Learning to Understand

Isabelle Tellier
LIFL and Université Charles de Gaulle-lille3 (UFR IDIST)
59 653 Villeneuve d'Ascq Cedex, FRANCE
tel : 03-20-41-61-78 ; fax : 03-20-41-61-71
E-mail : tellier@univ-lille3.fr

Abstract :

In this paper, we propose a unified framework for the syntactico-semantic learning of natural languages. This framework integrates recent results on grammatical inference with positive structural examples and an algebraic characterization of the Principle of Compositionality, introduced by linguists and logicians. The purpose is to give account of how children learn to understand their native language from data composed of correct sentences together with their meaning.

Key words :

Natural language learning, Structural Example, syntax and semantics, Principle of Compositionality, Classical Categorial Grammars

Résumé

Dans cet article, nous proposons un modèle de l'apprentissage syntaxico-sémantique du langage naturel. Ce modèle intègre des résultats récents sur l'inférence grammaticale à partir d'exemples structurés et une caractérisation algébrique du Principe de Compositionalité, introduit par des linguistes et des logiciens. L'objectif est de rendre compte de la façon dont les enfants acquièrent leur langue maternelle à partir d'exemples constitués de phrases correctes et d'indication sur le sens qu'elles véhiculent.

Mots clés :

Apprentissage du langage naturel, exemples structurés, syntaxe et sémantique, Principe de Compositionalité, Grammaires Catégorielles Classiques

1. Introduction

Since Chomsky's works and claims, the problem of how children manage to learn their native language has been recognized as a difficult and controversial matter ([Chomsky 65, 68], [Piatelli-Palmarini 79], [Pinker 94]). The question of what is innate and what is not crosses every proposal on the subject. Providing a computational model of natural language learning is then one of the main interesting challenge to Grammatical Inference.

This purpose imposes some constraints to the general framework of Grammatical Inference. First, the grammar to be learned must be at least context-free, as the power of natural languages is also at least context-free ([Chomsky 57], [Savitch 87]). Second, to correctly simulate natural learning, the examples provided to the learner must exclusively be positive ([Wexler & Culicover 80]).

But another more difficult constraint arises from natural situations. As a matter of fact, « no one believes that children learn the grammar of their native language independent of meaning (semantics) and use (pragmatics). » ([Feldman 98]). Taking this claim seriously imposes to take at minimum both levels of syntax and semantics into account. First attempts in this direction ([Hamburger & Wexler 75], [Anderson 77], [Langley 82], [Hill 83]) were based on peculiar conceptions of grammars and semantics, suspected of being *ad hoc* ([Pinker 79]).

On the other hand, the best known and admitted description of the articulation between syntax and semantics is called the Principle of Compositionality. Attributed to Frege and intensively used and studied by Montague ([Montague 74], [Dowty 81]) and his inheritors (for example [Kamp 93], [Muskens 93]), this Principle has recently received a precise and universal formulation ([Janssen 97]). The purpose of this paper is to evaluate the consequences of this formulation on the conditions of natural language learning by linking learning and understanding.

After a review of the usual bases of Grammatical Inference, of recent useful results on the subject and of the formal version of the Principle of Compositionality, we will propose a new way of setting the problem of Grammatical Inference, taking this Principle into account. The proposition will be illustrated with a fully compositional syntactico-semantic framework based on Classical Categorial Grammars.

2. Grammatical Inference

After the introduction of classical basic notations, a unified definition of various kind of *Structural Examples* is provided and a brief review of the aspects of Grammatical Inference concerned by our proposal is given.

2.1 Basic definitions

In the following, $\mathbb{N}$ denotes the set of natural numbers. Σ denotes a finite alphabet and Σ^* is the set of finite concatenations of elements of Σ. For any string $w \in \Sigma^*$, $|w| \in \mathbb{N}$ denotes the number of symbols in w.

A context-free grammar G is a quadruplet $G=<N, \Sigma, P, S>$ where N is an alphabet of auxiliary symbols, Σ is an alphabet of terminal symbols with $N \cap \Sigma = \varnothing$, P is a set of production rules where every production is a pair (A, γ) with $A \in N$ and $\gamma \in \{N \cup \Sigma\}^*$ (also denoted $A \longrightarrow \gamma$) and $S \in N$ is the axiom of the grammar.

Given such a grammar $G=<N, \Sigma, P, S>$ and two strings $x, y \in \{N \cup \Sigma\}^*$, we say *that x derives y* and we note $x \longrightarrow^* y$ if y can be obtained by applying to x a finite set of production rules of P. The language generated by the grammar G, denoted as L(G), is defined as : $L(G)=\{w \in \Sigma^* ; S \longrightarrow^* w\}$.

The *Classical Positive Problem* that Grammatical Inference researchers deal with can be stated as : « Learning Syntax from Sentences » (subtitle of [ICGI 96]). It can be briefly described by Definition 1.

Definition 1 : the Classical Positive Problem

The Classical Positive Problem consists in identifying a grammar $G=<N, \Sigma, P, S>$ (or a grammar equivalent to G) from a learning sample composed of sentences $w \in L(G) \subseteq \Sigma^*$.

This definition does not state any learnability model. Nevertheless, this problem is difficult since regular (and therefore context-free) grammars are not learnable with positive examples in usual models ([Gold 67], [Valiant 84]).

In the following, we will illustrate our proposals with an instance of Classical Categorial Grammars, whose characteristics are given in Definitions 2 and 3.

Definition 2 : Classical Categorial Grammars

A Classical Categorial Grammar (or CCG) Γ is a 4-tuple $\Gamma=<\Sigma, C, f, S>$ with :

- Σ is the finite alphabet (or vocabulary) of Γ ;
- C is the finite set of *basic categories* of Γ ;

From C, we define the set of all possible categories of Γ, noted C', as the closure of C for the operators / and \. C' is then the smallest set of categories verifying :

* $C \subseteq C'$;

* if $X \in C'$ and $Y \in C'$ then $X/Y \in C'$ and $Y\backslash X \in C'$;

- f is a function : $\Sigma \longrightarrow P_f(C')$ where $P_f(C')$ is the set of finite subsets of C' , which associates with each element u in Σ the finite set $f(u) \subseteq C'$ of its categories ;
- $S \in C$ is the axiomatic category of Γ.

In the framework of CCGs, the set of syntactically correct sentences is the set of finite concatenations of elements of the vocabulary for which there exists an affectation of categories that can be « reduced » to the axiomatic category S.

Definition 3 : language recognized by a CCG

Let Γ be a CCG. The admitted reduction rules for any categories X and Y in C' are :

- R1 : $X/Y \ . \ Y \longrightarrow X$ also noted : $R1[X/Y \ . \ Y]=X$;
- R'1 : $Y \ . \ Y\backslash X \longrightarrow X$ or : $R'1[Y \ . \ Y\backslash X]=X$.

The language $L(\Gamma)$ recognized by Γ is then :

$$L(\Gamma)=\{w \in \Sigma^*; \exists n \in \mathbb{N} \ \forall i \in \{1,..., n\} \ u_i \in \Sigma, \ w=u_1...u_n \text{ and } \exists C_i \in f(u_i), \ C_1...C_n \text{—*—>} S\}.$$

where —*—> is the transitive closure of the relation —> defined by R1 and R'1.

Example 1:

Let us define a CCG for the analysis of a small subset of natural language whose vocabulary is : $\Sigma=\{a, man, John, Mary, runs, loves, is\}$. The set of basic categories is $C=\{S, T, CN\}$. In this grammar, T stands for « term » and is affected to proper names : $f(John)=f(Mary)=\{T\}$. Intransitive verbs are distinct from transitive ones : $f(runs)=\{T\backslash S\}$ and $f(loves)=f(is)=\{(T\backslash S)/T\}$. CN means « common noun », so that : $f(man)=\{CN\}$, and finally $f(a)=\{(S/(T\backslash S))/CN\}$. This grammar allows to recognize sentences like : « John runs », « a man runs » or « John loves Mary » as follows (with a

little abuse of notation, to make analyses clearer, R1 and R'1 are used as if they applied on couples (word, category) instead of on categories alone) :

R'1[(John, T).(runs, T\S)]=(John . runs, S)

R1[R1[(a, (S/(T\S))/CN).(man, CN)].(runs, T\S)]=R1[(a . man, S/(T\S)).(runs, T\S)]
=(a . man . runs, S)

R'1[(John, T).R1[(loves,(T\S)/T).(Mary, T)]]=R'1[(John, T).(loves . Mary, T\S)]
=(John . loves . Mary, S)

It is possible to associate trees to these analyses, as displayed in figure 1 for the last two example sentences.

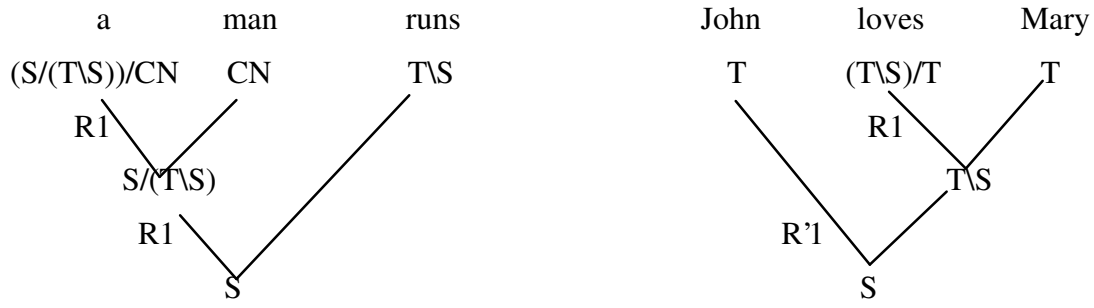


Figure 1 : syntactic analysis trees

From a Classical Categorial Grammar $\Gamma=<\Sigma, C, f, S>$, it is very easy to define a strongly equivalent context-free grammar $G=<N, \Sigma, P, S>$: N contains every possible sub-categories of $f(\Sigma)\subseteq C'$ and the set P includes every rule of the form $A \longrightarrow A/B\ \ B$ and $A \longrightarrow B\ \ B\backslash A$ for $A/B\in N$ and $B\backslash A\in N$ respectively and also every rule of the form $A \longrightarrow a$ for $A\in f(a)$ and $a\in\Sigma$. It can be noticed that G is in Chomsky Normal Form. As the simulation is also possible in the other direction, the class of languages recognized by CCGs is the class of context-free languages ([Bar-Hillel 60]).

CCGs are very lexically oriented as grammatical information are entirely supported by the categories associated with each word. They are thus very well adapted to natural languages ([Oehrle 88]).

2.2 Learning from Structural Examples

As the Classical Positive Problem has been proven impossible to solve in usual learnability models, variants have been defined and explored. One of these variants recently investigated consists in providing *Structural Examples* to the learner instead of strings of words. These Structural Examples are usually dependent on the framework used but we will now give a unified characterization of their format.

Definition 4 : Compositions

Let Σ be an alphabet and let $\{g_1, g_2 ... g_m\}$ be a finite set of function symbols. A *composition* over Σ^n relatively to the family $\{g_i\}_{1\leq i\leq m}$ is a tree containing n leaves each of which is taken among Σ and whose every internal node is a function symbol belonging to $\{g_i\}_{1\leq i\leq m}$.

In the following, g*(w) denotes any composition defined relatively to the family $\{g_i\}_{1\leq i\leq m}$ and whose corresponding terminal string is $w\in\Sigma^n$.

Example 2 :

Figure 2 shows two compositions over $\{a, b, c\}^4$ relatively to the family {g1, g2, g3}.

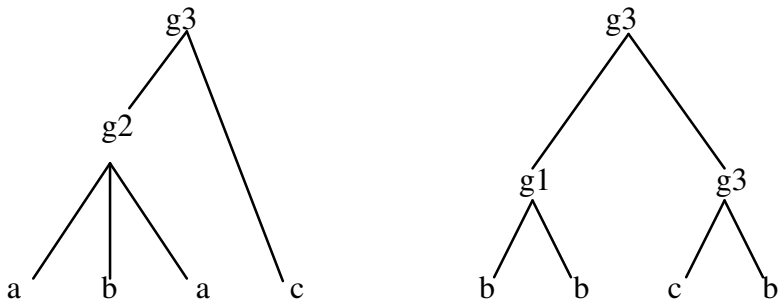


Figure 2 : compositions

A composition g*(w) can be seen as a well-parenthesized version of the string w where every parenthesis is indexed by a member of {1, ...,m}. The compositions of Figure 2 an also be written respectively as :

- g3(g2(a, b, a), c) ;
- g3(g1(b, b), g3(c, b)).

Definition 5 : Structural Examples

Let G=<N, Σ, P, S> be a context-free grammar and $\{g_i\}_{1\leq i\leq m}$ a set of function symbols. For any $w\in\Sigma^*$, a composition g*(w) is said to be a Structural Example for G relatively to $\{g_i\}_{1\leq i\leq m}$ if $w\in L(G)$ and there exists an application K : P —> {1, ..., m} satisfying the following condition : there exists a derivation tree for w in G so that the composition g*(w) is obtained by replacing every application of any rule A —> γ in this derivation tree by the function $g_j\in\{g_i\}_{1\leq i\leq m}$ where j=K((A, γ)).

It can be noticed that each symbol function in $\{g_i\}_{1\leq i\leq m}$ can be associated with a unique arity a_i if and only if we have : for every couple of rules (A, γ) and (B, δ) in P, $K((A, \gamma))=K((B, \delta))=i \Rightarrow |\gamma|=|\delta|=a_i$.

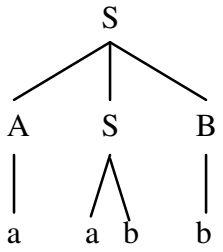
We will call a Regular Set of Structural Examples for G a set of Structural Examples for G relatively to $\{g_i\}_{1\leq i\leq m}$ and K associated with a unique function K.

Sentences which are ambiguous in G (i.e. which can be associated with various different derivation trees in G) can also be associated with various different Structural Examples for G.

It is obvious that if K is a one to one correspondence between P and {1, ..., m} then every Structural Example for G relatively to this function K is isomorphic with a derivation (or syntactic) tree of w in G (and the reciprocal), as illustrated in Example 3.

Example 3 :

Let G=<N, Σ, P, S> be a context-free grammar with N={S, A, B}, Σ={a, b} and P={(S, A S B), (S, a b), (A, a), (B, b)}. Let m=4 and K be defined by : K((S, A S B))=1, K((S, a b))=2, K((A, a))=3, K((B, b))=4.



is isomorphic with :

g1(g3(a), g2(a, b), g4(b))

Figure 3 : a Structural Example for G

In this case, every syntactic tree generated by G is isomorphic with a Structural Example for G and an arity a_i can be associated with each function g_i defined by : $\forall i \in \{1, ..., n\}$, $a_i=|\gamma|$ where $(A, \gamma)=K^{-1}(i)$. So, here : $a_1=3$, $a_3=a_4=1$ and $a_2=2$.

Example 4 :

Let G be the context-free grammar of Example 3 and let K associate with every rule of P the number 1. The only possible Structural Example for G corresponding with the analysis tree given in Figure 3 is then : g1(g1(a), g1(a, b), g1(b)).

This kind of Structural Examples are isomorphic with *skeletons* of analysis trees. Skeletons of trees display the branching without the intermediate non terminal symbols. It is obvious that every skeleton of a syntactic tree can be associated with a unique Structural Example based on this function K (and the reciprocal). It can also be noticed that in this case a unique arity cannot be associated with the function symbol g1.

Example 5 :

Let $\Gamma=<\Sigma, C, f, S>$ be a CCG and let $G=<N, \Sigma, P, S>$ be the strongly equivalent context-free grammar obtained by the process described in Example 1. Let K_Γ be defined in the following way :

- every rule of the form A —> A/B B is associated with 1 ;
- every rule of the form A —> B B\A is associated with 2 ;
- every rule of the form A —> a is associated with 3.

The Structural Examples corresponding with the analyses given in Example 1 are :

- g2(g3(John), g3(runs))
- g1(g1(g3(a), g3(man)), g3(runs))
- g2(g3(John), g1(g3(loves), g3(Mary)))

These compositions are very similar to the syntactic analyses obtained in Example 1, except that categories are lost. This kind of Structural Example is slightly more precise than a skeleton, as different classes of combination functions are distinguished, but far less precise than a syntactic tree, as intermediate symbols are lost.

As in Example 3, arities can be associated with the functions : $a_1=a_2=2$ and $a_3=1$.

The variant of the Classical Positive Problem noted Problem with Positive Structural Examples can now be given.

Definition 6 : the Problem with Positive Structural Examples

The Problem with Positive Structural Examples consists in identifying a context-free grammar G=<N, Σ, P, S> (or a grammar strongly equivalent to G) from a learning sample composed of elements of a Regular Set of Structural Examples for G.

The difficulty of the Problem with Positive Structural Examples depends on the function K. In the case of Structural Examples of the kind given in Example 3, i.e. analysis trees, the Problem with Positive Structural Examples is quasi trivial (the identification of non terminal symbols can be performed by merging).

This Problem has then mainly been studied in the context of Structural Examples of the kind given in Example 4, i.e. skeletons. This approach was first exemplified by Sakakibara who proposed algorithms to build a tree automaton (and consequently a corresponding context-free grammar) from such skeletons ([Sakakibara 90, 92]). Others have developed similar ideas ([Mäkinen 92a, 92b], [Sempere & Nagaraja 98]).

The Problem with Positive Structural Examples has also been adapted to the formalism of CCG, taking into account Structural Examples of the kind given in Example 5. In this way, Kanazawa ([Kanazawa 96], extending [Buszkowski & Penn 90]) has proved that the class of CCGs assigning a bounded number of different categories to individual worlds is identifiable in the limit from Positive Structural Examples. The algorithm proposed rely on the merging of intermediate label variables.

The algorithms providing a solution to the Problem with Positive Structural Examples are computationally efficient : they run in very reasonable polynomial time and space. But this model is not very satisfying from a psychological point of view. It is hardly arguable that in natural contexts, children are provided with Structural Examples. Of course, partial solutions to the Problem with Positive Structural Examples can always be adapted to partial solutions to the Classical Positive Problem by trying every possible composition corresponding with a given string of words *before* applying the inference algorithm. But this initial step is time and space highly exponential (see Proposition 5 in part 4.5) and the computational efficiency is then lost.

We have argued ([Tellier 98]) that, in the context of a special case of categorial grammars and logical semantics, the meaning representation could play a role similar to the one of a skeleton. Our idea is now to show that this strategy can be applied whatever syntactic and semantic formalism is used, provided that both representations are linked by a relation compatible with the Principle of Compositionality.

3. The Principle of Compositionality

3.1 Intuitive formulation

The Principle of Compositionality has been introduced by linguists and logicians to characterize the connection between the syntax and semantics of natural languages. It is usually attributed to Frege. Its contemporary version states that : « the meaning of a compound expression is a function of the meaning of its parts and of the syntactic rules by which they are combined » ([Partee 90]). It has been the basis of several theories, among which the best known may be Montague's semantics ([Montague 74], [Dowty 81]).

If the « parts » mentioned in this definition are assimilated with words, and the « compound expressions » with phrases, this formulation implies that :

- words have individual meanings ;
- the semantics of a phrase (and thus of a sentence) only depends of the meaning of its words and of its syntactic structure.

We will see in the next section how these constraints can be expressed as the mathematical properties of a function translating syntactic structures into meanings.

The Principle of Compositionality has strong psychological justifications. As a matter of fact, it « can explain how a human being can *understand* sentences never heard before » ([Janssen 97]). This argument can be compared to the one Chomsky used to support the claim that everyone needs to know a generative grammar of his/her mother tongue ([Chomsky 57]) : it is the only way to explain, he thought, the ability to *produce* sentences never heard before.

3.2 Formal version

The formal expression of the Principle of Compositionality proposed here is inspired by the one given in [Janssen 97], adapted to our notations. The definition of a Fully Compositional Set, useful further, is ours.

First, a semantic domain has to be defined. Let us note it D. Definitions 7 and 8 introduce other necessary tools.

Definition 7 : the Basic Translation Function

The Basic Translation Function specifies the meaning of individual words. It applies from Σ to D and is noted t.

It can be noticed that ambiguous words should receive several different meanings (eventually distinguished thanks to their different syntactic categories) but in first approximation, we restrict ourselves to *a non ambiguous meaning assignment for words*.

Now, the syntactic structure of a sentence must define the way its meaning elements are combined into the sentence meaning. As seen in part 2.2., the notion of syntactic structure is very close to the one of Structural Example. So, the combination of word meanings must parallel the special combinations of elements of vocabulary introduced in Definitions 4 and 5.

Definition 8 : The Rule Translation Function

The correspondence between syntactic structures and semantics is assured by a one to one correspondence between a family $\{g_i\}_{1\leq i\leq m}$ of function symbols each of which has a unique arity and another family $\{h_i\}_{1\leq i\leq m}$ of semantic functions. The Rule Translation Function noted T is this one to one correspondence and is defined by : $\forall i\in\{1, ..., m\}$, $T(g_i)=h_i$ where the arity of h_i is identical to the arity a_i of g_i.

Finally, the Principle of Compositionality can be mathematically characterized by the existence of a homomorphism between syntactic structures and meanings.

Definition 9 : the Global Translation Homomorphism

Let $G=<N, \Sigma, P, S>$ be a context-free grammar. Let $\{g_i\}_{1\leq i\leq m}$ be a family of function symbols and K a function from P to {1, ..., m} allowing to associate a unique arity to each of them. A Global Translation Homomorphism $H=<D, t, T>$ is composed of a semantic domain D, a Basic Translation Function t and a Rule Translation Function T as stated in Definitions 7 and 8.

H applies from $\{g^*(w)$ composition relatively to $\{g_i\}_{1\leq i\leq m}$; $w\in L(G)\}$ to D and for any given sentence $w=u_1...u_n$ in $L(G)\subseteq\Sigma^*$ and any composition $g^*(w)$ respecting the arities of $\{g_i\}_{1\leq i\leq m}$, it is defined by : $H[g^*(u_1...u_n)]=T(g^*)[t(u_1)...t(u_n)]$

where $T(g^*)[t(u_1)...t(u_n)]$ denotes the composition over D^n relatively to the family of functions $\{T(g_i)\}_{1\leq i\leq m}$ obtained from the composition $g^*(u_1...u_n)$ over Σ^n by replacing each u_i by $t(u_i)$ and each g_i by $T(g_i)$, for every $i\in\{1, ..., n\}$.

If the Global Translation Homomorphism H is correctly defined, then for any given *Structural Example* g*(w) for G relatively to $\{g_i\}_{1\leq i\leq m}$ and K (in the following, we will only say : Structural Example for G), we have : H(g*(w)) represents the (or, in the case of ambiguities, one of the) meaning(s) of w. Finally, we can define what we mean by a *fully compositional* syntactico-semantic framework.

Definition 10 : a Fully Compositional Set

A set <G, $\{gi\}_{1\leq i\leq m}$, K, H> where G=<N, Σ, P, S> is a context free grammar, $m\in\mathbb{N}$, $\{gi\}_{1\leq i\leq m}$ a family of function symbols, K a function from P to {1, ..., m} allowing to associate a unique arity to each of them and H=<D, t, T> a Global Translation Homomorphism is said to be Fully Compositional if for every $w\in\Sigma^*$ and every composition g*(w) over Σ^n relatively to the family $\{g_i\}_{1\leq i\leq m}$ and respecting its arities, the following holds : if $w\in L(G)$ and there exists a Structural Example g'*(w) for G satisfying : $H(g^*(w))=H(g'^*(w))\in D$ then g*(w) is a Structural Example for G.

The reciprocal of this condition always holds (you just have to take g'*(w)=g*(w)). This definition can be considered as stating that in a Fully Compositional Set, if w is a syntactically correct sentence then any composition based on w and translated by H into a correct meaning for w is a Structural Example for G.

3.3 Fully Compositional semantics for a CCG

The first Fully Compositional syntactico-semantic framework is due to Montague. But instead of projecting his syntactic structures into a semantic domain, he projected them into an intermediate logical language to be interpreted in a semantic domain later. In this context, the meaning of a sentence is represented by a logical formula. In the following, we will respect this tradition and apply our definition of the Principle of Compositionality to the example CCG previously introduced and to a new logical language called L. L is a typed language which extends the first order predicate logic by including typed lambda-calculus. It is characterized by :

- the set I of all possible types of L includes

  * elementary types : $e \in I$ (type of *entities*) and $t \in I$ (type of *truth values*) ;

  * for any types $u \in I$ and $v \in I$, $<u, v> \in I$ ($<u, v>$ is the type of functions taking an argument of type u and giving a result of type v).

- semantics of L : a denotation set D_i is associated with every type $i \in I$ as follows :

  * $D_e = E$ where E is the denumerable set of all entities of the world ;

  * $D_t = \{0,1\}$;

  * $D_{<u,v>} = D_v^{Du}$: the denotation set of a composed type is a function.

Example 6 :

run'(John'), where John' is of type e and run' of type < e,t>, is a formula of type t.

It is possible to provide a Fully Compositional semantics included into L to the CCG Γ defined in Example 1. The Structural Examples are supposed to be defined as in Example 5. Let us define the Global Translation Homomorphism H_L step by step :

- translation of the syntactic categories into logical types (function k : C' —> I) :

  * basic categories : k(S)=t, k(T)=e, k(CN)=<e,t> ;

  * derived categories : for any $X \in C'$ and $Y \in C'$, $k(X/Y)=k(Y\backslash X)=<k(Y),k(X)>$.

- the Basic Translation Function $t_L : \Sigma \longrightarrow L$: each word u in Σ is associated with a logical formula $t_L(u)$. For every $u \in \Sigma$ there exists a category $U \in f(u) \subseteq C'$ so that $t_L(u)$ is of type $k(U) \in I$. The logical translations of individual words are :

  * $t_L(a) = \lambda P_1 \lambda Q_1 \exists x[P_1(x) \wedge Q_1(x)]$

    where x and y are variables of type e, P_1 and Q_1 variables of type <e,t>, i.e. predicates of arity 1 (as indicated by the indexes).

  * the verb « to be », as a transitive verb is translated by

    $t_L(is) = \lambda x \lambda y[y=x]$ with x and y variables of type e.

  * every other word u in Σ is translated into a logical constant noted $t_L(u) = u_i'$ where i is the arity, only noted when $i \geq 1$ (conjugated verbs are first reduced to their infinitive form).

- the Rule Translation Function T_L is defined by :

  * $T_L(g1) = f1$ where for every couple (a, b) of formulas in L, we have : f1(a, b)=a(b) ;

  * $T_L(g2) = f'1$ where for every couple (a, b) of formulas in L, we have : f'1(a, b)=b(a) ;

  * $T_L(g3) = Id_1$ where for every formula a in L, $Id_1(a)=a$.

Example 7 :

Let us apply the Global Translation Homomorphism H_L=<L, t_L, T_L> just described to the Structural Examples given in Example 5.

$H_L[g2(g3(John), g3(runs))]=T_L(g2)[T_L(g3)(t_L(John)), T_L(g3)(t_L(runs))]$

$=f'1[\ Id_1(John'), Id_1(run_1')]$

$=f'1[John',\ run_1']$

$=run_1'(John')$

In the next two examples, function g3, translated into Id_1 and then unable to modify the final result, is not displayed for sake of simplicity.

$H_L[g1(g1(a, man), runs)]=T_L(g1)[T_L(g1)(t_L(a), t_L(man))], t_L(runs)]$

$=f1[f1(\lambda P_1\lambda Q_1\exists x[P_1(x)\wedge Q_1(x)], man_1'),\ run_1']$

$=f1[\lambda P_1\lambda Q_1\exists x[P_1(x)\wedge Q_1(x)](man_1'),\ run_1']$

$=(\lambda P_1\lambda Q_1\exists x[P_1(x)\wedge Q_1(x)](man_1'))(run_1')$

$=\lambda Q_1\exists x[man_1'(x)\wedge Q_1(x)](run_1')$

$=\exists x[man_1'(x)\wedge run_1'(x)]$

The *evaluation* of this formula needs usual lambda-conversions.

$H_L[g2(John, g1(loves, Mary))]=T_L(g2)[t_L(John), T_L(g1)[t_L(loves), t_L(Mary)]]$

$=f'1[John',\ f1(love_2',\ Mary')]$

$=f'1[John',\ love_2'(Mary')]$

$=love_2'(Mary')(John')$.

These structure-preserving translations can also be displayed using the analysis trees of Figure 1. The *translation trees* corresponding with the last two examples are given in Figures 4 and 5.

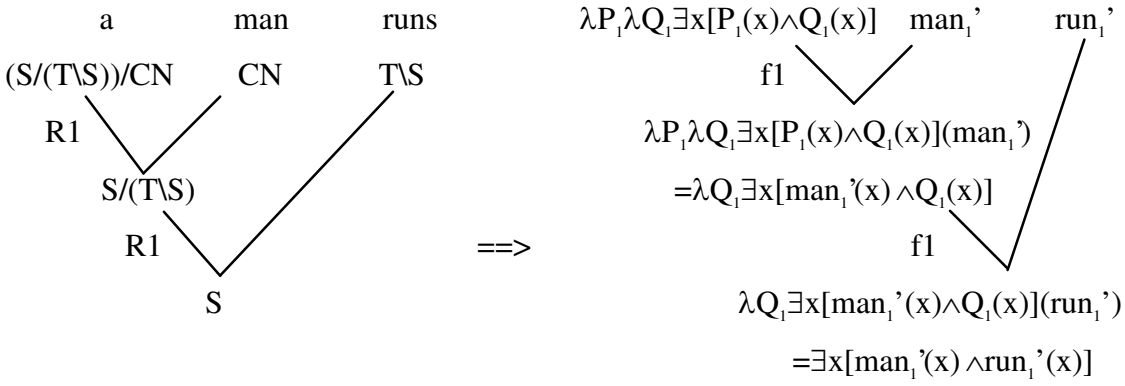


Figure 4 : semantic translation of the first analysis tree of figure 1

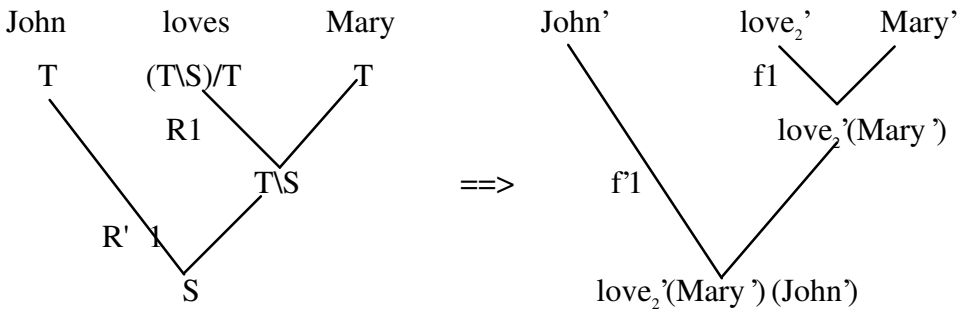


Figure 5 : semantic translation of the second analysis tree of figure 1

The logical formulas obtained by the *evaluation* of the result of the Global Translation Homomorphism H_L correctly represent the meaning of the corresponding initial sentences (by convention, in the case of verb-predicates of arity 2 like « $love_2$' », the first argument always represents the direct object of the verb and the second argument always represents its grammatical subject).

Proposition 1 :

For every $w \in L(\Gamma)$ and every g*(w) Structural Example for Γ relatively to the family {g1, g3, g3} and K_Γ, $H_L(g^*(w))$ is a correct representation in L of the meaning of w and the set $<\Gamma, \{g1, g3, g3\}, K_\Gamma, H_L>$ is Fully Compositional.

Sketch of the proof :

The set L(Γ) is finite, so the set of possible compositions g*(w) for every $w \in L(\Gamma)$ is also finite and Proposition 1 is then easy to verify by enumeration. □

This proof is trivial but more powerful arguments can convince that Definition 10 is relevant and that the set $<\Gamma, \{g1, g3, g3\}, K_\Gamma, H_L>$ can be extended while preserving the property of Fully Compositionality.

Let us first suppose that g*(w) is a Structural Example for Γ relatively to the family {g1, g3, g3} and K_Γ. By Definition 4, it implies that $w \in L(\Gamma)$. Furthermore, the correspondence k between the categories of the grammars and the types of the logic assures that $H_L(g^*(w))$ is a logical formula of type k(S)=t in L, i.e. a *proposition* which can be associated with a truth value. As a matter of fact, it must be noticed that both the Basic Translation Function t_L and the Rule Translation Function T_L respect this correspondence k. Recall that g1 only applies if its first argument is associated with a

category of the form A/B and its second argument is associated with a category of the form B ; in this case, it gives a result of category A. Similarly, the function f1 describes a *functional application*. Thanks to k, the first argument of f1 is a function of arity 1 whose type is <k(B), k(A)>, i.e. expecting an argument of type k(B), which is the type of the second argument of f1, and providing a result of type k(A), corresponding with the result of the rule g1. Of course, the same occurs for g2 and f'1 except for the order of the arguments. At every step of the semantic translation, the correspondence is then satisfied and so for every Structural Example for Γ g*(w), $H_L(g^*(w))$ can only be an admissible logical proposition of L.

Let us now illustrate that both conditions : $w \in L(\Gamma)$ and *there exists a Structural Example* $g'^*(w)$ *for* Γ *satisfying* $H_L(g'^*(w))=H_L(g^*(w)) \in L$ are necessary to expect g*(w) to also be a Structural Example for Γ.

Let w_1=man . a . runs and $g_1^*(w_1)$=g1(g2(g3(man), g3(a)), g3(runs)). We have :

$H_L(g_1^*(w_1))$=f1(f'1($Id_1(t_L(man))$, $Id_1(t_L(a))$), $Id_1(t_L(runs))$)

=f1[f'1(man_1', $\lambda P_1 \lambda Q_1 \exists x[P_1(x) \wedge Q_1(x)]$), run_1']

=$(\lambda P_1 \lambda Q_1 \exists x[P_1(x) \wedge Q_1(x)](man_1'))(run_1')$

=$\exists x[man_1'(x) \wedge run_1'(x)]$

$H_L(g_1^*(w_1))=H_L$[g1(g1(a, man), runs)] where g1(g1(a, man), runs) is a Structural Example for Γ (see Example 7) but $w_1 \notin L(\Gamma)$ and $g_1^*(w_1)$ is not a Structural Example for Γ. Now let w_2=John . loves . Mary and $g_2^*(w_2)$=g1(g2(g3(John), g3(loves)), g3(Mary)).

$H_L(g_2^*(w_2))$=f1(f'1($Id_1(t_L(John))$, $Id_1(t_L(loves))$), $Id_1(t_L(Mary))$)

=f1(f'1(John', $love_2'$), Mary')

=$love_2'$(John')(Mary')

This time, $w_2 \in L(\Gamma)$ but the only possible Structural Example for Γ associated with this sentence was given previously and it has a different semantic translation. As a matter of fact, $H_L(g_2^*(w_2))$ is an admissible proposition of L (it is of the good type) but, with our convention of notation, *it does not represent the meaning of the initial sentence* w_2. $g_2^*(w_2)$ is not a Structural Example for Γ either.

Remarks :

Other kinds of Structural Examples could also be used as inputs of a Rule Translation Function : those of the kind given in Example 3, isomorphic with syntactic trees, could perfectly fit. For example, the « Graph Deformation Condition » defined in [Anderson 77] can be re-formulated as the application of a certain Global Translation

Homomorphism on this kind of Structural Example. The Discourse Representation obtained in [Kamp 92] from syntactic trees belongs to the same family. On the contrary, skeletons (for which arities can be difficult to define) may not be precise enough to specify compositional semantics : as only one function symbol is used in compositions, only one semantic function could be associated with it, which may not be sufficient.

In fact, CCGs seem to be a particularly interesting compromise, because the same kind of Structural Examples can be used both for the Problem with Positive Structural Examples and as input of a Rule Translation Function, as illustrated before.

Furthermore, several variants of CCGs have been defined, which admit extra reduction rules whose corresponding translated rule are also known ([Moortgat 88], [Tellier 98]). But, as far as we know, the Problem with Positive Structural Examples has not been studied for Structural Examples defined relatively to a function K taking these new families of functions into account, so in the following we will stick to CCGs.

4. A new learning model

Our purpose is to provide a computational model of natural language learning and understanding. Now that the linguistic environment is set, it remains to fix what is supposed to be known by the learner and what is to be learned, under which conditions.

4.1 The learning framework

It is natural to suppose that when a child learns a language, she has at her disposal syntactically correct sentences together with their *meaning*. The corresponding situation in our model is an algorithm which takes as input a couple <w, f> composed of a syntactically correct sentence w together with its (or one of its) semantics f.

It can be noticed that this framework assumes that sensorial stimuli coming from a scene are coded into a semantic representation *before entering the learning module* (in the sense of [Fodor 83]) where it will be compared with the sentence describing the same scene : in this sense, *semantic learning precedes syntactic learning*. The way this semantic learning occurs is not our concern here but *it is supposed that the underlying set* $<G, \{g_i\}_{1\leq i\leq m}, K, H>$ *is Fully Compositional.*

The innate knowledge needed in this model is reduced to the function symbols $\{g_i\}_{1\leq i\leq m}$ allowed to define the Structural Examples and the corresponding set of functions $\{h_i\}_{1\leq i\leq m}$. The semantic domain D and the Rule Translation Function T

included in the definition of H are then considered as already known. As T is a one to one correspondence, its reverse function T^{-1}, which is trivial, is also admitted to be available. In usual semantic-based methods of learning ([Hamburger & Wexler 75], [Anderson 77], [Langley 82], [Hill 83]), word meanings are supposed to be already known when the grammatical inference mechanism starts. In first place, we will also make this supposition but in a second one, we will include the Basic Translation Function t into the target of the learning algorithm.

Finally, what does the learner have to learn ? The natural target includes the unknown part of H (i.e. t when it is unknown) and the characteristics of the unknown context-free grammar of the natural language to be learned. The function K is learned together with G (K is partly innate as m is supposed to be already known).

When the target is reached, the system, combining its knowledge, will be able to automatically parse a syntactically correct sentence, transform the syntactic tree built into a Structural Example and apply H to this Structural Example, so as to obtain a corresponding semantic representation. The algorithm thus learns to associate a meaning with a sentence, which is the exact cognitive definition of *learning to understand*. The new problem we deal with can now be precisely defined.

Definition 11 : the Problem of Learning to Understand from Positive Data

The Problem of Learning to Understand from Positive Data without (respectively with) word meanings consists in identifying $<G, \{g_i\}_{1\leq i\leq m}\ K, H>$, a Fully Compositional Set, from a learning sample composed of couples $<w, f>$ where $w \in L(G)$ is a sentence for which there exists a Structural Example for G noted $g^*(w)$ so that $f=H(g^*(w))$, from the knowledge of the family $\{g_i\}_{1\leq i\leq m}$, of the semantic domain D, of the Rule Translation Function T and of its reverse T^{-1} (respectively with the Basic Translation Function t).

This new problem generalizes the Classical Positive Problem described by Definition 1. The idea is to provide more information as input, but also to ask for more as output.

4.2 First results

Proposition 2 displays the link between the new problem of Definition 11 and the Problem with Positive Structural Examples stated in Definition 6. The solutions will make a crucial use of the Fully Compositionality introduced in Definition 10.

Proposition 2 :

If there exists an algorithm which is a solution for the Problem with Positive Structural Examples as stated in Definition 6, then the Problem of Learning to Understand from Positive Data proposed in Definition 11 can be reduced to a couple of sub-problems :

- the problem of inferring a Structural Example for G noted g*(w) from any input couple of the form <w, f> ;
- the problem of inferring every $t(u_i)$, $1 \leq i \leq n$ from any input couple <w, f> where $w=u_1...u_n$.

Proof :

Let us call A the algorithm which is a solution for the Problem with Positive Structural Examples. Let us suppose that both sub-problems can be solved and let us call B and C respectively algorithms that are solutions for them. Algorithm 1 displays the (obvious) strategy which provides a solution to the Problem of Learning to Understand from Positive Data from algorithms A, B and C.

- *$E \leftarrow \varnothing$; $T \leftarrow \varnothing$;*
- *for every input couple <w, f> where $w=u_1...u_n$ do :*
  - *apply algorithm B : the result is a Structural Example for G noted e ;*
  - *$E \leftarrow E \cup \{e\}$;*
  - *apply algorithm C : the result is $T_w=\{t(u_i)\}_{1 \leq i \leq n}$;*
  - *$T \leftarrow T \cup T_w$*
- *apply algorithm A to the set E : the result is G ;*
- *From G and E, infer K.*

Algorithm 1 : the reduction algorithm

Algorithm 1 combines algorithms A, B and C and produces the missing information : the grammar G and the extensions of the functions t and K. Its correctness relies on the correctness of A, B and C. If A exists, the Problem of Learning to Understand from Positive Data can thus be reduced to the problem of building B and C. □

It can be noticed that the complexity of Algorithm 1 is at least :

- the complexity of the algorithms B and C (solutions are studied below) ;
- the complexity of the algorithm A (as seen in part 2.2, known solutions are polynomial in time and space) ;
- the complexity of the inference of K : in the context of Structural Examples of the kind given in Example 5 (and even more in the context of skeletons), this point is trivial.

We can thus focus on the building of algorithms B and C. Proposition 3 treats the most favorable case to solve both sub-problems.

Proposition 3 :

Let $<G, \{g_i\}_{1\leq i\leq m}, K, H>$ be a Fully Compositional Set with $H=<D, t, T>$. If the semantic evaluation is an injection, i.e. if for every $n\in\mathbb{N}$, for every couple of sequences of n semantic items noted $d=(d_1.d_2 ... d_n)$ and $d'=(d'_1.d'_2 ... d'_n)$ in D^n and for every couple of semantic compositions h^* and h'^* over D^n relatively to the family $\{h_i\}_{1\leq i\leq m}$ we have : $h^*(d_1.d_2 ... d_n)=h'^*(d'_1.d'_2 ... d'_n) \Rightarrow h^*=h'^*$ and $\forall i\in\{1, ..., n\}\ d_i=d'_i$ then for every couple $<w, f>$ of input data where $w=u_1...u_n$, there exists a unique semantic composition $h^*(d_1.d_2 ... d_n)$ whose evaluation equals f and, in this case, we also have : $T^{-1}(h^*)[u_1...u_n]$ is a Structural Example for G and $\forall i\in\{1, ..., n\}\ t(u_i)=d_i$.

Proof :

By definition, for every input couple $<w, f>$ where $w=u_1...u_n$ there exists a Structural Example for G noted $g^*(w)$ so that $f=H(g^*(w))=T(g^*)[t(u_1)...t(u_n)]$. $\forall i\in\{1,..., n\}$, $t(u_i)\in D$ and $T(g^*)[t(u_1)...t(u_n)]$ defines a composition over D^n relatively to the family $\{h_i\}_{1\leq i\leq m}$. The semantic evaluation is thus a surjection on the domain of the input data. If it is also an injection, then it is a one to one function, so there exists a unique semantic composition $h^*(d_1.d_2 ... d_n)$ whose evaluation equals f. By hypothesis :

$T(g^*)[t(u_1)...t(u_n)]=f=h^*(d_1.d_2 ... d_n) \Longrightarrow T(g^*)=h^*$ and $\forall i\in\{1, ..., n\}\ t(u_i)=d_i$.

As T is a one to one function, $T(g^*)=h^*$ is equivalent with $g^*=T^{-1}(h^*)$.

As $<G, \{g_i\}_{1\leq i\leq m}, K, H>$ is a Fully Compositional Set, the fact that $w\in L(G)$ and $f=H(g^*(w))$ imply that $g^*(w)=(T^{-1}(h^*))(w)$ is a Structural Example for G. □

Example 8 :

The condition of Proposition 3 is very strong, as it implies that from a meaning representation, a structure can be inferred. Then, this structure fits both for the syntax and the semantics. Unfortunately, this property is not satisfied by the Fully Compositional Set <Γ, {g1, g2, g3}, K_L, H_L> defined in Examples 1, 5 and 7. As a matter of fact, it has already been shown in Example 7 that :

$f1(f1(\lambda P_1 \lambda Q_1 \exists x[P_1(x) \wedge Q_1(x)], man'), run') = \exists x[man_1'(x) \wedge run_1'(x)]$

and : $f1(f'1(man', \lambda P_1 \lambda Q_1 \exists x[P_1(x) \wedge Q_1(x)]), run') = \exists x[man_1'(x) \wedge run_1'(x)]$

The same meaning representation can be obtained from various semantic compositions applying on various strings (for this logical formula, others are possible).

It can nevertheless be noticed that Γ is a non ambiguous grammar : it means that *for a fixed sentence $w \in L(\Gamma)$,* there exists a unique Structural Example for Γ noted g*(w) so that H(g*(w)) represents the meaning of w. But this is not at all a necessary condition for being able to infer a Structural Example from any input couple <w, f>. As a matter of fact, it is very possible that a Fully Compositional Set <G, $\{g_i\}_{1 \leq i \leq m}$, K, H> contain ambiguous sentences. In this case, let w_0 be one of them. Then, there exist different possible analysis trees for w_0 in G corresponding with different Structural Examples for G respectively noted : $g_1^*(w_0)$,..., $g_k^*(w_0)$ and the set $\{H(g_i^*(w_0))\}_{1 \leq i \leq k}$, contains every possible meaning interpretation for w_0. The input data can be any couple of the form : <w, $H(g_i^*(w_0))$> with $1 \leq i \leq k$, and an admissible Structural Example for G is then any $g_j^*(w_0)$ with $1 \leq j \leq k$, satisfying : $H(g_j^*(w_0)) = H(g_i^*(w_0))$, even if $g_j^*(w) \neq g_i^*(w)$.

4.3 Learning Structural Examples with word meanings

When Proposition 3 does not apply (or when it does not lead to any efficient algorithm), other methods must be employed. Let us first consider the most simple case : when meaning assignment to words (i.e. the function t) is already known. H is, then, completely specified. From Proposition 2, the only remaining point is the inference of a Structural Example for G from any input couple. We will now investigate this sub-problem.

As already stated in 2.2, trying and storing every possible composition g*(w) compatible with every given string of words w is time and space exponential (the exact calculus is included further in proposition 5). Furthermore, usual models do not provide any criterion for selecting the real Structural Examples among them. Thanks to the

property of Fully Compositionality, semantic information will be able to play this selecting role.

The framework naturally leads to two different strategies respectively called *forward learning* and *backward learning* because of their similarity with usual forward chaining and backward chaining in deduction theory.

In forward learning, the idea is to try every possible syntactic composition applicable on the input sentence until one of them is translated into the correct meaning input. This strategy is described by Algorithm 2.

for an input couple <w, f> where w is a sentence and f its semantics do :

- *found <— false ;*
- *while not (found) do :*
  - ** try a composition e=g*(w) based on w ;*
  - ** if H(e)=f then found <— true ;*
- *return(e).*

Algorithm 2 : forward learning based on word meanings

The backward learning strategy, best exemplified by Proposition 3, works the other way : from the meanings back to the sentences. A semantic-based method of this kind but applicable whichever are the properties of semantic compositions is shown in Algorithm 3. It takes as starting point the word meanings and try every possible way to compose them by the functions $\{h_i\}_{1\leq i\leq m}$ until the meaning of the sentence is reached. T^{-1} is then applied on this semantic composition to obtain a Structural Example for G.

for a couple <w, f> where $w=u_1...u_n$ is a sentence and f its semantics do :

- *found <— false ;*
- *while not (found) do :*
  - ** try a composition $k=h^*(t(u_1)...t(u_n))$;*
  - ** if k=f then found <— true ;*
- *return($(T^1(h))^*(w)$)*

Algorithm 3 : backward learning based on word meanings

Proposition 4 :

If <G, $\{g_i\}_{1\leq i\leq m}$, K, H> is a Fully Compositional Set, then for every input couple <w, f>, Algorithms 2 and 3 always stop and return a Structural Example for G.

Proof :

By definition, for every input couple <w, f>, $w\in L(G)$ and there exists a Structural Example for G noted g*(w) so that f=H(g*(w)). Algorithm 2 tries every possible composition and Algorithm 3 tries all of their isomorphic semantic correspondence. They are in finite number, so both algorithms always stop. As <G, $\{g_i\}_{1\leq i\leq m}$, K, H> is a Fully Compositional Set, the composition returned is always a Structural Example for G. □

Of course, combined strategies crossing Algorithms 2 and 3 may be defined. Both algorithms are, *in the worst case* (i.e. if the only composition which provides a Structural Example is the last checked), exponential in time but *linear in space*, as only one composition is memorized at a time for every input couple.

Proposition 5 :

Let us suppose that every function$\{g_i\}_{1\leq i\leq m}$ used to define a composition is of arity 2 (Structural Examples of the kind given in Example 5 can easily be adapted to satisfy this property by removing the function g3, which is semantically unnecessary).

The number of different possible compositions g*(w) defined from m different functions $\{g_i\}_{1\leq i\leq m}$ of arity 2 on a sentence $w=u_1...u_n$ composed of n words with $n\geq 1$ equals : $C(n-1)*m^{n-1}$ where C(k) is the Catalan number defined by $C(k)=(2k)!/(k!(k+1)!)$ for any $k\in\mathbb{N}$ and the space occupied to store any of them is O(n).

Proof :

The Catalan number C(n-1) is the number of different possible binary trees built on n leaves. The only difference between a binary tree built on n words and a composition applying on these words is that in compositions, each internal node receives an index among {1, ..., m}. Each of the possible tree have exactly n-1 internal nodes, so for every tree there are m^{n-1} possible assignments of indexes. The total number of compositions is then $C(n-1)*m^{n-1}$.

Every composition can be defined by n words (whose length is bounded by $Max_{u\in\Sigma}(|u|)$) separated by n-1 couples of indexed parentheses and can then be stored in O(n). □

It can be noticed that the number m of different function symbols $\{g_i\}_{1\leq i\leq m}$ should be as small as possible. In this sense, skeletons (where m=1) are better than Structural Examples of the kind given in Example 5 (where m=2 after removing the function g3). Unfortunately, no compositional semantics has ever been defined from skeletons.

The strategies proposed allow to save much in space but are still *in the worst case* very expensive in time. It is nevertheless hoped that theoretical bounds can be much improved in practice, as natural and powerful heuristics should be available when the framework is completely specified.

Example 9 :

In our example framework, a simple heuristic should greatly improve the performance of Algorithm 3. It can be expressed by « when trying a composition on a sequence of semantic expressions, first try functional applications between two logical formulas whose arities are in decreasing order ».

This heuristic allows to find very quickly the semantic composition corresponding with a real Structural Example for Γ in every already seen example sentence :

- from the semantic sequence : « John' . run'_1 », the only semantic composition respecting the heuristic is the one which applies the function of arity 1 run'_1 on the argument (of arity 0) John' and defined by : $f'1(John', run'_1)=run'_1(John')$.
- from the sequence : « $\lambda P_1\lambda Q_1\exists x[P_1(x)\wedge Q_1(x)]$. man'_1. run'_1 », the heuristic engages to try in priority, among 8 possible semantic compositions, the one defined by $f1[f1(\lambda P_1\lambda Q_1\exists x[P_1(x)\wedge Q_1(x)], man'_1), run'_1]$ which is the only correct one.
- from the sequence : « John' . $love'_2$. Mary' », the heuristic leads in priority to two semantic compositions defined respectively by : $f'1[John', f1(love'_2, Mary')]$ and $f1[f'1(John', love'_2), Mary')]$ among which the first one is the correct one.

It is possible to imagine situations in which this heuristic would not be efficient. From a sequence of the kind : « $\lambda P_3[P_3(Mary')(Paul')(John')]$. $introduce'_3$ », it would lead to try a wrong functional application between the expression $introduce'_3$ (of arity 3) and the lambda expression (of arity 1). But this lambda expression is unnatural : it contains 3 logical constants and, in fact, must come from another one of arity 4...

4.4 Learning strategies without word meanings

When no word meaning is supposed to be initially known, then the target output of the learning algorithm includes the function K, the grammar G and the Basic Translation Function t. The problem is then much more difficult and nearly no one seems to have investigated it since now.

We suggest to treat it by introducing a *current hypothesis set* noted Z for the Structural Examples associated with a sentence and for the meaning assignment to words. This set is empty at the beginning of the learning process and then contains every possible solution. It is updated after each example presentation.

The global architecture of the learning framework proposed is displayed in Figure 6. The learning strategies used in Algorithms 2 and 3 can be adapted to this new framework, as shown in Example 10.

Example 10 :

Let us illustrate a simple learning sequence without word meanings, using an adapted version of Algorithm 3. At the beginning, Z=∅.

• Let us suppose that the first input couple is : <John runs, run_1'(John')>

- the only possible semantic compositions applying on the sequence of unknown word meanings t_L(John) . t_L(runs) are f1(t_L(John), t_L(runs)) and f'1(t_L(John), t_L(runs)).

- the evaluation of both hypotheses are :

  * f1(t_L(John), t_L(runs))=t_L(John)(t_L(runs))

  By hypothesis, this formula must equal the meaning input : run_1'(John'). By identification, this leads to : t_L(John)=run_1' and t_L(runs)=John'. Another solution would be : t_L(run)=run_1' and t_L(John)=$\lambda P_1[P_1$(John')] but we strictly stick to the simplest solution.

  * f'1(t_L(John), t_L(runs))=t_L(runs)(t_L(John))

  The same identification leads to : t_L(John)=John' and t_L(runs)=run_1'.

At this stage, we have no reason to prefer one of these hypotheses. The new current hypothesis set can then be written as :

Z={g1(John, runs) : (John, run_1'),(runs, John')}

OR {g2(John, runs) : (John, John'),(runs, run_1')}.

• Now, let us suppose that a second given example is <Mary runs, run_1'(Mary')>. The same process applies, except that *runs* now belongs to the current hypothesis.

- the two possible semantic compositions, similar to the previous ones, are : $f1(t_L(Mary), t_L(runs))$ and $f'1(t_L(Mary), t_L(runs))$.

- the corresponding evaluations are :

  * $f1(t_L(Mary), t_L(runs))=t_L(Mary)(t_L(runs))$

  It is clear that if $t_L(runs)=John'$, the identification with the formula $run_1'(Mary')$ is impossible. If $t_L(runs)=run_1'$ then we must have : $t_L(Mary')=\lambda P_1[P_1(Mary')]$.

  * $f'1(t_L(Mary), t_L(runs))=t_L(runs)(t_L(Mary'))$

  Similarly, if $t_L(runs)=John'$, the identification is impossible but if $t_L(runs)=run_1'$ this leads to : $t_L(Mary)=Mary'$. This simpler solution is then chosen.

So, the first subset of the current hypothesis is given up. It can be noticed that the hypothesis concerning *John* in this subset is also given up, although it was not concerned by the new example. A similar conclusion would have followed if the second example had been <John sleeps, $sleeps_1'(John')$>, which would have allowed to discard the first hypothesis concerning *John* and the associated one concerning *runs*. Only one repetition of a word is enough to select the correct meaning hypothesis concerning this word.

The new current hypothesis set is :

$Z=\{(g2(John, runs), g2(Mary, runs)) : (John, John'), (runs, run_1'), (Mary, Mary')\}$.

Without semantics, it would have been impossible to decide between the two initial possibilities. The only reason why *runs* must be translated by run_1' is that its translation behaves like a function, a predicate.

Now, if the new input couple is <a man runs, $\exists x[man_1'(x) \wedge run_1'(x)]$>, the learning process gives the following results (see [Tellier 98] for more details) :

- among the 8 possible semantic compositions, 7 are compatible with the arity of the translation of *runs* in Z ;

- among these 7 possible compositions, only 4 lead to an equation which has a solution (one of them has two solutions).

So five new hypotheses are built, among which three will clearly be discarded very easily at the next repetition (the two remaining ones are admissible).

Remarks :

From this simple example, it is clear that Z should have the structure of an AND/OR tree. The updating process should then be performed carefully.

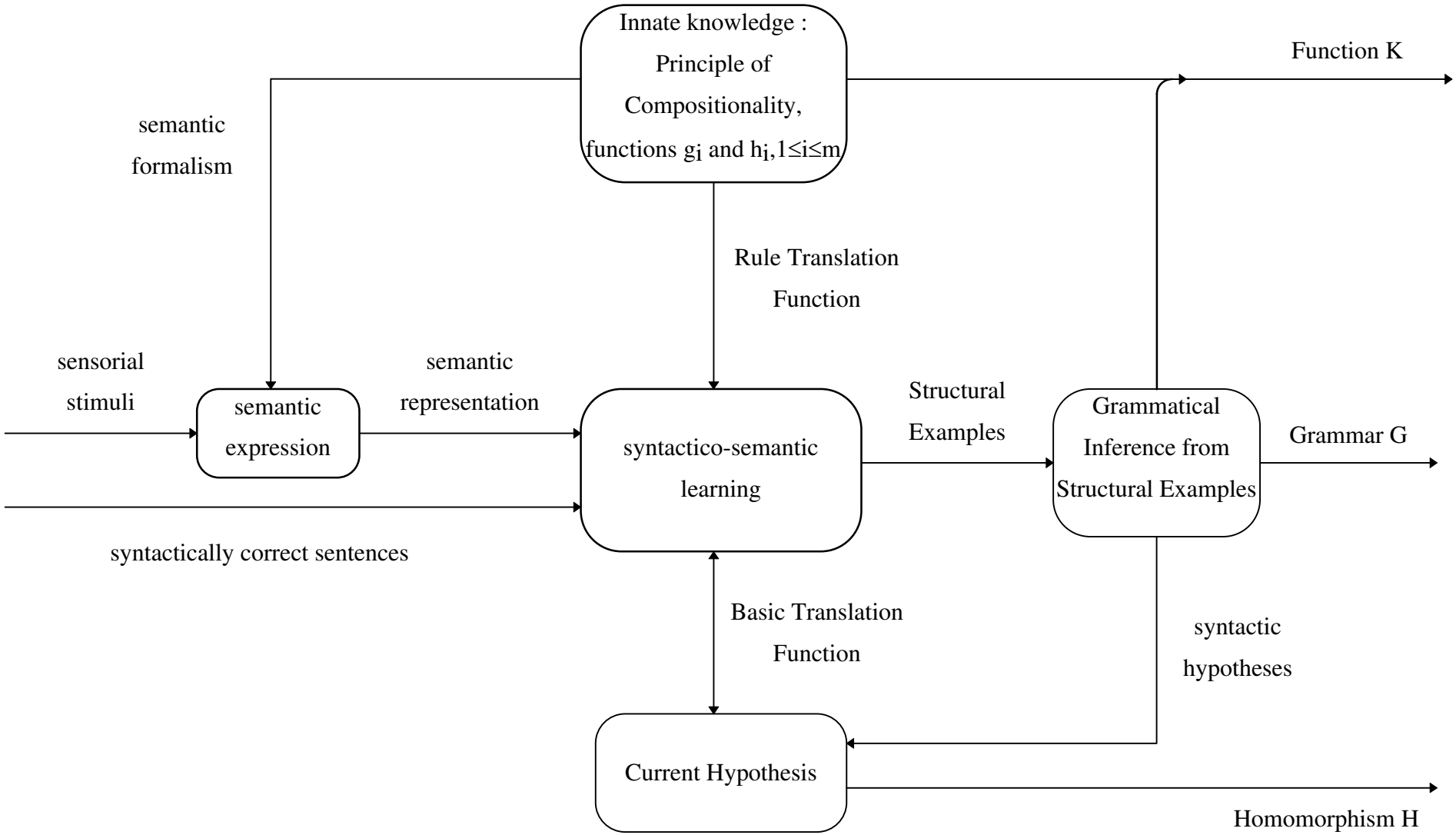


Figure 6 : the learning framework architecture

When the current hypothesis is empty, the algorithmic complexity of this strategy is equal to the worst calculated one. As a matter of fact, no heuristic is available and if at each step only the simplest solution is to be selected, every possibility has to be checked. But in fact the complexity of any new example couple *is relative to the current hypothesis*. This *relative complexity* can be measured by the number of new words in the example. The presentation order of the example couples (from the simplest ones to more and more complex ones) will then be crucial for the learning efficiency. To master this complexity, it is possible to put a bound on the number of new words appearing in a new sentence. An example which does not respect this bound should not be treated right away but saved so as to be treated later, when the current hypothesis is developed enough. It is reasonable to think that children also learn this way.

Finally, it may be efficient to sometimes send the part of the current hypothesis concerning the Structural Examples to the program able to learn a grammar from Positive Structural Examples. As a matter of fact, if one (or several) hypothesis(es) grammar(s) is (are) available, it (or they) may be used as heuristics during the search for syntactic compositions in learning algorithms adapted from Algorithm 2 (recall that the problem of the syntactic analysis is polynomial for grammars in Chomsky Normal Form). The conditions under which this strategy is incremental remains to be studied.

Syntactico-semantic language learning without initial word meanings seems to be possible under conditions on the order of presentation of the examples. In cases when this problem is nevertheless too difficult, it is possible to adopt an intermediate position, providing the knowledge of *content words* (whose semantic translation is elementary) but not of *grammatical words* (whose translation may include lambda abstractions).

4.5 A psycholinguistic interpretation

Our conception of language learning meets the psycholinguistic opinion that : « knowing a language is knowing how to translate mentalese into strings of words and vice-versa » ([Pinker 94]).

We believe that the treatment of the example couples in Example 10 reflects the natural learning process and that syntactico-semantic repetitions are necessary for this learning to take place. Imagine someone being told « John runs » in front of the scene of someone running. Even if the learner correctly infers that what he was told really meant « someone runs » (this is the « Gavagaï problem », introduced in [Quine 60], also see

[Pinker 94], chapter 5) and that this meaning is expressed by a predicate denoting the running action and a proper name denoting the subject of this action, he has absolutely no way of inferring which of the words he heard was the predicate and which one was the name. This inference only becomes possible if in front of a new scene of *someone else running* our learner is now told « Mary runs » (or, in front of *the same person asleep*, he is told « John sleeps »). By associating the common points between both scenes and the common points between both heard utterances, a first coherent current hypothesis can be built.

The exponential complexity *in the worst case* of the learning algorithms seems unavoidable as far as context-free grammars produce trees and as far as semantics under-determines the syntactic structures. In our example framework, from the logical formula : $love_2$'(Mary')(John'), it is possible to build a semantic composition corresponding with every possible ways of ordering a subject S, a verb V and an object O in a sentence :

- SOV : f'1(John', f'1(Mary', love $_2$'))
- VOS : f1(f1(love$_2$', Mary'), John')
- VSO : f1(f1(λxλy.love$_2$'(y)(x), John'), Mary')
- SVO : f'1(John', f1(love $_2$', Mary'))
- OVS : f1(f'1(Mary', love $_2$'), John')
- ...

The fact that nearly every possible ordering is represented in at least one natural language proves that Proposition 3 surely does not apply for natural languages.

If we take seriously the mentalese hypothesis ([Fodor 75]), the universals of natural languages can be interpreted in our model as coming from properties of this universal language of thought. The logical language L used here can be considered as an approximation of this mentalese, noted M, supposed to be innate or, at least, learned before language acquisition starts. The set of every possible natural language is then : $\{H^{-1}(M)$; H Global Translation Homomorphism$\}$, which may not be equivalent with the set of every context-free language.

5. Conclusion

In this paper, we first provide a unified definition for various kinds of Structural Examples used in the Grammatical Inference literature. We then develop a formal

expression of the Principle of Compositionality and introduce the new important notion of Fully Compositional Sets. Both frameworks are designed to be compatible and thus allow to throw a bridge between two previously separated domains. In this context, Structural Examples appear to be an interesting intermediate representation between the syntax and the semantics of a language. The result is the introduction of a new problem called « Learning to Understand » which generalizes the Classical Positive problem of Grammatical Inference and has strong theoretical and cognitive interests.

Thanks to these definitions, a crucial sub-problem of Grammatical Inference (i.e. ; the problem of associating a *structure* with a string of word) is shown to be closely related to a completely different one (i.e. the problem of building a global meaning representation from semantic items). Finally, we propose general learning strategies and heuristics which we think should be the basis for every solution to this problem. When word meanings are known, they allow an exact learning of context-free grammars. Putting these pieces of a puzzle together gives rise to a new architecture for the modeling of Natural Language Learning linked with psycholinguistic hypotheses.

The definitions are illustrated and proven relevant thanks to a simple example of Fully Compositional Set based on a formalism appreciated in the community of computational linguists and whose power can easily be extended. The paper then provides new arguments for the interests of Classical Categorial Grammars in the domain of Natural Language.

We thus hope to propose a safe basis for the study of a very complex domain which opens new directions of research. For example, for the learning of a given grammar G, a minimal kernel of Structural Examples generating G can first be identified, and another minimal kernel of input couples allowing to produce the first kernel can then be looked for. Providing such kernel would ensure the learning to take place.

The author thanks François Denis and Alain Terlutte for their very helpful advice.

6. References

[Anderson 77] : J. R. Anderson, "Induction of Augmented Transition Networks", *Cognitive Science* 1, p125-157, 1977.

[Bar-Hillel 60] : Y. Bar-Hillel, C. Gaifman, E. Shamir, "On Categorial and Phrase Structure Grammars", *Bulletin of the Research Council of Israel*. 9F, p1-16, 1960

[Buszkowski & Penn 90] : W. Buszkowski, G. Penn, "Categorial grammars determined from linguistic data by unification", *Studia Logica* 49, p431-454, 1990.

[Chomsky 57] : N. Chomsky, *Syntactic Structures*, Mouton & Co, the Hague, 1957.

[Chomsky 65] : N. Chomsky, *Aspects of the Theory of Syntax*, Cambridge, MIT Press.

[Chomsky 68] : N. Chomsky, *Language and Mind*, Brace & World, 1968.

[Dowty 81] : D. R. Dowty, R. E. Wall, S. Peters, *Introduction to Montague Semantics*, Reidel, Dordrecht, 1989.

[Feldman 98] : J. A. Feldman, "Real Language learning" in [ICGI 98], p114-125.

[Fodor 75] : J. Fodor, *The Language of Thought*, Thomas Y. Crowell, New-York, 1975.

[Fodor 83] : J. Fodor, *The Modularity of Mind*, MIT Press, Cambridge, 1983.

[Gold 67] : E. M. Gold, "Language Identification in the Limit", *Information and Control* 10, P447-474, 1967.

[Hamburger & Wexler 75] : H. Hamburger, K. Wexler, "A mathematical Theory of Learning Transformational Grammar", *Journal of Mathematical Psychology* 12, p137-177, 1975.

[Hill 83] : J. C. Hill, "A computational model of language acquisition in the two-year-old", *Cognition and Brain Theory* 6(3), p287-317, 1983.

[ICGI 96] : Proceedings of the third International Colloquium on Grammatical Inference, LNAI 862, Springer Verlag, 1996.

[ICGI 98] : Proceedings of the fourth International Colloquium on Grammatical Inference, LNAI 1433, Springer Verlag, 1998.

[Janssen 97] : T. M. V. Janssen, "Compositionality", in : *Handbook of Logic and Language*, Elsevier, Amsterdam and MIT Press, Cambridge, J. Van Benthem & A. ter Meulen (Eds), p417-473, 1997.

[Kamp 93] : H. Kamp, U. Reyle, *From Discourse to Logic ; Introduction to the Modeltheoretic Semantics of natural language*, Reidel, Dordrecht, 1993.

[Kanazawa 96] : M. Kanazawa, "Identification in the Limit of Categorial Grammars", *Journal of Logic, Language & Information*, vol 5, n°2, p115-155, 1996.

[Langley 82] : P. Langley, "Language acquisition through error discovery", *Cognition and Brain Theory* 5, p211-255, 1982.

[Mäkinen 92a] : E. Mäkinen, "On the structural grammatical inference problem for some classes of context-free grammars", *Information Processing Letters* 42, p1-5, 1992.

[Mäkinen 92b] : E. Mäkinen, "remarks on the structural grammatical inference problem for context-free grammars", *Information Processing Letters* 44, p125-127, 1992.

[Montague 74] : R. Montague, *Formal Philosophy; Selected papers of Richard Montague*, Yale University Press, New Haven, 1974.

[Muskens 93] : R. Muskens, "A compositional discourse representation theory", in P. Dekker & M. Stokhof, (Eds), Proceedings of the 9th Amsterdam Colloquium, p467-486, 1993.

[Oehrle 88] : R.T. Oehrle, E. Bach, D. Wheeler (Eds), *Categorial Grammars and Natural Language Structures*, Reidel, Dordrecht, 1988.

[Partee 90] : B. Partee, A. ter Meulen, R. E. Wall, *Mathematical methods in Linguistics*, in "Studies in Linguistics and Philosophy" n°30, Kluwer, Dordrecht, 1990.

[Piatelli-Palmarini 79] : M. Piatelli-Palmarini (Ed), *Théories du langage, théories de l'apprentissage, Le débat entre Jean Piaget et Noam Chomsky*, Le Seuil, Paris, 1979.

[Pinker 79] : S. Pinker, "Formal models of language learning", *Cognition* 7, p217-283, 1979.

[Pinker 94] : S. Pinker, *The Language Instinct*, Penguin Press, London, 1994.

[Quine 60] : W. V. O. Quine, *Words and objects*, MIT Press, Cambridge, 1960.

[Sakakibara 90] : Y. Sakakibara, "Learning context-free grammars from structural data in polynomial time", *Theoretical Computer Science* 76, p223-242, 1990.

[Sakakibara 92] : Y. Sakakibara, "Efficient Learning of Context-Free Grammars from Positive Structural Examples", *Information and Computation* 97, p23-60, 1992.

[Savitch 87], W. J. Savitch, E. Bach, W. Marsh, G. Safran-Naveh, *The Formal Complexity of Natural Language*, Studies in Linguistics and Philosophy, vol. 33, Reidel, Dordrecht, 1987.

[Sempere & Nagaraja 98] : J. M. Sempere, G. Nagaraja, "Learning a Subclass of Linear Languages from Positive Structural Information", in [ICGI 98], p162-173.

[Shinohara 90] : T. Shinohara, "Inductive inference of monotonic formal systems from positive data", p339-351 in : proceedings of Algorithmic Learning Theory, S. Arikara, S. Goto, S. Ohsuga & T. Yokomori (eds), Tokyo : Ohmsha and New York and Berlin, Springer, 1990.

[Tellier 98] : I. Tellier, "Meaning helps learning syntax", in [ICGI 98], p25-36.

[Valiant 84] : L. G. Valiant, "A theory of the learnable", *Communication of the ACM*, p1134-1142, 1984.

[Wexler & Culicover 80] : K. Wexler, P. Culicover, *Formal Principles of Language Acquisition*, Cambridge, MIT Press, 1980.