# Sign Grammar

J.J. Sarbo, J.I. Farkas

Computing Science Institute/

**CSI-R9922 October**

*KUN, CSI Technical Report, no. 9922, pp. 1–18*

# SIGN GRAMMAR

**Janos Sarbo**[*] **József Farkas**[**]

**Abstract**

Sign Grammar is a dependency based approach to syntactic structure in which the dependency relations are derived on a semiotic basis, from C.S. Peirce's theory of signs. We illustrate the potential of Sign Grammar by using English as an example. We argue that, by means of its syntactic structures, the English language implements signs analogous to 'real' world signs introduced by C.S. Peirce. No familiarity with semiotic is assumed in the paper.

**Keywords**

Language modelling, dependency, semiotic, C.S. Peirce, parsing algorithm.

[*]CSI : Computing Science Department, Faculty of Science, University of Nijmegen, The Netherlands. Email : janos@cs.kun.nl
[**]CSI , Email: jfarkas@cs.kun.nl

## INTRODUCTION

Sign Grammar (SG) is a dependency grammar (Tesnière L. 1959; Hudson R. 1984) in which the dependency relations are derived on the basis of a semiotic theory. We will argue that by virtue of its deeper foundation SG is capable of handling syntactic phenomena which are traditionally found difficult for NLP, like coordination and discontinuous modification.

Because language is based upon signs, we argue that it can be more adequately modelled if we take the properties of signs and, in particular, the necessary conditions for signs to function as signs as a starting point. The sign theoretic foundation of our approach is due to C.S. Peirce (1839-1914). Peirce's semiotic theory is strongly related to his theory of categories. He advocates that sign function must take place according to those categories. We will argue that Peirce's semiotic, in particular, his classification of signs can be used as a specification of a model of natural language. We will illustrate the potential of such a model by using English as an example.

Peirce's theory is difficult to comprehend, and this might require some effort from the reader. However, we do encourage her/him to take the challenge. Like Tamino from Die Zauberflöte, who first was put through a number of trials, but eventually got his reward, so will the reader, having passed the trial of Peirce's semiotic, be rewarded in the end, we hope, by the simplicity and power of our model of language.

The structure of this paper is as follows. After a brief exposition of Peirce's semiotic, a specification of Sign Grammar is derived stepwise from the properties of signs. This part is followed by an application of the specification developed to the English language. Finally, an attempt is made to clarify the correspondence between Peirce's signs and English syntactic structures.

A condensed version of this paper can be found in (Farkas J. & Sarbo J. 1999). A comparison with the 'generative' approach, from a philosophical point of view, is given in (Debrock G. & Sarbo J. 1998) and (Debrock G. *et al.* 1999).

## 1. LANGUAGE AND PEIRCE'S SEMIOTIC VIEW

Language is a tool for communication, for sending messages from speaker to hearer. Messages are signs which tell us about phenomena in the broad sense of the word including thoughts and knowledge. According to the pragmatist view of Peirce, phenomena can only be known by signs. This emphasis on knowledge through signs amounts to a correction of the traditional view of knowledge, inasmuch as it points out that no knowledge of reality or of phenomena is possible without mediation.

Let us recall that according to the traditional classification of things and phenomena in categories, which goes back to ancient Greek history and was first criticised by I. Kant (1724-1804), categories have an immutable structure, and phenomena have immutable properties. Language, according to this view, is a quasi-thing with a certain immutable structure and immutable properties.

### 1.1. Triadic structure

The simplest way of understanding how signs function is presumably via an example. Let our sign be a flashing blue light. In general, a sign stands for something other than itself. This other is called the *object* of the sign. For example, the object of our sign can be an ambulance, which may, or may not be visible for us. The sign represents its object to some agent in some way. This mediation is called the sign's effect, or its *interpretant.* For example, the effect of our sign can be a thought like 'the flashing blue light coming from the ambulance is a warning'.

Functionally, the perceived sign is connected to its object and triggers an interpretant. The sign, its object and interpretant are in this way triadically related to each other, and it is their relation that we call a *Sign.* We must emphasise that the components of a Sign are themselves Signs. This follows from Peirce's assumption that only Signs are there.

### 1.1.1. Structural characterisation

From the triadic relation of Sign we can draw conclusions about its types of structure. As Signs can only be known via signs, we will characterise these types from the sign's point of view. A sign can inform us about three types of structure:

1) a sign in itself, unrelated to anything else;
2) a sign related to another thing;
3) a sign triadically related to two other things.

First, because a sign stands for its object, the 'sign in itself' can be said to be equivalent to 'the object in itself'. Second, the 'another thing' can only be the object of the sign, because if we know the interpretant of the sign, then we must know its object as well. Finally, the 'two other things' must be the sign's object and interpretant.

According to the above classification, we can distinguish between unary, binary and tertiary relations. Because, structurally, a tertiary relation is already sufficient for a sign to function as a sign, it follows that higher arity relations are not necessary (n.b. nullary relations cannot be perceived).

The unary structure is about a thing presenting itself, e.g a substance. The binary one is about a thing related to another thing, e.g. a representation. The tertiary one is about a thing representing another thing, according to some rule or habit, e.g. a formal law. In virtue of these properties, the structural types of sign are also called: presentational or material, representational or relational, and interpretational or formal.

### 1.1.2. Functional characterisation

The triadic relation of sign also allows us to characterise Signs functionally. Again, from the sign's point of view, a sign can inform us about three types of function:

1) a sign's reference to itself;
2) a sign's relation to its object;
3) a sign's relation to its interpretant.

First, a sign can refer to itself, or equivalently, to an inseparable object. Such a reference, which lacks the aspect of relatedness, is called a quality. Second, a sign can also point to its object, because it stands for that object. Third, a sign can refer to the mode, object and sign are mediated by the interpretant, because the sign represents its object to some agent in some fashion. In virtue of these properties, the functional types of sign are also called: quality, indexicality and mediation.

By combining the orthogonal characterisations of signs, structural and functional, we get nine types of signs (see fig. 1). These signs, which are also classified in Peircean triads, material, relational and formal, can be briefly characterised as follows (Corrington R. 1993):

**Qualisign** a pure quality which is a sign.
**Sinsign** an actually existing thing or event (occurring only ones) which is a sign.
**Legisign** a law or a general type which is a sign.
**Icon** a reference by virtue of some similarity to its object.
**Index** a reference by virtue of being affected by its object.
**Symbol** a reference by virtue of some law or association to some more basic and recurrent features.
**Rheme** a sign of qualitative possibility, a possible object for its interpretant.
**Dicent sign** a sign of actual existence, a referent to a particular rheme.
**Argument** a sign by law that embodies the qualities of a rheme and the particularities of a dicent sign.

### 1.1.3. Peirce's categories

By analysing the two dimensions of fig. 1 we can observe that they share a number of properties which correspond to the mode of existence of phenomena. Peirce calls these modes,

| | | Structural type | | |
|---|---|---|---|---|
| | | 1 Material | 2 Relational | 3 Formal |
| Functional type | 1 Quality | Qualisign | Icon | Rheme |
| | 2 Indexicality | Sinsign | Index | Dicent sign |
| | 3 Mediation | Legisign | Symbol | Argument |

**Figure 1:** Peirce's classification of signs

or aspects, the *categories*: firstness, secondness and thirdness. These categories represent three different viewpoints or perspectives for characterising phenomena.

Firstness is determined by inherent *qualities* the phenomenon has *independent* of anything else. Secondness involves a *relation* or reaction directed *towards* something else. Thirdness is determined by *mediation* that makes multiple phenomena into a community, by virtue of a *rule* or habit (Sowa J. 1998).

We can also observe that both dimensions of fig. 1 exhibit a sort of dependency and growing complexity which must be reflected in the dependency of categories. Indeed, secondness cannot be reduced to firstness, but it presupposes firstness, and, similarly, thirdness cannot be reduced to either firstness or secondness, but it presupposes both firstness (through secondness) and secondness. Conversely, the element of firstness remains a mere possible, unless it be actualised by some interaction; and the element of secondness remains brutal interaction unless it derives its meaning from thirdness. The dependency of categories is formalised by the ordering firstness<secondness<thirdness, where "<" is a total order on categories. This order applies naturally, as a polymorphic relation, to the dimensions of fig. 1.

### 1.1.4. Example

The sign of the flashing blue light is a quality, having the aspect of firstness (qualisign). When this light is pointing in the direction of an ambulance (an actually existing thing), then it has the aspect of secondness (sinsign). Last, when the flashing blue light is a sign of warning, then it has the aspect of thirdness (legisign).

## 2. LANGUAGE AND ONTOLOGICAL PERSPECTIVE

The goal of this section is to derive a specification of Sign Grammar from the properties of signs. Because this derivation refers to language in general, the examples illustrating the application of the derived model to a particular language are postponed to the next section.

### 2.1. Language signs

Language consists of symbols which are signs. From this it follows that language only uses a particular type of signs. We do experience, however, that language is able to describe all types of Signs in terms of signs. This indicates that language is able to 'implement' signs, analogous to those of fig. 1, by means of symbols. In this regard we only assume that the function of symbols, like the one of 'real' world signs, takes place according to the three categories.

### 2.1.1. Events and processes

Language appears as a form of interaction, whether we speak it, write it, or read it. Interactions involve events. In terms of Peirce's categories, they represent the category of secondness. But the mere fact *that* something happens says nothing whatever about *what* happens. The latter aspect is the aspect of thirdness. *What* happens in an event requires that the event be embedded in a context of events which are related to each other. Such web of related events is what is called a *process*. Because signs are generated from signs, and in turn generate other signs, every sign must be related to an event. From this it follows that language symbols are sign-events which, by virtue of their interpretants, are embedded within a process.

## 2.2. Syntactic signs

Language as a process is generated by symbol-events which are themselves generated according to rules which, in Peircean terms, are habits evolving from interaction with other symbol-events. Language processes involve both syntactic and semantic rules or habit.

Linguistic symbols may be considered as gesture-events within a process of interactive responses. Syntactic symbols may be considered from two angles: the messenger and the receiver. From the point of view of the messenger, a syntactic symbol is a gesture announcing other gestures to be generated in view of the interpretant of the entire unit of meaning (e.g. a sentence). From the point of view of the receiver, a syntactic symbol-event elicits an abduction regarding a range of possible subsequent symbol-events. Thus, in English, the symbol-event 'the' elicits an indefinitely large field of possible subsequent symbol-events, but excludes, for instance, the possibility of the next symbol-event being 'is'.

On the analogy to Peirce's characterisation of signs, we can pursue the analysis of symbol-events according to their syntactic value, and introduce a classification of language symbols likewise. Indeed, from a syntactic point of view, symbol-events have a specific function, regardless of their semantic function. The syntactic value of the language symbols making up the unit of meaning may be seen in the function of the value which they have in forming such a unit.

By virtue of their secondness, events are marked by a binary relation. Therefore, linguistic symbol-events must be also binary. This is why, strictly speaking, one lexical item by itself has no meaning. The syntactic value of symbol-events will therefore depend upon the *sort* of relation that obtains between two language symbols. If one of the symbols has by itself no information content and therefore is a mere quality (a phoneme or a visible character), it will need another symbol to actualise its 'potential' content. Such nexus of two symbols, one of which is self-sufficient, but the other has mere potential content, may be called a *proto-symbol* (P) which corresponds to the category of firstness. An example of this is the symbol-nexus of free morpheme and affix.

Similarly, when the nexus is constituted by an asymmetrical relation between one language symbol which derives its full content from its association with another language symbol which is in principle self-sufficient, it may be called a *deutero-symbol* (D) which corresponds to the category of secondness. An example of this is the symbol-nexus of adjective and noun, or the one of determiner and noun.

Finally, when the nexus consists of two language symbols which are self-sufficient but together generate the interpretant of the unit formed by the string, e.g. a sentence, it will be called a *trito-symbol* (T) which, by its aspect of thirdness, mediates between the language symbols constituting a unit of meaning, or a thought, in the Fregean tradition. An example of this is the symbol-nexus between verb and subject.

To complete the picture, it is necessary to say a word about the *triadic relation* characterising each of these signs, because without such relation, they would not be signs, let alone syntactic signs. But precisely what makes them *syntactic* signs is the very fact that they stand for specific *rules* or habits. Thus, the object of syntactic signs is the rule for which they stand. Their interpretant on the other hand is the generation of the selection of the next symbol-event. The interpretant of the entire string of language symbols is, from a syntactic point of view, the establishment of the correctness of the string, regardless of its semantic content.

### 2.2.1. Argument and functor

The dual characterisation of signs can also be applied to syntactic signs. By virtue of their aspect of secondness, symbol-events refer to a binary relation, structurally, and, to a pair of different functions, one for each participant symbol, functionally. We call these functions argument and functor, which, according to their dependency exhibited, define the ordering argument<functor.

## 2.3. Towards a classification of syntactic signs

Inasmuch as linguistic symbols are also syntactic symbols, proto-, deutero-, and trito-symbols constitute a Peircean triad in the class of linguistic symbols. By virtue of their category exhibited, these signs define the ordering P<D<T which in turn defines the *levels* of syntactic signs. In the remaining, we will denote by X a level of syntactic signs, and by $X'$ the level subsequent to X. A sign (or symbol-event) of some level X will be called an X-level sign (or symbol-event).

Language implements syntactic signs basically by lexical items and their relations. These are called *syntactic structures* or, equivalently, *language units*, depending on whether we want to emphasise their structural or linguistic properties. In the mapping of syntactic signs to syntactic structures, called *syntactic mapping*, the notion of argument and functor plays a crucial role. Argument and functor, the types of sign we have derived on the semiotic basis, appear in language due to the combinatorial properties of lexical items. These combinatorial properties can be characterised as relational or argumental need. A lexical item has *relational need* if it can be a functor, and *argumental need* if it can be an argument in some relation.

By analysing the functional structure of the three types of syntactic sign, we can easily recognise an argument and a functor symbol in each of them. In the case of trito symbols, the functor is that symbol which has the most relational need in the determination of the interpretant. We assume that such a distinction can always be made.

We denote the constituents of an X-level symbol-event, the argument and the functor symbol, and the syntactic symbol itself as $X_1$, $X_2$ and $X_3$. By virtue of the category and dependency which different signs respectively exhibit, syntactic signs may be said to define the ordering $X_1<X_2<X_3$ which in turn defines the *classes* of level X. The total order on levels and on classes can be merged, by flattening, to a total order on syntactic signs.

The syntactic sign emerging from a symbol interaction is called its *descendant*. A syntactic sign that has no combinatorial need is called a *completed* or well-formed sign. Two symbols are said *incompatible* if they cannot establish a relation syntactically, and *compatible*, otherwise. A completed sign is incompatible with any symbol. A sign of class $X_i$ (i=1,2,3) of some level X is denoted an $X_i$ sign.

## 2.4. The emerging syntactic sign

From a receiver's point of view, input symbols have merely potential content according to the receiver's (parser's) hypothesis. The set of such hypotheses is called the parser's dictionary. Input symbols appear one after the other and syntactic signs emerge as a result of their interactions, by *symbol relation*. This might be called the 'automatic' type of sign generation. Because the descendant contains, besides the meaning of its constituents, the additional meaning of the relation itself, the syntactic signs generated by symbol relation are monotonously increasing.

But there are also cases of a degenerate symbol interaction. One of them is the interaction between incompatible symbols. In such a case, one of the interacting symbols, which is a sign generated in one symbol-event, is coerced to an argument or a functor, but *not* both, in another symbol-event. The other symbol is left unchanged. Accordingly, we will say that the sign coerced (the descendant of symbol coercion) increases its meaning, and enters a higher class and/or level of syntactic signs. This type of sign generation is called *symbol coercion*. By virtue of their more developed interpretants, the syntactic signs generated by symbol coercion are monotonously increasing.

Syntactic signs are composite signs which meet certain criteria. Thus, if a syntactic sign consists of related signs, the signs involved must in principle be *contiguous* to one another. This requirement is based upon the triadic structure of a syntactic sign the object of which is always a rule expressive of the expectation that a certain type of language unit must be followed by another type of language unit. The contiguity property can be defined as a *covering* relation on the ordering of syntactic signs (Davey B. & Priestley H. 1990).

The contiguity property is the driving force behind symbol coercion. Let us assume that two symbols $S_1$ and, subsequently, $S_2$ are recognised, which are incompatible. Assume, furthermore, that $S_2$ has combinatorial need on some higher level. Then, $S_1$ is forced to enter a higher class without symbol relation. This follows from the assumption that the entire input, e.g. a sentence, is a well-formed syntactic sign. Due to the monotonicity property, if $S_2$ (or its descendant) enters a higher level, then the contiguous signs $S_1$ and $S_2$ (or its descendant) will only be able to establish a symbol relation on a higher level. Therefore $S_1$ must enter a higher class, as well, conform to its combinatorial need.

Syntactic sign generation respects, besides the combinatorial need of the symbols, also their syntactic properties. For example, an interaction between an X-level sign, and a sign entering the X-level subsequently, is syntactically different from the one in which the two signs are interchanged. From the monotonicity property of syntactic signs, it follows that a lower level combinatorial need must have priority over a higher level one. The handling of one combinatorial need may require the elaboration of another, recursively.

## 2.5. Towards an algorithm for syntactic signs

The properties of symbol coercion are formalised as follows ($X_i \rightarrow Y_j$ denotes that a sign of class $i$ of level X may enter class $j$ of level Y, and $X_i \rightarrow Y_j \vee Y_k$ is a shorthand for $X_i \rightarrow Y_j \vee X_i \rightarrow Y_k$):

($\alpha_1$) $X_1 \rightarrow X_3$;  ($\alpha_2$) $X_1 \rightarrow X_1' \vee X_2'$;  ($\alpha_3$) $X_3 \rightarrow X_1'$.

In sum, $X_1$ and $X_3$ signs can increase their meaning without symbol relation, but an $X_2$ sign, due to its indexicality (aspect of secondness), presupposes an $X_1$ sign, and must relate with it. This meets our expectation that a relational need must be fulfilled always, though an argumental need can be optional.

Because language possesses only a finite number of lexical items, some syntactic signs must be generated *incrementally*, via degenerate symbol relations. In such a relation, the mediation aspect is incomplete, and the descendant of the X level symbol interaction will become an $X_1$ or $X_2$ sign on the same level. Accordingly, the rules of symbol relation are formalised as follows (the symbol relation of $X_1$ and $X_2$ is denoted as $X_1$-$X_2$):

($\beta_1$) $X_1\text{-}X_2 \rightarrow X_1 \vee X_2$;  ($\beta_2$) $X_1\text{-}X_2 \rightarrow X_3$;  ($\beta_3$) $X_1\text{-}X_2 \rightarrow X_1' \vee X_2'$.

## 2.6. Cumulative signs

Because of the incremental nature of syntactic signs there may be encountered simultaneously more than one sign of the same class. By virtue of its aspect of firstness, an $X_1$ class may contain a number of signs which are *unrelated*, but the collection of which is a sign (cf. section 4.1). By virtue of its aspect of secondness, an $X_2$ class may contain a number of signs which are *unrelated*, but which share a *common referent*. By virtue of its aspect of thirdness, an $X_3$ class may contain a completed sign which must be a single sign. In each case, the signs belonging to a class must have the same combinatorial need, and must be consistent with each other, with respect to their meaning (i.e. the interactions involved in the signs).

Based upon the above properties of sign classes we can refine our definition of incompatibility as follows. Symbols will be said *incompatible* if (i) they are of different classes of a level and cannot establish a relation syntactically, or (ii) they share the same class, but have different combinatorial need or meaning, or (iii) one of the symbols is a completed sign.

The third type of sign generation, *symbol stacking*, which is another case of a degenerate symbol interaction, is the interaction between compatible symbols of the same class. In such a case, the symbols involved are accumulated on a stack. The need for a stack is explained as follows. Assume that the subsequent symbols, $S_a$ and $S_b$, enter the same class, $X_1$ or $X_2$, of some level X. Let $S_a$ and $S_b$ be such that they cannot enter a higher level, because their combinatorial need on level X is not yet fulfilled. As a consequence, both symbols have to be stored, temporarily. Let another symbol, $S_c$, enter the other class, $X_2$ or $X_1$, of level X. Now, $S_c$ and $S_a$ are *not* contiguous symbols, and, therefore, should not interact, however, $S_c$ and $S_b$ must interact. These requirements can be satisfied by storing $S_a$ and $S_b$ on a stack. From the sign generation's point of view, a stack of signs will be considered a single sign represented via the topmost item of the stack. Symbols generated by stacking are, trivially, monotonously increasing.

The stacking of symbols is subject to restrictions implied by the property of contiguity, the binary nature of interactions, and the properties of symbol classes. For example, the sequence of input symbols $S_a$, $S_b$, $S_c$ cannot yield the analysis: $X_1 = S_a/S_c$ (where '/' is a leftassociative stack constructor) and $X_2=S_b$, because $S_b$ must have a different (binary) relation with $S_a$ and with $S_c$, and therefore the collection of $S_a$ and $S_c$ cannot function as a common referent of $S_b$.

Let us define the state of a level X by the pair of stacks of $X_1$ and $X_2$ (the $X_3$ stack has no effect on the stacking of symbols and therefore omitted). Let $s_i$ ($i$=1,2) denote the stack of $X_i$, $\perp$ the empty stack, and $s$ the next symbol entering level X. Then, symbol stacking must respect the following rules, specified as transitions on the state of a level X:

$(s_1,\perp)\to(s_1/s,\perp)$; $(\perp,s_2)\to(s,s_2)$; $(s_1,s_2)\to(s_1,s_2/s)$.

In sum, an $X_1$ stack can grow if the $X_2$ stack is empty, or otherwise, by a single item only (the common referent); and an $X_2$ stack can grow if the $X_1$ stack is constant. Stacking is a hypothesis which, due to a next symbol interaction can be further developed. It may turn out, for example, that the stacked signs have *different* combinatorial needs, and therefore, they cannot form part of the same sign, or cannot share a common referent. In that case, the stack has to be split and a segment of it, which must be a tail segment, by the contiguity property, has to leave the stack (as a single sign) via symbol relation or symbol coercion. We will exemplify such a case in section 4.1 and 5.1. (N.B. the above rule does not allow us to have a tail segment of $X_1$ enter $X_3$, because a completed sign has no combinatorial need, let alone a different one.)

## 2.7. Primary signs

We assume that the input symbols enter a lowest class (prm) as a sequence of primary signs, e.g. phonemes or characters. Prm, which has the aspect of firstness, is by definition a class of syntactic signs. The input primary signs, which have no combinatorial need, are collected in prm, as long as their sequence forms a morphological symbol which is a dictionary entry. When this happens, the symbol receives its combinatorial need from the dictionary, and enters the lowest level, in particular, $P_1$ if it has no P-level relational need; and $P_2$, otherwise:

($\gamma$) $\text{prm}\to P_1 \vee P_2$.

## 2.8. Mediating evaluation

Syntactic signs are yielded by symbol interaction. But the decision as to *when* the mediation must take place depends upon the type of evaluation, which can be lazy or greedy. In general, we will assume lazy evaluation of relations, because it can be more economic in some cases. However, there is always the possibility to choose between both types of evaluation which can become a source of ambiguity, for example, in the case of signs known as PP attachment structures.

The lazy evaluation of syntactic sign-events affects the modelling of the terminator symbol (e.g. the point symbol) which, therefore, will be treated as an incompatible argument and a nullary functor on each level, thereby forcing the realisation of *pending* relational needs.

Summarised, syntactic signs arise in language due to (1) the quality of contiguity, (2) symbol interaction, and (3) mediating evaluation which, respectively, have the aspect of the three categories, firstness, secondness and thirdness.

The emerging syntactic sign may become part of a cumulative sign, or change its aspect of correspondence with its object, or establish a relation with another sign. In sum, symbol interactions do emerge by (1) symbol stacking, (2) symbol coercion, and (3) symbol relation, which, in view of their descendants, exhibit the aspects of Peirce's three categories.

## 3. SYNTACTIC MAPPING

We illustrate the syntactic mapping of language by using English as an example. In this mapping we capitalise on the semiotic properties of syntactic signs, and on the syntactic and conceptual distinctions that may be expressed in English (Aarts F. & Aarts J. 1982), (Farkas J. *et al.* 1997).

Trito-symbols correspond to the symmetric relation between two constituents which are both self-sufficient and require the presence of the other, e.g. the relation between noun and verb (subject and predicate). Such a relation is called *predication*(p).

Deutero-symbols correspond to the asymmetric relation between two self-sufficient constituents, one of which requires the presence of the other, but the reverse does not hold. In English, two instantiations of this type of relation can be identified: *modification*(m), e.g. the relation between adjective and noun; and *qualification*(q), e.g. the relation between determiner and noun.

The third type of symbols, proto-symbols, correspond to the morphological phenomenon of *affixation*(a). An affixation relation distinguishes between a root (or base), e.g. a free morpheme, and an affix: the root is self-sufficient, and the affix has only potential meaning actualised by the root. The affix may relate, for example, to the number aspect of nouns (e.g. singular vs. plural), or to tense and aspect information expressed by the verb.

From the semiotic point of view, q- and m-signs form subsets of deutero-symbols. A qualification adds extra information to its argument, e.g. definiteness, but the resulting sign is similar (as to its meaning) to the one qualified. There is however, no such likeness in a modification, which is a sign pointing in the direction of its argument and selecting it from other signs via a general property. Using the analogy of the relational triad, q- and m-signs represent iconic and indexical meaning, respectively. Therefore, these signs may be said to define the order relation q<m. Eventually, this yields the ordering a<q<m<p, which in turn defines the *levels* of the English syntactic signs.

Though the above characterisation of the English syntactic mapping is based upon simple considerations, it may not be language model independent. In section 5 we will show however, that this mapping is certainly not accidental. It will be argued that the English language implements signs, by means of syntactic structures, analogous to those of Peirce's semiotic triads.

## 3.1. Syntactic relations

Syntactic relations emerge due to the combinatorial need of syntactic symbols. In general, a syntactic symbol can have argumental need, optionally, but its relational need is a function of that of its constituents, or, in the case of a lexical item, it is a constant value. Lexical items can contribute to the relational need of syntactic signs on each level.

A lexical item has a potential combinatorial need, which is a finite set. The combinatorial need of a syntactic symbol generated by a symbol interaction is the disjoint *union* of the combinatorial need of its constituents, possibly modified (i.e. restricted) by the interaction itself. The potential relational need of the types of lexical items is exemplified in fig. 2 (respectively, a '+' or '-' represents the presence or absence of a relational need on the level indicated by the column). The relational need of a particular lexical item is the subset of that of its type.

For example, the q-level relational need of adjectives and adverbs allows symbol-events like *keep awake*, or *walk by*; and their m-level relational need the modification like *happy girl*, or *walk quickly*. In the case of a preposition, the q-level relational need contributes to the relation with the obligatory argument (qualification); and the m-level one to the modification between the optional argument and the qualification yielded. From the semiotic point of view, the q-level relation yields a qualified preposition (a PP, syntactically), whereas the m-level one completes it with the aspect of selection (indexicality).

Verb–complement relation is classified as modification. Indeed, such a symbol-nexus fits the definition of a deutero sign: verb and complement are both self-sufficient, but the verb derives its full content from the complement. Because the descendant of a verb–complement relation has an indexical character (the verb points in the direction of its complement it is acting on), this type of relation is identical with modification (we admit that the term modification might be confusing for the linguist). The common framework for modification with a PP, and for verb–complement relation indicates the possibility of a Peircean semiotic treatment of verb–argument structure.

In sum, a verb relates with its complement(s) due to its m-level relational need (which is fulfilled when all necessary complements are found), and with the subject, due to its p-level one. A copula or an auxiliary relates with its complement due to its q-level relational need, but the copula relates with the subject due to its p-level relational need. The SV(O) rule of English is modelled by demanding that, a sign having p-level relational need entering some level X, is incompatible with any $X_1$ sign except for a $p_1$ one, potentially.

The development of the relational need of syntactic signs can be illustrated as follows. The potential m-level relational need of a preposition will be actual if the q-level relation it is involved in does not disallow that. For example, there will be such need in the case of *in London*, and there won't be, in the case of *drive in*.

| | a | q | m | p |
|---|---|---|---|---|
| primary | - | - | - | - |
| affix | + | - | - | - |
| noun | - | - | - | - |
| determiner | - | + | - | - |

| | a | q | m | p |
|---|---|---|---|---|
| preposition | - | + | + | - |
| adjective | - | + | + | - |
| adverb | - | + | + | - |
| verb | - | + | + | + |

**Figure 2:** Potential relational need

### 3.1.1. Example

The following example illustrates the parsing of a sentence with a Sign Grammar of English. The result of the analysis is displayed in fig. 3.

(1) *Mary drank some wine yesterday.*

We leave out the morphological analysis (also in later examples), and assume that the input signs leave the a-level and enter $q_1$ or $q_2$ conform to their combinatorial need. The potential relational need of the lexical items is as follows: *Mary*={}, *drank*={m,p}, *some*={q}, *wine*={}, *yesterday*={m}. Symbols having a non-empty relational need are written in capitals. In the table of fig. 3 below, an item represents the content of the storage of a class (column) prior to the evaluation of an input symbol (row).

| step | next input | $q_1$ | $q_2$ | $q_3$ | $m_1$ | $m_2$ | $m_3$ | $p_1$ | $p_2$ | $p_3$ |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Mary(m) | | | | | | | | | |
| 1 | drank(D) | m | | | | | | | | |
| 2 | some(S) | D | | m | | | | | | |
| 3 | wine(w) | | S | | | D | m | | | |
| 4 | yest(Y) | w | S | | | D | m | | | |
| 5 | . | Y | | s-w | | D | m | | | |
| 6 | . | . | | . | D-s-w | Y | m | | | |
| 7 | . | . | | . | D-s-w-y | | m | | | |
| 8 | . | . | | . | . | | . | m | D-s-w-y | |
| 9 | . | . | | . | . | | . | . | | m-d-s-w-y |

**Figure 3:** "*Mary drank some wine yesterday*"

In $\text{step}_1$ the symbol *Mary* enters $q_1$. Next, in $\text{step}_2$, the symbol *drank* enters $q_1$ (it has no q-level relational need, but it has argumental need, optionally). Because *Mary* and *drank* are incompatible on this level, $q_1$ (*Mary*) must increase its class by $\alpha_1$ and enter $q_3$. The next input, *some*, which is incompatible with $q_1$ (*drank*), forces the $q_1$ sign to increase its level by $\alpha_2$ which, in turn, forces the $q_3$ sign (*Mary*) to do the same by $\alpha_3$ and $\alpha_1$. In $\text{step}_4$, the symbol *wine* enters $q_1$. The appearance of the next symbol, *yesterday*, triggers the symbol relation between *some* and *wine* yielding the symbol *some–wine* by $\beta_2$ (the relational need of *some* is now satisfied).

The rest of the evaluation is as follows (we specify for each step which rule is applied to which sign, but we omit the treatment of the point symbol): $\text{step}_5$ $\alpha_2(q_1)$, $\alpha_3(q_3)$, $\beta_1(m_1,m_2)$;

step$_6$ $\beta_1$(m$_1$,m$_2$); step$_7$ $\alpha_2$(m$_1$), $\alpha_3$(m$_3$); step$_8$ $\beta_2$(p$_1$,p$_2$).

### 3.2. SG meets dependency

In this section we briefly compare Sign Grammar and Dependency Grammar (DG), on the basis of N.M. Fraser's definition of DG (Fraser N. 1996). His article also contains a list of references to relevant DG literature (this is not included here).

In SG and DG, syntactic structure is expressed in terms of relations, which are, respectively, symbol interactions and dependency relations. Both sorts of relations are based on grammatical and syntactic relations (but SG does not make use of syntactic functions).

The 'governor' of a dependency relation matches SG's notion of a functor (i.e. an X$_2$) sign. Whereas DG allows multi-arity relations, symbol interactions are binary. SG, like most DGs, is subject to the 'single application constraint'. The rules of a SG, which are the $\alpha_i$, $\beta_i$ and $\gamma$ rules ($i$=1,2,3) instantiated by the corresponding dictionary entries and by the levels of syntactic signs, are additive, except for a finite number of non-recursive rules (cf. symbol coercion).

Fraser mentions four constraints which are usually assumed in dependency grammar: (i) there is one root, (ii) all symbols are connected, (iii) each symbol must depend on exactly one other symbol, and (iv) governors (i.e. heads) and dependents are adjacent. With respect to these constraints, the following can be said.

SG's demand that, a well-formed input must be a completed sign, corresponds with the single 'root' and connected dependency structure conditions of DG. The 'one head' requirement of DG is automatically satisfied by SG, due to the monotonicity of syntactic signs. Finally, the contiguity property of SG can be said to be the *semiotic* counterpart of the notion of 'adjacency' of DG.

## 4. PARSING WITH SIGN GRAMMAR

### 4.1. Structural variations

We mentioned in section 2.6 that the collection of signs of a class can be considered a single sign. In the case of an X$_1$-stack, this possibility turns out to be useful in the parsing of signs known as compound nouns, and phenomena, like 'marked' word order, apposition and discontinuous modification.

In the case of 'marked' word order, the stack can reorder signs as formally explained in section 2.6. An example of this is given in fig. 4. In step$_3$ the two symbols *Flowers* and *Mary* are stacked in m$_1$, according to the abductive hypothesis that the two symbols can form part of an m$_1$ sign. It turns out, however, due to the symbol interaction with the next symbol (*likes*), that these signs cannot be unrelated. This triggers the postulation of another hypothesis, according to which, a tail segment of m$_1$ (*Mary*) is popped and coerced to m$_3$ (in step$_4$)

Compound nouns and apposition structures can be parsed likewise, except that in such cases the stacking hypothesis need not have to be revisited.

| step | next input | q$_1$ | q$_2$ | q$_3$ | m$_1$ | m$_2$ | m$_3$ | p$_1$ | p$_2$ | p$_3$ |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Flowers(f) | | | | | | | | | |
| 1 | Mary(m) | f | | | | | | | | |
| 2 | likes(L) | m | | f | | | | | | |
| 3 | . | L | | m | f | | | | | |
| 4 | . | L | | . | f/m | | | | | |
| 5 | . | . | | . | f | L | m | | | |
| 6 | . | . | | . | . | | . | m | L-f | |
| 7 | . | . | | . | . | | . | . | | m-l-f |

**Figure 4:** "*Flowers, Mary likes*"

The analysis of a sample discontinuous modification is depicted in fig. 5. Notice the presence of the $p_1$ sign, *A boy came in*, in step$_{10}$. This sign, which is a clause, syntactically, is stacked with the next symbol, *who*, in step$_{11}$. Now, $p_1$ contains two signs, which are unrelated, but the collection of which is a sign (the evaluation of anaphoric relations, like wh-dependencies, is not part of our analysis). Finally, in step$_{13}$, the symbol relation between $p_1$ and $p_2$ yields the completed sign in $p_3$: [*A boy came in/who*] *was covered with mud* (the operator [_] converts a stack of signs to a single sign).

| step | next input | $q_1$ | $q_2$ | $q_3$ | $m_1$ | $m_2$ | $m_3$ | $p_1$ | $p_2$ |
|---|---|---|---|---|---|---|---|---|---|
| 0 | A(A) | | | | | | | | |
| 1 | boy(b) | | A | | | | | | |
| 2 | came(CA) | b | A | | | | | | |
| 3 | in(I) | CA | | a-b | | | | | |
| 4 | who(wh) | CA | I | a-b | | | | | |
| 5 | was(WA) | wh | | | CA-i | | a-b | | |
| 6 | covered(CO) | | WA | wh | CA-i | | a-b | | |
| 7 | with(WI) | CO | WA | wh | CA-i | | a-b | | |
| 8 | mud(m) | | WI | | WA-co | | wh | a-b | CA-i |
| 9 | . | m | WI | | WA-co | | wh | a-b | CA-i |
| 10 | . | . | | . | WA-co | WI-m | wh | a-b | CA-i |
| 11 | . | . | | . | WA-co | WI-m | wh | a-b-ca-i | |
| 12 | . | . | | . | WA-co | WI-m | | a-b-ca-i/wh | |
| 13 | . | . | | . | | | | a-b-ca-i/wh | WA-co-wi-m |

**Figure 5:** "*A boy came in who was covered with mud*"

Also an $X_2$-stack can be considered a single sign. This can be useful, amongst others, in the parsing of compound verbs, and complex modifiers etc. Due to their common referent, the symbol relation between $X_2$ and $X_1$ satisfies the corresponding relational need of each of the signs of $X_2$, simultaneously.

## 4.2. Coordinate structures

Coordinate structures are traditionally considered too complex to be described adequately both in terms of dependency and constituency. We argue that, due to its semiotic foundation, Sign Grammar is capable of modelling such structures. We will restrict ourselves to coordination with the *and* coordinator.

From the semiotic point of view coordination starts on the appearance of the coordinator symbol. This implies that only the signs present at that stage, and their descendents, can be coordinated with. Coordination is an operation on signs which are contiguous to the coordinator. Coordination starts with the lowest sign following the coordinator, and proceeds, according to the monotonicity property of syntactic signs, with the signs of higher classes, in increasing order. As a result, all coordinate structures will be found, eventually.

The analysis of a coordinate structure proceeds as follows. First, the signs preceding the coordinator are analysed (non-deterministically), and saved, temporarily. Second, the input following the coordinator is analysed stepwise. Whenever a sign of some class is found, such that, there is a sign among the saved ones, of the same class, and consistent with the sign found, then the two signs are coordinated. This involves the inheritance of relations between the saved sign and the coordinated one, in agreement with their combinatorial properties (n.b. inheritance, in general, can also be triggered by the descendants of a coordinated sign).

Third, upon a successful coordination, the analysis of signs preceding the coordinator is resumed starting from last (highest class) sign coordinated. If, eventually, all signs preceding the coordinator are known, the analysis proceeds with the parsing of the remaining input.

Technically, a coordinated sign is treated as a single sign, the future relations of which must be checked for both signs involved, separately. Information for keeping track of corresponding signs of a coordinate structure is maintained (but omitted in the examples).

## 4.3. Coordination algorithm

Coordination is a meta-level operation affecting the parsing algorithm itself. A formal account of coordination requires that the parser itself be specified. Coordination refers to signs from the simultaneous analysis of two input strings. Let us denote the state of the parser as a pair $(\lambda,\rho)$ where $\lambda$ and $\rho$, respectively, refer to the signs generated from the input on the left- and righthand side of the coordinator (but a sign shared by both $\lambda$ and $\rho$, as a result of coordination, will be represented only once). A reference to a sign S is denoted as $\lambda$(S), or $\rho$(S), and the coordinated sign of S and T as S&T. Let $r=\{\alpha_i,\ \beta_i\ | i=1,2,3\}$, $p{\in}r$, $q{\in}r{\cup}\{\gamma\}$, and let $p.\lambda$ ($q.\rho$) denote the application of a rule $p(q)$ to some uniquely determined sign(s). Then, coordination must respect the following rules, specified as transitions on the state of the parser:

($\delta_1$) $(\lambda,\rho)\to(p.\lambda,\rho)$  ($\delta_2$) $(\lambda,\rho)\to(\lambda,q.\rho)$  ($\delta_3$) $(\lambda(\mathrm{S}),\rho(\mathrm{T}))\to(\lambda(\mathrm{S\&T}),\rho(\mathrm{S\&T}))$

where ($\delta_1$) and ($\delta_2$) refer to the analysis, respectively, on the left- and righthand side of the coordinator, and ($\delta_3$) to the coordination of signs.

### 4.3.1. Examples

Sign Grammar's approach to coordination is illustrated with the analysis of the following sentence.

(2) *Mary drank some wine yesterday and a coke today.*

A dependency analysis of this coordinate structure (Hudson R. 1995) yields crossing dependencies, which are forbidden, in general. But this example is also problematic for a constituency based account, because the conjuncts *some wine yesterday* and *a coke today* do not form a single unit at any level of analysis (see (Kamphuis V. 1998) for discussion).

Because the above sentence is an extension of (1), we will catch up with the analysis from step$_5$ (cf. fig. 6). The signs present at this stage are sufficiently analysed for a successful coordination and can be saved temporarily. Following the analysis of the input on the righthand side of the coordinator (cf. step$_7$–step$_9$), the coordination of the q$_1$ and q$_3$ signs (step$_9$) is effectuated in step$_{10}$.

| step | next input | q$_1$ | q$_2$ | q$_3$ | m$_1$ | m$_2$ | m$_3$ | p$_1$ | p$_2$ | p$_3$ |
|---|---|---|---|---|---|---|---|---|---|---|
| 5 | and | Y | | s-w | | D | m | | | |
| 6 | a(A) | | | | | | | | | |
| 7 | coke(c) | | A | | | | | | | |
| 8 | today(T) | c | A | | | | | | | |
| 9 | . | T | | a-c | | | | | | |
| 10 | . | . | | . | D-$\&^{\mathrm{s-w}}_{\mathrm{a-c}}$ | $\&^{\mathrm{Y}}_{\mathrm{T}}$ | m | | | |
| 11 | . | . | | . | D-$\&^{\mathrm{s-w}}_{\mathrm{a-c}}$-$\&^{\mathrm{y}}_{\mathrm{t}}$ | | m | | | |
| 12 | . | . | | . | . | | . | m | D-$\&^{\mathrm{s-w}}_{\mathrm{a-c}}$-$\&^{\mathrm{y}}_{\mathrm{t}}$ | |
| 14 | . | . | | . | . | | . | . | | m-d-$\&^{\mathrm{s-w}}_{\mathrm{a-c}}$-$\&^{\mathrm{y}}_{\mathrm{t}}$ |

**Figure 6:** "*Mary drank some wine yesterday and a coke today*"

The next example, taken from (Kamphuis V. 1998), is an illustration of the role of inheritance in coordination. The result of the analysis is depicted in fig. 7.

(3) *the belief in and fear of a vindictive God*

Coordination takes place in step$_5$, where the qualification of *the* is inherited; and in step$_6$ and step$_8$, where the qualification of *a*, and the modification of *vindictive* are inherited, respectively.

| step | next input | $q_1$ | $q_2$ | $q_3$ | $m_1$ | $m_2$ | $m_3$ |
|---|---|---|---|---|---|---|---|
| 0 | the(T) | | | | | | |
| 1 | belief(b) | | T | | | | |
| 2 | in(I) | b | T | | | | |
| 3 | and | | I | tb | | | |
| 4 | fear(f) | | | | | | |
| 5 | of(O) | f | | | | | |
| 6 | a(A) | | O | f | | | |
| 7 | vindictive(V) | V | [O/A] | f | | | |
| 8 | God(g) | g | | | $\text{t-}\&^{b}_{f}$ | $\&^{I}_{O}\text{-a-V}$ | |
| 9 | . | | | | g | $\text{t-}\&^{b}_{f}\text{-}\&^{i}_{o}\text{-a-V}$ | |
| 10 | . | . | | . | . | | $\text{t-}\&^{b}_{f}\text{-}\&^{i}_{o}\text{-a-v-g}$ |

**Figure 7:** *"the belief in and fear of a vindictive God"*

## 5. ENGLISH SYNTACTIC SYMBOLS AND PEIRCE'S SIGNS

In this section we show that, by its syntactic structures, the English language implements signs, analogous to those introduced by Peirce (cf. fig. 1).

The ordering of the classes of a level is depicted in fig. 8a (the order relation "<" is represented by an edge). In the case that degenerate signs are allowed, this graph can be paraphrased as a two-level scheme consisting of a finite automaton (FA) and a number of stacks. A state of the FA corresponds to a sign class, and a transition to an application of an $\alpha$ or $\beta$ rule which is represented by solid and dashed lines, respectively (but in later graphs we will use solid lines for both types of transition). The resulting graph is depicted in fig. 8b. An edge which is a cycle is omitted.

In virtue of the syntactic mapping, the classes of English syntactic signs define a total ordering as shown in fig. 8c. By using the interpretation of fig. 8b to fig. 8c, we get a two-level system (this is not illustrated). We map the $X_3$ and $X_1'$ classes, e.g. $a_3$ and $q_1$, to same states (same signs, different interpretants). The initial state is prm, all others are final states; each state has a stack. The output language of the system is the set of signs in the different stacks, upon termination.

We argue that the above system is terminating. This follows from the properties of the $\alpha$ and $\beta$ rules. Indeed, an $\alpha$ rule increases the class of the sign it is applied to, but there are finitely many classes; and a $\beta$ rule fulfils a combinatorial need, but the combinatorial need of a syntactic sign is finite.

The priority of a lower level combinatorial need over a higher level one implies a stack-like behaviour of the levels of syntactic signs; the properties of symbol stacking imply a stack-like behaviour of the classes of a level of syntactic signs. Accordingly, the individual stacks of the classes can be simulated by a single stack. Eventually we get that our two-level system is equivalent to a context free (CF) grammar, hence it has the same complexity as CF parsing.

The algorithm presented is capable of analysing multiple signs, which are independent of each other, simultaneously, by recursion. This potential of the algorithm might be necessary, for example, in the case of signs known as subordination and insertion. Because SG rules are additive, recursion will not affect the termination properties. The algorithm's complexity will not change either, because recursion can be implemented by means of a single stack.

Below we describe an equivalency transformation of our two-level system with respect to its input and output languages. In step1 (see fig. 9), we remove $m_2 \to p_1$. If the descendant sign $m_1$-$m_2$ has no m-level combinatorial need and enters the p-level, then, equivalently, we can push it to the $m_1$-stack, and let the m-level sign generated subsequently force it to enter the p-level by $\alpha_2$.

In step2 (cf. fig. 10), we close the $q_1 \to m_1 \to p_1$ transitions by $q_1 \to p_1$. Because these transitions only apply when the m-level stacks are empty, the storages are not affected by this transformation. Furthermore, we merge $q_2$ and $m_1$, because of the orthogonality of their signs