Completeness of Compositional Machine Translation for Context-Free Grammars *

Willem-Olaf Huijsen
Willem-Olaf.Huijsen@let.ruu.nl

Abstract

A machine translation system is said to be *complete* if all expressions that are correct according to the source-language grammar can be translated into the target language. This paper addresses the completeness issue for compositional machine translation in general, and for compositional machine translation of context-free grammars in particular. Conditions that guarantee translation completeness of context-free grammars are presented.

1 Introduction

Systems for translation of controlled language[1] require the source text to be expressed within severe syntactical and lexical limits. One of the objectives of such systems is that an author who fully conforms to the imposed restrictions is rewarded with a reliable and fully automatic translation of his text into one or more target languages. Therefore a proof of their *completeness* is of great importance. A machine translation system is said to be *complete* if all expressions that are correct according to the source-language grammar can be translated into the target language.

The starting point of this research has been the compositional approach to machine translation developed in the Rosetta project, (Rosetta, 1994). An important difference is that Rosetta made use of a rather complex grammar formalism, M-grammars, for which completeness could not be proven, whereas the current research focuses on the provability of completeness for relatively simple grammar formalisms, which may be more appropriate for machine translation of controlled languages.

---

*The research presented here is part of my PhD-project on the completeness of compositional machine translation. In this PhD-project I address the completeness issue for several grammar formalisms, describing and comparing them in terms of an abstract, algebraic formulation of compositional grammars and compositional translation. This paper is restricted mainly to the context-free grammar formalism.

[1] For more information on controlled language, see: `http://wwwots.let.ruu.nl/Controlled-languages/`.

First, Sections 2 and 3 describe our view and definitions of respectively compositional grammar and compositional machine translation. Section 4 presents the theme of this paper, viz. *completeness* of compositional machine translation. Subsequently, Section 5 works out completeness conditions for compositional grammars based on context-free grammars. These conditions are rather restrictive, and may therefore find application primarily in areas such as controlled languages. One of the objectives of ongoing research is to relax the conditions. Section 6 concludes the paper and discusses ongoing and future research.

2 Compositional Grammars

This Section defines *compositional grammars* (Subsection 2.1), and the auxiliary notions *syntactic derivation tree* (Subsection 2.2) and *semantic derivation tree* (Subsection 2.3).

2.1 A Definition of Compositional Grammars

Compositional machine translation assumes that the source language (SL) and the target language (TL) are defined by means of compositional grammars, i.e. grammars that obey the well-known compositionality principle (cf. (Partee, ter Meulen, and Wall, 1993; Janssen, 1986; Gamut, 1991, p. 315 ff.)). Abstracting away from the details of any specific syntactic formalism, we define a *compositional grammar* G as consisting of (i) a syntactic component, (ii) a semantic component, and (iii) an interpretation from the syntactic component to the semantic component (cf. Montague's Universal Grammar, (Thomason, 1974)). Roughly, the syntactic component consists of a set of basic expressions (words), each having a syntactic category, and a set of syntactic rules that build larger expressions from basic expressions. Likewise, the semantic component consists of a set of basic meanings, each having a semantic category, and a set of semantic rules that build larger meanings from basic meanings. The interpretation associates with every basic expression a *set* of basic meanings, and with every syntactic rules a *set* of semantic rules.

There now follows a more detailed description of these components, which the eager reader may wish to skip on a first pass.

The *syntactic component* specifies a finite set of basic expressions BE, a finite set of syntactic rules $SynR$, a finite set of syntactic categories $SynCats$, and a syntactic type-assignment function $SynType(\cdot)$. *Basic expressions* are, roughly, the smallest meaningful units in a language (more or less the stems of content words). *Syntactic rules* are operations that recursively build *derived expressions* from basic expressions. *Syntactic categories* describe the syntactic properties of expressions. Basic expressions b

all have a syntactic category $SynCat(b)$; syntactic rules restrict their arguments in their categories, and specify the category of the derived expression they yield. The *syntactic type-assignment function* associates every syntactic rule R with a 2-tuple $SynType(R)$ consisting of a so-called argument list $SynAL(R)$ of the categories of its arguments and its resultant category. The arity $arity(R)$ of a syntactic rule is the number of categories in the rule's argument list. We require that all syntactic and semantic rules are total: They must be applicable for any combination of arguments that matches their argument lists. Note that this is not a real restriction of expressiveness: Any partial function can be made into a total function by an appropriate tuning of the set of categories.

The *semantic component* has the same structure as the syntactic component: It specifies a finite set of basic meanings BM, a finite set of semantic rules $SemR$, a finite set of semantic categories $SemCats$, and a semantic type-assignment function $SemType(\cdot)$. *Basic meanings* are expressions of the semantic domain of some logical language. *Semantic rules* are operations in the logical language that recursively build *derived meanings* from basic meanings. For the purpose of compositional translation the choice of this logical language is not very important. However, the semantic rules must be total. *Semantic categories* describe the semantic properties of semantic expressions. Basic meanings m all have a semantic category $SemCat(m)$; semantic rules restrict their arguments in their semantic categories, and specify the category of the derived meaning they yield. The *semantic type-assignment function* associates every semantic rule M with a 2-tuple $SemType(M)$ consisting of a so-called argument list $SemAL$ of the categories of its arguments and its resultant category. The arity $arity(M)$ of a semantic rule is the number of categories in the rule's argument list.

The *interpretation*, denoted $[\![.]\!]$, associates every basic expression with a *set* of basic meanings, and every syntactic rule with a *set* of semantic rules. The arities of associated syntactic and semantic rules must match. Note that our approach differs here from Montague grammar, in which a basic expression (syntactic rule) is associated with *exactly one* basic meaning (semantic rule).

2.2 Syntactic Derivation Trees

Derivational histories of syntactic expressions are represented using so-called syntactic derivation trees:

DEFINITION 1 (Syntactic Derivation Tree)
A syntactic derivation tree t *is either a tree consisting of a single node* b*, where* b *is the name of a basic expression, or a tree of the form* $R[t_1, \ldots, t_n]$*, where* R *is the name of a syntactic rule, and* $t_1, \ldots, t_n$ *is an ordered list of syntactic derivation trees.*

We define the syntactic category of a syntactic derivation tree t, denoted $SynCat(t)$, to be the resultant category of its top syntactic rule. For convenience, we will sometimes annotate syntactic derivation trees with their syntactic category, e.g. $t : C$.

Intuitively one may think of a syntactic derivation tree as the derivational history of a syntactic expression. However, not all syntactic derivation trees actually describe expressions: The definition given above does not require the syntactic rules to be applicable to their arguments. This distinction is described by the concept of well-formedness.

Definition 2 (Well-Formedness of Syntactic Derivation Trees)
A syntactic derivation tree t is well-formed *if and only if it consists of a single basic expression or otherwise if all the syntactic rules in the tree are applicable to their arguments as specified by tree t, i.e. if and only if for all the syntactic rules in tree t (i) the number of arguments (subtrees) matches the rule's arity, and (ii) the arguments satisfy any conditions on the syntactic categories that may be made by the syntactic rule.*

Since there is generally more than one way to derive an expression, expressions are in general assigned a *set* of corresponding syntactic derivation trees.

2.3 Semantic Derivation Trees

The meaning of a derived expression is derived in parallel with the syntactic derivation process. Thus this semantic derivation process may be represented in a tree with the same geometry as the syntactic derivation tree, but labelled by basic meanings and semantic rules. This tree is called a *semantic derivation tree.*

Definition 3 (Semantic Derivation Tree)
A semantic derivation tree *d is either a tree consisting of a single node m, where m is the name of a basic meaning, or a tree of the form $M[d_1, \ldots, d_n]$, where M is the name of a semantic rule, and $d_1, \ldots, d_n$ is an ordered list of semantic derivation trees.*
We define the semantic category *of a semantic derivation tree d, denoted $SemCat(d)$, to be the resultant category of its top semantic rule. Semantic derivation trees may also be annotated with their semantic category, e.g. $d : C$.*

Since every syntactic derivation tree is associated with a *set* of semantic derivation trees, every syntactic derivation tree is associated with a *set* of semantic derivation trees. A semantic derivation tree is well-formed if and only if there is a corresponding well-formed syntactic derivation tree.

3 Compositional Machine Translation

In our definition of compositional translation, the semantic component is used as an interlingua: source and target-language expressions are *translation equivalent* if and only if they have at least one well-formed semantic derivation tree in common.

DEFINITION 4 (Compositional Translation)
For two compositional grammars G and G', the compositional translation *of a source-language utterance e is a set of target-language utterances, derived as follows:*

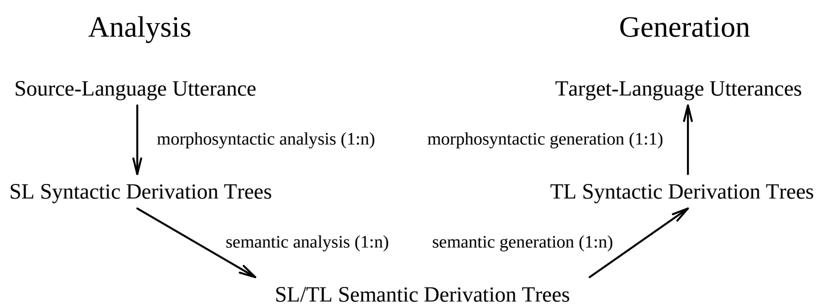

Figure 1: The Process of Compositional Translation

First, *morphosyntactic analysis* performs morphological and syntactic analysis of a SL utterance, yielding the set of all syntactic derivation trees that correspond to the utterance:

$$morsynan(e) = \{b \mid b = e, b \in BE\} \cup \{R[t_1, \dots, t_n] \mid e = R(e_1, \dots, e_n)$$

with $\forall i\ (1 \leq i \leq n),\ t_i \in morsynan(e_i),\ R \in SynR$, and where $R(e_1, \dots, e_n)$ denotes the result of applying rule R to expressions $e_1, \dots, e_n$.

Then, *semantic analysis* of a syntactic derivation tree yields the set of all corresponding semantic derivation trees:

$$seman(b) = [\![b]\!]$$
$$seman(R[t_1, \dots, t_n]) = \{M[d_1, \dots, d_n] \mid M \in [\![R]\!]\}$$

with $\forall i\ (1 \leq i \leq n)\ d_i \in seman(t_i)$.

Next, *semantic generation* from a semantic derivation tree yields the set of all corresponding syntactic derivation trees:

$$semgen(m) = \{b \mid m \in [\![b]\!]\}$$
$$semgen(M[d_1, \dots, d_n]) = \{R[t_1, \dots, t_n] \mid M \in [\![R]\!]\}$$

with $\forall i\ (1 \leq i \leq n)\ t_i \in semgen(d_i)$.

Finally, *morphosyntactic generation* for a well-formed syntactic derivation tree produces the corresponding utterance:

$$morsyngen(b) = b$$
$$morsyngen(R[t_1, \ldots, t_n]) = R(e_1, \ldots, e_n)$$

where $\forall i \, (1 \leq i \leq n) \; e_i \in morsyngen(t_i)$.

4 Completeness of Machine Translation

An important question regarding the reliability of compositional translation is what we call the *completeness*[2] issue: *Can the translation process be guaranteed to produce at least one translation?* In Subsection 4.1, we first make this notion of completeness precise. Then, in Subsection 4.2, we investigate what conditions must be satisfied to guarantee completeness. In Section 5, conditions are elaborated for compositional grammars based on context-free grammars.

4.1 Three Levels of Completeness

Completeness is about the guaranteed generation of well-formed translations, given a specific SL and TL grammar, and translation process. However, this description does not make precise from which stage on the translation process must be guaranteed to succeed. Depending on this, one may distinguish (at least) three levels of completeness (cf. Figure 1):

1. *Utterance Completeness*: For each well-formed SL utterance, the translation process yields at least one well-formed TL utterance.

2. *Syntactic Completeness*: For each syntactic derivation tree of each well-formed SL utterance, the translation process yields at least one well-formed TL utterance.

3. *Semantic Completeness*: For each semantic derivation tree of each syntactic derivation tree of each well-formed SL utterance, the translation process yields at least one well-formed TL utterance.
   Note: Semantic completeness subsumes syntactic completeness, which in turn subsumes utterance completeness.

Naively, one would like a machine translation system to produce at least one translation for every SL utterance. This requirement is included in the definition of utterance completeness above. However, it is well-known

[2]The term 'completeness' was taken from (Whitelock, 1994, pp. 342–343). In the Rosetta framework completeness is known as 'strict isomorphism', and is discussed in (Landsbergen, 1987) and (Rosetta, 1994).

that natural-language utterances are often ambiguous. For each of its interpretations, such an ambiguous utterance may have a different translation. Therefore, a machine translation system should be able to provide at least one translation *for each of the interpretations* of the SL utterance. Natural-language ambiguity takes on two forms: structural ambiguity and lexical ambiguity. The notion of syntactic completeness takes care of the structural ambiguity: it is formulated in terms of structurally unambiguous syntactic derivation trees. However, syntactic completeness is still unsatisfactory, as syntactic derivation trees are often lexically ambiguous. This is due to the fact that basic expressions may have more than one meaning, and syntactic rules may have more than one semantic rule associated with them. What is needed is a formulation of completeness in terms of a structure that is both structurally and lexically unambiguous. The solution is provided by the notion of semantic completeness. Therefore, from now on, the term 'completeness' will be taken to refer to semantic completeness only.

DEFINITION 5 (Completeness)
For a pair of compositional grammars $\langle G, G' \rangle$, compositional translation from G to G' is complete *if and only if for each well-formed semantic derivation tree, the translation process yields at least one well-formed TL utterance.*

4.2 Guaranteeing Completeness

The central issue of this paper is the question of how to guarantee completeness. Or, stated in terms of the process of compositional translation described above: What conditions on the SL and TL grammars are sufficient (and necessary) to guarantee that, after successful analysis, generation can produce a well-formed TL expression? Generation comprises *morphosyntactic* generation and *semantic* generation (cf. Figure 1).

Morpho-syntactic generation evaluates syntactic derivation trees which result from semantic generation by recursive rule application. As stated in Section 2, we assume that all syntactic rules are total for the categories of their arguments. Rule application therefore succeeds if and only if the arguments are of the correct categories. To ensure this, we must move upstream to semantic generation.

Semantic generation simply replaces the basic meanings and semantic rules in the semantic derivation tree with corresponding syntactic elements of the TL grammar, forming the TL syntactic derivation trees. An obvious necessary and sufficient condition for completeness of semantic generation is that there be *at least one* translation-equivalent counterpart in the TL grammar for each possible semantic element in the SL semantic derivation trees. A compositional grammar pair satisfying this condition is called a *homomorphic grammar pair* (see also (Rosetta, 1994, p. 368)):

Definition 6 (Grammar Homomorphism)
A compositional grammar pair $\langle G, G' \rangle$ *is* homomorphic *from* G *to* G' *if and only if* G' *is* attuned *to* G*:*

i. *For each SL basic expression* b*, for each of the basic meanings* m *of* b*, there is at least one TL basic expression* b' *such that basic meaning* m *is also a basic meaning of* b'*.*
Formally: $\forall b \in BE, \forall m \in [\![b]\!], \exists b' \in BE: \ m \in [\![b']\!]$

ii. *For each SL syntactic rule* R*, for each of the semantic rules* M *of* R*, there is at least one TL syntactic rule* R' *such that semantic rule* M *is also a semantic rule of* R'*.*
Formally: $\forall R \in SynR, \forall M \in [\![R]\!], \exists R' \in SynR: \ M \in [\![R']\!]$

However, to demand grammar homomorphism is only a necessary condition for completeness, and not a sufficient one. It merely guarantees that for every well-formed SL semantic derivation tree there is a corresponding TL syntactic derivation tree, and does not guarantee that this syntactic derivation tree is well-formed. The next section is about such sufficient conditions for context-free grammars.

5 Completeness for CFG-Based Compositional Grammars

This section presents completeness conditions for translation between compositional grammars based on the context-free grammar (CFG) formalism. We assume that the reader is familiar with this formalism. Subsection 5.1 explicates how a compositional grammar can be based on context-free grammars. Subsections 5.2 and 5.3 subsequently develop completeness conditions for such compositional grammars.

5.1 CFG-Based Compositional Grammar

A compositional grammar consists of a syntactic component with basic expressions and syntactic rules, a semantic component with basic meanings and semantic rules, and an interpretation from the syntactic component to the semantic component. Here we model the syntactic component as a CFG. The semantic component and the interpretation are as defined above.

In the syntactic component we let basic expressions correspond to rewrite rules that do not have right-hand side (RHS) nonterminals. The rule's RHS corresponds to the lexical material of the basic expression; the rule's left-hand side (LHS) nonterminal corresponds to the syntactic category of the basic expression. We let syntactic rules correspond to rewrite rules that *do* have RHS nonterminals. The type of a syntactic rule is a 2-tuple consisting of a list of categories of the arguments it expects and the category of the expression it produces. The list of categories corresponds to an ordered list of the rewrite rule's RHS nonterminals; the resultant category corresponds to the rewrite rule's LHS nonterminal. The operation performed by the syntactic rule is the in-order concatenation of its RHS terminals and nonterminals, where the nonterminals are replaced with the lexical material of the expressions which are provided as arguments to the syntactic rule. An example illustrates this:

EXAMPLE 1 (CFG-Based Compositional Grammars)
In this example, we briefly illustrate CFG-based compositional grammars. Consider Table 1, which shows the syntactic component of a CFG-based compositional grammar and its interpretation in the semantic component. Observe that the order of syntactic categories in the argument list need not

CFG Rewrite Rule	*Syntactic Rule* *Name : Type*	*Basic Expression* *Name : Category*	*Interpretation*
$A \rightarrow B\ C$	$R_1 : \langle\langle B, C\rangle, A\rangle$		$\{M_1\}$
$A \rightarrow a\ B\ d$	$R_2 : \langle\langle B\rangle, A\rangle$		$\{M_{2a}, M_{2b}\}$
$A \rightarrow e\ C\ B$	$R_3 : \langle\langle B, C\rangle, A\rangle$		$\{M_{3a}, M_{3b}\}$
$B \rightarrow b$		$b : B$	$\{m_1\}$
$C \rightarrow c$		$c : C$	$\{m_{2a}, m_{2b}\}$

Table 1: CFG-based compositional grammar

be the same as the order in the rewrite rules (see R_1, R_3). Syntactic rules R_1 and R_3 have two arguments. As a consequence semantic rules M_1, M_{3a} and M_{3b} are binary operators. Syntactic rule R_2 and semantic rules M_{2a} and M_{2b} have one argument.

The notion of well-formedness can be made more precise now:

DEFINITION 7 (CFG-well-formedness)
A CFG syntactic derivation tree t is CFG-well-formed *if and only if it is either the name of a basic expression, or a tree of the form $R[t_1, \ldots, t_n]$, such that:*

i. rule R's argument list matches the list of syntactic categories of the subtrees $t_1, \ldots, t_n$: $SynAL(R) = \langle SynCat(t_1), \ldots, SynCat(t_n)\rangle$

ii. subtrees $t_1, \ldots, t_n$ are CFG-well-formed.

What about the 'translation power' of CFG-based compositional grammars? The compositional translation method described in Section 3 demands that basic expressions of the source language correspond to basic expressions in the target language, and that the syntactic rules of the source-language correspond to syntactic rules of the target language with the same arity. This restricts the translation power considerably. The main degrees of freedom in the translation relation are the following. In the syntactic rules, the nonterminals need not occur in the same order as in the argument list. This allows translation-equivalent rules to describe word-order differences between languages. Syntactic rules may also introduce lexical material other than that of the arguments. This is called *syncategorematic introduction* (cf. syntactic rules R_2 and R_3 in the example above, where basic expressions a, d and e are left out). The third degree of freedom relates to the correspondence between categories of source- and target-language grammars.

Subsection 5.2 now develops a completeness condition for CFG-based compositional grammars. Subsection 5.3 then shows that this condition is rather restrictive and presents a way to relax it.

5.2 CFG Completeness for Many-to-One Category Correspondence

In this section, we show how a restriction of the correspondence between syntactic and semantic categories of the target language can lead to completeness. First, we formally define a restriction of this correspondence:

Definition 8 (N–1 Category Correspondence)
There is an N–1 category correspondence *between a semantic component and a syntactic component of a compositional grammar if and only if there is a function* $f : SemCats \rightarrow SynCats$ *such that:*

i. $\forall m \in BM, \forall b \in BE :$

$$m \in [\![b]\!] \quad \Rightarrow \quad SynCat(b) = f(SemCat(m))$$

ii. $\forall M \in SemR, \forall R \in SynR :$

$$((M \in [\![R]\!]) \ \wedge \ (SemType(M) = \langle\langle c_1, \ldots, c_n\rangle, c\rangle)) \Rightarrow \\ \text{SynType}_((R)) = \langle\langle f(c_1), \ldots, f(c_n)\rangle, f(c)\rangle$$

The restriction of compositional grammars to such an N–1 category correspondence together with the grammar homomorphism condition gives us completeness:

Theorem 1 (CFG Completeness for N–1 Category Correspondence)
For any CFG-based compositional grammar pair $\langle G, G'\rangle$*, compositional translation from* G *to* G' *is* complete *if*

i. the grammar pair is homomorphic from G *to* G'

ii. there is an N–1 *category correspondence between the semantic and the syntactic categories of* G'*.*

Proof: As we are concerned with semantic completeness, we have to prove that for every grammatical SL utterance, for every one of its well-formed semantic derivation trees, there exists at least one grammatical TL utterance. As we assume it to be trivial that morphosyntactic generation succeeds for CFG-well-formed syntactic derivation trees, we focus on semantic generation. We must show that every well-formed semantic derivation tree always yields at least one *CFG-well-formed* TL syntactic derivation tree. We do this by induction on the depth of the semantic derivation trees.

Induction Base A semantic derivation tree of depth 1 is a basic meaning. Homomorphism from G to G' guarantees that there is at least one TL basic expression that is associated with that basic meaning. Basic expressions are trivially CFG-well-formed syntactic derivation trees.

Induction Hypothesis For every well-formed semantic derivation tree derivable in G which is of depth m or less, compositional translation yields at least one CFG-well-formed TL syntactic derivation tree in G'.

Induction Step Assuming the induction hypothesis holds for arbitrary depth m, we must prove that it also holds for depth $m+1$. Every well-formed semantic derivation tree of depth $m+1$ is of the form $M[d_1, \ldots, d_n] : A$, where each subtree d_i is of the form $M_i[\ldots] : A_i$ (see Figure 2 below). Because of the given well-formedness of the semantic derivation tree we know that M is applicable to its arguments, so that its argument list $\langle A_1, \ldots, A_n \rangle$ matches the semantic categories of the arguments A_i. Homomorphism guarantees that M has at least one associated syntactic rule R', which has some argument list $\langle B_1, \ldots, B_n \rangle$. The induction hypothesis guarantees that every tree d_i has at least one CFG-well-formed TL syntactic derivation tree $t'_i = R'_i[\ldots] : C_i$ associated with it. Note that the induction hypothesis says nothing about the categories C_i of these trees.

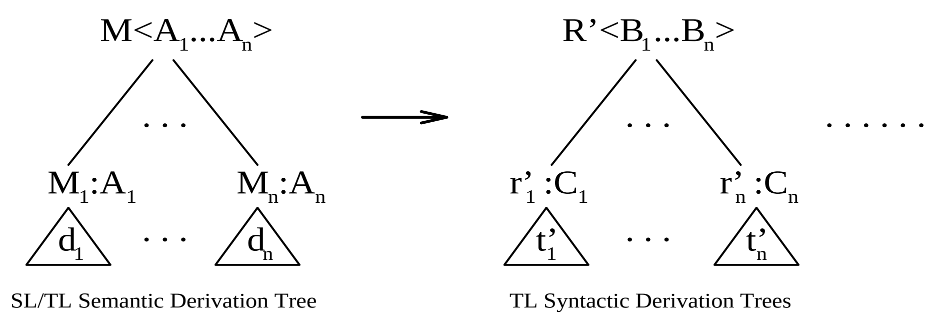

Figure 2: Induction Step: Generating Syntactic from Semantic Derivation Trees

The remaining question is whether there is at least one TL syntactic derivation tree formed in this way which is CFG-well-formed, i.e. for which, according to Definition 2, (i) rule R' is applicable to its arguments, and (ii) all subtrees t'_i are CFG-well-formed. Condition (ii) is covered by the induction hypothesis. Condition (i) requires that the argument list of rule R' matches the syntactic categories of the subtrees $t'_1, \ldots, t'_n$:

$$SynAL(R) = \langle B_1, \ldots, B_n \rangle = \langle SynCat(t'_1), \ldots, SynCat(t'_n) \rangle$$

From the condition in the theorem, we know that there is an N–1 category correspondence f between the semantic categories and the syntactic categories of G'. Because rule R' is associated with rule M, we know that, for all $1 \leq i \leq n$, $B_i = f(A_i)$. Since for all $1 \leq i \leq n$, we also know that tree t'_i is associated with tree d_i, it holds that $C_i = f(A_i)$. Since f is a function, it must hold that for all $1 \leq i \leq n$, $B_i = C_i$, so that the argument list of R' matches the categories of its arguments. Therefore, every such rule R' is applicable to its arguments, so that completeness is guaranteed. □

5.3 Many-to-Many Category Correspondence

The N–1 category correspondence condition is rather restrictive. It implies that a semantic category of the source language must be translated into exactly one syntactic catgory of the target language. We would like to have a looser category correspondence.

EXAMPLE 2
Consider the following grammar rules for translating between English and French noun phrases, where French uses agreement on determiners and nouns: Here we would like to relate semantic category $\underline{DET}$ to syntactic

English Syntax	*Semantics*	*French Syntax*
$R_1 : NP \rightarrow DET\ N$	$M_1 : \underline{NP} \rightarrow \underline{DET}\ \underline{N}$	$R'_{1a} : NP' \rightarrow DET'_m\ N'_m$
		$R'_{1b} : NP' \rightarrow DET'_f\ N'_f$

Table 2: English/French noun phrases

categories DET'_m and DET'_f, and semantic category $\underline{N}$ to syntactic categories N'_m and N'_f. To be able to do so, we could allow every semantic category to be associated with a number of syntactic categories, instead of with just one. This corresponds to an N–N category correspondence.

DEFINITION 9 (N–N Category Correspondence)
There is an N–N category correspondence *between a semantic component and a syntactic component of a compositional grammar if and only if there is a function* $f : SemCats \rightarrow SynCats$ *such that:*

i. $\forall m \in BM, \forall b \in BE :$

$$m \in [\![b]\!] \quad \Rightarrow \quad SynCat(b) \in f(SemCat(m))$$

ii. $\forall M \in SemR, \forall R \in SynR$:

$$((M \in [\![R]\!]) \wedge (SemType(M) = \langle\langle c_1, \ldots, c_n\rangle, c\rangle)) \Rightarrow$$
$$SynType(R) = \langle\langle c'_1, \ldots, c'_n\rangle, c'\rangle$$

where $\forall i\, (1 \leq i \leq n)\ c'_i \in f(c_i)$ *and* $c' \in f(c)$

For a semantic category C the set of corresponding syntactic categories $f(C)$ is called the category correspondence set *of C and is denoted $\tilde{C}$.*

For this new situation we must adjust the completeness condition. Referring to Figure 2, it now is the case that each syntactic category C_i may be any category in the set $f(A_i)$. As the induction hypothesis guarantees only one successful translation for each subtree d_i — and it is not known which one — to guarantee completeness is to guarantee that there is a syntactic rule R' for *every* argument list in $f(A_1) \times \ldots \times f(A_n)$. This is an unrealistic condition: In the English/French example (Example 2), it corresponds to the demand that there must be a French syntactic rule for all four argument lists $\langle DET_m, N_m\rangle, \langle DET_m, N_f\rangle, \langle DET_f, N_m\rangle, \langle DET_f, N_f\rangle$. But, to demand that there is e.g a syntactic rule R' that combines a masculine determiner DET_m and a feminine noun N_f, as this would imply, is nonsensical. The underlying problem is that the agreement dependencies cannot be expressed explicitly in the CFG grammar formalism. The lesson to be learned from this example is that the dependencies between the categories should be taken into account.

We will now present a way of encoding information about the dependencies between categories in CFG-based compositional grammar. To this end, we distinguish two kinds of category correspondence:

DEFINITION 10 (Conjunctive/Disjunctive Correspondence Category)
For a compositional grammar, a semantic category N is a conjunctive (correspondence) category *if and only if for every well-formed semantic derivation tree d of category N, for* every *corresponding category N' in $\tilde{N}$, there exists at least one corresponding well-formed syntactic derivation tree t' of category N'. Any semantic category that is not a conjunctive correspondence category is called a* disjunctive (correspondence) category. *Semantic categories that have only one syntactic category in their category correspondence set are trivially conjunctive categories.*

For example, in the case of the English/French NP rules, the semantic category $\underline{DET}$ corresponds *conjunctively* to categories DET'_m and DET'_f (any determiner has both a masculine and a feminine form), whilst semantic category $\underline{N}$ corresponds *disjunctively* to categories N'_m and N'_f (nouns usually have either masculine or feminine gender). Semantic category $\underline{NP}$ corresponds to only one category, NP', and is therefore a conjunctive category.

How can we use this to establish a condition for completeness? The key idea is that some of the CFG-well-formed syntactic derivation trees of some category A may be guaranteed to translate into at least one CFG-well-formed TL syntactic derivation tree *for all categories* in $\tilde{A}$, instead of *for at least one*. Category A is then said to *correspond conjunctively* to the categories in $\tilde{A}$. As opposed to disjunctive categories, a conjunctive category does not require every rule R' to have translation-equivalent variants for all categories in $\tilde{A}$. Thus, the distinction between conjunctive and disjunctive categories allows for a more realistic condition on the grammars.

We adjust the definition of N–N category correspondence, taking into account the distinction between conjunctive and disjunctive categories. As for the basic meanings and basic expressions: for every basic meaning m, if its category C is a disjunctive category, there must be at least one associated basic expression b' with category C' *for at least one category* C' in $\tilde{C}$. If category C of basic meaning m is a conjunctive category, then there must exist at least one associated basic expression b' with category C' *for every category* C' in $\tilde{C}$.

As for the semantic and syntactic rules, for every semantic rule M with type $\langle\langle A_1, \ldots, A_n\rangle, A\rangle$, we establish conditions on the syntactic rules with which they are associated. Again referring to Figure 2, when generating a syntactic derivation tree from a semantic derivation tree, for subtrees d_i that have a conjunctive category C we can guarantee a tree t'_i *for every* category in $\tilde{C}$. For subtrees d_i that have a disjunctive category C we can guarantee a tree t_i *for only one* category in $\tilde{C}$, and we do not know which one. Therefore, we must guarantee that for every tuple[3] $D \in X_{i\in I_d}\ \tilde{A}_i$ of the syntactic categories corresponding to disjunctive categories of M, there exists *at least one* syntactic rule R' with type $\langle\langle B_1, \ldots, B_n\rangle, B\rangle$ such that:

1. The tuple of the syntactic categories corresponding to the disjunctive categories of the argument list of M is equal to D: $\langle B_i \mid i \in I_d\rangle{=}D$.

2. Every syntactic category B_i that corresponds to a conjunctive category A_i of the argument list of M is in the category correspondence set of A_i: $\forall i \in I_c\ \ B_i \in \tilde{A}_i$.

3. In addition, the resultant category A of semantic rule M must be taken into account. If this is a disjunctive category, then it suffices if the resultant category B of the syntactic rule R' is in $\tilde{A}$. If category A is a conjunctive category, then there must be at least one syntactic rule R' with resultant category N for all categories N in $\tilde{A}$.

Using this condition we again obtain completeness:

[3] Consider the following auxiliary definitions. For any argument list $\langle A_1, \ldots, A_n\rangle$, define sets I_c and I_d as consisting of the indices of its conjunctive and disjunctive categories, respectively. Define $\langle A_i \mid\ i \in I_c\rangle$ and $\langle A_i \mid i \in I_d\rangle$ as the corresponding subtuples.

Theorem 2 (CFG Completeness for N–N Category Correspondence)
For any CFG-based compositional grammar pair $\langle G, G' \rangle$, compositional translation from G to G' is complete *if:*

i. *the grammar pair is homomorphic from G to G'*

ii. *there is an N–N category correspondence between the semantic and the syntactic categories of G', where every semantic category of G' has been declared conjunctive or disjunctive and the sets of categories of G' satisfy the condition described above.*

Because of space limitations we do not include the proof; we trust that the description of the condition above gives the reader an insight into how the proof can be given.

Example 3
Returning to the English/French example discussed earlier, we declared $\underline{DET}$ a conjunctive, $\underline{N}$ a disjunctive, and $\underline{NP}$ a conjunctive category. Checking the condition formulated above, this amounts to the requirement that for every tuple D in $\{\langle N'_m \rangle, \langle N'_f \rangle\}$, there exists a syntactic rule R' such that $\langle B_i \mid \ i \in I_d \rangle = D$ and $\forall i \in I_c,\ B_i \in \tilde{A}_i$, which is indeed the case.

6 Conclusion and Future Research

In this paper, we presented the issue of completeness for compositional translation, and discussed how conditions for compositional translation could be found. In Section 5, we examined the completeness issue for context-free grammars. We established completeness conditions for grammars with an N–1 category correspondence. As this condition is rather restrictive, we relaxed this condition to an N–N category correspondence condition. The first attempt however led to unrealistic conditions on the grammar rules, so that it was necessary to introduce the distinction between conjunctive and disjunctive categories. We adjusted the N–N category correspondence condition accordingly, and obtained a completeness condition for grammars with an N–N category correspondence.

The central issues in ongoing and future research are (i) the completeness issue for some other grammar formalisms, (ii) the algebraic formulation of completeness, and (iii) polynomial compositional translation.

- Completeness for Other Grammar Formalisms — The definite-clause grammar formalism (DCG, see e.g. (Pereira and Shieber, 1987)) extends the CFG grammar formalism with attributes added to the nonterminals. Attributes have a variety of uses, one of the most prominent being the enforcement of agreement relations. As for the completeness condition for DCG, we assume the same conditions on the nonterminals as we did for CFG. In addition, we formulate restrictions on the use of attributes. A proof has been established for completeness of grammars that satisfy these restrictions.

Future research will also address the completeness issue for Tree-Adjoining Grammars. Tree-Adjoining Grammars are interesting because they are somewhat more expressive than CFGs (they are so-called mildly context-sensitive), and it enables expressing linguistic phenomena such as long-distance dependencies.

- Algebraic Formulation of Compositional Translation — Compositional grammar, compositional translation and the completeness issue can be formulated algebraically. Such an algebraic formulation has a number of advantages: (i) it abstracts away from the details of specific grammar formalisms, thus revealing the essentials of compositional translation and completeness, (ii) this abstraction provides a basis for the comparison of different grammar formalisms, and (iii) an algebraic formulation gives access to well-investigated mathematical theory, the results of which may be readily carried over. I hope to use the algebraic formulation as a basis for the investigation of the combination of the use of features and completeness. For other work on algebraic description of natural language, see (Janssen, 1986; Hendriks, 1993). An algebraic view on compositional translation is presented in (Rosetta, 1994, Chapter 19).

- Polynomial Compositional Translation — Another line of work is concerned with an extension of the method of compositional translation for grammar formalisms that use only concatenative operations. The basic idea here is a generalization of the unit of translation-equivalence from single elements to combinations of these (polynomials). This improves 'translation power', as it becomes possible to overcome all kinds of translation problems due to structural divergencies between languages. For example it becomes possible to relate a structure like $[A\ [B\ C]]$ with a structure like $[A'\ B'\ C']$. I hope to show that, as polynomially derived algebras are algebras again, completeness conditions found for compositional translation will carry over to polynomial compositional translation.

References

Gamut, L.T.F. 1991. *Logic, Language and Meaning (2 volumes).* University of Chicago Press, Chicago.

Hendriks, Herman. 1993. *Studied Flexibility: Categories and Types in Syntax and Semantics.* ILLC Dissertation Series 1993-5.

Janssen, T.M.V. 1986. *Foundations and Applications of Montague Grammar.* Ph.D. thesis, Mathematisch Centrum (now CWI), Amsterdam, The Netherlands.

Landsbergen, S.P.J. 1987. Isomorphic grammars and their use in the rosetta translation system. In Margaret King, editor, *Machine Translation Today: The State Of The Art.* Edinburgh University Press, pages 351–372. Philips Research M.S. 12.950.

Partee, Barbara H., Alice ter Meulen, and Robert E. Wall. 1993. *Mathematical Methods in Linguistics (second edition).* Kluwer Academic Publishers, Dordrecht, The Netherlands.

Pereira, Fernando C.N. and Stuart M. Shieber. 1987. *Prolog and Natural-Language Analysis.* Center for the Study of Language and Information (CSLI), Ventura Hall, Stanford University, Stanford, CA 94305.

Rosetta, M.T. 1994. *Compositional Translation.* Kluwer Academic Publishers, Dordrecht, The Netherlands.

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

Whitelock, Pete J. 1994. Shake-and-bake translation. In C.J. Rupp, M.A. Rosner, and R.L. Johnson, editors, *Constraints, Language and Computation.* Academic Press, London, pages 339–359.