A Peircean ontology of language

Janos Sarbo and József Farkas

University of Nijmegen, The Netherlands
janos@cs.kun.nl

Abstract. Formal models of natural language often suffer from excessive complexity. A reason for this, we think, may be due to the underlying approach itself. In this paper we introduce a novel, semiotic based model of language which provides us with a simple algorithm for language processing.

1 Introduction

Formal models of natural language often suffer from excessive complexity which, in our opinion, may be due to the underlying approach itself. Their formal character implies that they are doomed to reflect what is 'natural' in language in an ad hoc fashion only.

In this paper we introduce – on the bias of logic – an alternative model of language which is built on the assumptions that (1) language symbols are signs, (2) the meaning of a sign emerges via mediation, and (3) signs arise from a dichotomous relation of perceived qualities. We argue that on the basis of these assumptions and the properties of signs, a simple parsing algorithm can be defined.

2 Sign and perception

In our analysis we follow the principles of Peirce's semiotic ([5], [7]). Accordingly, a sign signifies its object to an agent in some sense, which is called the interpretant of the sign. The inseparable relation of sign, object and interpretant (each of which is a sign, recursively) is called the *triadic relation* of sign. In this paper we start from the observation that the ground for any sign is a *contrast* in the 'real' world. Because sign and object are the primary representation of such contrast, sign and object must be *differ* from each other.

How can we know about signs? We have discussed this problem in [3] and here we will only recapitulate the main results. Following cognition theory ([4]), the recognition of any sign must begin with the sensation of the physical input. Physical stimuli enter the human receiver via the senses which transform the raw data into internal sensation continuously. The output of the senses, a bioelectric signal, is processed by the brain in percepts. The generation of such a percept is triggered by a change in the input, typically, or by the duration of some sampling time, e.g. in the case of visual perception.

The brain compares the current percept with the previous one, and this enables it to distinguish between two sorts of input qualities: one, which was there and remained there, something stable, which we will call a *continuant*; and another, which, though it was not there, is there now (or the other way round), something changing, which we will call an *occurrent.* The collections of continuants and occurrents, which are inherently related to each other, form the basis for our perception of a phenomenon as a sign. We also assume that, by means of *selective attention*, we recognise in these collections coherent sets of qualities: the qualities of the *observed* and those of the *complementary* part of the phenomenon. We will refer to these sets collectively as the *input.*

2.1 The variety of signs

In Peirce's view, the most complete signs are the icon, index, and symbol which represent their object on the basis of, respectively, similarity, causality and arbitrary consensus. Besides this taxonomy, Peirce also distinguishes signs, respectively, according to the categorical status of the sign, and according to the relationship between object and interpretant. From a categorical perspective, signs can be qualisigns, sinsigns or legisigns, which correspond, respectively, to firstness, secondness and thirdness. In other words, a sign can be a quality, an actual event, or a rule. Seen from the perspective of the relationship between object and interpretant, a sign may be a rheme, a dicent or an argument. In other words a sign may signify a qualitative possibility, an actual existence, or a proposition. Thus we obtain nine kinds of sign which may be arranged in a matrix as shown in fig. 1 (the meaning of the horizontal lines and directed edges will be explained later). Although Peirce defined more complex systems of signs, we hold that his 'simple' classification is the most practical.

Here, the expressions 'class of a sign' or 'type of a sign' will be used interchangeably. In our specification of logical and language signs we will make use of Peirce's classes. A comparison between our use of them and his definitions may be found in ([3]).

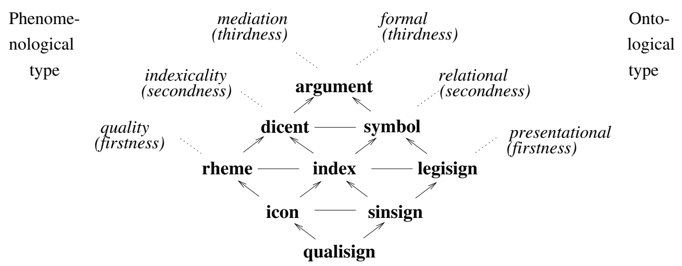

Fig. 1. Peirce's classification of signs

3 Logic and semiosis

We argue that semiosis begins with our experience of the input collections (which are qualities themselves) as signs. Such signs, which are called in Peirce's terminology a qualisign, are special signs for which we have no denotation. Although qualisigns are coherent, by definition, we experience them as independent signs. In order to be able to refer to the qualisigns, we will make use of logical symbols which are the most general of that type of sign. Logical symbols are signs from the logical point of view ([2]). We will represent such signs as *logical functions* on two variables A and B, respectively, for the continuants and the occurrents, over two values 0 and 1, for the complementary and the observed part.

How do complex signs emerge? Again, we refer to [3] where we introduced a semiotic model of signs and elaborated it for logical signs. Briefly, that model is based on a process in which trichotomic relations are generated recursively revealing gradually more accurate and clear *approximations* of the full richness of a sign of an observed phenomenon. Accordingly, in this paper we argue that the proposition of the input as a sign arises from the input qualisigns via a number of signs. By virtue of the fast and continuous nature of cognition we may assume that such signs are not recognised isolatedly, but only as 'temporary' signs. Such signs, which are approximations of the final assertion, are *re*-presentations of the input qualisigns. Their classes are identical to those defined by Peirce. We will argue that language is based on a similar mechanism.

The qualisigns form the ground for our semiosis. Because such signs are perceived as independent signs, but it is their unity that signifies the contrast as a whole, we may assume that there exists a *need* for the representation of the full richness of the relation of the qualisigns, eventually as a proposition. We will argue that this 'representational need' also appears in language in the form of the relational need of symbols.

The sign *mediates* between object and interpretant. In the case of logical signs, the interpretant is defined as the *application* of sign to object, both of which are logical functions. Here, the notion 'application' is used in a broad sense. In the particular case we allow that sign and object 'merge' as a result of such operation. Therefore, this form of semiosis will be called an *interaction*; sign and object will be referred to as its *constituents*.

A derivation of those 'temporary' signs, and, eventually, the proposition of the input as a sign proceeds as follows ([3]). We denote the signs of the collections of the *observed* part by the functions A and B (and those of the complementary part as $\neg A$ and $\neg B$). These collections are similar to the input and appear simultaneously, by definition. Something which is similar to something else can be a sign of it (icon). Such a sign, A *or* B, must refer to an object which includes both A *and* B (sinsign). The interpretant of such a sign and object can refer either to their common origin (via the complementary signs), which is called the *context* (index), or, to their relative difference. The latter provides us with a representation of the input collections independent of each other, the 'abstract' continuants (rheme), and how they co-occur, the 'abstract' occurrents (legisign).

The index can be represented by the Shäffer and Peirce functions, the rheme and the legisign, respectively, by the inhibition and exclusive-or functions.

The application of the index to the rheme and to the legisign yields the complementation of those abstract signs by the context, i.e. the actual constituent 'parts' of the input (dicent), and, the 'property' characterising their co-occurrence (symbol). Dicent and symbol can be represented, respectively, by the implication and the equivalence functions. Finally, by merging this property (as sign) with the sign of those actual parts (as object) we get a proposition of the input as a sign (argument) which can be represented by a syllogism (degenerately).

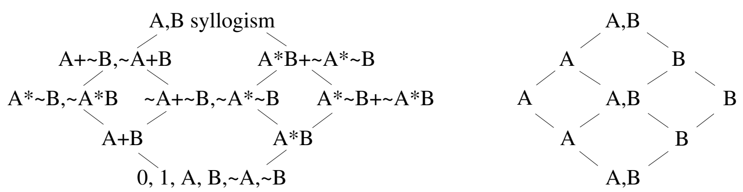

Fig. 2. The classification of logical signs

The logical representation of the qualisign can be completed with two more functions: 0 ('not valid input'), and 1 ('valid input'). As a result, we can conclude that in the semiosis of logical signs *all* Boolean functions (on two variables) can emerge. The resulting classification is depicted in fig. 2 (on the left-hand side).

Notice that in the derivations of the logical signs the interpretant always emerges from *neighbouring* sign and object (in the sense of the triadic relation). Such signs are connected in fig. 1 by a horizontal line. By virtue of the fundamental character of logic it may be conceived that semiosis can always be explained in a similar vein.

4 Language

We will argue that language is logic, *sequentially.* In this section we will show that by means of this single condition a model of language can be derived which is isomorphic to the one of logic. Language signs are symbols which are subject to syntactic and semantic rules. In this paper we will consider only syntactic rules, and restrict language to syntax ([6]).

'Sequential' means that the input signs appear one after the other as qualisigns. Earlier we have pointed out that the qualisigns are the (first) representation of a contrast. In our cognitive model we defined such a contrast between continuant and occurrent. Because language symbols are about 'real' world phenomena (typically), we may assume that an underlying contrast, analogous to the one of cognition, does exist in language, too. The language equivalent of continuant and occurrent is identified, respectively, in the aspects 'thing' and 'change', typically represented by nominals and verbs. Because language signs are inherently related to memory, a 'change' can also refer to an 'appearing' new fact, which is a relative change of some thing or event (i.e. an alteration), typically represented by adjectives and adverbs.

In what follows, we will refer by a sign class to the classification of *syntactic* symbols. In as much as the logical equivalent of the sign of a collection of 'things' and 'changes' must be the functions A and B, respectively, we will use the logical and syntactic names of the qualisigns interchangeably.

4.1 A preliminary classification

We will derive a model of language by transforming our specification of logical signs to a sequential one. Such a derivation is partly technical. In order not to get drown in the details, we make a preliminary attempt at a semiotic classification of the main syntactic concepts.

The argument, which is a proposition, must correspond to the notion of a sentence in as much as both are expressive of a statement. Hence, the dicent must be subject, and the symbol predicate. By virtue of its factual meaning (cf. 'modification') the index can be an adjective, an adverb, or a complemented preposition (in short, prep-compl). A rheme is a possible for the subject, for example, a noun; a legisign is an actual event in the syntactic sense, that is, a 'structure' event defined as a rule, e.g. a verb(-complement).

4.2 Towards a sequential version of logic

Qualisigns are the representation of a contrast. In the sequential case when each qualisign consists of a single symbol, also the contrast itself will appear sequentially. Accordingly, a qualisign will have either the aspect of a 'thing' (logically A), or a 'change' (logically B). From this it follows that in language we cannot distinguish between asserted and negated signs (at least not at this level), and this implies that language phenomena must include their own context. Hence, a part of the input may have to be devoted to the representation of the context as a sign. Akin to the general case of signs, we can have access to memory knowledge, e.g. a lexicon, but such knowledge is *not* related to the perceived input in any way.

Syntactic qualisigns consist of a single symbol and define a unique universe, therefore different qualisigns cannot be merged. Because input symbols appear continuously (i) *'place' has to be created for the appearing next qualisign.*

What can be done with the previous qualisign? The answer is simple, we have to re-present it by another sign. Following our classification of logic, such a sign can be an icon or a sinsign.

We mentioned that in the case of language (ii) *a qualisign is either A or B, but not both.* Because a sinsign is a representation of an event, it must include the aspect of a 'change'. Accordingly, the re-presentation of a qualisign which is B can be a sinsign, hence the one which is A must be an icon. Such a re-presentation involves the generation of a new sign, the denotation of which is *identical* to the one of the qualisign. From the bottom-up 'nature' of our logical model it follows that the new sign will include the older one degenerately (in the semiotic sense).

Icon and sinsign are different representations of the same qualisigns. Because of (ii) and the uniqueness of signs[1], it follows that icon and sinsign symbols typically will not be adjacent, therefore (iii) *icon and sinsign implement a 'sorting' re-presentation of qualisigns.* In particular, an icon and a sinsign can define a shared universe, for example, in the case of compound (multi-word) symbols, and idiomatic expressions.

By virtue of (i), we have to represent the previous qualisign by a new sign, which is an icon or a sinsign. The appearance of this new sign, in turn, may force us to do the same for the previous icon or sinsign and, eventually, signs may have to be generated in any class. In sum, the conclusion can be drawn that in the sequential model of logic there is sign generation *by need.*

4.3 Sign generation by need

The logical interpretant is defined as the application of sign to object. In language, such application will be called a 'binding'. Due to the sequential nature of language signs, we will also have to consider two degenerate variants of such interaction which are the following.

a) *Accumulation*: in which case an existing sign is combined with another sign of the same type. Such an interaction assigns the same meaning to both constituents thereby rendering them indistinguishable.

b) *Coercion*: in which case a new sign is generated for the denotation of an existing sign (which is said 'coerced'). Coercion applies if the signs, which are to interact, are incapable for accumulation or binding.

Accumulation is possible in any class, except for the qualisign. For example, a series of adjectives can be merged to a single sign via accumulation. Coercion is applicable in any class, except for the argument. In this form of an interaction we refer by the 'constituent' to the sign triggering the interaction.

Coercions play an important role in syntactic sign recognition, more specifically, in what we call as, the *default* scheme. This type of semiosis can be characterised as follows. By virtue of (i), signs are generated by need. The coercion of a qualisign, A or B, yields either an icon or a sinsign. Due to subsequent applications of (i), such an icon or sinsign is coerced, respectively, to a rheme or a legisign, and eventually to a dicent or a symbol. An index cannot emerge this way, because there are no negated signs (so far). By virtue of the appearing different kinds of qualisigns, in the end, we may have a dicent and a symbol sign which are adjacent and generate the argument sign (the sentence as a sign).

Notice that also a dicent or a symbol can be coerced to an argument, but such a sign will be a degenerate one, semiotically, because an argument must

[1] Lexical ambiguity of a symbol is treated by introducing a unique denotation for each meaning.

represent the observed phenomenon in its *character* ([5]) and has to include both A and B. Such signs of the input as a whole, are the dicent and the symbol signs.

In as much as the argument arises from dicent and symbol (subject and predicate), we may conclude that, in the default case, *only* subject and predicate are recognised and their relation represented as a sentence. It will be argued that language sign recognition always follows this scheme and any deviation from it may only occur if otherwise a successful parse cannot be found. Such a case will be described in the next section.

4.4 The genesis of the context

In language we are burdened by the task of the recognition of the entire string of input symbols as a single sign. Although, in some cases, the sign generation operations (coercion, accumulation and binding) may be unsatisfactory, the above goal can yet be achieved if we allow for a sign, which is potentially subject or predicate, to be represented degenerately (in the semiotic sense). Such signs define, what we called as, the *context*. The degenerate representation of symbols also plays an important role in the 'stepwise' construction of signs.

Sign degeneration ($\downarrow$) can be explicated by the phenomenological and ontological types of signs indicated in fig. 1 as follows: dicent$\downarrow$index if subject meaning is not present 'formally'; symbol$\downarrow$index if predicate meaning is not present 'mediationally'; dicent$\downarrow$rheme if subject meaning is unfinished 'indexically'; symbol$\downarrow$legisign if predicate meaning is unfinished 'relationally'.

Sample context signs are, for example, dicent$\downarrow$index in Mary, John likes (the potential subject Mary becomes a context sign for likes); symbol$\downarrow$index in Mary with flowers (the potential property with flowers is represented degenerately as a context sign for Mary).

The existence of degenerate signs is related to our ability of analysing a segment of input symbols (*nested input*) independently from the rest of the input. When such a segment is recognised, its meaning relative to the input as a whole is represented degenerately. Because of this degeneration such a symbol *will not* appear as an isolatedly recognised sign. The semiosis of such *nested signs* is implemented by a recursive application of the sign recognition 'machinery' (for example, in the case of coordination, subordination etc.).

In certain cases we can know in advance if a sign eventually will become a context sign. If a nested sign consists of a single input symbol, the recursive analysis may be replaced by a coercion. For example, an index sign can be directly generated by coercion from such an icon or sinsign. This kind of optimisation is typical for adjective, adverb and prep-compl symbols and it is lexically specified as their syntactic property. The 'stepwise' construction of a sign can be optimised by the immediate generation of the degenerate representation of the interpretant. For example, the interaction of such rheme and index can be directly represented as a rheme or an index sign, without explicitly generating its meaning as a dicent sign.

The representation of logical signs in the sequential case is depicted in fig. 2 (on the right-hand side). Although the same denotations, A and B, appear many times, each of the occurrences has a different meaning. A language implementation of these signs involves the mapping of the logical functions to their syntactic equivalent. The definition of such a mapping is the subject of the next section.

4.5 Relational need

The requirement that all symbols have to be 'merged' to a single sign and single universe is in conflict with their individual character and unique universe property. How can such symbols be combined?

An answer can be found in the properties of syntactic qualisigns. Such symbols are representations of one 'half' of a contrast. Because we can only reason about a contrast if both of its 'parts' are known, syntactic symbols can be said to be 'longing' for finding their complementary part. This inherent property of syntactic symbols is the language equivalent of the 'representational need' of signs introduced in sect. 3. This interaction 'potential' forms the *ground* of the relational properties of syntactic signs. In as much as the 'parts' of a contrast are different, syntactic sign interactions always arise between symbols of a different type of relational properties.

If, as we argued, logical and language signs are analogous, this must also apply to syntactic symbols. We map continuant and occurrent (via 'thing' and 'change'), respectively, to the *relational types* 'passive' (p) and 'active' (a). Formally, we define the type 'neutral' (n). The logical qualisigns, A and B, are mapped to a syntactic *relational need*, or valency, represented as a pair consisting of a relational type and a set of relational qualities (or, syntactic properties). In this paper it will be assumed that the set of such qualities is *finite*.

Because any sign is a re-presentation of the qualisigns, the relational qualities may contribute to different relational properties in each class. These qualities, which are lexically defined, will be omitted in the specification. Accordingly, we will refer to the relational need of any sign by its type only (and call it an a-, p- and n-need, ambiguously). A sign, which has an n-need, is finished (relationally). Such a sign cannot take part in any interaction, except for a coercion and, vice versa, only such a sign can be subject to a coercion (typically).

The above mapping defines an initial representation of the relational need of signs in the various classes. By virtue of the properties of logical signs, this mapping has to be further developed as follows. Notice that each modification amounts to an adjustment of the definition of the qualisigns.

We introduce an n-need in the qualisign, icon, sinsign and argument classes. Indeed, qualisigns are independent signs which cannot interact; icon and sinsign symbols typically do not establish a relation; and finally, the argument which is the sentence as a sign, must be 'complete' (and neutral, relationally). Because icon, index and symbol do function as sign, in the sense of the triadic relation, we introduce an (optional) p-need for each class which functions as object (e.g. for the legisign class). Finally, when sign and object, respectively, are initially assigned to a p- and an a-need, their relational needs are exchanged in the mapping. The reason for this modification can be explained as follows.

The predicate of the sentence can arise from a legisign (a verb) via the complementation of an index. Traditionally, such complement is lexically specified in the verb's entry. Semiotically, however, it is the complement that points to the verb and selects its actual meaning. This interpretation is conform with the default scheme of sign recognition, according to which, verbs only function as a sign in the predication symbol interaction. Because also adjectives, adverbs etc. function as sign (in the sense of the triadic relation), in our model of language the concept of 'modifier' and 'complement' amalgamate.

In sum, the relational need of a qualisign can be defined as a set, an element of which is a reference to a class in which the qualities of the input symbol can contribute to a relational need (of a sign) via re-presentation. For example, the valency of a transitive verb can be defined as {legisign,symbol} referring to an *a*-need, respectively, for the complement and the subject. A *p*-need, which is optional (typically) is omitted (also in the examples).

We demand that a relational need is always satisfied. In particular, a binding satisfies a pair of *a*- and a *p*-needs by resolving them; an accumulation merges a pair of needs to a single one; a sign generated by coercion inherits the valency of the sign coerced. In as much as a *p*-need is optional, when such a need is not present, it is equivalent to an *n*-need.

Because the index sign does not partake in the default scheme of syntactic sign recognition, the generation of such signs is subject to special conditions. We demand that a symbol can become an index having a *p*-need, either if any other analysis of that symbol eventually fails, or, if there is an existing *a*-need of a symbol in the legisign class.

4.6 A Peircean model of language

In this section we will specify language signs and their valency. Instead of using a set representation, we directly refer to the classes of fig. 1. With respect to the lexical denotation of language signs we will refer to the types of speech.

Syntactic qualisigns are defined as follows: A=noun; B=verb, adjective, adverb, prep(-compl), where 'compl' can be a noun, verb, adjective or adverb. Formally, we also define 0='no input' and 1='end of input'. The latter can represent the 'dot' symbol which is considered an A *and* B sign, having an *a*-need in any class, but incompatible with any sign except for itself. Hence, the dot symbol is capable of 'forcing' the realisation of pending interactions. In the examples we will assume that the input is closed by a finite number of dots. The specification is depicted in fig. 3 (on the left-hand side). An occurrence of a *p*-need may also denote an *n*-need, except for the dicent sign class (cf. subject).

4.7 Morphology

We argue that morphology can be modelled analogously to syntax and logic. Due to space, we can only run through the essential ideas behind such a specification. We characterise morphological symbols as follows (now we refer by a sign class to the Peircean classification of morphological signs).

Dicent and symbol signs are finished morphologically and represent only a *sorting* of signs. This is justified by the different syntactic properties of the signs of these classes. A symbol, for example, an adjective, is a sign which, *syntactically*, requires a complement adjacent to it in the input (on 'surface' level). A dicent, for example, a noun or a verb, does not have such a property.

Rheme, index and legisign, respectively, represent a qualitative possibility, a factuality (e.g. definiteness), and something rule-like (e.g. argument-structure). These signs are involved in the creation of the syntactic relational needs, passive and active (N.B. such a need should not be mixed up with the signs' morphological valency which, akin to syntax, determine the *morphological* sign interactions). Finally, icon and sinsign are either sorting, alike to syntax, or, when they are adjacent, are generating a new syntactic property, e.g. an adjective from a verb (such signs represent the same phenomenon and have a shared universe).

Morphological signs and their *morphological* relational needs are displayed in fig. 3 (on the right-hand side). The dicent sign is defined as a coerced rheme, or a rheme-index interaction, whereas the symbol as a coerced legisign, or an index-legisign interaction. The morphological argument sign coincides with the syntactic qualisign.

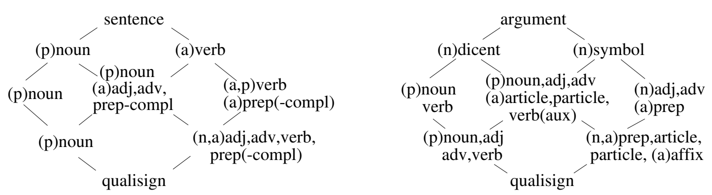

Fig. 3. Classification of syntactic and morphological symbols

Morphological qualisigns are defined as follows: A=noun, verb, adjective, adverb, coordinator; B=preposition, article, particle, affix. Formally, we also define 0='no input' and 1='separation'. The latter can represent the 'space' symbol which is considered an A *and* B sign, incompatible for any interaction, except for accumulation (with any sign triggering the interaction). Hence, the space symbol can 'sweep' out the morphologically finished signs. Dot symbols are treated similarly as in the case of syntax.

4.8 Formal definition

We specify a recogniser for our model as a pushdown automaton. Formally, the automaton is defined as a 8-tuple $M = (K, C, I, \Gamma, \rho, s, F, \Delta)$ where K is a finite set of states, C is a finite set of sign classes, I is a finite set of input symbols, Γ is a finite set of stack symbols, ρ is a function defining the relational need of input symbols, s is the initial state, $F \subseteq K$ is a set of final states, Δ is a transition relation consisting of is a finite set of transition rules.

A transition rule is a mapping $(p, u, \beta) \to (q, \gamma)$ where $p, q \in K$ are, respectively, the states before and after the transition, $u \in I^*$ are the symbols to be read, and $\beta, \gamma \in \Gamma^*$ are the symbols to be popped and pushed. There is a single transition rule for reading the next input symbol onto the stack, all other rules are 'internal' transition rules which operate only on the stack. Such rules will be used for the specification of the syntactic sign interactions as follows.

Because such rules always refer to the set C, in the definitions we will replace p and q by C, but we will specify only those classes of C (by listing them) which are involved in the transition. A further simplification is possible due to the observation that the stack is used in 'frames'. Such a frame (β) has a finite number of locations for each element of C. The first two of these locations will be called the next and the existing sign of a class, all others are temporary locations. A temporary location can be necessary, for example, for the evaluation of a condition. The specification of such computations may require a number of internal rules which we alternatively define as a (logical) expression. Accordingly, the temporary locations are removed from the transition rules.

The value of a next and existing sign can be an input symbol (only in the qualisign class), or a relational need $r = (t, y)$ where $t \subseteq C$ and y is a finite set (of syntactic properties). The logical type of r is defined by the function τ as follows: $\tau((t, y)) = A$ if $t = \emptyset$, and B, otherwise.

Nondeterminism is assumed to be implemented by backtracking ([1]). In the definition of Δ we will allow a reference to the current value of the evaluation mode, forward('f') or backward('b'), via the function *mode*. Finally, we will make use of a graph $G = (C, E)$ where $E = E_d \cup E_h$, and $E_d, E_h \subseteq C \times C$ are, respectively, the set of directed edges and horizontal lines (undirected edges) as shown in fig. 1 (a formal definition is omitted). The successors and neighbours of a class are defined, respectively, by the functions $succ(c) = \{c' | (c, c') \in E_d\}$ and $adj(c) = \{c' | (c, c') \in E_h\}$. An element of $succ(c)$ and $adj(c)$ is denoted, respectively, as c^s and c_a.

In a transition rule, state and stack will be merged. The stack is read non-destructively ($\gamma = \beta' \beta$ where β' and β are, respectively, the next and the current frame). Because input symbols (u) are only related to the qualisign class of C, the treatment of these symbols is specified by a separate rule (as mentioned earlier) in which u is defined to appear in the next sign location of the qualisign class. In sum, we will refer to triples (c, s, s') where c is a class, and s and s' are, respectively, its next and existing signs (any of s and s' may not be specified, in which case, it is denoted by a "_" symbol). The triples on the left- and right-hand side of a rule, respectively, refer to the current and next frame on the top of the stack. In the rules below (and also in the examples) the names of the sign classes are abbreviated; ε denotes the empty value.

input :
$(qual, u, \varepsilon) \to \quad (qual, \varepsilon, \rho(u)).$
$(qual, \varepsilon, r), (icon, \varepsilon, \_) \to \quad (qual, \varepsilon, \varepsilon), (icon, r, \_) \quad \text{IF } \tau(r) = A.$
$(qual, \varepsilon, r), (sins, \varepsilon, \_) \to \quad (qual, \varepsilon, \varepsilon), (sins, r, \_) \quad \text{IF } \tau(r) = B.$

The internal rules will be given by rule schemes for the class variable X ($X \in C\backslash qualisign$). The class of a symbol generated by binding is denoted as X^b. Because of space, the specification of degeneration is omitted and X^b is restricted to X^s (in general, $X^b \in \{X, X_a, X^s\}$). In virtue of the special conditions required by the index class, the triple corresponding to the legisign class is explicitly defined in some of the rule schemes. We will make use of the functions *cmpacc* and *cmpbind* which, respectively, yield true if their arguments can syntactically accumulate and bind in the class specified. Furthermore, we will refer to the functions *acc*, *coerce* and *bind* which, respectively, determine the relational need of the symbols yielded by accumulation, coercion and binding. The function *condix* checks if the special conditions of the index class hold.

accumulation :

$$(X, r, r') \rightarrow (X, \varepsilon, acc(X, r, r')) \quad \text{IF } cmpacc(X, r, r').$$

coercion$_1$:

$$(X, r, r'), (X_a, \varepsilon, \varepsilon), (X^s, \varepsilon, \_), (legi, \_, \lambda) \rightarrow (X, \varepsilon, r), (X^s, r^c, \_)$$
$$\text{IF } \{p, X\} \not\subseteq t' \ \wedge \ \neg cmpacc(X, r, r') \ \wedge \ condix(X^s, r^c, \lambda)$$
$$\text{WHERE } r^c = coerce(X, r', X^s), \ r' = (t', y').$$

coercion$_2$:

$$(X, \varepsilon, r'), (X_a, r_a, \varepsilon), (X^s, \varepsilon, \_), (legi, \_, \lambda) \rightarrow (X, \varepsilon, \varepsilon), (X_a, \varepsilon, r_a), (X^s, r^c, \_)$$
$$\text{IF } \{p, X\} \not\subseteq t' \ \wedge \ \neg cmpbind(X, r', X_a, r'_a) \ \wedge \ condix(X^s, r^c, \lambda)$$
$$\text{WHERE } r^c = coerce(X, r', X^s), \ r' = (t', y').$$

binding :

$$(X, r, r'), (X_a, \varepsilon, r'_a), (X^b, \varepsilon, \_), (legi, \_, \lambda) \rightarrow (X, \varepsilon, r), (X_a, \varepsilon, \varepsilon), (X^b, r^b, \_)$$
$$\text{IF } (p \in t' \wedge X_a \in t'_a) \ \wedge \ cmpbind(X, r', X_a, r'_a) \ \wedge \ condix(X^b, r^b, \lambda)$$
$$\text{WHERE } r^b = bind(X^b, r', r'_a), \ r' = (t', y'), \ r'_a = (t'_a, y'_a).$$

$condix(X, (t, y), (t_l, y_l))$:

$$X = indx \ \wedge \ ((p \in t \ \wedge \ (mode = \text{'b'} \ \vee \ legi \in t_l)) \ \vee \ X \in t) \ \vee \ \text{TRUE} .$$

A parser can be defined by using temporary locations. Such a location may contain an input symbol, or, one or two constants which are used as pointers to locations of the previous frame.

4.9 Example

In this section we show the analysis of the sentence Mary eats pizza with a fork. The sign matrix will be represented in a tabular form. A column corresponds to a sign class, and a row to the recognition of an input symbol. The treatment of space and dot symbols is omitted. The final step of sign recognition (the generation of an argument sign) is not displayed. The accumulation of symbols is denoted by a "/" sign. Rule names are abbreviated as follows: input(i), accumulation(a), coerce$_1$(c$_1$), coerce$_2$(c$_2$) and binding(b); degeneration is indicated by a subscript 'd'.

The morphological analysis is depicted in table 1. In step 8, the index sign 'a' binds to the rheme 'fork' and complements it with the property of 'definiteness'. The morphological sign 'a fork' is represented degenerately as an index which, then, complements the legisign 'with'. Their interaction yields the sign 'with a fork' (a prep-compl) having adjective- or adverb-like syntactic properties.

Table 1. Morphological analysis

nr.	qual	icon	sins	rhme	indx	legi	dcnt	symb	*rule*
0	Mary(M)								i
1	eat(e)	M							i, c_1
2	-s	e		M					i, c_1, c_1
3	pizza(p)	e	-s	M					i, c_1
4	with(w)	p		e-s			M		i, c_1, b, c_1
5	a(a)		w	p			e-s		i, c_1, c_1, c_1
6	fork(f)		a	p		w	e-s		i, c_1, c_1
7		f			a	w	p		c_1, c_1, c_1, c_1
8				f	a	w	p		b_d
9					a-f	w	p		b, c_1
10								w-a-f	c_1

The final output is (Mary)(eats)(pizza)(with a fork) where an item enclosed in parentheses denotes an argument sign generated.

For the syntactic analysis, the relational needs are defined as follows: 'eats'= {legisign,symbol}, 'with a fork'={index}. The parses are displayed in table 2 and table 3 (in the latter only the steps deviating from the first analysis are given). Notice that, due to the sequentiality of the model, the reference of a sign is fixed when the sign of its object appears.

Table 2. Syntactic analysis (alternative #1)

nr.	qual	icon	sins	rhme	indx	legi	dcnt	symb	*rule*
0	Mary(M)								i
1	eats(e)	M							i, c_1
2	pizza(p)		e	M					i, c_1, c_2
3	with a fork(w_{af})	p		M		e			i, c_1, c_2
4			w_{af}	M	p	e			c_1, b_d
5				M	w_{af}	e-p			c_1
6					w_{af}	e-p	M		b
7							M	e-p-w_{af}	b

Because a prep(-compl) can have an *a*-need in the legisign class (but only if it is accumulated with a verb), there is a third analysis possible, in which, eats and with a fork combine to a single legisign representing a common meaning, e.g. 'with-a-fork-eating'. This analysis ('with a fork'={legisign}) is shown in table 4.

Table 3. Syntactic analysis (alternative #2)

nr.	qual	icon	sins	rhme	indx	legi	dcnt	symb	*rule*
3'	with a fork(w_{af})	p		M		e			i, c_2, c_1
4'			w_{af}	p		e	M		c_1
5'				p	w_{af}	e	M		b_d
6'					p-w_{af}	e	M		b
7'							M	e-p-w_{af}	b

Table 4. Syntactic analysis (alternative #3)

nr.	qual	icon	sins	rhme	indx	legi	dcnt	symb	*rule*
3"	with a fork(w_{af})	p		M		e			i, c_2
4"			w_{af}	M	p	e			c_1, a
5"				M	p	e/w_{af}			c_1
6"					p	e/w_{af}	M		b
7"							M	e/w_{af}-p	b

Further research

The simplicity of the algorithm introduced in the paper allows the generation of efficient parsers. In virtue of the simple 'structure' of its syntactic (and logical) analyses, such parsers can be particularly suitable for text summarisation and for handling incomplete sentences.

References

1. Aho, A.V., Ullman, J.D.: *Parsing*, volume 1 of *The Theory of Parsing, Translation and Compiling.* Prentice-Hall, (1972)
2. Farkas,J.I., Sarbo, J.J.: A Peircean framework of syntactic structure. In *7th International Conference on Conceptual Structures (ICCS'99)*, Lecture Notes in Artificial Intelligence, Vol. 1640. Springer-Verlag, (1999) 112–126
3. Farkas, J.I., Sarbo, J.J.: A Logical Ontology. In: G. Stumme (ed.): *Working with Conceptual Structures: Contributions to ICCS2000.* Shaker Verlag. (2000) 138–151
4. Harnad, S.: *Categorical perception: the groundwork of cognition.* Cambridge University Press, Cambridge (1987)
5. Peirce, C.S.: *Collected Papers of Charles Sanders Peirce.* Harvard University Press, Cambridge (1931)
6. R. Quirk, S. Greenbaum, G. Leech, and J. Svartvik. *A Comprehensive Grammar of the English Language.* Longman, London and New York, 1985.
7. V. Tejera. *Semiotics from Peirce to Barthes.* E.J. Brill, Leiden, 1988.