[From: P.Whitelock et al. (eds.) Linguistic theory & computer applications
(London: Academic Press, 1987)]

Montague Grammar and Machine Translation

Jan Landsbergen

Introduction

In this paper I will examine the possibilities of using Montague Grammar for machine translation. I will discuss briefly the various ways in which this theory could be used, but most attention will be given to one actual application: the Rosetta translation system. The paper is organized as follows. After a short introduction to Montague Grammar, its strong and weak points with respect to computer applications will be discussed. Then a syntactically powerful and computationally viable version of Montague Grammar, called M-grammar, will be described. Subsequently I will discuss various ways in which Montague Grammar may be used directly for machine translation and pay special attention to the problems that arise in these cases. Finally I will outline the isomorphic grammar approach to machine translation, followed in the Rosetta project, in which the compositionality principle of Montague Grammar plays an important role.

Montague Grammar

It is not possible to give in a few words a fair account of Montague Grammar and this holds in particular for

The Rosetta project is partially sponsored by NEHEM (Nederlandse Herstructureringsmaatschappij).

I would like to thank Jeroen Groenendijk, Kees van Deemter, Rene Leermakers and Jan Odijk for their comments.

its semantic power. In this section I will restrict myself to introducing some basic concepts and the corresponding terminology, which are needed for a good understanding of the rest of the paper. The terminology and the notation may deviate a little from "standard" Montague Grammar.

Montague's most important papers on language are 'The Proper Treatment of Quantification' (1973), 'Universal Grammar' (1970a), and 'English as a Formal Language' (1970b). They have been collected together with other papers in Thomason (1974). A good introduction to the theory is Dowty et al (1981). The 1973 'PTQ' paper, as it is usually called, is the best known and contains the most influential example of a Montague grammar. The paper 'Universal Grammar' describes the general algebraic framework (cf Janssen 1986 for a better insight into and an elaboration of this framework). 'English as a Formal Language' (EFL) is interesting because it shows how natural language can be interpreted directly, without intervention of a logical language.

The main characteristic of Montague Grammar is the attention that is given to semantics. Montague grammars have to obey the **compositionality principle**, which says that the meaning of an expression is a function of the meaning of its parts. What the parts are has to be defined by the syntax, so the principle prescribes a close relation between syntax and semantics.

The syntax of a Montague Grammar specifies (i) a set of basic expressions and (ii) a set of syntactic rules. The basic expressions are the smallest meaningful units, the syntactic rules define how larger phrases and ultimately sentences can be constructed, starting with the basic expressions. The rules are applied in a compositional ("bottom-up") way.

A simple example:

The basic expressions are: the noun boy and the verb sleep.

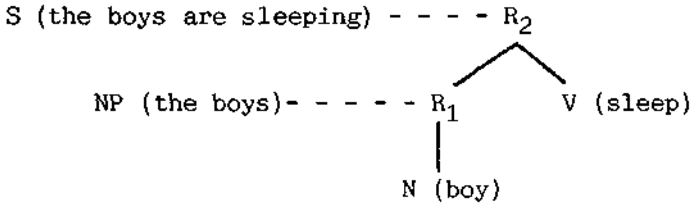
The rules are:

R_1 : this rule is applicable to a noun, e.g. boy, and makes a definite plural noun phrase, by adding the article the and the suffix -s; e.g. the boys.

R_2 : this rule is applicable to a noun phrase and a verb and makes a sentence with the NP as its subject, in the present progressive tense, e.g. the boys are sleeping.

The process of deriving a sentence from basic expressions by recursive application of rules can be made explicit in a **syntactic derivation tree.** In (1) an example of a syntactic derivation tree is given: it shows the derivation of the sentence the boys are sleeping according to the example grammar.

(1)

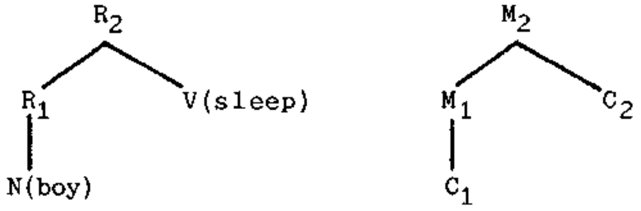


In Montague's example grammars the basic expressions and the expressions generated by the rules have a syntactic category, but no explicit internal structure, they are just symbol strings. Actually, Montague used a version of categorial grammar. However, these restrictions are in general not considered essential properties of the theory. Already in the seventies Partee (1976) proposed an extension in which the rules operate on syntactic structures (or - equivalently - labelled bracketings) in which syntactic transformations may occur.

The semantic component of Montague Grammar assigns a semantic interpretation to the language as follows. First a semantic domain is defined, consisting of individual entities, truth values, special indices and functions defined in terms of these objects. Characteristic of Montague Grammar is the use of a special kind of indices, usually called "possible worlds". They are important for the power of the semantic system, which is often referred to as "possible-world semantics", but will not be discussed here.

The assignment of semantic values to expressions of the language can be done in two ways: directly and indirectly. In a direct interpretation (a method explored in the paper EFL) basic expressions and syntactic rules are immediately interpreted in terms of the semantic domain; each basic expression is associated with an object in the domain (e.g. an individual, a function from individuals to truth values, etc.) and with each rule an operation on objects in the domain (e.g. function application) is associated. The semantic value of an arbitrary expression is then defined with the help of the syntactic derivation tree. In parallel with the application of the syntactic rules the semantic operations associated with these rules are applied to the semantic values of their arguments, starting with the values of the basic expressions. The final result is the semantic value of the complete expression. So the process of derivation of the semantic value runs parallel with the syntactic derivation process and can be represented in a tree with the same geometry as the syntactic derivation tree, but which is labelled by names of semantic values and semantic operations. This representation, called **semantic derivation tree**, is introduced here because it will be useful in the sequel; it is not explicitly used by Montague. If we assume that the rules of our example grammar correspond to meaning rules, named M_1 and M_2, and the basic expressions to meanings with the names C_1 (for boy) and C_2 (for sleep), the relation between syntactic and semantic derivation tree is as in (2).

(2) syntactic derivation tree --> semantic derivation tree



A simplified example:

C_1 is a property of individuals (equivalently: a set of individuals), i.e. the property "being a boy".

C_2 is also a property of individuals, i.e. the property "sleeping".

M_1 operates on a property P and yields the set of properties that all individuals with property P have, in this case the properties all boys have. (In this example it is assumed - wrongly - that "the + plural" can be interpreted as universal quantification.)

M_2 operates on a set of properties S and a property P and yields true if P is in S, else false.

So the semantic value of the sentence is true if the property of "sleeping" is a property that all boys have, else it is false.

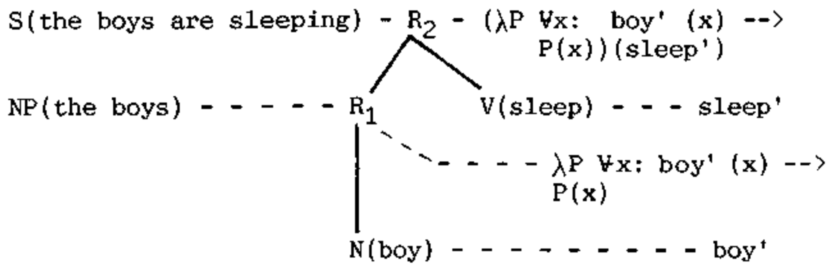
The more usual way of assigning interpretations (pursued in PTQ) is the indirect one, which proceeds in two steps. First an expression of the language is translated into an expression of a logical language (in PTQ higher-order intensional logic). Then the logical expression is assigned a semantic value by interpreting the logical language in the standard way.

The translation from natural language into logical language is defined in a similar - syntax-directed - way as the direct interpretation. For each basic expression its translation into the logic is given, each syntactic rule corresponds to a (possibly complex) operation on logical expressions. In parallel with the application of the syntactic rules the logical operations associated with these rules are applied to the logical expression associated with their arguments, starting with the logical expressions corresponding to the basic expressions.

The final result is the logical representation of the complete sentence. Note that in the indirect way of assigning interpretations, the form of the logical expressions themselves is not relevant; they are only a means to express in a convenient way the model-theoretic interpretation.

In (3) I illustrate this process by showing in parallel the derivation of the sentence the boys are sleeping and of its (extensional) logical representation, but without further explanation. The derived logical expression for the complete sentence is equivalent to the reduced form: ∀x: boy' (x) --> sleep' (x)

(3)



Montague Grammar and computer applications

What are the strong and the weak points of Montague Grammar with regard to its use in computer applications that involve natural language processing?

Two important application areas in the field of natural language processing are natural language question-answering and machine translation. A strong point of Montague Grammar in these two areas is the attention that is given to semantics. In both application areas a sound semantic base is needed for determining what a correct answer or a correct translation is.

Another advantage of Montague Grammar in comparison with some other linguistic theories is its exactness and its "constructiveness". By "constructiveness" I mean that there is a clear step-by-step construction of phrases and - in parallel - of their meanings, thanks to the compositionality principle. Since for each rule both the syntactic and the semantic operation must be defined, the correctness of the rule can - to a large extent - be judged locally. This advantage is lost in a grammar with several syntactic levels, where the semantics is defined at the deepest level (whatever other virtues these levels may have). Local correctness criteria are important in the design of large systems in general and in particular in the design of large grammars.

A supposed weak point of Montague Grammar is that it treats only small fragments of language in a syntactically simplistic way. As for the fragmentariness, this is a consequence of exactness. Dealing with small - but non-trivial - fragments completely, in full detail is to be preferred - from the point of view of computer applications - to making interesting, but imprecise claims about natural languages in general. The syntactic simplicity of the framework is certainly a weak point, but it is more an incidental property of Montague's example grammars than an inherent property of the theory. The problem is not a lack of formal power, but a lack of linguistic power: the rules operate on strings and not on structured objects, e.g. syntactic trees. I have already referred to the syntactic extensions proposed by Partee (1976), and other work has been done in this direction, but nevertheless it is a correct observation that most workers in the field are primarily interested in semantics and less in syntax.

Another objection against Montague Grammar is that intensional logic and possible-world semantics are complicated and therefore hard to put to practical use in large systems. This is a correct observation. Montague needed the power of intensional logic to solve several difficult semantic problems, but these problems do not necessarily occur in all applications. For instance, in most data base question-answering systems a simple extensional semantics is sufficient. It is not in conflict with the spirit of Montague Grammar to use a simpler logic, as long as there is a compositional and model-theoretic semantics. The specific system of intensional logic may indeed be difficult, but model-theoretic semantics in itself is very easy to understand and to use; by imagining a particular interpretation it is possible to get a fast insight into the semantic correctness (and especially the incorrectness) of a particular rule or of a larger part of the grammar.

The most important obstacle to the application of Montague Grammar is that it is a purely generative framework. The theory defines how sentences and their meaning representations are generated in parallel, but it does not define how for a given sentence a meaning representation can be constructed effectively. This weakness can only be overcome by restricting in some way the class of possible Montague grammars. This will be the topic of the next section. There I will define M-grammars, which are less powerful than unrestricted Montague grammars from a purely formal point of view, but more powerful from a linguistic point of view, in the sense that the rules operate on structured objects instead of strings.

M-grammars

To my knowledge, two different ways of defining parsers for Montague Grammars have been described: by Friedman and Warren (1978) and by Landsbergen (1981). The approaches differ strongly in what they consider to be a Montague grammar. Friedman and Warren remained as close as possible to the PTQ grammar and designed a parser which can be characterized as a context-free parser with some specific extensions for phenomena falling outside the context-free framework, in particular the quantification rules. My own proposal defines a parser for a class of grammars, called M-grammars, which are syntactically more powerful and which are in accordance with Partee's transformational extensions (Partee 1976). Since 1981 a few changes in the definition of M-grammars have been made, of which the most important is the introduction of a separate morphological component. The new version is described in Landsbergen (1985). I will recapitulate it here briefly.

An **M-grammar** consists of three components: a syntactic component, a morphological component and a semantic component.

The **syntactic component** of an M-grammar defines a set of surface trees of sentences. The specific kind of surface trees generated by M-grammars - and the intermediate results - are called **S-trees**. An S-tree is an ordered tree of which the nodes are labelled by syntactic categories and attribute-value pairs and of which the edges are labelled by syntactic relations.

Formally, an S-tree **t** is an object of the form

$$N\ [r_1/t_1,\ \ldots,\ r_n/t_n],\ (n \geq 0)$$
$$\text{with } N = C\ \{a_1\text{: } v_1,\ \ldots,\ a_k\text{: } v_k\}$$

where N is a node,

t_1, ..., t_n are S-trees, the immediate constituents of **t**,

r_1, ..., r_n are syntactic relations, between t and its constituents (if n = 0, t is a terminal S-tree)

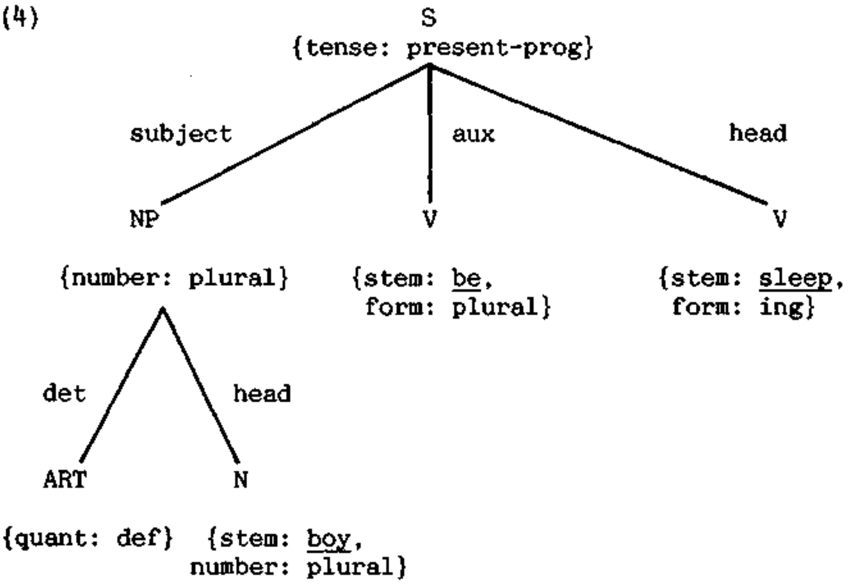
C is a syntactic category,

a_1, ..., a_k are attributes,

v_1, ..., v_k are values of these attributes.

An example of an S-tree in the more familiar graphical representation is given in (4). It is a - simplified and unrealistic - example of a surface tree, for the sentence *the boys are sleeping*.

(4)



In the sequel I will often use an abbreviated notation, as in (5).

(5) S (the boys are sleeping)

The leaves of an S-tree correspond to words. For example, the terminal node

N {stem: *boy*, number: plural}

corresponds to *boys*. This relation between terminal nodes and words as symbol strings is defined by the morphological component.

An M-grammar defines a language (in this case a set of surface trees) in the same way as a Montague grammar, i.e., by specifying a set of basic expressions and a set of syntactic rules. But here the basic expressions are S-trees (in general S-trees consisting of one node) and the rules are defined for S-trees as arguments and yield S-trees as their results.

The derivation process of a surface tree from basic S-trees by application of rules can be represented by a syntactic derivation tree in the way described earlier. If we reformulate the example grammar of the previous section in terms of S-trees, the syntactic derivation tree of the boys are sleeping (i.e. of its surface tree) is the same as (1).

In principle all rules of an M-grammar have "transformational power": they can perform fairly arbitrary operations on S-trees. However, this power is restricted by three conditions that M-grammars have to obey in order to make effective parsing possible: the reversibility condition, the measure condition, and the surface syntax condition. I will describe them here informally (cf. Landsbergen 1985 for more precise definitions).

The **reversibility condition** states that a rule should not only define a compositional ("generative") function (with a tuple of S-trees as argument and an S-tree as result), but also an analytical function (which operates on an S-tree and yields a tuple of S-trees). The compositional and the analytical function should be each other's reverse (the term reverse is used instead of inverse, because a rule produces a set of results, possibly the empty set, if the rule is not applicable). If the compositional function is applied to a tuple (t_1, ..., t_n) and t is in the set of results, then application of the analytical function to t must yield a finite set containing the tuple (t_1, ..., t_n), and vice versa.

Given a set of basic S-trees and a set of reversible rules, two functions, M-PARSER and M-GENERATOR, can be defined:

M-GENERATOR operates on an arbitrary syntactic derivation tree (i.e. an arbitrary tree labelled by rules and basic expressions) and yields a set of S-trees, by applying the compositional versions of the rules in the derivation tree.

in a "bottom-up" way. The resulting set may be empty if some rule is not applicable.

M-PARSER operates on an arbitrary S-tree. It tries to apply the analytical versions of the rules in a "top-down" way until it arrives at basic S-trees. If this is successful, the result is a syntactic derivation tree (more than one derivation tree in case of ambiguities; the empty set if the input was not a correct S-tree).

M-GENERATOR and M-PARSER are each other's reverse: they define the same relation between S-trees and derivation trees.

In order to guarantee that M-PARSER is a computable function, an M-grammar has to obey the **measure condition**. It says: there is a measure on S-trees (a function from S-trees to integers, with a minimum) such that application of an analytical rule to an S-tree t yields S-trees smaller than t with respect to this measure. An example of a measure is the number of nodes in an S-tree, but in practice more subtle measures are needed. Thanks to the measure condition, application of M-PARSER always ends after a finite number of rule applications.

As it is our purpose to generate and analyze sentences, not surface trees, additional functions are needed. In the generative direction this is no problem: a function LEAVES can be defined which yields the sequence of leaves (the terminal S-trees) of an S-tree. For analysis purposes we need the third condition on M-grammars, the **surface syntax condition**. It says that for each M-grammar a set of "surface rules" must exist which define for each sentence a finite set of surface trees of which the set of correct surface trees is a subset. So this surface syntax has to be "weaker" than the real syntax and the surface rules can be simpler than the actual syntactic rules. A surface rule is applied in a bottom-up way to a sequence of S-trees; if it is applicable, the result is an S-tree with a new top node and with the input sequence of S-trees as its immediate constituents. Thanks to this, conventional parsing strategies can be used for the application of the surface rules, e.g. a variant of the CKY or the Earley Parser. The function applied by the parser is called S-PARSER.

The **morphological component** of an M-grammar relates terminal S-trees to actual words, symbol strings. It makes use of a dictionary and of various kinds of morphological rules, not to be discussed here. The morphological component defines two functions:

A-MORPH ´converts words into (sets of) terminal S-trees.

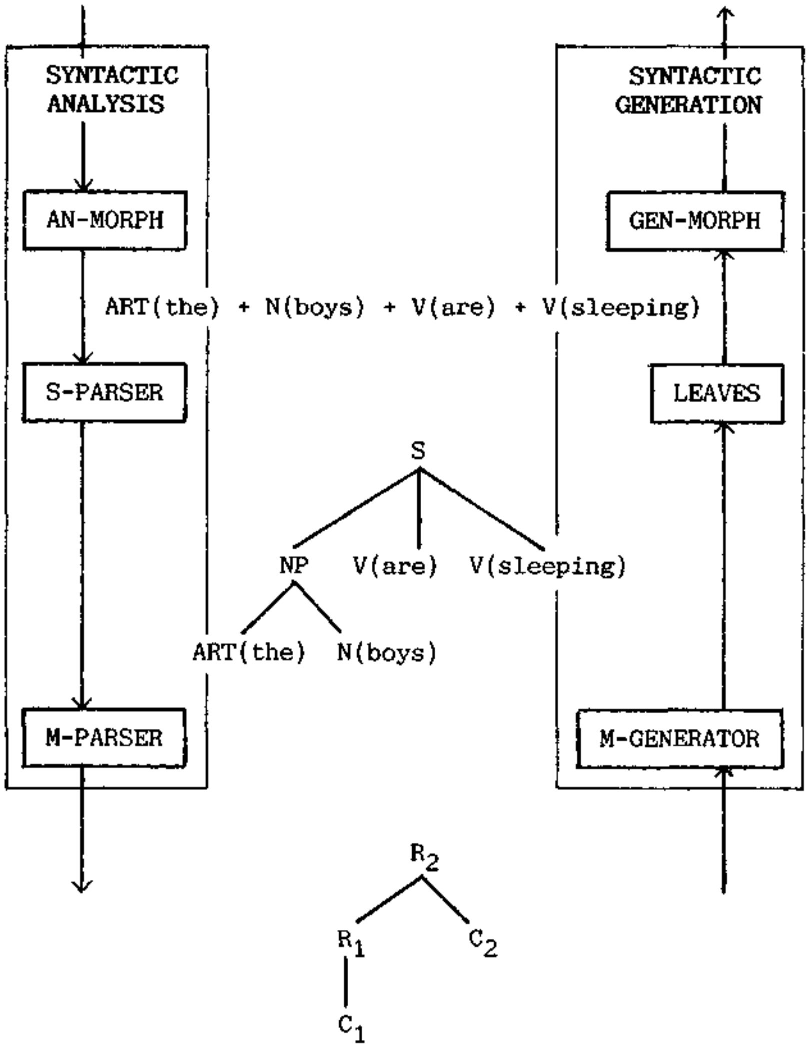
G-MORPH converts terminal S-trees into (sets of) words.

A-MORPH and G-MORPH are each other's reverse.

The syntactic component and the morphological component together define a function SYNTACTIC ANAYLYSIS and a function SYNTACTIC GENERATION, which are each other's reverse. The function SYNTACTIC ANALYSIS is the composition of A-MORPH, S-PARSER and M-PARSER, the function SYNTACTIC GENERATION is the composition of M-GENERATOR, LEAVES and G-MORPH. In (6) the two functions are shown with example expressions. Note that the examples are a bit misleading as they suggest that these functions always give a unique result, which is the case for our example grammar, but not in general.

The **semantic component** of an M-grammar defines for each syntactic rule a "meaning rule" and for each basic expression a set of "basic meanings". As it depends on the application what the most appropriate way is to express these meanings - in an intensional logic, in an extensional logic or in some other way - this is left open here. A minor difference from standard Montague Grammar is that in an M-grammar a basic expression may have more than one meaning. This has the practical advantage that during analysis purely semantic word ambiguities can be "postponed" until after the syntactic analysis.

(6) the boys are sleeping



Montague Grammar and machine translation

I arrive now at the central topic of my paper: the use of Montague Grammar in translation systems. In the previous section I have defined M-grammars, syntactically powerful versions of Montague grammars, for which an effective analysis procedure can be defined. In what way can they be used in a translation system? In order to be able to discuss the application of a linguistic theory in a translation system, I assume that in such a system the linguistic aspects can be clearly separated from the other aspects (e.g. the use of extra-linguistic information, robustness measures, etc.). Then it is possible in principle to consider a "stripped" system that makes use of linguistic information only. In addition I restrict the discussion to systems that translate isolated sentences. Such systems are in general not able to translate sentences unambiguously, but they define a set of **possible translations**. I define the function F-PTR as the function that operates on a sentence of the source language and yields the set of possible translations into the target language. F-PTR has the property that it is reversible: if s' is a possible translation of s, then s is a possible translation of s'.

(7) s' in F-PTR(s) <--> s in F-PTR'(s')

The "correct" or "best" translation of s (chosen on the basis of extra-linguistic information) should be an element of the set F-PTR(s). Obviously, the function that yields this best translation is not reversible.

I would like to impose the following requirements on such a "possible translation" system.

1. It must be defined clearly what are correct sentences of the source language (SL) and the target language (TL). In other words, the system must be based on explicit grammars of SL and TL.

2. The translation function F-PTR must be defined in such a way that correct sentences of SL are translated into correct sentences of TL.

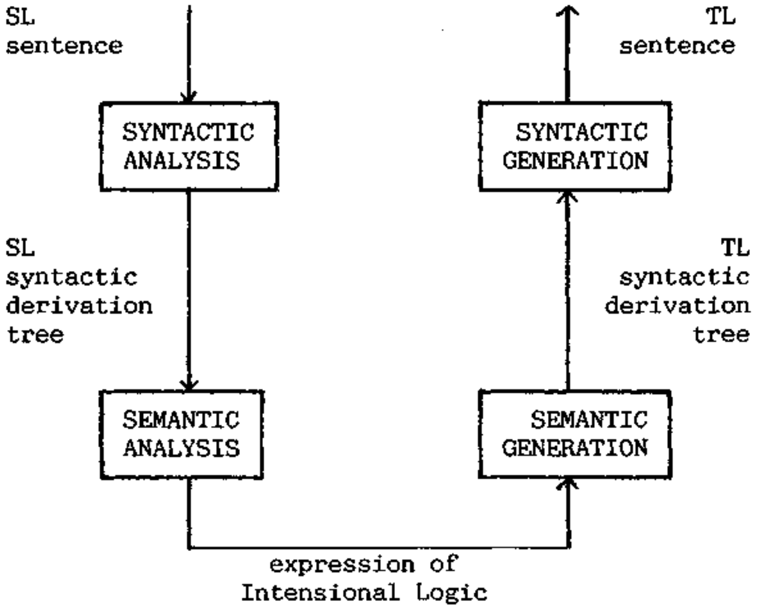
For me these requirements define the domain in which a theoretical discussion on machine translation makes sense. It is hard to compare - on a theoretical level - translation systems that do not obey them or at least try to obey them.

3. There must be some definition of the information that has to be conveyed during translation. Only if there is a

clear definition of information content that a sentence and its translation should have in common, is it possible to evaluate a translation system in this respect. Unfortunately, there appears to be no theory of translation that offers a satisfactory definition.

The obvious way to use Montague Grammar (i.e. M-grammar or some other analyzable version) in a "possible translation" system appears to be the following. Define a Montague grammar for the source language and for the target language. From these grammars analysis and generation components are derived. Then we extend the analysis with a component which translates a syntactic derivation tree into the logic according to the semantic component of the grammar. The generation component is extended with a component which performs the reverse function. So in this approach Intensional Logic is used as an interlingua. This type of system is outlined in (8).

(8)



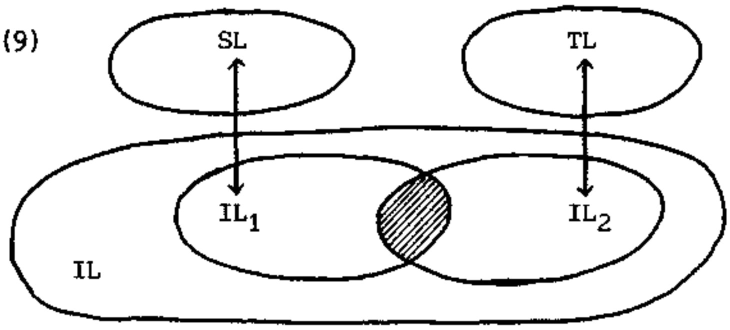
This approach obeys the three requirements: a correct sentence of SL is translated into a correct sentence of TL according to explicit grammars and the information that is conveyed is the meaning in the model-theoretical sense. At first sight this is a very attractive method. It has the additional advantage that knowledge of the world can in principle be formulated in the same logical language as the interlingua, that inferences can be made, etc. I think that long-term research along these lines would be very useful. But in the Rosetta project we have chosen a different approach. Why? Because of the following problems with intensional logic as an interlingua.

1. Montague Grammar has been successful in defining the semantics of a number of natural language constructions, but a lot of work has to be done yet. For translation purposes it is in general not necessary to define in detail what a certain term or construct means, it is sufficient to know that a term or construct of one language means the same as a term or construct of another language. For example, the semantics of belief-sentences may be a problem, but the translation of the verb believe into the Dutch geloven is probably not at all problematic. This is not really a fundamental objection against the use of some kind of intensional logic. The problem is mainly that there is a discrepancy between the actual research in the field of Montague Grammar (directed to a detailed semantic analysis, for small fragments) and what is needed for machine translation (a fairly superficial analysis, with a wide coverage).

2. The second problem is more fundamental. In this approach the information that is conveyed during translation is the meaning in the model-theoretic sense. This is a nice basis for machine translation and certainly preferable to a purely syntactic approach, but there is other information to be conveyed as well, e.g. information on pragmatic and stylistic aspects. In general it seems to be wise to stay as close to the original form as possible (in some sense of the word "form"). Intensional logic is not adequate for carrying this information. One might object that the form of the logical expression expresses information about the form of the sentence too, and this is correct to a certain extent, but making use of the form of logical expressions is in fact in conflict with the spirit of Montague Grammar. As I already mentioned in the introduction, the logical expressions are only a way to define the model-theoretic meaning, their form is not relevant.

3. The third problem is the most delicate one: Montague grammars translate natural languages into a subset of intensional logic. There is no guarantee that two Montague grammars for two languages map them onto the same subset. In (9) the situation is sketched. The grammar of SL maps onto a subset IL_1 of IL. The grammar of TL maps onto a subset IL_2, and consequently the generation component based on this grammar is only applicable to expressions of IL_2. So translation is only possible for the sentences that are mapped onto the intersection of IL_1 and IL_2.

(9)



Notice that there is no independent definition of IL_1 and IL_2. They are only defined indirectly by the mappings that follow from the grammars of SL and TL. Therefore it is very difficult to get to grips with this problem. For solving it, it is not sufficient that the terms of IL_1 and IL_2 are the same, but in addition sentences that are to be translated into each other should get exactly the same logical structure and not just equivalent logical structures.

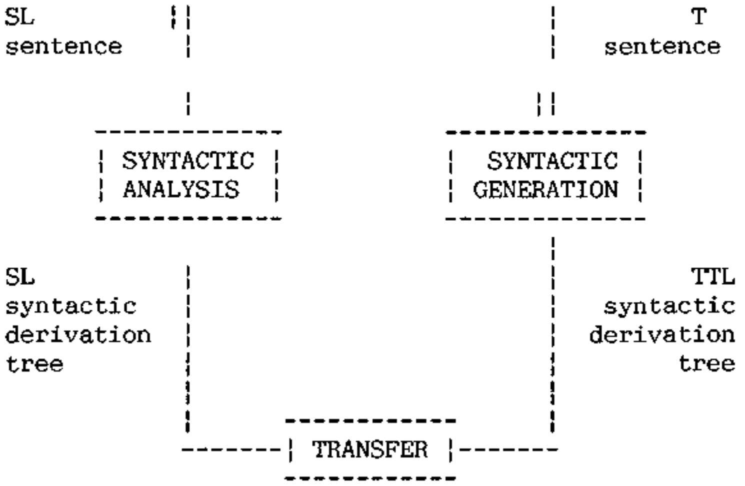
This "subset problem" arises in some guise in all systems - both interlingual and transfer sytems - that translate via deep structures of some kind. In general it is not possible to define the translation for all "possible" deep structures (many of them will not correspond to any sentence at all), but on the other hand it is not possible to characterize what the subset of relevant deep structures is and to guarantee their translation. (Of course this problem does not arise in systems where the correct translation operations cannot be distinguished from the robustness measures.) The only fundamental way to solve this problem appears to be that the grammars of SL and TL are not developed independently, but in close cooperation.

This possibility will be exploited in the next section, but will be left out of consideration here.

There are various other ways in which Montague grammars can be used for machine translation. One of them is to make a transfer system at the level of the intensional logic. In terms of figure (9) the transfer component has to translate from IL_1 into IL_2. Godden (1981) has done work along these lines for Thai to English, making use of Friedman and Warren's parser. The transfer rules have the status of meaning postulates, which gives them a sound semantic foundation. This is very interesting, but has only been worked out for the small fragment grammar of PTQ and does not appear to be easily extensible to larger fragments. Godden wrote in fact a PTQ-like grammar for Thai (i.e. the grammars for the two languages have not been written independently of each other) and added transfer rules for the small set of discrepancies betwen this grammar and the English PTQ grammar. Apart from the problem of the growing set of discrepancies for larger grammars (which ultimately comes down to the earlier-mentioned problem 3), problems 1 and 2 with regard to the use of intensional logic in machine translation are valid here too.

Another possibility of basing a translation system on Montague Grammar is to design a transfer system as outlined in (10) with transfer at the level of syntactic derivation trees.

(10)



In this approach there is an analysis component based on a grammar of SL and a generation component based on a grammar of TL; the transfer component converts syntactic derivation trees of SL into syntactic derivation trees of TL. In the most general version of this approach the transfer rules would convert arbitrary parts of SL derivation trees into arbitrary parts of TL derivation trees. Problems 1 and 2 do not arise here, as intensional logic is not used explicitly. However problem 3, the subset problem, returns here in a different form. The point is that the rules of the TL derivation tree that is yielded by the transfer component need not be applicable.

A different type of Montague-based transfer system is described by Nishida and Doshita (1982). In this system the transfer component converts the logical expression yielded by the analysis component (of which the terms are source language dependent) into a function-argument structure of which the application (in the generation component) yields target language expressions. There is no separate grammar of the target language in this approach.

I discussed the various Montague-based approaches under the assumption that the grammars of source language and target language are developed independently. Some of the problems are alleviated or disappear completely if these grammars are coordinated in some way. One, rather drastic, way of doing this will be discussed in the next section.

Isomorphic M-grammars

After the introduction of M-grammars, compositional grammars that can be used for both analysis and generation, only a relatively small, but essential, step has to be made to arrive at the isomorphic grammar approach. This step is that the grammars of the various languages are not developed independently, but more or less in parallel and are attuned to each other as follows.

For each basic expression in one language there must be at least one corresponding basic expression in the other language with the same meaning. For each syntactic rule in one language there must be at least one corresponding syntactic rule in the other language with the same meaning operation. Grammars that are attuned in this way are called

isomorphic grammars, if the rules obey applicability conditions to which I will come back later.

Given two isomorphic grammars, the translation relation is - informally - defined as follows: two sentences are translations of each other if they are derived from corresponding basic expressions by application of corresponding rules.

Before giving more precise definitions, I will give a simple example of isomorphic grammars for English and Dutch, in (12). The grammar is the same as the one described before. In the middle column of (12) the names of the basic meanings and meaning rules that the two grammars share are given. The grammars define a translation relation between sentences (11a) and (11b).

(11a) The boys are sleeping.
(11b) De jongens slapen.

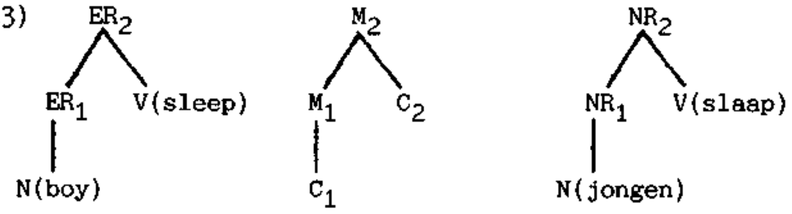
(12) DUTCH		ENGLISH
basic expressions	basic meanings	basic expressions
N(jongen)	C_1	N(boy)
V(slaap)	C_2	V(sleep)
syntactic rules	meaning rules	syntactic rules
NR_1:	M_1	ER_1:
N(jongen) -> NP(de jongens)		N(boy) -> NP (the boys)
NR_2	M_2	ER_2:
NP(de jongens) + V(slaap) ->		NP(the boys) + V(sleep) ->
S(de jongens slapen)		S(the boys are sleeping)

In the example grammar I use the abbreviated notations for S-trees; the rules are characterized by means of an example application.

Note that the relation between basic expressions of Dutch and English need not be one-to-one, although the example may suggest this. For each basic meaning there is a set of basic expressions in each language. The same holds for the rules. For example, NR_2 might also correspond to a rule ER_3, which generates a sentence in the simple present tense. Then the grammars would also define a possible translation relation between de jongens slapen and the boys sleep.

The definition of the translation relation given above can be reformulated more precisely as follows. Two sentences are each other's translation, if they have the same semantic derivation tree, i.e. if they have syntactic derivation trees with the same geometry, of which the nodes are labelled by corresponding rules and basic expressions. The syntactic derivation trees of the example sentences and their semantic derivation tree are given in (13).

(13)



syntactic derivation tree of (11a)

semantic derivation tree of (11a) and (11b)

syntactic derivation tree of (11b)

There are several possible ways of using isomorphic grammars in a translation system; one of them is a transfer system like the one sketched in diagram (10). The global design is the same, but the difference is that the TRANSFER component is now much simpler. The syntactic derivation tree of the source language can be converted into a derivation tree of the target language by a straightforward node-by-node transfer of basic expressions and rules.

Here I will discuss another possibility: the use of semantic derivation trees as interlingual expressions. This lies at hand, since a semantic derivation tree is exactly what translations have in common according to our definitions. In the section on M-grammars I described how a function SYNTACTIC ANALYSIS and a function SYNTACTIC GENERATION can be defined on the basis of the syntactic and the morphological component of an M-grammar. The semantic component of an M-grammar relates basic expressions to basic meanings and syntactic rules to meaning rules. On this basis two additional functions can be defined:

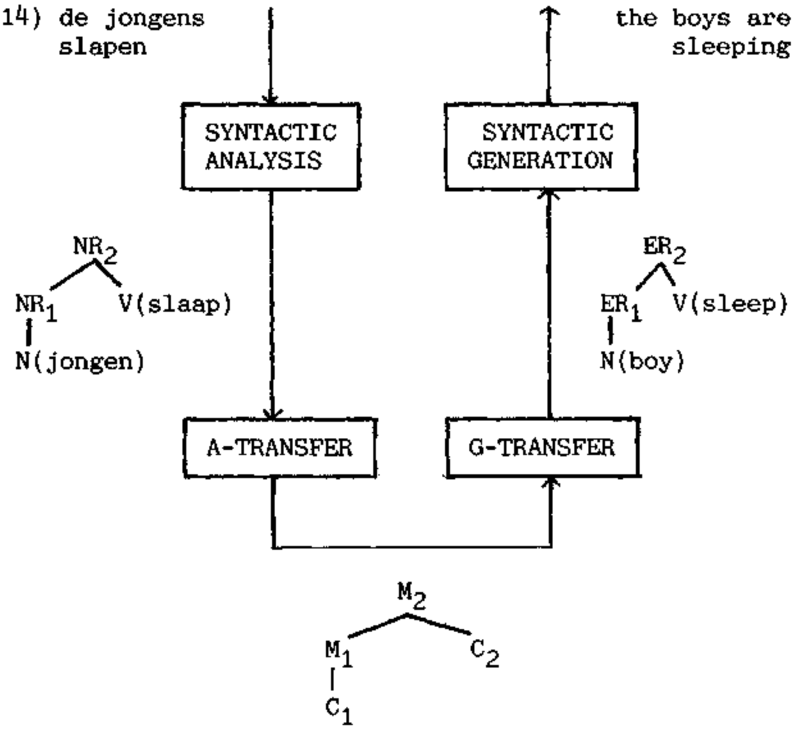
A-TRANSFER applies to a syntactic derivation tree and yields the set of corresponding semantic derivation trees.

G-TRANSFER applies to a semantic derivation tree and yields the set of corresponding syntactic derivation trees.

Both A-TRANSFER and G-TRANSFER are simple functions, defined in terms of local operations on nodes.

The result is an interlingual system as outlined in (14) for Dutch to English.

(14)



I will now give a more precise definition of isomorphy. First, a syntactic derivation tree is called **well-formed** if it defines at least one sentence (i.e. the rules in the derivation tree are applicable). A semantic derivation tree is called well-formed if one of the derivation trees to which it corresponds (according to the semantic component) is well-formed. Two grammars G and G' are called **isomorphic** if each semantic derivation tree that is well-formed with respect to G is also well-formed with respect to G', and vice versa. Note that isomorphy is an equivalence relation between grammars and that the definition can be extended easily to sets of more than two grammars.

The definitions imply that if a translation system as outlined in (14) is based on isomorphic grammars, we know that the analysis of a sentence in the source language yields a semantic derivation tree, the generation component will always yield a correct sentence of the target language. Translations defined in this way have the same meaning, they have the same semantic derivation tree, they have similar syntactic derivation trees, and they may have completely different surface trees. So in this framework the information that is conveyed during translation is not only the model-theoretical meaning, but also the way in which this meaning is derived. This could be called the **compositionality principle of translation.**

This approach avoids the earlier mentioned problems with intensional logic as an interlingua. The hardest of these problems was the "subset problem", which arises not only in a system with a logical interlingua, but also in a system with transfer on syntactic derivation trees (as in (10)), if the grammars of source and target language are developed independently. In principle this problem is solved in a system based on isomorphic grammars, but it would be somewhat misleading to state it that way. A remaining problem is that in the syntactic framework we use, it is not yet possible to prove **formally** whether two grammars are isomorphic or not. For various kinds of grammars a formal proof is possible, but not yet for grammars with the syntactic power of M-grammars. However, even without a formal proof the approach is an important step forward.

In practice the process of grammar writing proceeds as follows. A set of compositional rules R is written for handling a particular phenomenon in language L, a corresponding set of compositional rules R' is written for handling the corresponding phenomenon in language L'. The rules R should be complete for the expected set of input expressions, the rules R' should be complete for the corresponding set of input expressions of L' (their "translations"). The most important practical difference between this and other approaches may be that here the grammars are written with translation in mind. Because of the reversibility of the grammars the rule writers can focus their attention on writing compositional (i.e. generative) rules in parallel and on the applicability of these rules to the expected inputs.

Diagram (14) shows the global design of the systems which are being developed in the Rosetta project. This is a research project on machine translation at Philips Research Laboratories, Eindhoven. A few years of preparatory research resulted in the isomorphic grammar approach outlined here and in two experimental systems based on this approach, Rosetta1 and Rosetta2. A fairly large six-year project has started this year (1985), in which more sophisticated systems, Rosetta3 and Rosetta4, will be developed, for Dutch, English and Spanish.

The Rosetta approach is interlingual. Since interlinguality can be defined in various ways, this statement may cause misunderstandings. Therefore I will give three possible definitions of interlinguality and indicate which of them are applicable here.

1. A system is interlingual if there is an intermediate meaning representation which has the "same distance" to the sentences of the source language and the target language. Note that according to this definition even a bilingual one-direction translation system may be interlingual. This definition is clearly applicable to the Rosetta systems.

2. A system is interlingual if an interlingua is defined for a given set of languages in such a way that for each of these languages an analysis component can be defined that translates from that language into the interlingua and a generation component that does the reverse. So the combination of an analysis component for language L and a generation component for language L' is a translation system from L into L'. This definition is also applicable to the Rosetta systems.

3. A system is interlingual if it uses an interlingua

which is "universal", i.e. which can be used for expressing any meaning of any sentence in any natural language. Obviously, the Rosetta approach is not interlingual in this sense.

In the Rosetta project we aim at developing an interlingual system, according to definition 2. This is certainly more ambitious and more difficult than developing a purely bilingual system. Rosetta3 is being developed for three languages in order to find out what the price of this multilingual approach is, in comparison with the bilingual approach according to definition 1.

Concluding remarks

In the section on Montague Grammar and machine translation I formulated three requirements on translation systems: explicit grammars of source language and target language, translation of correct sentences into correct sentences according to these grammars and a definition of what has to be conveyed during translation. The isomorphic grammar approach satisfied the first two requirements; with respect to the third requirement a step forward has been made in comparison with using Intensional Logic as interlingua. In the Rosetta systems it is not only the model-theoretical meaning that is conveyed, but also the way in which this meaning is derived from basic meanings.

I mentioned three problems with using Intensional Logic as an interlingua. The first problem was that a meaning representation in Intensional Logic may require a more detailed meaning analysis than is needed for translation purposes, because for translation we are mainly interested in equality of meanings. This problem is solved by using semantic derivation trees as interlingual meaning representations, in which the unique names of basic meanings and meaning rules serve exactly to express the equality of meaning of basic expressions and syntactic rules, respectively. The second problem was that expressions of Intensional Logic only convey the meaning in the strict model-theoretical sense. As I pointed out, semantic derivation trees indicate in addition the way in which the meaning is derived. They may also be used to convey other information than the meaning. If two basic expressions or two syntactic rules (of the same language) have the same meaning, but differ in some other aspect which is relevant to translation, we may assign different names to the corresponding basic meanings and meaning rules. The solution of the third problem, the subset problem, has been the main motivation for the isomorphic grammar approach. If the grammars of the source and the target language are isomorphic, each interlingual expression generated by the analysis component can be processed by the generation component.

In this paper I have illustrated the isomorphic grammar approach by means of very simple examples. This may leave you with the impression that isomorphic grammars can only define very trivial translation relations. The following remarks should indicate the potential power of the approach.

1. First and foremost it is important to notice that the rules and the basic expressions of the grammars are chosen with translation in mind.

2. Syntactic rules may perform powerful operations on syntactic trees, e.g. permutations, substitutions and deletions, as long as the conditions on M-grammars are obeyed. So the correspondence between syntactic rules of different grammars as required by the isomorphy relation does not imply similarity of the surface structures.

3. Basic expressions need not be terminals (i.e. S-trees consisting of one node), but may also be complex S-trees. This is especialy useful for idiomatic expressions (e.g. to make up one's mind), which are primitive from a semantic point of view, but complex from a syntactic point of view. The same mechanism is used in the case where a word in one language corresponds to a complex expression in the other language, even if this complex expression would not be considered as an idiom in that language (e.g. the translation of the Spanish verb madrugar into the English expression to get up early). On the other hand basic expressions may correspond to "deeper", possibly more abstract, notions than those denoted by words.

4. Corresponding basic expressions of two languages need not have the same syntactic category, under the conditions that these different categories correspond to the same semantic type. Obviously allowing such a mismatch of categories imposes conditions on the rest of the grammars

which are not always easy to fulfill. In Landsbergen (1985) I dealt with a particular example of this: the translation of the English verb to like into the Dutch adverb graag.

I hope that these points make it clear that the isomorphic grammar approach is in principle quite powerful. The practical feasibility should be shown and has to a certain extent been shown already by the actual systems developed in the Rosetta project.

In conclusion, I hope to have shown that application of Montague Grammar in machine translation may yield the best results if it is applied in a "creative" way. The main influence on the Rosetta systems has been exerted by the compositionality principle. This plays an important role in Rosetta, not only by relating form and meaning of one language, but also by inspiring us to formulate a compositionality principle of translation which relates form and meaning of various languages. These principles should not be interpreted as refutable theories of language or translation (cf. Partee 1982 on the status of the compositionality principle), but as **guiding** principles for the construction of grammars and translation systems.

And there I would like to stop.

Pete WHITELOCK: Well, I would like to ask a question. In your approach, if you have a sentence which is ambiguous in translation but non-ambiguous in the source language as far as we can tell, do you have to essentially give it two analyses so that you can get the two translations?

LANDSBERGEN: If a sentence is ambiguous in translation, i.e. if it has more than one translation, there are two possibilities. The first one is that these translations are paraphrases, corresponding to the same meaning. In that case there is only one analysis of the SL sentence and the ambiguity arises only in the generation component. The second possibility is that these translations correspond to different meanings. In that case there must be two analyses of the SL sentence. It is not always easy to decide if for a particular phenomenon we have to create a semantic ambiguity or if it can be described as having one "encompassing" meaning. In Rosetta this decision will not only depend on what is most elegant in one language, but it will also be influenced by the other languages.

Doug ARNOLD: The language that the grammar defines is something rather close to the surface of the languages - it's something, I imagine, like morphologically and syntactically analyzed English, or morphologically and syntactically analyzed Dutch, and so on. That's right, isn't it? You have only one set of ...

LANDSBERGEN: One level of representation, yes. However, during the generation process of a sentence, we start with rather abstract representations, which are gradually transformed into surface representations. But they are all S-trees, so essentially there is one level of representation.

ARNOLD: What is your feeling about having more levels of representation, so that in fact the 'tuning' of the grammars would be between grammars that essentially generate semantic representions of appropriate languages or, let's say, F-structures of the languages, logical forms of the languages, something like that? Do you have an argument against using other levels of representation, for instance?

LANDSBERGEN: Well, in the first place it is the other way round. There should be arguments for having more levels. But leaving that aside: in Rosetta the syntactic rules have a clear effect on both the form and the meaning. If there are more levels between form and meaning the effect of the rules may be harder to understand. But the main problem with having more levels is the "subset problem" I discussed in my presentation. If there are more levels, the representations at the deepest level will be the result of a number of translation steps between the various levels. It is hard to charaterize independently the subset of deep representations that correspond to sentences. This makes it difficult to guarantee that this subset is actually translated.

ARNOLD: I think that the subset problem is one of the major problems. Could I just say what the argument for having other levels is: there are more superficial differences between languages than there are non-superficial ones; so languages configure differently, let's say. So a non-configurational representation makes translation easier. You can phrase that within a different theory if you want, but there is that sort of intuition around. That would motivate having other levels than one.

LANDSBERGEN: I forgot to mention another objection against having deeper levels. After going to a deeper level of analysis, information that is useful for translation may get lost. E.g., at the F-structure level of LFG there is no information about the surface order of constituents, although this may be important for choosing the most plausible interpretation with regard to scope. Of course, the idea that languages have more in common at a deeper level of analysis than at the surface is an argument in favour of having more levels. But in our approach the derivational history is such a level; our assumption is that languages have much in common at the level of derivational history.

But I interrupted you - please continue.

ARNOLD: My point really relates to the subset problem. Why don't you just say, for the cases where there is a failure of intersection between source language and target language ILs, that there is no translation in those cases? Why don't you adopt a more restrictive view of translation, distinguishing, say, between translation and paraphrase?

LANDSBERGEN: There are two reasons. The first reason is a practical one. We make an interlingual system with interaction with the user during analysis, in case of ambiguities. If in such a system the analysis has been successful and has yielded an interlingual expression, one wants to be sure that the generation component provides a translation.

ARNOLD: Why? If what you are doing is translation why don't you ...

LANDSBERGEN: Well, I think of the application of this system in an electronic mail environment. It is unacceptable if an analyzed message is not translated.

ARNOLD: No, I was pressing you for a theoretical argument.

LANDSBERGEN: OK, that was a practical point.

ARNOLD: Why do you call the result of that sort of activity 'translation' and not something else? If the source text and a target text don't share at least one IL representation, why do you want to claim that they are translations?

LANDSBERGEN: The theoretical argument is that if the source text and the target text do not share an IL representation, it may still be the case that they have logically equivalent representations. So in that case they have the same meaning and may be called each other's translations, but due to fairly arbitrary differences in the two grammars, they are not recognized as such by the system.

Henry THOMPSON: I suspect that really the right place to get an answer to this is in Partee's work, but on a quick understanding of what you said, can you disabuse me of the notion that a Montague grammar with constraints imposed on it which you refer to to ensure parsability is any different from a context-free grammar? Is there an obvious way to characterize the difference between a Montague grammar so restricted, particularly the S-rules that are associated with it, and something that I would think of as a context-free grammar with a rule-to-rule relationship between the syntactic rules and some compositional semantics? Is there anything that really remains of Montague in this? That is, I guess, what it comes down to.

LANDSBERGEN: Montague's own example grammars are more or less context-free, but in Rosetta we use a transformational extension of Montague Grammar (cf Landsbergen 1981). Our rules are powerful, they can perform permutations, deletions etc. Indeed, our surface grammar is context-free in its weak generative capacity, but the grammar as a whole defines a non-context-free subset of this. Actually, our formalism is undergoing some changes at the moment. We are going to make a distinction between meaningful rules that contain information relevant for translation and on the other hand purely syntactic transformations. These transformations are not involved in the isomorphy relation and can be defined for each language separately.

THOMPSON: What does the parser then look like as a result of all this?

LANDSBERGEN: The parser consists of two parts: the surface parser and the M-parser. The surface parser produces a set of candidate surface trees for the input sentence. The M-parser applies the analytical rules of the M-grammar to a surface tree and breaks it down into smaller parts, ultimately into basic expressions. If the M-parser is successfully applied, i.e. if the surface tree is correct, the result is a derivation tree. The surface grammar is weakly equivalent to a context-free grammar, it is similar to a recursive transition network grammar. The rules of the M-grammar are more powerful.

THOMPSON: Thank you.

Graeme RITCHIE: Could I ask you about idioms? I'm a bit puzzled about what you said about idioms. It sounded from what you said as if, if one of the languages had a phrasal, idiomatic expression of some concept, there had to be a basic concept in the logic and a basic expression in the semantics corresponding to that which had that semantic compositional structure.

LANDSBERGEN: No no no. Not that.

RITCHIE: Well you said that idioms may have whole semantic derivation trees.

LANDSBERGEN: I said that idioms correspond to compound basic expressions. I am sorry about all these different kinds of trees, but here we have to make clear distinction between S-trees and derivation trees. All basic expressions are S-trees, but usually they consist of one node. An idiom is a compound S-tree, consisting of more than one node. It is a basic expression from a semantic point of view, but it is a compound expression from a syntactic point of view. For example, *to lose one's temper* will be represented as an S-tree with *lose* and *temper* in it, but its meaning is not derived compositionally from these parts.

RITCHIE: For the semantics that's derived from it, to do the translation the other language has to have some expression which has that as its semantics?

LANDSBERGEN: Yes, the expression in the other language may be atomic or may be an idiomatic expression. It may also be a compound expression that one would not be inclined to call an idiomatic expression in that language. For instance, a possible translation of *to lose one's temper* into Dutch is kwaad worden, an idiomatic expression of Dutch, but in the translation system it has to be treated in the same way as an idiom.

RITCHIE: I can understand that. I didn't see what the adjective 'compound' implied with your various levels.

Karen SPARCK JONES: You said quite explicitly you're not dealing with ill-formed text at the moment, fragments and things like that. Is it perfectly obvious how, when you've got around to it, in principle you would do this in this kind of approach?

LANDSBERGEN: I did not deal with ill-formed input in my paper, but in the actual system Rosetta2 we try to deal with it. For sentences that do not fit into the system's grammar, there are several robustness measures, partially similar to those in other systems. For instance, if the surface parser is not able to make a complete parse, it will look for a "cover" of the sentence by the largest constituents it has found. It puts them together under a special node with category UG (for "Ungrammatical"). In the next phase, the M-parser, there is an analytical rule that is able to cope with a UG. At the moment this rule is very simple: it splits up the tree into its immediate subtrees. Each of the subtrees is then analyzed and translated further in the usual way. In the generation component the translated subtrees are combined again by a rule corresponding to the beforementioned analysis rule for a UG. So the net result of all this is that an incorrect sentence is split up into correct parts which are translated separately.

Nick OSTLER: Do you have any experience of working practically with, say, three languages? I don't know whether it's only in the future that you are going to bring in Spanish, but it seems that you envisage a real-time interaction between linguists working together drafting these grammars, and presumably that's just about feasible when you've got two languages. If you've got three, establishing your isomorphisms will be twice as difficult again, I suppose, and if you were to add more languages of course it would rapidly become completely infeasible.

LANDSBERGEN: I have some experience with writing isomorphic grammars for Dutch, English and Italian, for Rosetta1, but these grammars were small and I did that on my own, so there

I did not encounter the problems you are talking about. The second version of the system, Rosetta2, has larger grammars, which have been designed for the same three languages, but they have been worked out only for Dutch and English, due to a change in our planning. We are now working with a group of linguists and the actual writing of the rules has to start yet. We will first make global isomorphic schemes for the three grammars. Then these grammars will be worked out in detail, separately. If serous problems arise in that phase, there may be feed-back to the isomorphic scheme.

OSTLER: But you haven't done it very much as yet? This is your plan for the six-year project.

LANDSBERGEN: Yes. The six-year project itself is very young. It started at the beginning of this year [1985].

OSTLER: So your experience is just of doing English and Dutch. There has been the PHLIQA project (Landsbergen 1976).

LANDSBERGEN: That was in a way the predecessor of this project.

OSTLER: Did that involve multilingual or just bilingual ...

LANDSBERGEN: No, it was just English. PHLIQA was a question-answering system. So we have experience with building large systems, but not with building a large interlingual translation system with a group of linguists. Note that the isomorphic approach is also feasible for bilingual translation. We have chosen to work on three languages, because we are interested in interlingual applications and want to investigate to what extent the multilingual approach is feasible. One of the goals of the project is to find out what the price of this multilinguality is. I hope to report on this in a few years.

References

Dowty, D.R., Wall, R.E. & Peters, S. 1981. **Introduction to Montague Semantics.** Dordrecht: D. Reidel.

Friedman, J. & Warren, D.S. 1978. A Parsing Method for Montague Grammars. **Linguistics and Philosophy** 2, 347-372.

Godden, K. 1981. **Montague Grammar and Machine Translation between Thai and English.** PhD dissertation, University of Kansas.

Janssen, T.M.V. 1986. **Foundations and Applications of Montague Grammar, Part** I. CWI Tract 19. Amsterdam: Centre for Mathematics and Computer Science.

Landsbergen, S.P.J. 1976. Syntax and formal semantics of English in PHLIQA1. In L. Steels, ed., **Advances in Natural Language Processing.** Antwerp: University of Antwerp.

Landsbergen, S.P.J. 1981. Adaptation of Montague Grammar to the Requirements of Parsing. In Groenendijk, J.A.G., Janssen, T.M.V. & Stokhof, M.B.J. **Formal Methods in the Study of Language, Part 2.** MC Tract 136, Mathematical Centre, Amsterdam, 399-420.

Landsbergen, S.P.J. 1985. Isomorphic Grammars and their Use in the Rosetta Translation System. In King, M., ed., **Machine Translation Today.** Edinburgh: Edinburgh University Press.

Montague, R. 1970a. Universal Grammar. **Theoria** 36, 373-398. Reprinted in Thomason, ed., 222-246.

Montague, R. 1970b. English as a Formal Language. In B. Visentini et al, eds., **Linguaggi nella società e nella technica.** Milan: Edizioni di Comunità, 189-224. Reprinted in Thomason, ed., 108-221.

Montague, R. 1973. The Proper Treatment of Quantification in Ordinary English. In Hintikka, J., Moravcsik, J. & Suppes, P., eds. **Approaches to Natural Language.** Dordrecht: D. Reidel, 221-242. Reprinted in Thomason, ed., 247-270.

Nishida, T. & Doshita, S. 1982. An English-Japanese Machine Translation System based on Formal Semantics of Natural Language. In Horecky, J., ed., **COLING 82: Proceedings of the Ninth International Conference on Computational Linguistics**. Amsterdam: North-Holland, 277-282.

Partee, B.H. 1976. Some Transformational Extensions of Montague Grammar. In Partee, B.H., ed., **Montague Grammar**. New York: Academic Press, 51-76.

Partee, B.H. 1982. Compositionality. In Landman, F. & Veltman, F., eds. **Varieties of Formal Semantics** (Proceedings of the 4th Amsterdam Colloquium). Dordrecht: Foris, 281-312.

Thomason, R.H., ed. 1974. **Formal Philosophy: Selected Papers of Richard Montague**. New Haven: Yale University Press.