A linearly complex model for knowledge representation

Janos Sarbo and József Farkas

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

Abstract. We present two results which complete our Peircean semiotic model of signs introduced in [10]. The first result is concerned with the potential of our model for the representation of knowledge. The second one consists of a formal proof about the model's complexity.

1 Introduction

In this paper we argue that Peirce's pragmatic philosophy ([8]) can be effectively used for knowledge representation. Because knowledge emerges from a cognitive process, a Peircean approach must respect the properties of human cognition. In as much as knowledge arises via the mediation of signs, such a model must be based on a semiotic theory.

Earlier we introduced such a cognitively based semiotic model for Boolean logic ([3]), language, in particular, for syntax and morphology ([10]), syllogistic logic ([11]), and most recently, also for semantics ([4]). The purpose of this paper is twofold. First, we show by example that our model can be adequately used for the *specification* of any problem which appears as a phenomenon (i.e. which is observable). Second, we make an attempt to formally prove that the complexity of such a specification can be *linear*.

An example for a problem which is a phenomenon, is natural language. We experience language, hence it must be a phenomenon, and we are capable of recognizing its meaningful units, therefore it must appear as a problem. This also illustrates that, in our conception, a *problem* is equivalent to the cognitive process of perception of a phenomenon.

Language is strongly related to knowledge. Some philosophers have even suggested that "it is *learning language* that makes a mind systematic" ([5]). Perhaps we do not falsely interpret their idea by assuming that the representation of language can be isomorphic to the representation of knowledge in general. If, as we argue, language can be linearly complex, the conclusion may be drawn that knowledge representation might have the same complexity, as well.

In complexity theory linearity is equivalent to real-time complexity ([7]). Because signs are inherently related to a contrast which, according to our model, can be observed as a change in the 'real' world, the results of this paper imply that the well-known 'frame problem' of cognition ([9]), which is traditionally considered exponential, might have a simple solution as well.

Peirce's semiotic theory entails that, ontologically, everything must be a sign. Apart from the possible implications, also including the philosophical one that we, human, must be a sign as well, the above conclusion has also practical consequences. If, as Peirce maintains, there can be distinguished in the 'real' world between nine kinds of signs ([2]), then such signs or aspects must be present in *any* phenomenon also including cognition which is a phenomenon as well. This means that any observable problem could be *specified* in terms of Peirce's signs. The question is, precisely *how* those signs are called in a given problem.

Peirce's classification of signs is depicted in fig. 1 (the meaning of the horizontal lines and the labels on the right-hand side will be explained later). The different types of signs are defined by means of a set of aspects. A brief characterization of such aspects is given in fig. 2.

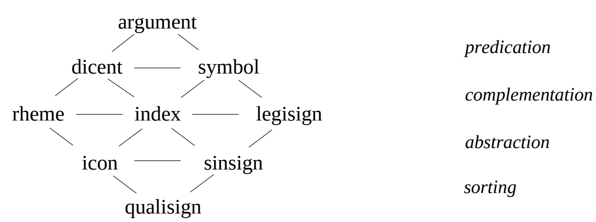

Fig. 1. Peirce's classification of signs

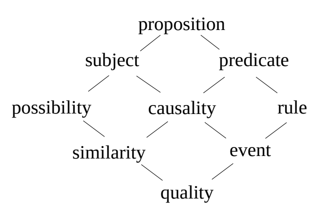

Fig. 2. The aspects of Peirce's signs

2 The cognitive model of signs

In our model ([3], [10]) we assume that signs emerge from the sensation of the physical input. Such stimuli are sampled by the brain in percepts typically due to a change in the input ([6]). A percept may also contain qualities from the memory (how sensory and memory qualities can be merged is described in [4]).

Because percepts arise by virtue of a change in the input, subsequent percepts must be different from one another. By comparing the previous percept with the current one the brain can distinguish between two sorts of input qualities: one, which was there and remained there, which we will call a *continuant*; and another, which was not there, but is there now (or vice versa), 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. By means of *selective attention*, the qualities of these collections are further classified as *observed* and *complementary*. We will collectively refer to the perceived qualities as the *input*. We will assume that these qualities *are* the elementary signs we observe: qualities which are signs. Peirce called them a *qualisign* ([8]2.244).

2.1 The generation of complex signs

In this section we briefly summarize the stages of sign generation ([3]). When it is clear from the context, we will uniformly refer to a sign class and an element of it. For example, a reference to an icon may denote an icon sign, or the icon class itself. By virtue of the isomorphism between the classification of the different sign phenomena ([10]), we will denote a sign by the corresponding Boolean expression. The classification of Boolean logical signs is displayed in fig. 3.

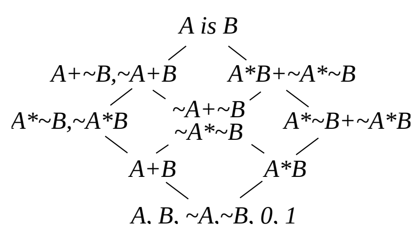

Fig. 3. The classification of Boolean signs

The source of sign generation is the set of qualisigns. Qualisigns are special signs for which we have no denotation except on the level of description. We will refer to them by the logical expressions A, B, $\neg A$, $\neg B$ (0 and 1, respectively, represent the absence and presence of input). The process of sign generation is initialized by a *sorting* representation of the input qualisigns (which are qualities). This yields two different views of the input: the listing of the qualities as the parts of the observation, and the simultaneous occurrence of those parts, as an event. These signs are, respectively, the icon $(A+B)$ and the sinsign[1] $(A*B)$. The remaining signs are generated via *sign interactions* between adjacent signs, in subsequent stages. In fig. 1, the adjacent signs are connected by horizontal lines; the stages are indicated by labels on the right-hand side.

[1] "The syllable *sin* is taken as meaning 'being only once', as in *single*" ([8]2.245).

In the first stage (*abstraction*), the interaction of the icon and sinsign is interpreted as a rheme[2], index and legisign[3]. The rheme signs ($A{*}\neg B$, $\neg A{*}B$) refer to the abstractions of the individually observed input collections. The legisign ($A{*}\neg B{+}\neg A{*}B$) signifies the compatibility of these abstractions. The index signs ($\neg A{+}\neg B$, $\neg A{*}\neg B$) represent the complementary qualities as a phenomenon (context) in two different ways. Via the DeMorgan rules (not displayed), the index signifies the relation between the observed and complementary phenomena.

The second stage (*complementation*) is concerned with the representation of the actual subject and predicate of the observation. These signs are generated from the abstract concepts of the rheme and legisign via complementation, by means of the index. The resulting signs are the dicent[4] ($A{+}\neg B$, $\neg A{+}B$) and the symbol[5] ($A{*}B{+}\neg A{*}\neg B$).

Finally, in the third stage (*predication*), the subject and predicate of the observation are merged and their interaction represented as a proposition, which is an argument sign (A *is* B).

2.2 A Peircean model of language

Language consists of signs which are symbols. A language phenomenon, for example, a sentence appears as a sequence of words. In [10] we argue that a Peircean model of syntactic signs can be derived from a *sequential* version of the sign generation process described above. The interaction between syntactic signs is called a *binding*. By virtue of the sequential nature of syntactic sign generation, there may exist degenerate cases of a binding which are *accumulation* and *coercion*. In an accumulation, an existing sign is combined with another sign of the same class. In a coercion, 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.

Syntactic sign interactions are characterized by the *relational need* of the interacting symbols (which are called the *constituents* of the interaction; in the case of a coercion we refer by the constituent to the sign triggering the interaction. The relational need of a sign is a finite set of qualities; such a set is either lexically defined, or computed from the relational needs of the constituent symbols of a binding. We distinguish between three types of syntactic relational qualities: *active*(a), *passive*(p) and *neutral*(n). It is assumed that a binding resolves, and an accumulation merges a pair of relational qualities, whereas a coercion inherits the need of the sign coerced.

For example, a verb can be characterized by the number and type of its complements, a noun by the properties that allow for it to become a verb complement. The interaction of a verb and a noun can be represented as a syntactic

[2] Greek for 'word'.
[3] Latin for 'law' (gen.).
[4] Latin for 'speak' or 'say'.
[5] Latin for 'put together'.

sign, the relational need of which is defined as a combination of the relational needs of the verb and the noun from which the relational qualities which are satisfied, are removed.

Finally we mention that our model of language allows for a contiguous segment of input symbols to be analyzed recursively as a *nested* sign ([10]). When such a segment is recognized as a *single* sign, its meaning relative to the input as a whole is represented, degenerately.

3 A Peircean specification of concepts

Having recapitulated our theory, we are now ready to illustrate its application for the specification of a sample problem which is the phenomenon of selling a bike. We will argue that the meaningful units of this problem can be found by recognizing the different aspects that can be distinguished in a 'real' world phenomenon. In the end, we will have a set of signs which will constitute our specification of the given problem. Because such meaningful units, or concepts, arise from qualities that are perceived, our Peircean approach to specification can be said a first step towards a theory of *real concept analysis*.

Selling a bike is a process. We assume that in the beginning of this process the purchaser has a general idea (knowledge) about the kind of bike (s)he wishes to buy. This idea, then, is confronted with the 'real' assortment of bicycles that can be bought. As a result, the purchaser comes to a decision and buys some product. We will assume that our sample process takes place in a bike shop. That such a context contains a diversity of information, is illustrated by fig. 4.

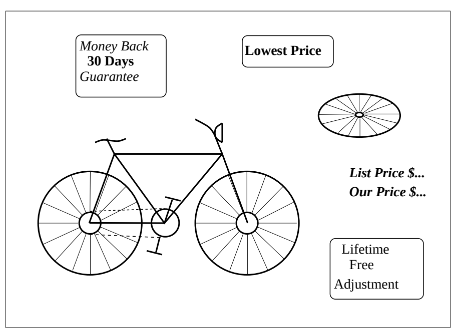

Fig. 4. Illustration for the phenomenon of selling a bike

Earlier we mentioned that a percept may contain memory qualities (which are thought signs). The hidden agenda of this paper is an attempt to demonstrate that our cognitive approach to signs equally applies to the qualities of the physical stimulus, as well as, to those of the abstract concepts of the mind. We will tacitly assume the existence of a uniform representation for signs, also including memory signs.

Qualisign We may consider a phenomenon a 'story' which we tell by means of signs. Like any story, also the one of selling a bike is based on primary observations which are the qualisigns. What is experienced in the given phenomenon and recognized as a qualisign, can be defined as follows.

A: product; B: differences; $\neg A$: ownership; $\neg B$: conditions.

A refers to an observed bicycle which is the possible subject of the actual sale, for example, a red bicycle with green mudguard. B denotes the properties, or facts which refer to the differences between the imagined and real product. For example, if the purchaser wanted to buy a red bicycle with white mudguard, then there is a difference which lies in the color of the mudguard. In sum, A and B refer to those sets of qualities which respectively identify the possible 'thing' and 'property' that the purchaser might want to buy. These qualities may not determine a product presented in the showroom. Qualisigns are possibles which are a first approximation of the meaning of the observed phenomenon.

The complementary qualisigns, $\neg A$ and $\neg B$, refer to those 'things' and 'properties' or 'facts' which are there, but which are not in the focus of attention. Such qualities are typically due to memory knowledge. Such knowledge may include written specifications, handbooks etc. as well. In our example we will assume that $\neg A$ denotes the form of product ownership, e.g. a sales contract, or leasing; $\neg B$ refers to the judicial and organizational basis of a sale, for example, guarantee and service, or the conditions for recompensation in the case of a damage. Formally, we also define 1 and 0, respectively denoting the case of an effective sale and no selling.

Icon and sinsign In the first towards the recognition of the given phenomenon as a proposition the input qualities are sorted yielding an icon and sinsign representation of the input.

$A{+}B$: that, what the purchaser is focusing on, given as a listing of the observed product ('thing') and the observed discriminating attributes ('properties').

$A{*}B$: the signification of the simultaneous observation of the product and the attributes as an event that happens 'now'.

Rheme, index and legisign By virtue of selective attention, any perception may refer to two collections of qualities which are interrelated: one, which is selected by our attention, and another, that we are not focusing on. From this it follows that any observation must be embedded in the context of other qualities.

$\neg A{+}\neg B$, $\neg A{*}\neg B$: the Shäffer and Peirce functions, respectively, refer to the icon- and sinsign-like representation of the qualities of the 'background' of the observation. Such a context consists of complementary 'things' (ownership) and 'facts' (conditions) that are beyond the purchaser's attention.

As indicated above, the index signifies the complementary qualities analogously to those of the observed phenomenon. From this it follows that the context of the observation must be a phenomenon as well. The two phenomena, observed and complementary, are interrelated.

The observed product can be related to complementary properties, and the other way round, the observed differences can be used to identify a product which is complementary with respect to the actual observation. Such completion is precisely the meaning of the rheme signs.

$A{*}\neg B$: the observed product completed by features, for example, following the manufacturer's specification. Such features may include technical data, price lists etc. Because such features are not part of the actual observation, the resulting sign can only signify an abstract concept of a bicycle, or briefly, an *abstract product*.

$\neg A{*}B$: the concept of *abstract differences* which is defined analogously. An example for such a sign is the specification of the formal conditions for replacing the green mudguard with a white one, also including the possible effects of such an adjustment on the price, guarantee conditions etc.

Because, in the end, the purchaser will buy a bicycle which is a single product, the different sorts of abstractions of the input cannot be independent from each other. Their relatedness is the meaning of the legisign which, by representing a listing of such abstract views as a sign, signifies their *compatibility*. By virtue of the abstract meaning of such views, the compatibility indicated by the legisign has the aspect of a rule.

$A{*}\neg B{+}\neg A{*}B$: that the abstraction of the observed product ($A{*}\neg B$) and differences ($\neg A{*}B$) are compatible, is the meaning of the concept of an abstract sale, or the notion of *bargaining*.

The above interpretation of the legisign perfectly illustrates what has been suggested in the beginning of this paper. Peirce's signs *are there* in any problem, the task of specification is to find out *how* such signs are denoted. Sometimes we may be familiar with the name of such a sign, and sometimes there may be no proper denotation available. In such a case, we may define one by ourselves and thereby extend language. Because, in our approach, any sign is a representation of the input qualities, such a denotation will always be meaningful.

Dicent and symbol The two abstractions of the observed qualities, and the relatedness mediated by the index allow for a further approximation of the given phenomenon: that, what the purchaser observed as a possible bicycle and the conditions that complete it to a 'real' product define the subject of the actual purchasing.

$A{+}\neg B$: the observed bicycle and the formal conditions of the sale are logically related to each another (cf. implication).

$\neg A{+}B$: the relation of the actual differences and the corresponding possible product which is defined analogously.

The dicent sign amounts to the two views of the subject of the actual selling event. These views are *different* interpretations of the *same* object, however the dicent is only emphasizing the last aspect. That the two views are related and their contrast defines a property is the meaning of the symbol sign.

> $A{*}B{+}\neg A{*}\neg B$: the simultaneous existence of the two views of the subject characterizes the selling of a bike as a 'real' event, as a property, or predicate (notice that $A{*}B{+}\neg A{*}\neg B$ is short for $(A{+}\neg B){*}(\neg A{+}B)$).

Argument What is being sold, is the combination of A and B embedded in the context of the complementary phenomenon signified by $\neg A$ and $\neg B$.

> $A(\neg A)$ *is* $B(\neg B)$: "The selling of (*such and such*) a bike" is a proposition of the observed phenomenon.

3.1 Remarks

Earlier we mentioned that, in our model, signs are generated via interactions. We argue that from the algorithmic content of sign generation an operational specification can be derived. In this section we briefly summarize the properties of such a specification.

A percept consists of a finite number of sensory and memory qualities. Hence, qualisigns can be represented by a finite set. Such a representation trivially applies to lexically defined qualities like the syntactic relational need of a symbol. By virtue of the independent nature of qualisigns, there may be introduced two types for the continuant and occurrent qualities, and two subtypes for the observed and complementary ones. From the logical meaning of the icon we may conclude that there must exist two data structures containing references (e.g. pointers) to the different types of qualisigns, as well as, corresponding access algorithms. From the meaning of the sinsign we may conclude that the different types of observed qualities may appear in any order. This implies the potential need for a parsing algorithm and a suitable representation of the parsed data.

From the linking meaning of the index a conversion algorithm between the different types of qualities can be derived. Such a conversion operation may be necessary for computing the abstract data of the rheme, and also, for the implemention of the type checking (cf. compatibility) involved in the meaning of the legisign. From the operational point of view, the dicent sign can be represented by a data structure generated from the rheme and the complementary signs via conversion by means of the index. From the interpretation of the symbol sign, the definition of a procedure specifying the steps of the selling process can be derived. Finally, the operational meaning of the argument sign can be specified as a program applying the procedure of the symbol to the data structure of the dicent.

This completes the illustration of the use of our Peircean semiotic approach to the specification of problems which appear as phenomena.

4 An analysis of the model's complexity

In this section we return to our model of syntactic signs. We define a formal specification for our language model introduced in [10] and discuss its complexity. We specify a recognizer for our model of signs as a pushdown automaton.

Formally, the automaton is defined as an 8-tuple $M = (K, C, I, \mathit{\Gamma}, \rho, s_0, F, \mathit{\Delta})$ where $K = \{s_0, s_1\}$ is a finite set of states, C is a finite set of sign classes, I is a finite set of input symbols, $\mathit{\Gamma}$ is a finite set of stack symbols, $\rho \in I \rightarrow \mathit{\Gamma}$ is a function defining the relational need of input symbols, s_0 is the initial state, $F \subseteq K$ is a set of final states, $\mathit{\Delta}$ is a transition relation consisting of is a finite set of transition rules.

A transition rule is a mapping $(p, u, \beta) \rightarrow (q, \gamma)$ where $p, q \in K$ are, respectively, the states before and after the transition, $u \in I^*$ are the symbols to be read, and $\beta, \gamma \in \mathit{\Gamma}^*$ are the symbols to be popped and pushed.

We will assume that the stack is divided into *frames*. A frame contains a storage area for each sign class, consisting of a class name, a location for the *next* and the *existing sign* of the class, and a constant number of locations for *temporary* values (see fig. 5).

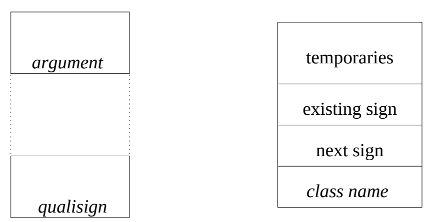

Fig. 5. Stack frame and storage area

The start rule and the rule handling the input of symbols are specified as follows (ε denotes the empty string, $\epsilon \in \mathit{\Gamma}$ stands for the empty value):

start : $(s_0, \varepsilon, \varepsilon) \rightarrow \quad (s_1, \iota_\epsilon)$
read : $(s_1, u, \iota_\epsilon) \rightarrow \quad (s_1, \iota_{\rho(u)})$

where ι_x denotes a frame in which the existing sign location of the qualisign class contains the value x (the next sign location of this class is not used). The other locations of ι_ϵ and $\iota_{\rho(u)}$ have an identical value in the two frames.

All other rules are 'internal' transition rules which only operate on the stack (ϕ and ϕ' denote frames):

transition : $(s_1, \varepsilon, \phi) \rightarrow \quad (s_1, \phi'\phi)$

We will simplify the specification of a transition rule by only defining ϕ and ϕ', and only specifying those locations of a frame which are involved in the transition (those not involved are assumed to have an identical value in ϕ and ϕ'). A further simplification is achieved by representing a frame as a *set* of storage areas (instead of a list).

Temporary locations 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 specification of temporary locations will be omitted. The value of the next and existing sign location of a class is a relational need which is a constant (cf. sect. 2.2).

Nondeterminism is implemented by backtracking ([1]). In a transition rule we allow a reference to the actual evaluation mode, which can be forward('f') or backward('b'), via the function *mode*. We will make use of a graph $G=(C,E)$ where $E=E_d \cup E_h$, $E_d, E_h \subseteq C \times C$. E_d and E_h are, respectively, the set of directed edges and horizontal lines (undirected edges) as shown in fig. 6 (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.

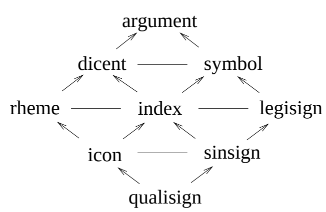

Fig. 6. Transition graph

In sum, in a transition rule we will refer to a *set* of triples (set brackets are omitted). An element of such a set is given as a triple (c, s, s') where c is a class, and s and s' are, respectively, the next and existing signs of c (any of s and s' may not be specified, in which case they are denoted by a "_" symbol). The triples on the left- and right-hand side of a rule, respectively, refer to the current(ϕ) and next frame(ϕ') located on the top of the stack (notice that a condition always refers to the current frame). The logical type of the next sign (r) of the qualisign class, $lt(r)$, is A if r has no *a*-need in any class; and B, otherwise. The names of the sign classes are abbreviated to a four letter name.

sorting

$(qual, \_, r), (icon, \epsilon, \_) \rightarrow \quad (qual, \_, \epsilon), (icon, r, \_) \quad \text{IF } lt(r) = A.$
$(qual, \_, r), (sins, \epsilon, \_) \rightarrow \quad (qual, \_, \epsilon), (sins, r, \_) \quad \text{IF } lt(r) = B.$

The remaining internal transitions are given by rule schemes for the class variable X ($X \in C \backslash \{qual\}$). In virtue of the special conditions required by the index class ([10]), the triple corresponding to the legisign class is explicitly defined in some of the rule schemes. These conditions require that a symbol can become an index having a *p*-need, either if any other analysis of that symbol eventually fails, or, if there already exists an *a*-need due to a legisign symbol.

We make use of the functions *cmpacc* and *cmpbnd* which, respectively, yield true if their arguments can syntactically accumulate and bind in the class specified. We also apply the functions *ntrl*, *pssv* and *actv* which, respectively, succeed if their argument has a *n*-, *p*- and an *a*-need in the class given. Additionally we make use of the functions *acc*, *coerce* and *bind* which, respectively, determine the relational need of the symbols yielded by accumulation, coercion and binding. The function *cndix* checks if the special conditions of the index class hold. The degenerate variants of the rule 'binding' are omitted (in such a case, the result of binding emerges in the class of one of the constituents). The sentence as a sign arises in the next sign location of the argument class. The rule schemes are illustrated in fig. 7-9.

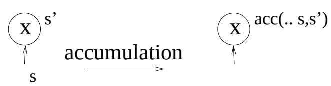

Fig. 7. Accumulation

accumulation
$(X, r, r') \rightarrow\ (X, \epsilon, acc(X, r, r'))$ IF $cmpacc(X, r, r')$.

coercion$_1$
$(X, r, r'), (X_a, \epsilon, \epsilon), (X^s, \epsilon, \_), (legi, \_, r_l) \rightarrow\ (X, \epsilon, r), (X^s, r^c, \_)$
IF $ntrl(X, r') \wedge \neg cmpacc(X, r, r') \wedge cndix(X^s, r^c, r_l)$
WHERE $r^c = coerce(X, r', X^s)$.

coercion$_2$
$(X, \epsilon, r'), (X_a, r_a, \epsilon), (X^s, \epsilon, \_), (legi, \_, r_l) \rightarrow\ (X, \epsilon, \epsilon), (X_a, \epsilon, r_a), (X^s, r^c, \_)$
IF $ntrl(X, r') \wedge cndix(X^s, r^c, r_l)$
WHERE $r^c = coerce(X, r', X^s)$.

binding
$(X, r, r'), (X_a, \epsilon, r'_a), (X^s, \epsilon, \_), (legi, \_, r_l) \rightarrow\ (X, \epsilon, r), (X_a, \epsilon, \epsilon), (X^s, r^b, \_)$
IF $pssv(X, r') \wedge actv(X_a, r'_a) \wedge cmpbnd(X, r', X_a, r'_a) \wedge cndix(X^s, r^b, r_l)$
WHERE $r^b = bind(X^s, r', r'_a)$.

$cndix(X, r, r_l)$:
$X = indx \wedge (pssv(X, r) \wedge (mode = \text{'b'} \vee actv(legi, r_l)) \vee actv(X, r)) \vee$ TRUE .

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

When a segment of input symbols is to be analyzed recursively, transition may proceed until no rules apply. Then, the current frame is pushed to the stack. Upon return from a recursion, the current frame and the saved one are 'merged' according to the properties of the nested sign.

4.1 Complexity

The directed edges of fig. 6 define a partial ordering on Peirce's classes of signs. Earlier we mentioned that a binding resolves, and an accumulation merges a pair

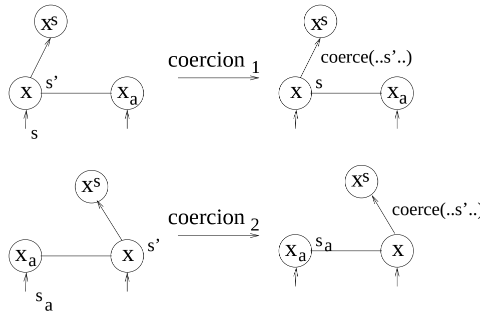

Fig. 8. Coercion

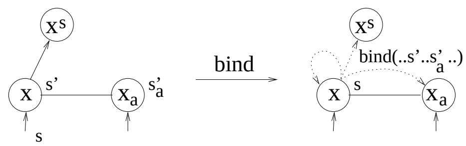

Fig. 9. Binding

of relational qualities, whereas a sign generated by coercion inherits the need of the sign coerced. In sum, there is no increase of the relational qualities in any interaction. The class of a sign yielded by binding and accumulation is not lower, and the one yielded by coercion is definitely higher in the partial ordering, than the class of its constituent(s).

In the conditions of the transition rules we make use operations on sets which are *intersection*, e.g. for testing the compatibility of symbols, and *union*, e.g. for the generation of the sign yielded by binding (we may need copy operations, but which can be implemented by means of a finite number of intersections and unions, as well). The evaluation of a condition may require a constant number of set operations. Because the sets are finite (they cannot exceed the size of the lexicon, which is a constant), the complexity of the conditions is $\mathcal{O}(1)$ in the size of the sets and the number of set operations. In as much as the number of classes as well as the relational need of input symbols are finite, the processing of an input symbol (which terminates when no transition rule applies) requires a constant number of transitions which are $\mathcal{O}(1)$ complex each. Eventually we get that the complexity of our model, if nesting is not allowed, is $\mathcal{O}(n)$ where n is the number of input symbols.

The complexity remains unchanged if additionally we allow nesting. We assume that a syntactic symbol's potential for initiating and terminating a nested segment (which is analyzed recursively) is defined analogously to the symbol's syntactic relational properties. Accordingly, an input symbol may 'start' or 'end' a nested analysis only a *finite* number of times, which is lexically defined.

We assume that the stack frames are linked to each other via a 'previous frame' pointer which is stored in a temporary location. Upon entering a recursion

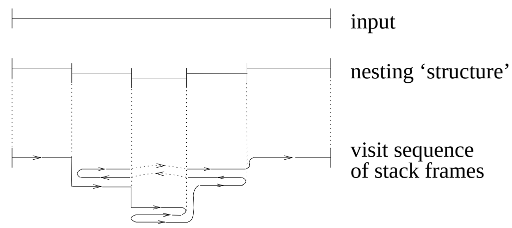

Fig. 10. Sample nested input analysis

the current frame is saved. Upon return, there will be a *single* (nested) sign in the topmost frame of the stack. Let k denote the number of input symbols involved in the recursively analyzed segment. Then, to find and fetch the values of the last saved frame needs at most $\mathcal{O}(k)$ steps, but the frames involved in this process will *not* be visited anymore. This can be solved by adjusting the previous frame pointers of the frames of the recursively analyzed segment. Accordingly, any frame will be visited at most three times (cf. fig. 10) and the complexity of the algorithm will be $3{*}\mathcal{O}(n)$ which is equivalent to $\mathcal{O}(n)$.

5 Conclusion

In the first part of this paper we argue that any problem (which is a phenomenon) can be specified in terms of Peirce's sign. Contrary to the traditional way of specification which, by virtue of its formal character is doomed to be ad hoc, the promise of the Peircean approach is that the nine kinds of signs are always there and we only need to identify them.

In the second part, we prove that the complexity of the Peircean model of language introduced in [10] is linearly complex. We argue that this result applies to other sign phenomena as well. Because the perceived qualities of a phenomenon can always be represented by a finite set, the complexity of sign generation can be linear in general (the sequentiality assumption used in the language model does not affect this result). Because knowledge emerges from the perception of ‘real’ world phenomena, the results of this paper imply that human knowledge can possess a linearly complex representation. This, of course, does not impose any restrictions on the complexity of ‘real’ world phenomena which can be arbitrary. Understanding a problem and knowing all its solutions are different. Knowledge representation is only concerned with the first of these.

References

1. Aho, A.V., Ullman, J.D.: *The Theory of Parsing, Translation and Compiling*, Vol. 1. Prentice-Hall (1972)
2. Farkas, J.I., Sarbo, J.J.: A Peircean framework of syntactic structure. In: Tepfenhart, W., and Cyre, W. (eds.): *ICCS'99*, Lecture Notes in Artificial Intelligence, Vol. 1640. Springer-Verlag, Berlin Heidelberg New York (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. Farkas, J.I., Sarbo, J.J.: A Peircean Ontology of Semantics. In: Priss, U., Corbett, D. (eds.): *ICCS'2002*, Lecture Notes in Artificial Intelligence (this volume), Springer-Verlag, Berlin Heidelberg New York (2002)
5. Fodor, J.A.: *Concepts: Where Cognitive Science Went Wrong*. Clarendon Press, Oxford (1998)
6. Harnad, S.: *Categorical perception: the groundwork of cognition*. Cambridge University Press, Cambridge (1987)
7. Paul, W.: On heads versus tapes. *Theoretical Computer Science* **28** (1984) 1–12
8. Peirce, C.S.: *Collected Papers of Charles Sanders Peirce*. Harvard University Press, Cambridge (1931)
9. Pylyshyn, Z.W.: *The robot's dilemma: the frame problem in artificial intelligence*, *Theoretical issues in cognitive science*, Vol. 4. Norwood, N.J. (1987)
10. Sarbo, J.J., Farkas, J.I.: A Peircean Ontology of Language. In: Delugach, H., Stumme, G. (eds.): Lecture Notes in Artificial Intelligence, Vol. 2120. Springer-Verlag, Berlin Heidelberg New York (2001) 1–14
11. Sarbo, J.J., Hoppenbrouwers, S., Farkas, J.I.: Towards thought as a logical picture of signs. *International Journal of Computing Anticipatory Systems* **8** (2001) 1–16 (in press)