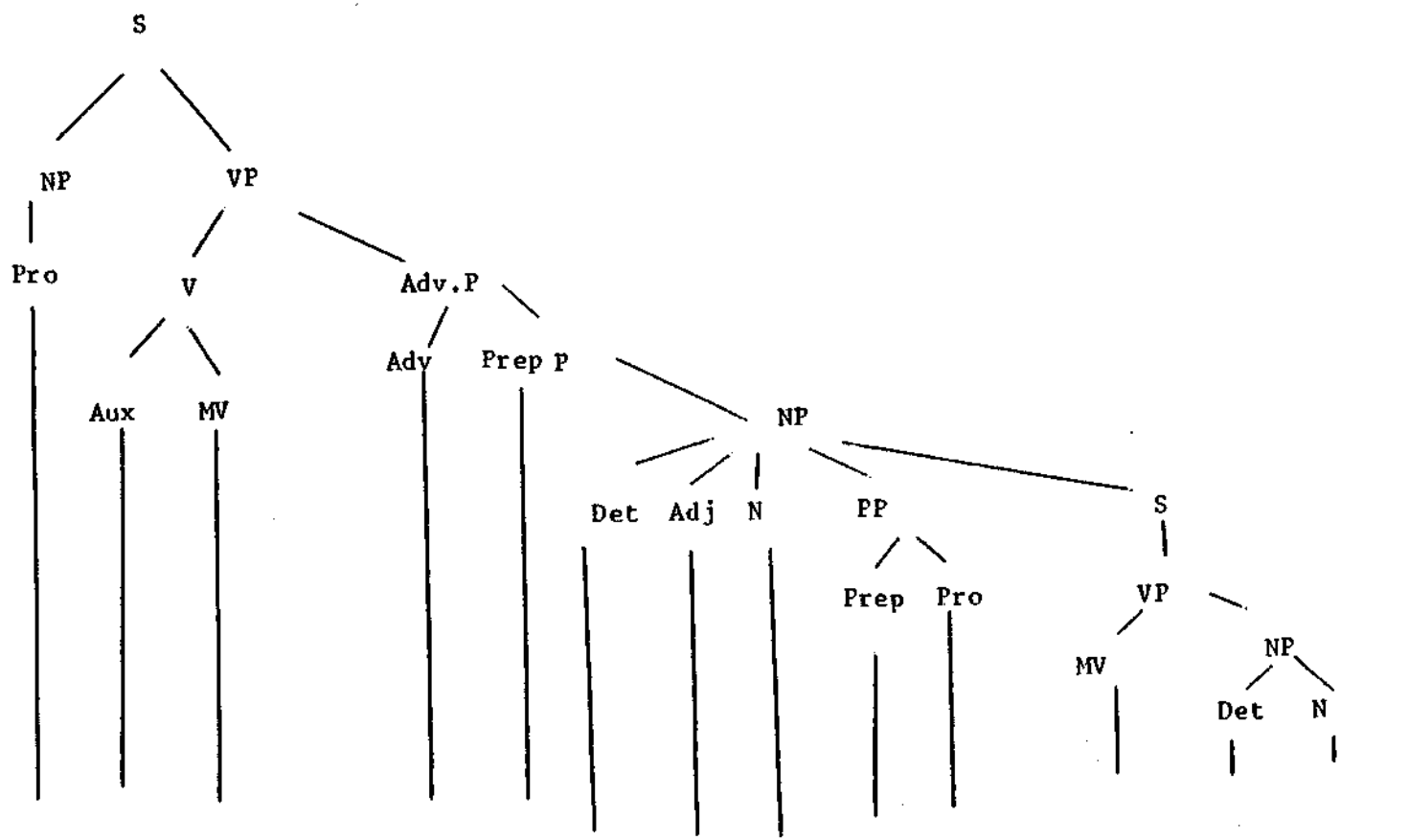
S
NP
VP
Pro
V
Adv.P
Aux
MV
Adv
Prep P
NP
Det
Adj
N
PP
S
Prep
Pro
VP
MV
NP
Det
N
I am writing further to my earlier letter to you accepting your paper.

Figure 1.

Let us assume that one wishes to reconfigure the original syntax so that only the simple infinitive forms of the main verbs are immediately apparent, together with the nouns, and that the more incidental (though no less significant) aspects are reduced to something akin to an index of their grammatical category thus:

write(T)(Pro)(Adv) letter(Adj)(Pos) accept(T) paper(Pos).

The designation (T) indicates only that the preceding verb is to assume a particular tense whilst the phrase "to you" is abandoned as redundant.

A SYNICS routine to effect this transformation is listed in the appendix. It operates upon what it takes to be the 'grammar' for the sentence in question, in this case the syntactic structure shown in figure 1., which is made explicit in the program. The commands within square brackets, "[. . .]" , operate on the relevant syntactic structures after the sentence has been successfully parsed. Any such command of the form Cn constitutes an instruction to the system and means: "generate a result associated with n next" whilst statements of the form Ln mean: "copy, literally, the word(s) associated with n into the result." Characters within quotes, "...", are copied directly into the resulting output stream. $n statements are calls to specialized hypothetical subroutines for dictionary searches, verb conjugation, or whatever. Used during the parsing process these routines must indicate success or failure.

The resultant transform for our sample retains the words from the original sentence and is really no more than a sequential list of the major parts of speech drawn from the original syntax. Each of these components is followed by one or more indeterminate qualifiers having a particular bearing on that part of speech. As the SYNICS program illustrates, these qualifiers would be addressed separately, either as literal copies of the original, or as an instruction to the system as a whole to do some 'reasoning' about their syntactic/semantic significance. In this last respect our example, although trivial, attempts to take account of some of the obvious pitfalls in the translation exercise.

Assuming one’s intention were to translate the original English sentence into Spanish, say, then our hypothetical system would have recourse to the transformed representation shown earlier. The converse operation of translating from this format into the correct syntax for the target language could be handled by SYNICS also. The complementary synthesis module would translate all recognisable English words literally yielding:

escribir(T)(Pro)(Adv) carta(Adj)(Pos) aceptar(T) papel(Pos).

Unravelling the rest of the sentence, i.e., handling the qualifiers in a serial left-right fashion, would have to be accomplished with reference not only to the grammatical rules of Spanish but also to some fairly specific contextual constraints, since even the literal translation of "paper" into "papel" is incorrect in this context. In order to arrive at an acceptable translation (Le escribo para continuar mi anterior carta aceptando su disertacion) our hypothetical system would require knowledge about such factors as formal modes of address, admissible paraphrases, and implied objects, to mention but a few.

The intermediate representation which we have chosen to discuss here is clearly inadequate in many respects but to dwell on such a criticism would be to miss the point, which is that certain forms of linguistic transformation, not necessarily that offered here, could be beneficial in facilitating the process of translation by machine. Where such transformations are required, be they logic formalisms, pointers to grammatical status or whatever, then SYNICS offers a tool which could save considerable programming effort on the part of those researchers seeking a means of passing efficiently, to and fro, between conventional syntactic expressions in natural language and their chosen optimisation.

In the final analysis the task for automatic translation is to extract and reformulate meaning. Verbrugge (1977) provides convincing support for the dictum "Meanings are in people, not in words." Nonetheless it remains the case that syntactic structure is a function of meaning. That being so then certain aspects of meaning should be expected to reveal themselves as a consequence of an analysis of sentence structure. The SYNICS system is ideally suited to handling such structure. It is not claimed that it can open Pandora's box, but it could be a useful guide to finding the key.

The intermediate representation which we have chosen to discuss here is clearly inadequate in many respects but to dwell on such a criticism would be to miss the point, which is that certain forms of linguistic transformation, not necessarily that offered here, could be beneficial in facilitating the process of translation by machine. Where such transformations are required, be they logic formalisms, pointers to grammatical status or whatever, then SYNICS offers a tool which could save considerable programming effort on the part of those researchers seeking a means of passing efficiently, to and fro, between conventional syntactic expressions in natural language and their chosen optimisation.

In the final analysis the task for automatic translation is to extract and reformulate meaning. Verbrugge (1977) provides convincing support for the dictum "Meanings are in people, not in words." Nonetheless it remains the case that syntactic structure is a function of meaning. That being so then certain aspects of meaning should be expected to reveal themselves as a consequence of an analysis of sentence structure. The SYNICS system is ideally suited to handling such structure. It is not claimed that it can open Pandora's box, but it could be a useful guide to finding the key.

APPENDIX. SYNICS routine for transforming the Sentence: "I am writing further to my earlier letter to you accepting your paper."

$S = {}^{3}Np\ {}^{2}V\ {}^{1}Adv.p.\ [C3\ C2"("\$104")"\ C1]\ /{}^{1}Vp.[C1]$ Remarks = Collapsed form.
$104 generate "person flag."

$Vp = {}^{2}Mv\ {}^{1}Np\ [C2\ C1]$

$V = {}^{2}Aux\ {}^{1}Mv\ [C1"("C2")"]\ /{}^{2}Mv\ {}^{1}Np\ [C2\ C1]\ /\ {}^{1}Mv\ (C1)$

$Adv.p. = {}^{2}Adv\ {}^{1}Prep\ p.[" ("C2")"\ C1]$

$Prep.p. = {}^{2}Prep\ {}^{1}Np\ ["("C2")"C1]\ /{}^{2}Prep.pro$ Remark:no transformation; phrase ignored.

$Np = {}^{5}Det\ {}^{4}Ad.\ {}^{3}N\ {}^{2}Prep.p.\ {}^{1}S\ [C3"("C4")("C5")"C1]\ /{}^{2}Det.N\ [C1"("C2")"]\ {}^{1}Pro.[C1]$

$Mv = {}^{1}\$1\ [\$101]$ Remark:$1ook up a verb
$101 generate its root

$N = {}^{1}\$2\ [L1]$ Remark:$2 look up a noun

$Prep = {}^{1}\$3\ [L1]$ Remark:$3 look up a preposition

$Pro = {}^{1}\$4$ Remark:$4 look up a pronoun

$Aux = {}^{1}\$5\ [\$105]$ Remark:$5 look up aux. $105 generate "time flag"

$Adv = {}^{1}\$6\ [L1]$ Remark:$6 look up an adverb

$Adv = {}^{1}\$7\ [\$107]$ Remark:$7 look up an adjective $107 generate "form flag"

$Det = {}^{1}\$8\ [L1]$ Remark:$8 look up determiner

Reference Note

1. Durham, T. Interpreting the job of the translators. Article written for the trade weekly, "Computing," 24.11.1983.

REFERENCES

Edmonds, E.A. and Guest, S.P. (1978). SYNICS - A Fortran subroutine package for translation. Man-Computer Interaction Research Group Report No.6, Leicester Polytechnic.

Pereira, F.C.N. and Warren, D.H.D. (1980). Definite clause grammars for language analysis - a survey of the formalism and a comparison with augmented transition networks. Artificial Intelligence, 13, 231-278.

Pylyshyn, Z.W. (1973). What the mind's eye tells the mind's brain : a critique of mental imagery. Psych. Bull., 80, 1-24.

Ritchie, G. and Thompson, H. (in press) . Natural language processing. To appear in: Eisenstadt and O'Shea (eds.). Artificial intelligence skills: tools, techniques and applications : Harper and Row.

Tversky, B. (1969). Pictorial and verbal encoding in a memory search task. Perception and Psychophysics, 4, 225-233.

Verbrugge, R.P.. (1977) . Resemblances in language and perception. In: R. Shaw and J. Bransford (eds.). Perceiving, Acting, and Knowing. Toward and Ecological Psychology : Lawrence Erlbaum.