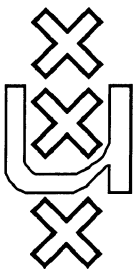
Institute for Language, Logic and Information

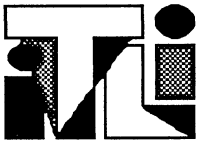
A MATHEMATICAL MODEL FOR THE CAT FRAMEWORK OF EUROTRA

Theo M.V. Janssen

ITLI Prepublication Series
for Logic, Semantics and Philosophy of Language LP-88-09



University of Amsterdam



Institute for Language, Logic and Information
Instituut voor Taal, Logica en Informatie

A MATHEMATICAL MODEL FOR THE CAT FRAMEWORK OF EUROTRA

Theo M.V. Janssen
Department of Mathematics and Computer Science
University of Amsterdam

Abstract. Using universal algebra, a mathematical model is developed for the CAT framework of Eurotra (the EEC translation project). It is shown that there are several advantages of such a mathematical approach. The model turns out to be closely related to Montague's Universal Grammar.

Received July 1988

To be published in:
Computerlinguistik und ihre theoretische Grundlagen
Proceedings Symposium Saarbrücken, 1988
Informatik Fachberichte, Springer Verlag, Berlin

Correspondence to:

Faculteit der Wiskunde en Informatica
(Department of Mathematics and Computer Science)
Roetersstraat 15
1018WB Amsterdam

or

Faculteit der Wijsbegeerte
(Department of Philosophy)
Grimburgwal 10
1012GA Amsterdam

A mathematical model for the CAT framework of Eurotra

Theo M.V. Janssen
Dept. of Mathematics and Computer Science
University of Amsterdam
Nieuwe Achtergracht 166
1018 WV Amsterdam
The Netherlands

1. Introduction

Eurotra is the machine translation project of the EEC. The basic ideas for the design of the system are given by the CAT framework, which is, together with various relaxations, presented in several publications (e.g. Arnold e.a., 1985, 1986, des Tombes e.a. 1985, Arnold & des Tombes 1987). In the present paper a mathematical model for the CAT framework will be developed. This will be a model of the structural aspects of the framework, such as the structure of the grammars and of the translation steps. The model uses notions and results from universal algebra; a branch of mathematics which deals with structures and their relations. The model is in a certain sense the same as the CAT framework, but it is build with different tools. Eurotra is a project of ongoing research with continuous practical experience, and this might cause changes in the original framework. Since the present paper is mainly based upon the publications mentioned above, it does not necessarily describe the present situation correctly (for your information, the author is not personally involved in the project). The aim of this paper is, however, not to present some version of Eurotra, but to argue for a more abstract and more mathematically based approach to Eurotra (and other machine translation systems). It will be shown that there are several advantages of such a mathematical approach. It brings new insights in the framework (see sections 4 and 5), and gives us a new appreciation of certain Eurotra proposals (see sections 6 and 7). Furthermore, the mathematical model for Eurotra will, I expect, be a good starting point for investigating later stages of the Eurotra system.

The aspects of the Eurotra system that are relevant for the discussion of the present paper are the following. It is a transfer system that translates sentences. In the course of this translation process the sentence is analysed in different ways according to different criteria. Each of these analyses is considered as an expression in some analysis language. The process of translating a sentence is a process which transputs the sentence through the several analyses for the source language, and next in reversed order for the target language. Each of these steps from analysis language to analysis language is considered as a translation step of the same nature as the 'real' transfer step from the last source language analysis to the first target language analysis.

The different analysis languages are mentioned below (using the terminology from Arnold & des Tombes 1987). In the Eurotra publications the discussion usually is restricted to the analyses

3, 4 and 5, and so will be done in the present paper.

1. *ENT (=Eurotra normalized text)*

The input and output of the system are unanalysed expressions, presented in some normalized form.

2. *EMT (=Eurotra morphologically analysed text)*

At this level the words of the expressions are morphologically analysed. So instead of *works* a expression will contain something like *work*[third person singular present tense].

3. *ECS (=Eurotra Constituent Structures)*

Constituent structures are assigned to morphologically analysed expressions. The order of the words in the structure is the same as in the surface expression.

4. *ERS (=Eurotra Relational Structures)*

The syntactic relations of an expression are given in a labelled tree. The surface order needs not to be respected; for instance a direct object of a verb is connected immedeately with the verb it belongs to.

5. *IS (=Interface Structures)*

The semantic relations of a phrase are given by means of a labelled tree.

2. Algebras as syntax

Several considerations have influenced the design of the CAT framework. One of these is the principle of compositionality of translation. It reads, in my formulation, as follows:

The translation of an expression is a function of the translations of its parts
and of the way they are syntactically combined.

This principle I will take as point of departure for the development of a mathematical model. The other considerations will not be mentioned here, since the compositionality principle is sufficient for that purpose.

The principle of compositionality speaks about the parts of an expression. So there has to be in the model a formal source for determining what the parts of an expression are. The information on how expressions are formed is given by the syntax of a language, and consequently the rules of the grammar determine in our model what the parts of an expression are. This means that the rules build new expressions from old expressions, and we will call these old expressions *parts*.

Let us consider an example. Suppose that a rule, called S_1, builds *John takes the apple away* from *John* and *take away* and *the apple*. Then these three expressions are the (immediate) parts of this sentence. If one would prefer to consider this sentence as consisting of two parts, then one should not have rule S_1 in the grammar, but a rule S_1' that builds this sentence from the two parts *John* and *take the apple away*.

A syntax with the kind of rules as described above is a very specific example of what is called in mathematics 'an algebra'. Informally stated, an algebra is a set with functions defined on that set. After the formal definitions some examples will be given.

Definitions.
A an **Algebra** *A*, consists of a set A called the **carrier** of the algebra, and a set F of functions defined on that set. So *A* = <A,F>. The elements of the carrier are called the **elements of the algebra**. A function is called **n-ary** if it takes n arguments. Instead of function, we often use the name **operator**. If an operator is not defined on the whole carrier, it is called a **partial operator.** If $F(E_1,E_2, ..,E_n)=E$, then E_1,E_2, .., and E_n are called **parts** of E.

The notion *set* is a very general notion, and so is the notion *algebra* which has a set as one of its basic ingredients. I will give three examples of a completely different nature. The first is the algebra with as carrier the set **N** of natural numbers {0, 1, 2, 3, ... } and with addition and multiplication as operators. The second example has a more linguistic character. The carrier is the set of all finite strings of words which can be formed from the entries in a given dictionary, and the operator is concatenation. A third example consists of the set of trees (consituent structures) and as operation making a new tree from two old ones by giving them a common root. In order to avoid the misconception that everything is an algebra, finally a non-example. Take the second algebra (finite strings of words with concatenation), and add an operator that counts the length of a string. Then it is not an algebra any more, since the lenghts (natural numbers) are not elements of the algebra.

As argued above, it is a consequence of the principle of compositionality of translation that the grammars have to be algebras. And indeed, in the CAT framework for Eurotra all grammars are algebras (although this terminology is not used). The first two levels of analysis in Eurotra (unanalysed sentences and morphologically analysed sentences) are algebras with concatenation as operator. The three other levels (ECS, ERS and IS) concern labelled trees and the operators mostly combine two or more trees to a new tree by providing them with a new common root.

In the linguistic examples we have met operators of different nature. In the second example of the above paragraph the operator was concatenation of strings, whereas in the example in the beginnings of this section it was a substitution: *the apple* is placed between *take* and *away*. An operator which introduces a new word, viz. a determiner, is the Eurotra operator S_{def} that produces *the apple* from *apple* . We have defined the notion *part of E* as the inputs of the operator producing E. Hence, according to rule S_1, *take away* is a part of *John takes the apple away,* whereas it does not occur as substring of that sentence. And *the*, which intuitively might be considered as a part, is according to S_{def} not a part of *the apple* . Rules that involve unification are frequently used in Eurotra. They give us other examples of rules that build a compound expression from parts that are not parts in the naive sense. These examples show that *part* is a now a theoretical notion and not an empirical one; the formal notion and the intuitive notion coincide if the syntactic rules are concatenation rules.

Next we will meet a subclass of the algebras, viz. the finitely generated algebras. All Eurotra algebras belong to this class. To give an example, consider in the subset {1} in the algebra of natural numbers defined above. By application of the operator + to elements in this subset, that is by calculating 1 + 1, one gets 2. From the then obtained set one can produce 3 (by 2+1, or 1+2), and in this way the whole carrier can be obtained. Such a subset is called a *generating set* for the algebra. If an algebra has a finite generating set, the algebra is called *finitely generated.* If we have in the same algebra the subset {2}, then only the even numbers can be formed. Therefore the

subset {2} not a generating subset of the algebra of natural numbers. On the other hand, the even numbers form an algebra. This fact that can be explained as follows. If one starts with some set, and add all elements that can be produced from the given set and from already produced elements, then one gets a set that is closed under the given operators. Hence it is an algebra. This method can be applied to any subset in any algebra.

Definitions

Let A = <A,F> be an algebra, and H be a subset of A. Then <[H],F> denotes the smallest algebra containing H, and is called the by H **generated subalgebra.** If <[H],F> = <A,F>, then H is called a **generating set** for A. The elements of H are called **generators.** If H is finite, then A is called a **finitely generated algebra.**

So for the first example of an algebra, a finitely generated algebra, holds <**N**, {+, ×}> = <[{1}], {+, ×}>. Another example of an algebra was the set of all strings of entries in a lexicon; this algebra is finitely generated with the lexicon as generating set. An algebra that is not finitely generated is <**N**, × >, the natural numbers with multiplication.

The terminology of Eurotra is different from the algebraic terminology. They use the name *constructor* instead of *operator*, *atom* instead of *generator*, and write <C,A> where we would write <[A],C>. If we may understand T as an abbreviation for translation, the name CAT framework (or <C,A>-T framework) can now be understood.

3. Terms as production processes

The compositionality principle states that the translation of an expression is determined by the translations of its parts and the way in which they are syntactically combined. The latter clause accounts of course for the the fact that the same parts can be used in different ways, yielding different expressions (e.g *John loves Mary* vs *Mary loves John*). So from compositionality it follows that this 'way of production' is crucial for the purpose of translating. Therefore it is useful to have a representation for such a production process or derivational history. Below an example of a derivational history and its representation will be given.

Consider the sentence *John finds the apple* . According to the Eurotra rules this sentence is formed as follows. The operator C_{def} is applied to the noun *apple,* forming the noun phrase *the apple.* Next the operator C_{VP} is applied to the just formed noun phrase and the verb *find,* yielding the verb phrase *finds the apple.* Finally C_S is applied to this verb phrase and *John* . This production process is represented by the following sequence of symbols:

$C_S(John,(C_{VP}(find, C_{Def}(apple)))$.

This method for representing a formation process, viz. by means of bracketing, operator symbols and generators, can be used in any algebra. Such expressions are called *terms.*

Definition

Let B = <[B], F> be an algebra. Introduce for each element in B a distinct symbol b, and for each

operator in F a distinct symbol f. Then $T_{B,F}$, the set of **terms** over <[B], F> is defined as follows

1) for element in B the corresponding symbol $b \in T_{B,F}$

2) if f corresponds with an n-ary operator, and if $t_1, t_2, \ldots t_n \in T_{B,F}$,

then $f(t_1,t_2, \ldots t_n) \in T_{B,F}$.

In case we do not want to be explicit about the set of constants, we may use the algebra itself as subscript (as in T_B).

Terms can be combined to form new terms. An example was the combination of the term $C_{Def}(apple)$ with *find* to form the term $C_{VP}(find, C_{Def}(apple))$. Thus the terms over an algebra form an algebra again, and this algebra is called a *termalgebra*. There is a simple relation of the terms to the elements in the original algebra. With the term $C_{Def}(apple)$ corresponds an element which is found by evaluating the term, i.e. executing the operator on its arguments. Note that different terms may evaluate to the same element, and the evaluation of a term can be very different from the term itself.

As I argued, it follows from the principle of compositionality of translation that the terms give the relevant information for translation. And indeed, the translations between the analysis languages of Eurotra are defined not between the algebras for ECS etc. themselves, but on the corresponding termalgebras. Hence translations are mappings from termalgebras to termalgebras; such mappings will be considered in the next section. The evaluation of term in such an termalgebra (in Eurotra) is a linguistic analysis tree. A difficulty with the role of terms in the Eurotra framework is that linguists are not used to them, and have therefore no intuitions about their linguistic acceptability. They probably prefer to read evaluated terms, called *inspection trees* by the proposers of the CAT framework. But in the translation process itself such inspection trees play no role.

4. Homomorphisms as compositional translation

The principle of compositionality of translation does not only tell us which objects are to be translated, but also in which way this translation has to be performed. Suppose we have an expression obtained by application of operation f_A to arguments $a_1, \ldots, a_n$. Then the translation into B should be obtained from the translations of its parts, hence by application of an operator g_B (corresponding with f) to the translations of $a_1, \ldots, a_n$. So, if we let T denote the translation function, we have

$T(f_A(a_1, \ldots, a_n)) = g_B(T(a_1), \ldots T(a_n))$.

In Eurotra such a translation mapping is called a *strictly compositional translation*, and in algebra it is called an *homomorphism*. In the CAT-framework the translations indeed are homomorphisms between termalgebras.

A homomorphism h from an algebra A to algebra B is, intuitively speaking, a mapping which respects the structure of A in the following way. If in A an element a is obtained by means

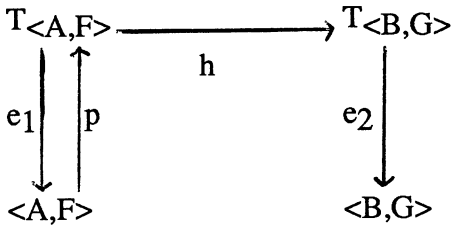
of application of an operator F , then the image of *a* can be obtained in *B* by application of an operator corresponding with F. The structural difference that may arise between *A* and *B* is that two distinct elements of A may be mapped to the same element of B, and that two distinct operators of *A* may correspond with the same operator in *B*.

Definition

Let <A,F> and <B,G> be algebras. A mapping h: A →B is called a **homomorphism** if there is a mapping h': F → G such that for all f ∈F and all $a_1, \ldots a_n \in A$ holds $h(f(a_1, \ldots ,a_n)) = h'(f)(h(a_1), \ldots ,h(a_n))$.

As a matter of fact, we have already met a mapping with these properties. The operator which evaluates aterm is a homomorphism, or, in the Eurotra case, the operator that produces an inspection tree from a given derivational history. But the more important in our approach is the role homomorphisms have in the translation procedure.

By the introduction of terms and homomorphisms all ingredients are present which are needed in order to define what compositional translation is. A compositional translation from algebra <A,F> to algebra <B,G> is an homomorphism from $T_{<A,F>}$ to $T_{<B,G>}$. So the translation of an element a ∈ A is obtained by first finding its derivational history in $T_{<A,F>}$, then homomorphically translating it into $T_{<B,G>}$ and finally evaluating the thus obtained expression. This process is summarized in figure 1.



h : translation homomorphism; e_1,e_2: evaluation homomorphisms,
p: parsing, finding a corresponding term for a given element in <A,F>

Figure 1. *The basic model for compositional translation*

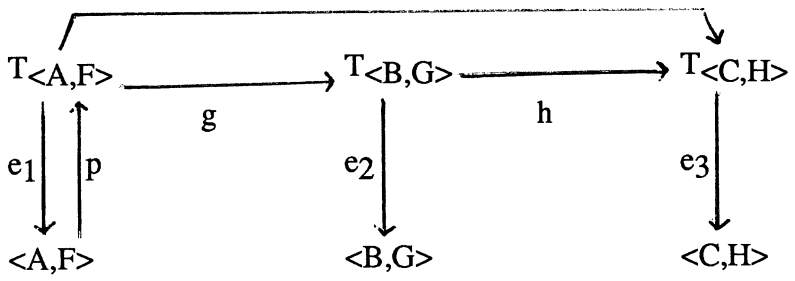
Indeed, in the CAT framework all translation steps are homomorphisms. It requires, however, some further steps to arrive at the model for the CAT framework when starting from the above model for compositional translation. An obvious difference is that compositionality speaks about translating from one language to the other, whereas in a Eurotra translation several analyses languages are involved. In building the model, we need a mathematical result, stating that the composition of two homomorphisms is again an homomorphism.

Theorem

Let <A, F>, <B, G> and <C, H> be algebras, and let g: A → B and h: B → C be homomorphisms. Let the composition g∘h : A → C be defined by g∘h(a) = h(g(a)).

Then g∘h is a homomorphism from <A, F> to <C, H>.

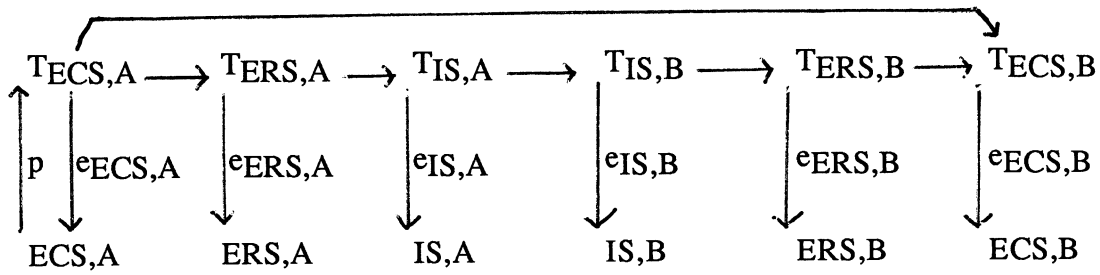
This theorem allows us to extend the model as presented in figure 1 with a second translation step, see figure 2. The theorem states that executing the two translations consecutively amounts to a homomorphic translation from <A, F> to <C, H>. The decomposition of this translation into two steps as well as the use of the intermediate language B, can be considered as auxiliary. In principle the intermediate stage could be eliminated and the the homomorphism g∘h could be defined directly without reference to B.



g,h : translation homomorphisms; e_1,e_2,e_3: evaluation homomorphisms,
p: parsing, finding a corresponding term for a given element in <A,F>

Figure 2. *Compositional translation with one intermediate language*

In the CAT framework of Eurotra all translations are homomorphisms between termalgebras. And the theorem we have applied once, can be applied again: composing several homomorphisms to a single one. The many translation steps in Eurotra define together one translation homomorphism. All intermediate translation steps can be considered as auxiliary means for defining the translation from source to target. Our mathemtical model shows that the CAT framework is in essence a framework for obtaining a compositional translation (a conclusion that is not explicit in the Eurotra publications). The mathematical model for the CAT framework is given in figure 3. The main result from the algebraic theory is the presence of the uppermost arrow: the direct translation homomorphism.



A: source language, B: target language

$e_{ECS,A}$, $e_{ERS,A}$, $e_{IS,A}$, $e_{IS,B}$, $e_{ERS,B}$, $e_{ECS,B}$: evaluation homomorphisms,

p: parsing, finding a corresponding term for a given element in ECS,A

figure 3. *The mathematical model of the CAT framework, for translating from source language A to target language B.*

5. Polynomials as relaxations

Up till now there was a parallelism between the mathematical model and the Eurotra CAT framework. We obtained a new perspective and have seen some features that were hidden in the framework. But in the present section we will investigate a proposal where the mathematical model leads us to a different appreciation. This proposal might be considered as a reflection of the Eurotra opinion that compositional translation is a beautiful ideal, but unattainable in practice. One is willing to take it as a starting point, but relaxations are considered as indispensable. In the present section the proposal will be investigated, and it will turn out that the proposed relaxations are, with one exception, relaxations that fit completely into the mathematical model. Hence they are no relaxation at all of the notion of a compositional translation.

The relaxations that will be discussed below are proposed in several Eurotra publications. We will follow the formulation in Arnold e.a. 1985. Immediately after the introduction of the notion 'strictly compositional' translation (= homomorphic translation) the relaxation is introduced. For the ease of the discussion, the relevant passage is quoted.

A translation relation T between G_i= $<C_i,A_i>$ and G_j = $<C_j,A_j>$ is strictly compositional if T maps A_i into A_j, and there is a mapping t from C_i into C_j such that if exp = $c[u_1, . . u_n]$ then the translation of exp is: $t(c)[T(u_1), . . ,T(u_n)]$. In addition to strict compositionality the following relaxations are allowed:

1) The number and/order of arguments of c and t(c) may differ

2) Rather than being a an actual member of the constructors for a given G, either c, or t(c) may be a function made up of variables, and atoms, and constructors of G.

In order to discuss this proposal, the algebraic theory has to be developed somewhat further. What is needed is a method to define new operators in a given algebra. A simple example of an operator defined from given operators is *composition*: if f and g are operators which take one argument then f∘g is defined by first applying f to the argument, and next applying g to the result. So for all a f∘g(a) = g(f(a)). A less elementary example concerns the algebra of natural numbers with + and × as operators. The new operator takes two arguments and is represented by the expression $x_1 \times x_1 + x_2 \times x_2$. The operator assigns to the arguments 1 and 2 (given in this order) the value $1 \times 1+2 \times 2$, i.e. 5, and it assigns to the arguments 2 and 3 the value $2 \times 2 + 3 \times 3$, i.e. 13. An expression like $x_1 \times x_1 + x_2 \times x_2$ is called a polynomial. Given two arguments, the resulting value is obtained by substituting the first argument for x_1, the second argument for x_2, and performing the calculations which are indicated in the expression. Informally stated, a polynomial is a term with variables, and it defines an operator. This method of defining new operations by means of polynomials can be used in every algebra, the relevant formal definition are given below.

Definitions.

The set $Pol^n_{<[B],F>}$ of n-ary **polynomial symbols,** shortly **polynomials,** over algebra <[B],F> , henceforth abbreviated as Pol^n, is defined as follows.

1) For every element in B there is a distinct symbol $b \in Pol^n$. These symbols are called constants.

2) For every number i , with $1 \leq i \leq n$, the symbol $x_i \in Pol^n$. These symbols are called variables.

3) For every operator in F there is a distinct symbol f. If F is a m-ary operator, and we have that if $p_1, p_2, \ldots, p_m \in Pol^n$ then also $f(p_1, \ldots, p_m) \in Pol^n$.

The set Pol of **polynomial symbols** over algebra <[B],F> is defined as the union for all n of the n-ary polynomial symbols, i.e. by $Pol = \cup_n Pol^n$.

A polynomial symbol $p \in Pol^n$ defines an n-ary **polynomial operator**; its value for given arguments is found by evaluating the term that is obtained by replacing x_1 by the first argument, x_2 by the second etc..

Given an algebra <[B],F>, and a set P of polynomial symbols over *A.*, we obtain a new algebra <[B],P> by replacing the original set of operators by a set of polynomial operators. An algebra obtained in this way is called a **polynomially derived algebra**, or shortly a **derived algebra.**

Note that a symbol like x_1 is a member of Pol^1, Pol^2, etc, and analogously for all other symbols. The polynomial x_1+x_2 might be a 3-ary polynomial, and the corresponding operator has the property that its value is independent of its third argument. This polynomial illustrates that the form of a polynomial does not determine completely the arity of the corresponding operator. If it is necessary to mention explicitly the number of arguments a polynomial takes, this can be done by a superscript indicating the arity (but in most contexts the arity will be evident).

The relaxation presented in the beginnings of this section was divided into two clauses. Each of the two can be split into several subcases. Below we will consider them separately, and show that their effects can (with one exception) be obtained by means of translating an operator into a polynomial. This means that there is a strictly compositional translation into a polynomially derived algebra, i.e. into an algebra of which the operators are defined by means of polynomials.

1a) the order of arguments of c and t(c) differs

An example of change of order of arguments arises when the translation of c[u_1,u_2] is defined by t(c)[u_2,u_1]. The same effect can be obtained by translating operator c into the polynomial symbol **t(c)(x2,x1).**

1b) the number of arguments in t(c) is less than in c

A simple example is that c[u_1,u_2] is translated into t(c)[u_1]. This effect is obtained by translating c into the polynomial symbol $\mathbf{t(c)(x_1)}$ from Pol^2. Recall that this corresponds with a two place operator for which the value of the second argument is irrelevant.

1c) the number of arguments in t(c) is more than in c

It is of course not meant by the proposal that c[u_1,u_2] can be translated into t(c)[u_1,u_2,u_3], since there is no u_3 that can serve as argument of t(c). Presumably those situations are intended where the main operator after translation has more arguments than the original operator, and the extra arguments are known. An example is reduplication, e.g. when the translation of c[u_1,u_2] is defined as t(c)[u_1,u_2,u_1]. Another possibility is that the extra argument is a constant. An example (not from Eurotra) arises if we translate from Latin (which has no articles) into English. We might then translate C_{NP}(*pater*) into C'_{NP}(*the, father*). The effects of these two examples are obtained by the polynomials $\mathbf{t(\ C_{NP}\ )(x_1,x_2,x_1)}$ and $\mathbf{t(C'_{NP})}$ **(*the,* $\mathbf{x_1}$)** respectively.

2a) t(c)is a function made up of variables, and atoms, and constructors

The description of what is meant by a function, learns us that it is the same as a polynomial. In the light of the cases 1a . . 1c) we see that relaxation 1) is in fact a special case of relaxation 2b).

2b) c is a function of variables, and atoms, and constructors

A special case of this relaxation is the following: variables are allowed in c. An example arises when c_{27}[1, 2, 3] is translated as c_{38}[2, 3]. The same effect is obtained by means of the polynomial $\mathbf{c_{38}(x_2,x_3)}$. The general case that c is a function does not fit into the idea that the relaxations are in fact homomorphisms to a derived algebra. This exception will be discussed in the next section.

The investigations in the mathematical model show that the original Eurotra division of the relaxation into two cases is not correct. The first clause is in fact a special case of the second one. Furthermore, the relaxations allowed for in the second clause can, with one exception, be formulated by means of polynomials. So they consitute variants of the compositional framework, and do no disturbe compositionality at all.

6. Deviations

Most Eurotra relaxations can be considered as the introduction of polynomially derived operators. The single exception is that the left hand side of a translation rule can be a term. In the present section we will discuss an example this relaxation. First its linguistic background will be sketched. Sentences like

(1) *John seeks Mary*

are in Eurotra, as well as in many linguistic theories, syntactically analysed as consisting of two parts, a Noun Phrase (*John*) and a Verb Phrase(*seek Mary)* . But semantically this sentence is considered as a ternary structure, with relation *seek* and arguments *John* and *Mary*. This explains why in ECS it is given a binary structure and in ERS a ternary one. The involved operator in ECS is C_S and in ERS it is $C_{Subj/Obj}$. So one whishes to get the following translation

(2) C_S(*John*, C_{VP}(*seek, Mary*)) ===> $C_{subj/obj}$(*seek, John, Mary*)

For this purpose the following translation rule is proposed (Arnold e.a. 1985).

(3) $C_S(X1,C_{VP}(X2,X3)$) ===> $C_{subj/obj}(X2,X1,X3)$

So one term is translated into another one. The output of an homomorphic translation rule is fully determined by the operator and the translations of its parts, whereas in this proposal the form of the parts play a crucial role. Therefore it is not a homomorphic translation. It might seem an innocent variant; however, in interaction with other rules the situation turns out to be harmful. An example of such an interaction is given below.

Consider the sentence

(4) *John gives Mary the book.*

In ECS this would probably have the structure

(5) C_S(*John*, $C_{VP}(C_{TVP}$(*give, Mary*),C_{Def}(*book*))))

In ERS it would probably have the structure

(6) $C_{Subj,Iobj,Obj}$(*give, John, Mary,* C_{Def}(*book*)).

The translation rule that performs this translation is

(7) $C_S(X_1,C_{VP}(C_{TVP}(X_2, X_3),X_4))$ ===> $C_{Subj,Iobj,Obj}(X_2, X_1,X_3,X_4)$.

The aim of these two translation is obvious: sentences like (1) have to be translated by rule (3), and sentences like (4) by (7). Unfortunately, translation rule (3) is applicable to structure (5) as well. This introduces an undesired nondeterminism, which was not realized when relaxation 2b was proposed. The rules (3) and (7) themselves do not tell uniquely what has to happen when (5) is given as input. Someone writing a computer program for these translation rules has to make a decision what to do. This is of course not acceptable: what the translations is, should not be determined by the programmers, but by the designers of the rules. And this example gives just one of the possible conflicts. There certainly will be many other rules for translating sentences and there might be a competition among such rules as well. So in the context of other rules (3) does not define a translation function at all, then this may disturbe the whole translation process.

One should not conclude from the above discussion that the proposed relaxation should be

rejected completely. It is only intended to show that the relaxation is not as innocent as the other ones. There are several strategies one might follow in order to avoid the problems. One might try to reformulate (3) in such a way that it is no longer applicable to a sentence like (4). That probably requires a further relaxation: the introduction of negative conditions. And even then,there is no guarantee that no conflicts will arise. An alternative strategy is to introduce an ordering of translation rules that tells which rule has to be tried first etc.. In this way a new component is introduced into the framework. I would prefer to stay in the realm of algebra, and consider (3) and (7) as instructions for termrewriting. Then they are not considered as instructions for going from the one algebra to the other, but as instructions for obtaining a normal form within one algebra. Methods from the fields of term rewriting systems can then be used to deal with the problems of interaction. For a survey of the field of termrewriting, see Klop 1987. Further investigations might answer the question whether,with one of these strategies, compositionality can be maintained.

7. Discussion

The mathematical model for the CAT framework presented in section 4, defines a structure which is about the same as the CAT framework in Eurotra. The main difference is that our model has been build from mathematical ingredients such as homomorphisms and algebras, whereas the CAT framework is presented with ad hoc definitions. The advantages of using well known mathematical tools are manifold. First of all, the definitions are more clear and more elegant than the Eurotra definitions. Secondly, the mathematical notions carry on their sleaves a treasure of mathematical knowledge, thus enabling us to prove properties of the system. We have employed a very elementary theorem: that the composition of homomorphisms is a homomorphism. Using this, we showed that the Eurotra framework produces a homomorphic, i.e. compositional, translation from the source language to the target language.

A third advantage of the mathematical model was met in sections 6 and 7.The mathematical model describes structure of the translation system independently of the accidental linguistic information it contains, and thus the essential aspects of the system become evident. In this way relaxations of the system can be distinguished in innocent variants and fundamental changes. This discussion in sections 6 and 7 of the Eurotra relaxations can be summarized as follows. The translation relation between two Eurotra algebras A and B is a homomorphism (strictly compositional) from T_A into an algebra that is polynomially derived from T_B. Only one relaxation constitutes an exception to this statement. That relaxation cannot be added to the framework in the proposed way, but requires further changes . The mathematical model enabled us here to separate innocent variants from harmful deviations.

It is interesting to compare the above sketched situation (translating into polynomially derived termalgebras) with the situation in PTQ (Montague 1974). There one aims at translating a fragment of English into intensional logic, since that logic is used to represent meanings of English phrases. The algebraic grammar for intensional logic has its own motivation, and its operators do not correspond with the operators in the algebraic grammar for English. So a direct homomorphism

from the termalgebra for English to the termalgebra for intensional logic is not possible. The meanings of operators for English correspond sometimes with complicated logical formulas containing variables where arguments have to filled in. Therefore $T_{English}$ is homomorphically translated into an algebra that is polynomially derived from the algebra for logic. This method of using polynomially derived algebras originates from Universal Grammar (Montague 1970). This observation is again an example of the benefit of a mathematical perspective: the essentials of the system become evident.

8. References

Arnold, D.J., L. Jaspaert, R.L. Johnson, S.Krauwer, M. Rosner, L. des Tombe, G.B. Varile and S. Warwick, 1985, 'A MU1 View of the <C,A>,T Framework in EUROTRA', in: *'Proceedings of the Conference on Theoretical and methodological Issues in Machine translation of Natural Languages,* Colgate University, Hamilton, NY. pp. 1-14.

Arnold, D.J., S. Krauwer, M. Rosner, L. des Tombe, and G.B Varile, 1986,' The <C,A>,T framework in EUROTRA: a theoretically commited notation for MT', in *Proceedings of Coling 86,* pp. 297-303.

Arnold, D. and L. des Tombe, 1987, 'Basic theory and methodology in Eurotra', in S. Nirenburg (ed.), 1987, *Machine Translation. Theoretical and methodological issues.* Cambridge University Press pp.114-134.

Klop, J.W., 'Term rewriting systems, a tutorial', *Bull. of the European Association for Theoretical Computer Science,* 32, p. 143-182. Also CWI Note CS-N8701, Centre for Mathematics and Computer Science, Amsterdam. To appear in Abramski, Gabbay and Naibaum, Handbook of logics and Computer Science.

Montague, R., 1970, 'Universal grammar', *Theoria* 36, 373-398. Reprinted in R.H. Thomason (ed.), 1974, pp. 222-246.

Montague, R., 1973, 'The proper treatment of quantification in ordinary English', in K.J.J. Hintikka, J.M.E. Moravcsik & P. Suppes (eds), *Approaches to natural language,* Synthese Library 49, Reidel, Dordrecht, 1973, pp. 221-242. Reprinted in R.H. Thomason, 1974, pp. 247-270.

Thomason R.H. (ed.), *Formal philosophy. Selected papers of Richard Montague,* Yale Univ. Press, 1974.

Tombe, L. des, D.J. Arnold, L. Jaspaert, R.L. Johnson, S.Krauwer, M. Rosner,, G.B. Varile and S. Warwick, 1985, A preliminary linguistic framework for Eurotra, In: *'Proceedings of the Conference on Theoretical and methodological Issues in Machine translation of Natural Languages,* Colgate University, Hamilton, NY. pp. 1-14.

The ITLI Prepublication Series

1986

86-01	The Institute of Language, Logic and Information
86-02 Peter van Emde Boas	A Semantical Model for Integration and Modularization of Rules
86-03 Johan van Benthem	Categorial Grammar and Lambda Calculus
86-04 Reinhard Muskens	A Relational Formulation of the Theory of Types
86-05 Kenneth A. Bowen, Dick de Jongh	Some Complete Logics for Branched Time, Part I Well-founded Time, Forward looking Operators
86-06 Johan van Benthem	Logical Syntax

1987

87-01 Jeroen Groenendijk, Martin Stokhof	Type shifting Rules and the Semantics of Interrogatives
87-02 Renate Bartsch	Frame Representations and Discourse Representations
87-03 Jan Willem Klop, Roel de Vrijer	Unique Normal Forms for Lambda Calculus with Surjective Pairing
87-04 Johan van Benthem	Polyadic quantifiers
87-05 Víctor Sánchez Valencia	Traditional Logicians and de Morgan's Example
87-06 Eleonore Oversteegen	Temporal Adverbials in the Two Track Theory of Time
87-07 Johan van Benthem	Categorial Grammar and Type Theory
87-08 Renate Bartsch	The Construction of Properties under Perspectives
87-09 Herman Hendriks	Type Change in Semantics: The Scope of Quantification and Coordination

1988

Logic, Semantics and Philosophy of Language:

LP-88-01 Michiel van Lambalgen	Algorithmic Information Theory
LP-88-02 Yde Venema	Expressiveness and Completeness of an Interval Tense Logic
LP-88-03	Year Report 1987
LP-88-04 Reinhard Muskens	Going partial in Montague Grammar
LP-88-05 Johan van Benthem	Logical Constants across Varying Types
LP-88-06 Johan van Benthem	Semantic Parallels in Natural Language and Computation
LP-88-07 Renate Bartsch	Tenses, Aspects, and their Scopes in Discourse
LP-88-08 Jeroen Groenendijk, Martin Stokhof	Context and Information in Dynamic Semantics
LP-88-09 Theo M.V. Janssen	A mathematical model for the CAT framework of Eurotra

Mathematical Logic and Foundations:

ML-88-01 Jaap van Oosten	Lifschitz' Realizabiility
ML-88-02 M.D.G. Swaen	The Arithmetical Fragment of Martin Löf's Type Theories with weak Σ-elimination
ML-88-03 Dick de Jongh, Frank Veltman	Provability Logics for Relative Interpretability

Computation and Complexity Theory:

CT-88-01 Ming Li, Paul M.B.Vitanyi	Two Decades of Applied Kolmogorov Complexity
CT-88-02 Michiel H.M. Smid	General Lower Bounds for the Partitioning of Range Trees
CT-88-03 Michiel H.M. Smid, Mark H. Overmars Leen Torenvliet, Peter van Emde Boas	Maintaining Multiple Representations of Dynamic Data Structures
CT-88-04 Dick de Jongh, Lex Hendriks Gerard R. Renardel de Lavalette	Computations in Fragments of Intuitionistic Propositional Logic
CT-88-05 Peter van Emde Boas	Machine Models and Simulations (revised version)