The Version of Record of this manuscript has been published and is available in the *International Journal of General Systems* Vol. 46, No. 5, July 2017. p. 569-585 (http://www.tandfonline.com, http://dx.doi.org/10.1080/03081079.2017.1349655)

Semiotic-Conceptual Analysis: A Proposal

Uta Priss*

Zentrum für erfolgreiches Lehren und Lernen, Ostfalia University of Applied Sciences,Wolfenbüttel, Germany
orcid: 0000-0003-0375-429X

(*received July 15, 2016*)

This paper provides the basic definitions of Semiotic-Conceptual Analysis (SCA), which is a mathematical modelling of signs as elements of a triadic relation. FCA concept lattices are constructed for each of the three sign components. It is demonstrated how core linguistic and semiotic notions (such as synonymy and icon) can be represented with SCA. While the usefulness of SCA has already been demonstrated in a number of applications and several propositions are proven in this paper, there are still many open questions as to what to do next with SCA. Therefore this paper is meant as a proposal and encouragement for further development.

Keywords: semiotics; triadic relations; sign; formal concept analysis; interpretation function

1. Introduction

From a semiotic viewpoint, signs are the core units of information and communication. The American philosopher C.S. Peirce and the Swiss linguist F. de Saussure both independently developed theories of semiotics. Contrary to Saussure who proposed a binary view of signs (consisting of sign vehicle and meaning), Peirce saw signs as triadic (the third aspect being interpretation). Some previous attempts at mathematically formalising Peirce's theory exist (e.g. Marty 1992; Zalamea 2010). The goal of such attempts is usually to stay close to Peirce's original theory, which is challenging since Peirce developed his theory over many years but never formalised it himself. Furthermore, many advances of modern mathematics were not yet available in Peirce's time. Even the understanding of core mathematical concepts such as "function" was different then. In our opinion, Peirce's original idea of modelling signs with triadic relations is intriguing. But instead of a philosophical investigation into the details of Peirce's writings, our goal for this paper is to focus on his core idea (i.e. triadic signs) and to examine how this can be modelled from a modern perspective. Thus Semiotic-Conceptual Analysis (SCA) starts with a very simple mathematical definition of signs as elements of a triadic relation with the condition that two of the components uniquely determine the third one and investigates a theory based on that definition. We are translating existing semiotic and linguistic terminology into SCA, but we are more interested in developing a mathematically coherent theory than in modelling exactly what Peirce or others discussed in the past. Thus this paper should be read as an applied mathematics paper, not as a philosophical paper[1].

*Corresponding author. Email: cf. www.upriss.org.uk
[1]Priss (2015) discusses how SCA relates to the semiotics of C. S. Peirce in more detail.

SCA extends and uses Formal Concept Analysis[2] (FCA). Concept lattices are employed as a means of modelling hierarchies of and implications among signs and their components. Mathematically interesting questions for SCA are whether and how structures among sets of signs relate to lattice theoretical structures. In the same manner as FCA defines "concept", SCA notions of "sign", "denotation", "interpretation" and so on should be understood as mathematically defined structures whose meanings can be different from how they are used in other disciplines. The abstract definition of signs in SCA does not require them to be "sign-like" in the sense of other theories. Thus the question should not be whether "SCA signs" are signs (in the sense of other theories), but whether SCA can successfully model semiotic structures in applications. The semiotic motivation for SCA is elaborated in more detail in the next section. We have used SCA in a few applications (Priss 2015, 2016) which appear promising. But we feel that further studies are needed in order to establish more precisely what SCA can or cannot do for applications. Furthermore, we are hopeful that the mathematical theory will, over time, progress beyond what is presented in this paper. Therefore, this paper should be seen as a proposal and starting point for future SCA research. Core definitions of SCA were already provided by Priss (2015), but they are more elaborated and somewhat enhanced in this paper. For example, Priss (2015) directly incorporated the sign components into the definition of concept lattices. We have now changed this to mappings from sign components into lattices instead of a fixed construction in order to provide more modelling flexibility for applications.

Contrary to our previous publications on SCA (Priss 2015, 2016) which focused on applications in the domain of formal languages and educational analysis, this paper does not focus on applications but develops the mathematical theory further and in more detail than the previous papers. Several definitions, propositions 2 and 3 and the description of the modelling with concept lattices in Section 4 are novel in this paper. Section 2 provides more details about the semiotic background of SCA. Section 3 presents the core definitions. Section 4 elaborates how the semiotic structures among the signs and their sets lead to structures among concept lattices. Section 5 discusses types of tasks that can be performed with SCA. The paper ends with a concluding section.

2. A semiotic motivation for SCA

From a semiotic viewpoint, SCA investigates the relationship between representations and their meanings in a mathematical framework. A sign is an item occurring in any means of communication, voluntary or involuntary, used by humans or non-human agents, in any medium. Signs can be words, pictures, pieces of written or spoken text, webpages, parts of computer programs, gestures, hormonal signals, traffic signs, brain scans that are showing neural activity and so on. A *sign* consists of a *representamen*, a *denotation* and an *interpretation*. A representamen is a sign vehicle or the physical manifestation of a sign. A denotation is its presumed meaning. An interpretation is a (partial) function that maps representamens into denotations.

Many interpretation functions might exist simultaneously. For example, when two people communicate they will each be forming their own interpretations in their minds. Obviously, interpretations in the minds of people cannot be directly analysed. Thus further interpretations are required in order to produce data that can be analysed. Such processes of compiling data can themselves be analysed using further interpretations resulting in a chain of related interpretations. For the purposes of mathematical modelling with SCA,

[2]Because this paper is published in a special issue on FCA, it does not provide an introduction to FCA. Information about FCA can be found, for example, on-line (www.upriss.org.uk/fca/) and in the main FCA textbook by Ganter and Wille (1999).

the sets of signs, representamens, denotations and interpretations and their relationships are considered to be fixed and available as data. How such data is collected and what other signs (or interpretations) might have been used in the data collection process is not a focus of SCA.

At first sight, semiotics might seem similar to formal semantics (which maps representamens into denotations), but the difference is that while semantics tries to provide a model for possible meanings, semiotics attempts to analyse how structures among representamens relate to structures among denotations with respect to structures among interpretations. Thus semiotics is broader than semantics. Furthermore, the types of applications for semantics and semiotics can be different.

Two signs are equal if all of their components are equal. But this definition of equality can be too strong because if the same representamen/denotation pair is used in another context (i.e. with another interpretation) it is a different sign. Therefore instead of equality, it is often more useful to consider equivalence and similarity of signs. Similarity can be modelled via tolerance relations, which are reflexive and symmetric relations. An equivalence relation is a tolerance relation that is also transitive. A tolerance relation can be derived from any similarity measure by defining a threshold. Items which are at least as similar to each other as the threshold are related via the tolerance relation. Items which are less similar to each other are not related via the tolerance relation. Synonymy among words is a typical tolerance relation because, for example, "vehicle" might be a synonym of "car" and "truck", but "car" and "truck" might not be synonyms.

If the meanings of words were taken to be real world objects, then words such as "unicorn" would not have a meaning. Therefore, the meanings of words are not objects but concepts, which have real world, imaginary or abstract objects in their extensions. Thus denotations (as meanings of signs) can be modelled as concepts. Because representamens and interpretations can have implicit relationships, it is advantageous to model them as conceptual structures as well. Similarity can then be investigated, not just with respect to individual data items but with respect to concepts. One might expect that similar representamen concepts are mapped onto similar denotation concepts by similar interpretation concepts. For example, traffic signs use certain shapes and colours to differentiate between warning (red triangle), regulatory (red circle) and informative (blue square) signs. Ideally the concepts for the representamens (built using red, blue, triangle, circle, square) should be mapped onto denotational concepts (warning, regulation and information) in a consistent manner. Although there are international standards for traffic signs, there are some differences between countries and some inconsistencies in place. For example, the signs for one-way streets in Germany are blue squares although they are not informative but regulatory. SCA provides a means for investigating this. The different usages of signs in different countries can be represented as different interpretations where signs that are not used in a particular country are mapped onto a null concept.

Goguen's (1999) theory of algebraic semiotics should be mentioned here. Translated into SCA terminology, Goguen describes (partial) morphisms between representamens based on their syntax, constituents, types and axioms. Such morphisms are then used to compare representamens, for example with respect to their similarity and "quality". While we consider Goguen's approach useful for specific types of representamens, we believe that it mostly works for representamens that are structurally reasonably similar. For example, it would be feasible to define morphisms among different vector graphics storage formats, but it might not be possible to define syntax-based morphisms between a vector graphics and a raster graphics representation of the same image because their construction mechanisms are very different (using vectors versus pixels). In contrast to the difficulty of constructing such morphisms, a human or a neural network program could easily conclude that two representamens "look" similar without having to explain why that is the case in terms of

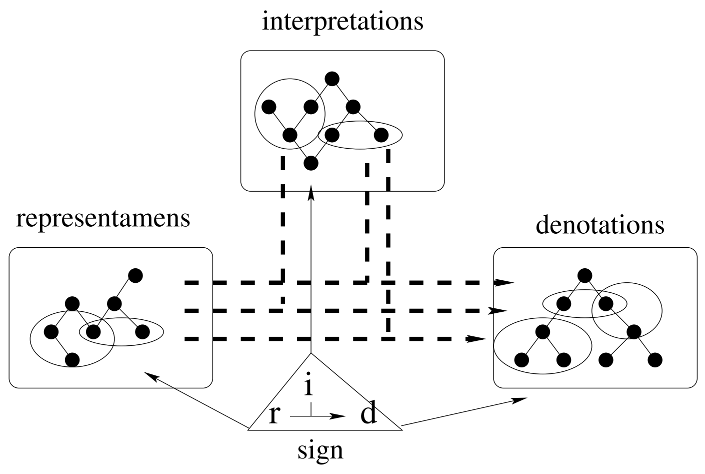

Figure 1. Overview of SCA

vectors and pixels. Thus using concept lattices and tolerance relations is more general than exactly specifying syntactic representations and their morphisms.

Fig. 1 provides an overview of SCA. As mentioned above, signs have three components: an interpretation i, a representamen r and a denotation d. Each of the components of a sign is mapped into a separate conceptual structure. For example, the representamens of a set of signs are mapped into a conceptual representamen structure, which is built from representamens collected from other signs or from background knowledge. A core question is then what kinds of relationships can be observed between the three conceptual structures. Fig. 1 will be more elaborated via an example in Fig. 3 below. The goal for SCA is to provide a framework in which sign relationships are mathematically described, analysed and compared. Some of SCA's core semiotic notions do not depend on which formalisation for conceptual structures is being used. In this paper we are using FCA as a mathematical formalisation of conceptual hierarchies. Other conceptual structures (such as conceptual graphs, formal ontologies, description logics or graph-based representations) could be used instead.

The reason for why the components of a sign are mapped into (instead of being part of) conceptual structures is that it allows to separate signs from concepts. It provides modelling flexibility, allows for the integration of novel signs and for the reuse of the conceptual structures, for example with respect to different languages. Furthermore, in applications where the set of representamens contains all the words of a natural language, it would be highly impractical if such a set would have to be listed explicitly as objects of a concept lattice.

3. Core semiotic definitions

This section defines signs as elements of a triadic relation. Although concept lattices are defined and somewhat discussed, this section focuses mainly on features of semiotic relations which do not depend on concept lattices or FCA. Aspects that depend on concept lattices and FCA are discussed in the next section.

Definition 1. *For a set R (called* representamens*), a set D (called* denotations*) and a set I of partial functions $i : R \nrightarrow D$ (called* interpretations*) a* semiotic relation *S is a relation $S \subseteq I \times R \times D$. A relation instance $(i, r, d) \in S$ with $i.r = d$ is called a* sign.

We are using the notation $i.r = d$ in analogy to programming languages instead of

$i(r) = d$. The reason for this notation is explained below. The sets R, I, D and S should be considered finite but need not be disjoint. Because denotations and interpretations are represented by words or symbols they themselves can also be considered as representamens. For example in the equation $x = 5$, the denotation "5" has two representamens (letter x and number 5). Depending on the purpose of an application it is thus possible that I, D and S are all subsets of R. It should be pointed out that while Def. 1 covers the mathematical aspects of signs, it lacks semiotic aspects such as requiring a sign to be something that is used in communication. The following alternative definition would probably be more acceptable from a semiotic viewpoint:

(Definition 1a). *A* sign *is a unit of communication corresponding to a triple* (i, r, d) *consisting of an interpretation* i*, a representamen* r *and a denotation* d *with the condition that* i *and* r *together uniquely determine* d*.*

But Def. 1a lacks mathematical precision and is not the basis of SCA. The two definitions are similar but not equivalent. Mathematically sets come first in the definition of signs (as in Def. 1). In applications the direction is often the other way around: someone observes an act of communication, decides which are the signs and uses some form a data analysis method to extract the sets R, I, and D and their relationships from the data. Yet another possible definition of signs would be to describe them as rows of a three-column relational database table with a functional dependency. This view is more dynamic because in a database, rows (i.e. signs) can be added, modified and deleted. Thus there are different possibilities for the definition of signs. The commonality between them is the use of a triadic relation and the condition that i and r together uniquely determine d. This is captured in Def. 1 from a mathematical view.

It is possible to use total functions instead of partial functions by adding a NULL-element as shown in the next definition. Formalisations involving a NULL-element can be complex because NULL might correspond to negative, missing or contradictory information. In the remainder of the paper, NULL-elements are only mentioned when it is absolutely necessary and ignored otherwise.

Definition 2. *For a semiotic relation* S*, a* NULL-element $d_\perp$ *is a special kind of denotation with the following conditions: (i)* $i.r$ *undefined in* $D \Rightarrow i.r := d_\perp$ *in* $D \cup \{d_\perp\}$*. (ii)* $d_\perp \in D \Rightarrow$ *all* i *are total functions.*

In mathematics, representamens are usually ignored. In the example of $x = 5$ mentioned above, x and 5 are equal as denotations, but as representamens "x" is a letter and 5 a number. Understanding $x = 5$ requires an interpretation even though this is not normally mentioned. We define $\alpha \in I$ as a standard mathematical interpretation, which is always assumed but not explicitly mentioned. Technically, because there are differences in notation and practices in mathematical subdisciplines, there can be more than one α interpretation, but it is usually clear from the context which α is used. To make α explicit, one could write $\alpha.x = \alpha.5$ or even $\alpha.x\ \alpha.\!= \alpha.5$ instead of $x = 5$ to indicate that x, $=$ and 5 are to be interpreted in the usual mathematical manner. For programming languages, α is for example an interpretation for literals and numbers. For non-mathematical notions, α is undefined. Priss (2016) claims that one reason why students sometimes have problems with mathematics is because they might be incorrectly using interpretations other than α when they are reading mathematical texts.

We define $\rho \in I$ as an interpretation that explicitly maps into representamen features. For example $\rho.x =$ "letter x" and $\rho.5 =$ "number 5". We argue that equality is often not useful for representamens because they tend to display minute variations. For example, two spoken or handwritten words are never totally equal. Under a microscope, printed letters will look differently and even on a computer they might be stored differently. Because equality in mathematics is evaluated with respect to denotations, we use equivalence instead as the main relation on representamens. According to the next definition, representamen equivalence should be finer grained than denotation and interpretation equality because if, for example, “x” and “x” are considered indistinguishable as representamens, it would be difficult to communicate that they are different denotations. An interpretation of equivalent representamens should also result in equal denotations but not vice versa as demonstrated by the example $x = 5$. Representamen equivalence determines which differences are to be ignored. For example, different fonts might be ignored for words, zooming in might be ignored for pictures and rotation might be ignored for graphs. The next definition presents separate sets of equivalence relations for each of the three sign components. Using functions in combination with equivalence relations always poses the question whether these are well-defined.

Definition 3. *For a semiotic relation S with sets R, I, D, sets of tolerance relations $\mathcal{T}_R := \{T \mid T \subseteq R \times R\}$, $\mathcal{T}_I := \{T \mid T \subseteq I \times I\}$, $\mathcal{T}_D := \{T \mid T \subseteq D \times D\}$ and of equivalence relations $\mathcal{E}_R \subseteq \mathcal{T}_R$, $\mathcal{E}_I \subseteq \mathcal{T}_I$ and $\mathcal{E}_D \subseteq \mathcal{T}_D$ are defined with $(d, d_\perp) \in E \in \mathcal{E}_D \Rightarrow d = d_\perp$, $\forall_{d_1,d_2 \in R\cap D} : (d_1, d_2) \in E \in \mathcal{E}_R \implies d_1 = d_2$ and $\forall_{i_1,i_2 \in R\cap I} : (i_1, i_2) \in E \in \mathcal{E}_R \implies i_1 = i_2$.*
An interpretation $i \in I$ respects *$E \in \mathcal{E}_R :\iff (\forall_{r_1,r_2 \in R} : (r_1, r_2) \in E \implies i.r_1 = i.r_2)$.*
A semiotic relation is well-defined *for $E \in \mathcal{E}_R$ if the following conditions are fulfilled:*

a) $\forall_{i_1,i_2 \in I} \forall_{r \in R} : i_1 = i_2 \implies i_1.r = i_2.r$
b) $\forall_{i \in I} : i \in I$ respects $E \in \mathcal{E}_R$

A semiotic relation is well-defined *if it is well-defined for all $E \in \mathcal{E}_R$.*

In this case, $i_1.r = i_2.r$ implies that they are either both defined or both undefined. Unless mentioned otherwise, we always assume “=” $\in \mathcal{E}_I$ and “=” $\in \mathcal{E}_D$ and that condition a) is fulfilled in this paper. If required one could rewrite condition a) using another $E \in \mathcal{E}_I$. The tolerance relations are meant to be defined on the whole set (e.g. $\forall_{r \in R} : (r, r) \in T$) and not just on subsets. It follows that α respects equality but ρ does not. In programming languages, arrays respect equality for their indices. For example, in Python one can write `n=4` and then `employee[n]` in order to retrieve `employee[4]`. But `chr(101)+mployee[n]` produces a syntax error even though `chr(101)` results in “e” and $+$ is the string concatenation because the interpretation that is applied to this representamen does not respect equality. The fact that some interpretations do not respect equality is the reason for using the dot notation $(i.r)$, which emphasises that r is a representamen. For interpretations that respect equality one can write $i(r)$ instead of $i.r$. The next definition ensures that complex signs that are built up from other signs and involve multiple interpretations are well-defined. It is assumed that condition a) from Def. 3 is fulfilled.

Definition 4. *A* sign chain *is a sequence of elements of I and R, brackets and dots, which is recursively evaluated starting at the innermost brackets with the condition that whatever stands left of a dot resolves to an interpretation and what stands right of a dot to a representamen. Interpretations which have a bracketed expression on their right must respect equality.*

For example, a sign chain $(i3.((i2.r2).(i1.r1))).r$ has the condition that $(i2.r2)$ and $(i3.((i2.r2).(i1.r1)))$ resolve to interpretations and $(i1.r1)$ and $((i2.r2).(i1.r1))$ to representamens. In this case $i3$ and $i2.r2$ must respect equality. In general, brackets in sign chains cannot be omitted and sign chains may not fulfil the associative law. In programming languages brackets tend to be missing for the dot notation because of an implicit rule that evaluation is conducted in sequence starting from the left. A sign chain could for example consist of steps such as parsing and disambiguation in the process of language understanding. For example the English word “lead” could either mean a noun “metal” or a verb which is a synonym of “to guide”. Determining the part of speech

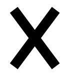

Figure 2. What kind of representamen is this?

disambiguates the word. This could be represented with two interpretations i_1 and i_2 as $i_2.(i_1.$"lead"$) = i_2.$"lead/noun" $=$ "metal lead". Again this is a mathematical modelling which ignores the fact that according to cognitive science, the two interpretations i_1 and i_2 are not independent and may be executed simultaneously.

The following definitions determine the basic structures for each of the three sign components. In each case a concept lattice is defined. The sign components are then mapped into such domain lattices. The sets of objects and attributes of the domain lattices are just further sets which again need not be disjoint to each other or to the sets, S, R, I and D.

Definition 5. *For a set $R_1 \subseteq R$, a formal context $(O_{R_1}, A_{R_1}, J_{R_1})$ and concept lattice $\mathcal{B}(O_{R_1}, A_{R_1}, J_{R_1})$ called* representamen domain context/lattice *are defined with the condition that there is a (total) function $\beta_R : R_1 \to \mathcal{B}(O_{R_1}, A_{R_1}, J_{R_1})$ with $\exists_{E \in \mathcal{E}_R} \forall_{r_1, r_2 \in R_1} : (r_1, r_2) \in E \iff \beta_R(r_1) = \beta_R(r_2)$.*

The condition means that β_R is well-defined and injective on the quotient set R_1/E. Thus the representamen domain lattice presents some modelling of the representamens as concepts in a lattice. Examples for representamen attributes in A_R are "vector graphics", "uppercase letter", "word" and "English sentence". Forming a representamen domain lattice constitutes by itself an act of interpretation which can be modelled using sign chains. For example, Fig. 2 could be interpreted to be a Latin or Greek letter X, a German traffic sign for railway crossings seen from a distance, a multiplication sign, a tick on a form, a word ("to x out"), and so on. What kinds of sets are chosen and how the representamens are represented are modelling decisions which depend on applications. A representamen can be a compound that is made up of other representamens, such as sentences consisting of words consisting of letters consisting of strokes. At what level of granularity signs are considered is another modelling decision. The following definitions for interpretations and denotations are analogous to the one for representamens.

Definition 6. *For a set $I_1 \subseteq I$, a formal context $(O_{I_1}, A_{I_1}, J_{I_1})$ and concept lattice $\mathcal{B}(O_{I_1}, A_{I_1}, J_{I_1})$ called* interpretation domain context/lattice *are defined with the condition that there is a (total) function $\beta_I : I_1 \to \mathcal{B}(O_{I_1}, A_{I_1}, J_{I_1})$ with $\forall_{i_1, i_2 \in I_1} : i_1 = i_2 \iff \beta_I(i_1) = \beta_I(i_2)$.*

Interpretation attributes might represent who is interpreting (a native speaker of a particular language, a teacher or a student, a programming language compiler, and so on) and when and where a sign is used. This could involve a containment hierarchy. For example, there could be an interpretation for a whole book, with separate interpretations for chapters and paragraphs. In applications, either a lattice can be derived from a context or vice versa. From a containment hierarchy for interpretations a lattice can be built using the method of DedekindMacNeille completion. The context can then be formed, for example, by setting the set A_I to correspond to the meet-irreducible lattice elements.

Definition 7. *For a set $D_1 \subseteq D$, a formal context $(O_{D_1}, A_{D_1}, J_{D_1})$ and concept lattice $\mathcal{B}(O_{D_1}, A_{D_1}, J_{D_1})$ called* denotation domain context/lattice *are defined with the condition that there is a (total) function $\beta_D : D_1 \to \mathcal{B}(O_{D_1}, A_{D_1}, J_{D_1})$ with $\forall_{d_1, d_2 \in D_1} : d_1 = d_2 \iff \beta_D(d_1) = \beta_D(d_2)$ and $d_\perp \in D_1 \implies \beta_D(d_\perp)$ is the bottom element of the lattice.*

The condition $d_\perp \in D_1$ is not redundant because there could be applications where all i are total functions without needing $d_\perp$. A denotation domain lattice represents the denotational knowledge of a domain. It could be derived from data or from textbook knowledge using any of the usual FCA techniques for encoding knowledge. The denotational knowledge could also be provided using other knowledge representation techniques (such as conceptual graphs, description logic or formal ontologies). But for the purposes of SCA, we are only interested in denotation domain lattices that are extracted from such knowledge. The next definition shows how some common linguistic terms are formalised in SCA.

Definition 8. *For a semiotic relation S with $T \in \mathcal{T}_D$, $I_1 \subseteq I$, $E \in \mathcal{E}_R$*

a) I_1 *is* compatible $\Leftrightarrow \forall_{i_1,i_2 \in I_1, i_1.r_1 \neq d_\perp, i_2.r_2 \neq d_\perp} : (r_1, r_2) \in E \Rightarrow (i_1.r_1, i_2.r_2) \in T$
b) I_1 *is* mergeable $\Leftrightarrow \forall_{i_1,i_2 \in I_1, i_1.r_1 \neq d_\perp, i_2.r_2 \neq d_\perp} : (r_1, r_2) \in E \Rightarrow i_1.r_1 = i_2.r_2$
c) $(i_1, r_1, d_1) = (i_2, r_2, d_2) \Leftrightarrow i_1 = i_2, (r_1, r_2) \in E, d_1 = d_2$.
d) (i_1, r_1, d_1) *and* (i_2, r_2, d_2) *are* strong synonyms $\Leftrightarrow d_1 = d_2$
e) (i_1, r_1, d_1) *and* (i_2, r_2, d_2) *are* equinyms $\Leftrightarrow (r_1, r_2) \in E$ *and* $d_1 = d_2$
f) (i_1, r_1, d_1) *and* (i_2, r_2, d_2) *are* synonyms $\Leftrightarrow (d_1, d_2) \in T$
g) (i_1, r_1, d_1) *and* (i_2, r_2, d_2) *are* polysemous $\Leftrightarrow (r_1, r_2) \in E$ *and* $(d_1, d_2) \in T$
h) (i_1, r_1, d_1) *and* (i_2, r_2, d_2) *are* homographs $\Leftrightarrow (r_1, r_2) \in E$ *and* $(d_1, d_2) \notin T$
i) (i, r, d) *is* ambiguous $\Leftrightarrow \exists_{(i_1, r_1, d_1) \in S} : (r_1, r_2) \in E$ *and* $d_1 \neq d_2$
j) (i, r, d) *is* anonymous $\Leftrightarrow r = d$ *and* i *respects equality.*

The notions in a) to i) depend on E, thus in cases of possible ambiguity, one can write "compatible with respect to E", "E-synonyms" and so on. The reason for excluding $d_\perp$ from a) and b) is because, for example, variables in programming languages may be undefined before they are assigned for the first time. In that case, "undefined" does not contradict a value that is assigned later. Mergeability means that all interpretations in I_1 can be merged into one interpretation because the result of the merger is still a well-defined function. As mentioned before, equivalence is often more useful for representamens than equality. Therefore two signs are equal if they have equal interpretations and denotations but equivalent instead of equal representamens. In linguistics, synonymous words tend to be expected to have different representamens. Thus a word might not be considered a synonym of itself. But in that case synonymy would just be a symmetric relation, not a tolerance relation. Anonymous signs are signs used in mathematics. As mentioned before, a standard interpretation α can be used for mathematical representamens. If $\alpha.r$ is defined then (α, r, r) with $\alpha.r = r$ is an anonymous sign. This means that in mathematics, only denotational information matters. Apart from mathematical variables, other examples of anonymous signs are programming language literals and numbers. Variables in programming languages are not anonymous and their interpretations are not always mergeable because variables tend to change their values in the course of a program being executed.

As mentioned before, equivalence and tolerance relations may be more important for signs than equality. As shown in Prop. 1 below, some of the notions from Def. 8 are forms of similarity among interpretations and among signs. Synonymy, polysemy and homographs are formalisations of the usual linguistic notions. They always depend on all three components of the signs including interpretations. For example, in some interpretations "car" and "vehicle" are strong synonyms because they are considered to have the same meaning. In other interpretations they are just synonyms because "vehicle" might imply a larger size than "car". Equinymy was coined by Priss (2004) and refers here to the same representamen being used with the same meaning under different interpretations. Thus two instances of a word or phrase used with the same meaning in different sentences are equinyms (for example, "car" in "My car is green. It is a great car."). A single representamen used with slightly different meanings corresponds to polysemous signs, such as "car" as a vehicle or the car of an elevator. Equinymy probably expresses what one might intuitively think it means for signs to be "the same". An example of homographs is presented by the verb "lead" and the metal "lead".

Proposition 1. *For a well-defined semiotic relation S the following statements are true:*

a) Compatibility and mergeability are tolerance relations on $I \times I$. If all interpretations are total functions then mergeability is an equivalence relation on $I \times I$.
b) Equinymy and strong synonymy are equivalence relations, synonymy, polysemy and homographs are tolerance relations on $S \times S$.
c) Strong synonymy implies synonymy. Equinymy implies strong synonymy and polysemy. Polysemy implies synonymy.
d) Compatible interpretations are free of homographs.
e) For mergeable interpretations, polysemy and equinymy are the same.
f) Two anonymous signs are equinyms with respect to "$=$" if they are strong synonyms.
g) If $E \in \mathcal{E}_R$ and $T \in \mathcal{T}_D$ are both the equality relation then two anonymous signs in mergeable interpretations $i_1, i_2 \in I$ can only be either equal (if $i_1 = i_2$), equinyms (if $i_1 \neq i_2$) or not equal. They cannot be homographs.

Proof. a) If denotations are undefined then mergeability need not be an equivalence relation because for mergeable $\{i_1, i_2\}$ and mergeable $\{i_2, i_3\}$ with $i_2.r_2 = d_\perp$ one cannot conclude that $i_1.r_1 = i_3.r_3$.
g) In that case strong synonymy, synonymy, polysemy and equinymy all coincide.
The rest of Prop. 1 follows directly from Def. 8. □

Statement g) describes the normal mathematical use of language where signs are anonymous and only one global interpretation α is used. Therefore mathematical variables tend to be precisely either equal or not equal. The following definition models some notions which are core to Peirce's semiotics. They are included here in order to show how SCA formalises such notions. They are probably only relevant for very specific applications.

Definition 9. *For a semiotic relation S the following are defined:*

a) $(i, r, d) \in S$ is an icon $\Leftrightarrow \exists_{T \in \mathcal{T}_R} : d \in R, (r, d) \in T$
b) $(i, r, d) \in S$ is an index $\Leftrightarrow \exists_{f: R \nrightarrow R} : d \in R, i.r = f(r) = d$
c) $(i, r, d) \in S$ is a symbol $\Leftrightarrow$ *(i, r, d) is neither icon nor index.*

In the case of an icon, the representamen and the denotation are both elements of the same set and similar to each other. Icons are more primitive than symbols because for modelling icons one does not need to differentiate between a set of representamens and a set of denotations. An anonymous sign is both an icon and an index. An index describes a functional relationship among representamens. This could be smoke being an index of fire or a finger pointing to an object. Although i and f are both (partial) functions, processing f is simpler because it only involves R. It should be emphasised that being an icon or index depends on i thus it cannot be concluded that $f(r) = i_1.r$ for any $i_1 \neq i$. In the example of arrays in programming languages, an array f with index number r is interpreted as $i_2.(f(r)) = i_2.d$ in order to retrieve the value $i_2.d$. A symbol is a sign where the relationship between representamen and denotation is "arbitrary" (using Saussure's terminology) because there is no similarity or functional relationship between the representamen and the denotation.

The notions of *syntax, semantics* and *pragmatics* can also be described using SCA. As Pinker (1991) observes, language employs both rules and associative relationships. For example, regular verbs follow rules whereas the patterns of irregular verbs have to be memorised explicitly. In SCA, syntactic rules can be expressed by representamen domain lattices whereas associative syntactic relationships are expressed by $\mathcal{E}_R$ and $\mathcal{T}_R$. Semantics investigates the relationships between a representamen domain and a denotation domain lattice where the interpretations are just treated as elements of a set. From a semantic viewpoint, one is only interested in the set of possible denotations for a representamen but not in how the interpretations themselves are structured. The structures and relationships between interpretations are the focus of pragmatics. For example, comparing how a representamen is used by different people or in different contexts is a question of pragmatics.

4. Modelling semiotic relations with concept lattices

So far we have mainly focused on aspects of semiotic relations which did not rely on concept lattices. This section discusses semiotic relations with concept lattices.

Definition 10. *A semiotic relation S with concept lattices as presented in Definitions 5-7 and $R = R_1, I = I_1, D = D_1$ is denoted by $S^\star$.*

A question is whether and how the relationships among signs and their tolerance and equivalence relations defined above lead to relationships among lattices. For example, any $T_1 \in \mathcal{T}_R$ induces a tolerance relation T_2 on the representamen domain lattice with $(c_1, c_2) \in T_2 :\Longleftrightarrow \exists_{r_1,r_2 \in T_1} : c_1 = \beta_R(r_1), c_2 = \beta_R(r_2)$. For a particular application, this raises the question as to how T_2 relates to the lattice structure, for example, whether one can define a distance metric on the lattice from which T_2 can be derived. Some tolerance relations might only be relevant for some concepts. For $c \in \mathcal{B}(O_R, A_R, J_R)$ and $T \in \mathcal{T}_R$, one can define $c \rhd T :\Leftrightarrow \forall_{r_1 \neq r_2 \in R} : (r_1, r_2) \in T \Rightarrow \beta_R(r_1) \leq c$ and $\beta_R(r_2) \leq c$. Thus T is only interesting for concepts below c and for all other representamens: $(r_1, r_2) \in T \Rightarrow r_1 = r_2$. The interpretation domain and the denotation domain lattices can be investigated in the same manner. The next proposition demonstrates that it is always possible to generate a well-defined semiotic relation $S^\star$ with concept lattices from any semiotic relation S.

Proposition 2. *The following construction yields a semiotic relation $S^\star$ that is well-defined for $E \in \mathcal{E}_{\mathcal{R}}$. For S use $(R, I \times D, J_1)$ as representamen domain context with $(r, (i, d)) \in J_1 :\Leftrightarrow (i, r, d) \in S$ and $(r_1, r_2) \in E :\Leftrightarrow r_1^{J_1} = r_2^{J_1}$, and $(I, R \times D, J_2)$ as interpretation domain context with $(i, (r, d)) \in J_2 :\Leftrightarrow (i, r, d) \in S$ and $i_1 = i_2 :\Leftrightarrow i_1^{J_2} = i_2^{J_2}$. The denotation domain context can be chosen in any manner consistent with Def. 7.*

Proof. The usual FCA definition implies: $r_1^{J_1} := \{(i, d) \in I \times D \mid r_1 J_1 (i, d)\} = \{(i, d) \in I \times D \mid (i, r_1, d) \in S\} = \{(i, d) \in I \times D \mid i.r_1 = d\}$. Therefore $(r_1, r_2) \in E \Leftrightarrow r_1^{J_1} = r_2^{J_1} \Leftrightarrow \forall_{(i,d)} : (i.r_1 = d \Leftrightarrow i.r_2 = d) \Longrightarrow \forall_{i \in I} : i.r_1 = i.r_2$.
Analoguously $i_1^{J_1} = \{(r, d) \in R \times D \mid i_1.r = d\}$. Therefore $i_1 = i_2 \Leftrightarrow i_1^{J_1} = i_2^{J_1} \Leftrightarrow \forall_{(r,d)} : (i_1.r = d \Leftrightarrow i_2.r = d) \Longrightarrow \forall_{r \in R} : i_1.r = i_2.r$. Thus the constructed semiotic relation is well-defined. □

Prop. 2 shows that a construction is always possible, but this construction may not be the most useful modelling for applications. A denotation domain lattice could be constructed in the same manner. Such constructions, which derive three binary relations out of a triadic relation, are already discussed by other authors and are related to the topic of "triadic FCA" (cf. Krolak-Schwerdt, Orlik and Ganter 1994; Lehmann and Wille 1995; Ignatov et al. 2015; Belohlavek and Osicka 2012; Belohlavek, Glodeanu, Vychodil 2013). For semiotic analyses, however, it might be more useful to construct the three lattices from background knowledge and not necessarily directly from the semiotic relation itself. Thus, while there is overlap between SCA and triadic FCA, they are separate theories.

A next step is to investigate how (and whether) the interpretations as partial functions from R to D give rise to interesting functions between the representamen and denotation domain lattices. Fig. 3 shows an example of three lattices. The data for this example are programming language variables. The interpretations are states of a running program. For example, a variable "error" has the value 0 at state "i2". The interpretation domain lattice

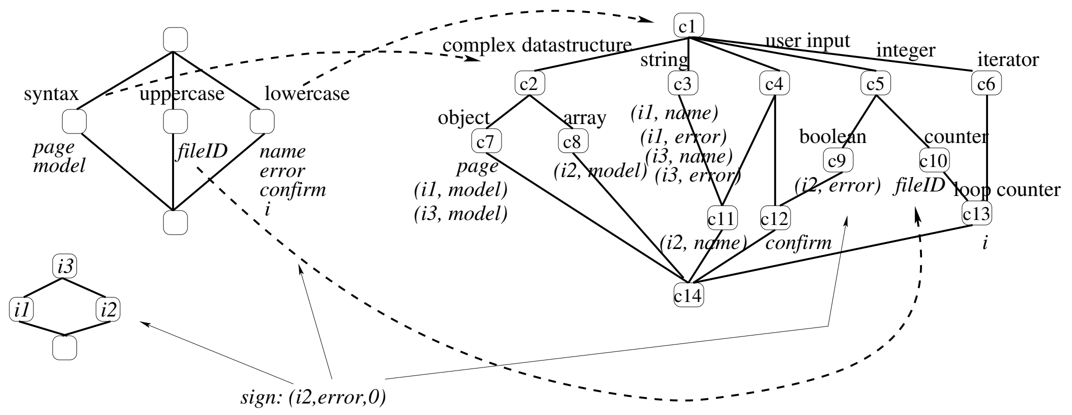

Figure 3. Representamen, interpretation and denotation domain lattices

is shown in the bottom left. It does not provide much information other than that there are three interpretations and i_3 is in some sense more general than the other two. In the representamen domain lattice on the left, the variable names (or representamens) are displayed as the formal objects. Thus β_R is a mapping into object concepts. The formal attributes in this lattice refer to whether a representamen contains uppercase letters or only lowercase letters. The attribute "syntax" characterises representamens which use a specific syntactic structure when they are defined or used, such as square brackets in the case of arrays or the word "new" for objects.

In the denotation domain lattice, the denotations in this example are values of the variables in a running program. They are mapped onto object concepts in the denotation domain lattice, but they are not shown in the lattice. Instead of the values, representamens are shown if they have the same value for all three interpretations (i.e. equinymous signs) and pairs of interpretations and representamens otherwise (i.e. for polysemous signs). For example, i1.page = i2.page = i1.model = i3.model. It is now of interest to investigate whether any representamen structures are preserved in the denotation domain. All representamens which have the representamen attribute "syntax" are below the denotation "complex datastructure". Thus there is a correspondence between the representamen domain and the denotation domain. This is shown by the dashed line in Fig. 3. The representamens with attribute "uppercase" also relate to a concept in the denotation domain, but the reason for this could be because this only affects one representamen. The join of the denotations of the representamens with attribute "lowercase" is the top concept in the denotation domain lattice. Thus no structure is preserved for them. The following definition describes functions from the representamen into the denotation domain lattice.

Definition 11. *For a semiotic relation $S^\star$ with subsets C_R, C_I and C_D of the sets of concepts of the representamen, interpretation and denotation domain lattices with $d_\perp \in D$, functions $Z_i : R \to \mathcal{P}(\mathcal{B}(O_D, A_D, J_D))$ and $\overline{Z}_{C_I} : \mathcal{P}(\mathcal{B}(O_R, A_R, J_R)) \to \mathcal{P}(\mathcal{B}(O_D, A_D, J_D))$ are defined as:*

a) $Z_i(r) := \{\beta_D(i_1.r_1) \mid \exists_{r_1 \in R, i_1 \in I} : \beta_R(r_1) = \beta_R(r), i_1 = i\}$
b) $\overline{Z}_{C_I}(C_R) := \bigcup_{i \in \{i \in I \mid \beta_I(i) \in C_I\}, r \in \{r \in R \mid \beta_R(r) \in C_R\}} Z_i(r)$
c) $\overline{Z}_i^\vee(C_R) := \bigvee_{\{r \in R \mid \beta_R(r) \leq C_R\}} Z_i(r), \qquad Z^\vee(r) := \bigvee_{i \in I} Z_i(r)$
d) $\overline{Z}_i^\wedge(C_R) := \bigwedge_{\{r \in R \mid \beta_R(r) \leq C_R\}} Z_i(r), \qquad Z^\wedge(r) := \bigwedge_{i \in I} Z_i(r)$

One can further define $\overline{Z}_{C_I}^{-1}(C_D)$ and $Z_i^{-1}(\{\beta_D(d)\})$ for the reverse images and $\overline{Z}^\diamond$ for the interval from $\overline{Z}^\vee$ to $\overline{Z}^\wedge$.

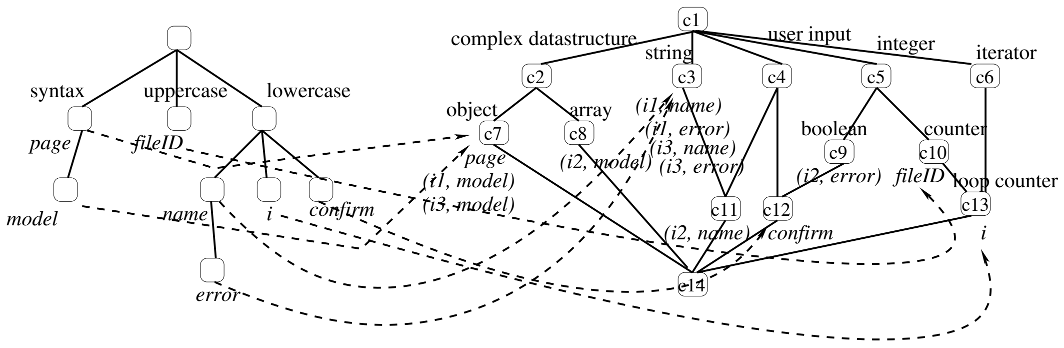

Figure 4. A well-defined semiotic relation

Because the join and meet relations in a lattice are commutative and associative it does not matter whether one first iterates through interpretations or through representamens. Thus all of the operations in Def. 11 are well-defined. The joins and meets in c) and d) are meant to be performed on the concepts of the denotation domain lattice, i.e. on the elements of the sets. The functions in c) and d) result in concepts and not sets of concepts as the other functions. The functions in b) are monotonic with respect to set inclusion: $C_{R1} \subseteq C_{R2}$ and $C_{I1} \subseteq C_{I2} \Rightarrow \overline{Z}_{C_{I1}}(C_{R1}) \subseteq \overline{Z}_{C_{I2}}(C_{R2})$. Some examples for Fig. 3 are: $Z_{i1}(\text{page}) = \{\text{c7}\}$, $Z_{i2}(\text{page}) = \{\text{c7}, \text{c8}\}$, $\overline{Z}^{\vee}_{i2}(\{\beta_R(\text{page})\}) = Z^{\vee}(\text{page}) = c2$ and $Z^{\wedge}(\text{page}) = c14$, $Z^{\diamond}(\text{page}) = \{\text{c2}, \text{c7}, \text{c8}, \text{c14}\}$. These different functions show that the representamen concept for "page" corresponds in some way to the interval between c2 and c14 in the denotation domain lattice. The exact location in the interval depends on which of the functions is chosen. The following proposition shows that in the case of well-defined semiotic relations, the functions are simpler. In this proposition $E \in \mathcal{E}_R$ is meant to be the same one as used for β_R.

Proposition 3. *For a semiotic relation $S^{\star}$ as in Def. 11, the following statements hold:*

a) $\overline{Z}_{C_I}(C_R) = \{\beta_D(i.r) \mid \exists_{r \in R, i \in I} : \beta_R(r) \in C_R, \beta_I(i) \in C_I\}$
b) If $S^{\star}$ is well-defined then $Z_i(r) = \{\beta_D(i.r)\}$
c) For $S^{\star}$ well-defined: (i, r, d) is ambiguous $\iff \exists_{i \in I} : Z_i(r) \neq \{Z^{\vee}(r)\}$

Proof. a) $x \in \overline{Z}_{C_I}(C_R) \iff \exists_{r \in R, i \in I} : \beta_R(r) \in C_R, \beta_I(i) \in C_I, x = \beta_D(i.r) \iff \exists_{r_1 \in R, i_1 \in I, r \in R, i \in I} : \beta_R(r_1) = \beta_R(r) \in C_R, \beta_I(i_1) = \beta_I(i) \in C_I, x = \beta_D(i.r) \iff \exists_{r \in R, i \in I} : \beta_I(i) \in C_I, \beta_R(r) \in C_R, \exists_{r_1 \in R, i_1 \in I} : \beta_I(i_1) = \beta_I(i), \beta_R(r_1) = \beta_R(r), x = \beta_D(i_1.r_1)$
b) Follows from the definition of "well-defined".
c) (i, r, d) is ambiguous $\iff \exists_{i_1} : i_1.r \neq i.r \iff \beta_D(i_1.r) \neq \beta_D(i.r) \iff Z_{i_1}(r) \neq Z_i(r)$. □

The semiotic relation in Fig. 3 is not well-defined because for example, "name" and "confirm" are not mapped onto the same denotations. Fig. 4 shows a version of Fig. 3 where the representation domain lattice has been modified so that the semiotic relation is well-defined. In this case only the representamen and denotation domain lattices are shown but without the bottom elements of the lattices. The dashed lines show the $Z_{i1}(r)$ functions. Some examples are $Z_{i1}(\text{page}) = Z_{i2}(\text{page}) = \{\text{c7}\}$, $Z^{\vee}(\text{page}) = c7$ but $\overline{Z}^{\vee}_{i2}(\{\beta_R(\text{page})\}) = c2$, $Z_{i1}(\text{model}) = \{\text{c7}\} \neq \{\text{c8}\} = Z_{i2}(\text{model})$ and $Z^{\vee}(\text{model}) = c2$ because "model" is ambiguous.

Another question is if and when any of the functions from Def. 11 are order-preserving. Many questions have not yet been answered. Closure operators could be investigated. Lattice theoretical properties could be used to define further structures on the sign components.

For example, $r_1 \leq r_2 :\Longleftrightarrow \beta_R(r_1) \leq \beta_R(r_2)$. But from an applied mathematics viewpoint, it would be of interest to first establish the usefulness of SCA in more applications before the theory is developed further. As mentioned in the title and introduction, this paper is intended as a proposal for further research.

5. How to use SCA in applications

SCA has already been used for a number of applications. The notion of "anonymous signs" mentioned in Section 3 was first introduced by Priss (2004) in order to explain why variables used in mathematics are quite different from variables used in programming languages. As mentioned above, Priss (2016) speculates that anonymous signs might be one reason why many students perceive mathematics as a difficult subject to learn. Priss (2015) uses SCA for analysing programming source code and to generalise some constructions from relational concept analysis by means of SCA. Priss (2016) shows that SCA is a possible tool for investigating conceptual learning. In that respect SCA can be compared to other educational analysis methods (such as Knowledge Space Theory [Albert and Lukas 1999; Falmagne et al. 2013] and APOS theory [Dubinsky and Mcdonald 2002]) because concept lattices facilitate the visualisation of student learning progress and misconceptions between the concepts of a student and of an expert.

The framework of SCA is quite open with respect to how the data is modelled and investigated. Based on the definitions provided in Section 3 there are, however, a number of typical tasks which are elaborated in this section. *Homograph detection* is an example of such a task. As defined above, homographs are signs with equivalent representamens and dissimilar denotations. In programming languages, homographs occur if a single variable or other identifier is used for several completely different purposes. An example is "error" in Fig. 3 in contrast to "name". Because $Z^{\vee}(\text{name}) = c3$ is not the top node, "name" is most likely polysemous whereas $Z^{\vee}(\text{error}) = c1$ is the top node, possibly indicating that "error" could be a homograph. While homographs are not necessarily programming errors, they can make it more difficult to read and understand code, which might lead to errors. Once homographs have been detected, they can be eliminated by renaming. In educational applications of SCA, homographs occur if, for example, a student uses a term inconsistently with how it is used by experts. Detecting homographs helps a teacher to determine which are difficult concepts of a domain. Homographs which are inherent in domain knowledge (when a term is used for two unrelated concepts even by experts) should also be highlighted to teachers because students should be alerted to such problems in the domain knowledge.

It is also useful to search for and identify synonyms by investigating how representamens relate to denotations. In programming languages, strong synonyms are pointers or references. In that case synonymy depends on interpretations because the values of pointers or references can change during the execution of a program. In order to avoid redundancy it might be necessary to detect strong synonyms and eliminate one of the redundant representamens.

For the three domain lattices, it may be of interest to investigate whether the β-functions provide sufficient coverage of the lattices. If a concept is neither an image of a β-function nor a meet- or join-combination of other concepts that are images of β-functions, it is not relevant for the semiotic relation that is being mapped. In natural language research, denotational concepts which are not an image of a β-function but a meet- or join-combination are called *lexical gaps* because they can be expressed but not by a single representamen. Two different natural languages which are mapped into a single denotation domain lattice might show lexical gaps as differences in lexicalisation between the languages.

One purpose of the Z-functions is to determine whether structures from the representamen domain are mapped into the denotation domain. Well-defined semiotic relations with order-preserving mappings are highly structurally aligned. In applications where the representamens are used to express information about a domain in a natural language a high degree of alignment between representamen and denotation structures would be desirable. This is because one would hope that natural language expressions present their meanings fairly consistently.

In many of the applications of SCA so far, the interpretation domain lattice has not been used much. As mentioned in Section 3, interpretations could present some form of containment hierarchy, such as a book, a chapter, a paragraph and a sentence or a whole program, a class or function, a loop and a line of code. In that case the order between finer- and broader-grained interpretations should be preserved. Instead of eliminating or renaming representamens in order to avoid synonyms and homographs, it is also possible to eliminate interpretations. For example, if representamens are provided in both German and English, then it is not surprising that many synonyms occur. Eliminating one of the interpretations (German or English) will remove such synonyms. More applications are needed in order to determine how to use the interpretation domain lattice most effectively.

6. Conclusion

In conclusion, this paper demonstrates that starting from a triadic relation for signs, one can develop a mathematical formalisation of core notions from semiotics and linguistics. Because it is convenient to model the meaning of signs as concepts, it is promising to combine this triadic relation with formalised conceptual structures. It can be challenging to clearly distinguish between denotations (which are differentiated by being equal or unequal), representamens (whose equivalence relation is finer grained than equality) and signs. In common language such distinctions are rarely expressed because a sign is usually referenced via its representamen and, especially in mathematical contexts, it only matters whether items are (denotationally) equal and not how they are represented. Thus it is an important contribution of SCA to provide a terminology and methodology for modelling and investigating the sign components independently. We are not aware of any other existing mathematical semiotics theory which accomplishes this.

SCA provides many possibilities for further modelling, such as how to characterise structures and how to define mappings among the conceptual structures. While the general usefulness of SCA has been demonstrated in a small number of applications, more applications are needed in order to decide which directions to take with further modelling. Thus this paper presents the core notions of SCA but is also a proposal for future development.

Acknowledgements

This work was partially supported by the German Federal Ministry of Education and Research (BMBF) under Grant 01PL16066H. The sole responsibility for the content of this paper lies with the author.

References

Albert, D., and J. Lukas, eds. 1999. *Knowledge Spaces: Theories, Empirical Research, Applications.* Mahwah, NJ: Lawrence Erlbaum Associates.

Belohlavek, Radim, and Petr Osicka. 2012. "Triadic concept lattices of data with graded attributes." *International Journal of General Systems* 41 (2): 93-108.

Belohlavek, Radim, Cynthia Glodeanu, and Vilem Vychodil. 2013. "Optimal factorization of three-way binary data using triadic concepts." *Order* 30 (2): 437-454.

Dubinsky, Ed., and Michael A. Mcdonald. 2002. "APOS: A Constructivist Theory of Learning in Undergraduate Mathematics Education Research." *ICMI Study Series* 7: 275-282.

Falmagne, J.-C., D. Albert, D. Doble, D. Eppstein, and X. Hu. 2013. *Knowledge Spaces: Applications in Education.* Berlin-Heidelberg-New York: Springer.

Ganter, Bernhard, and Rudolf Wille. 1999. *Formal Concept Analysis. Mathematical Foundations.* Berlin-Heidelberg-New York: Springer.

Goguen, Joseph. 1999. "An introduction to algebraic semiotics, with application to user interface design." In *Computation for metaphors, analogy, and agents*, edited by Chrystopher L. Nehaniv, 242-291. Berlin-Heidelberg-New York: Springer.

Ignatov, Dmitry I., Dmitry V. Gnatyshak, Sergei O. Kuznetsov, and Boris G. Mirkin. 2015. "Triadic formal concept analysis and triclustering: searching for optimal patterns." *Machine Learning* 101 (1): 271-302.

Krolak-Schwerdt, Sabine, Peter Orlik, and Bernhard Ganter. 1994. "TRIPAT: a model for analyzing three-mode binary data." In *Information Systems and Data Analysis*, edited by H.-H. Bock, W. Lenski, and M. Richter, 298-307. Berlin-Heidelberg-New York: Springer.

Lehmann, Fritz, and Rudolf Wille. 1995. "A triadic approach to formal concept analysis." In *International Conference on Conceptual Structures*, edited by G. Ellis, R. Levinson, W. Rich, and J. Sowa, 32-43. Berlin-Heidelberg-New York: Springer.

Marty, Robert. 1992. "Foliated semantic networks: concepts, facts, qualities." *Computers & mathematics with applications* 23 (6): 679-696.

Pinker, S. 1991. "Rules of language." *Science* 253: 530-535.

Priss, Uta. 2004. "Signs and Formal Concepts." In *Concept Lattices: Second International Conference on Formal Concept Analysis*, edited by Peter Eklund, LNCS 2961, 28-38, Berlin-Heidelberg-New York: Springer.

Priss, Uta. 2015. "An Introduction to Semiotic-Conceptual Analysis with Formal Concept Analysis." In *Proceedings of the Twelfth International Conference on Concept Lattices and Their Applications*, edited by Yahia and Konecny, 135-146.

Priss, Uta. 2016. "A Semiotic-Conceptual Analysis of Conceptual Learning." In *Graph-Based Representation and Reasoning*, Proceedings of the 22nd International Conference on Conceptual Structures, edited by Haemmerle, Stapleton, and Faron-Zucker, LNCS 9717, 122-136, Berlin-Heidelberg-New York: Springer.

Zalamea, Fernando. 2010. "Towards a Complex Variable Interpretation of Peirces Existential Graphs." In *Ideas in Action*, Proceedings of the Applying Peirce Conference, edited by Bergman, M., S. Paavola, A. Pietarinen, and H. Rydenfelt, 277-287.