Contents

1 Formal Logic with Conceptual Graphs **3**
Frithjof Dau
1.1 Introduction 3
1.2 Short Introduction to Conceptual Graphs 4
1.2.1 Simple Conceptual Graphs 4
1.2.2 Nested Conceptual Graphs 7
1.2.3 Negation in Conceptual Graphs 8
1.2.4 Conceptual Graphs and First Order Logic 8
1.2.5 Reasoning with Conceptual Graphs 9
1.3 Conceptual Graphs from a Formal Point of View 9
1.3.1 Sowa's Syntax 10
1.3.2 Sowa's Definition of $\mathbf{\Phi}$ 10
1.3.3 Sowa's Understanding of Negation 11
1.3.4 Sowa's Calculus 12
1.4 The General Approach for Formal Logic with CGs 13
1.4.1 Syntax 14
1.4.2 Semantics 17
1.4.3 Reasoning Facilities 18
1.5 Different Forms of Conceptual Graphs 20
1.5.1 Simple Conceptual Graphs 20
1.5.2 Nested Conceptual Graphs 24
1.5.3 Conceptual Graphs with Atomic Negation 25
1.5.4 Conceptual Graphs with Full Negation 27
1.6 Works not Cited 29

References **31**

0-8493-0052-5/00/$0.00+$.50

Chapter 1

Formal Logic with Conceptual Graphs

Frithjof Dau

1.1 Introduction 3
1.2 Short Introduction to Conceptual Graphs 4
1.3 Conceptual Graphs from a Formal Point of View 9
1.4 The General Approach for Formal Logic with CGs 13
1.5 Different Forms of Conceptual Graphs 19
1.6 Works not Cited 28

1.1 Introduction

This chapter aims to give an introduction into the formal theory of conceptual graphs (abbreviated by CGs), with an emphasis on how CGs can be understood as a diagrammatic approach to formal logic. Of course, as it can already easily be seen from the huge variety of topics covered in this book, formal logic is only one of many aspects of CGs. Anyhow, maybe not the majority, but at least a vast amount of CG research has focused on this specific facet, and even Sowa emphasizes the logic aspect of CGs.

CGs are based on diagrammatic representation of logic, namely Peirce's (1839-1914) existential graphs (for introductions into existential graphs, see [Zem64, Rob73, Shi02]). In [Sow97] Sowa writes that 'conceptual graphs (CGs) are an extension of existential graphs with features adopted from linguistics and AI. The purpose of the system is to express meaning in a form that is logically precise, humanly readable, and computationally tractable.'

The system of existential graphs is divided into three parts named *Alpha*, *Beta* and *Gamma*. Alpha corresponds to propositional logic, and Beta corresponds to first-order predicate logic. Gamma is more complicated: It covers features of higher order and modal logic, the possibility to express self-reference, and other features. Due to its complexity, it was not completed by Peirce. Peirce's existential graphs and Sowa's CGs cannot be directly compared. Sowa adopted many ideas of existential graphs, even from Gamma, but CGs have a different and richer syntax, and they are tailored to better suit the needs of contemporary knowledge representation systems. Anyhow, Sowa's goal CGs to be 'logically precise' and his reference to existential graphs

0-8493-0052-5/00/$0.00+$.50

clearly hints to the logical alignment of CGs. This is more explicitely expressed in [Sowa], where Sowa states that 'CGs have been developed as a graphic representation for logic with the full expressive power of first-order logic and with extensions to support metalanguage, modules, and namespaces.'

In the following, we scrutinize CGs in terms of formal logic. First, in Sec. 1.2, we give a short introduction into CGs, as they have been introduced by Sowa. In this section, some core notations of CGs are defined. Next, in Sec. 1.3, we shortly investigate whether Sowa's CGs are indeed 'logically precise', and we will see that they *not* suit the needs of contemporary formal logic. For this reason, much research on the formal theory of CGs aims to fix the formal gaps and flaws of CGs. In Sec. 1.4, an overview over different approaches to turn CGs into a mathematically precise system of logic is given, and a core notation for a formal theory is provided. In different works, different fragments of CGs are elaborated in a precise manner. An overview over these fragments is given in Sec. 1.5.

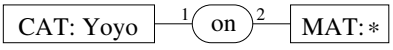
A final remark to the definitions in the following sections has to be made. Different authors have used different notations, and the following definitions aim to select the most convenient ones among these. A term which is defined as it is used in this chapter is set in small caps, like VOCABULARY. If we refer to a notation which is not used in this volume, it is set in italics, like *support*.

1.2 Short Introduction to Conceptual Graphs

In this section, an introduction into the theory of CGs, as they have been invented by Sowa, is provided.

1.2.1 Simple Conceptual Graphs

In the following, the broad range of CGs is illustrated by several examples. The first one, given below, is probably one of the best known CGs.

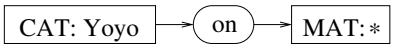


The boxes in this graph are called CONCEPT BOXES. We have to stress that usually, the term *concept* istead of *concept box* is used, but this term is often used in a much broader, often semantical understanding in knowledge representation. Moreover, it can be confused with the *formal concepts* of Formal Concept Analysis. Finally, by using the term 'box', we refer to some extent to the diagrammatic representations of CGs. For these reasons, this chapter sticks to the use of the term *concept box*. When in Section 1.4 the formalization of CGs by means of mathematical graphs is provided, we will speak of concept *vertices* instead. In each concept box, we have two entities: A

TYPE LABEL and a REFERENT. In the graph above, the two concept boxes contain two different kinds of referents. The referent 'Yoyo' in the left box is a name for an object. The referent '$*$' in the right box is called GENERIC MARKER, thus the right concept box is called GENERIC CONCEPT BOX. The generic marker denotes an unqualified object, i.e., it can be understood as an existential quantification. The type labels of the concept boxes are 'Cat' and 'Mat', respectively, and the concept boxes encode the information that the referents belongs to the type. Thus the left concept box encodes the atomar piece of information that Yoyo is a cat, and the right concept box encodes the information that there exists a mat. Type labels are ordered in a sub-type-relation, according to their level of generality. For example, CAT is a subtype of ANIMAL. An ordered set of types is often called *support*, *type hierarchy*, *taxonomy* or *alphabet*.

Besides the concept boxes, the graph contains an RELATION OVAL, labelled with 'on'. The concept boxes are LINKED with ARCS to the relation oval. The relation in the oval relates the referents of these concept boxes. In our example, the meaning of the relation oval is 'Yoyo is on the (unknown) mat'. So, the meaning of the whole graph is 'Yoyo is a cat, there is a mat, and Yoyo is on that mat', or 'the cat Yoyo is on a mat' for short.

Of course, the order of the arguments of the relations matter. In the diagrammatic representation, the order is represented by indexing the arcs which link the concept boxes to the relation oval with numbers. Another way to depict the order of the arguments is the use of use arrows instead of arcs in the diagrams. The corresponding diagram is then



.

This approach works fine if only relations of arity 1 or 2 are used, but for relation with arities higher than 2 (e.g., 'between', which is a ternary relation), this approach fails. So indexes are preferrable.

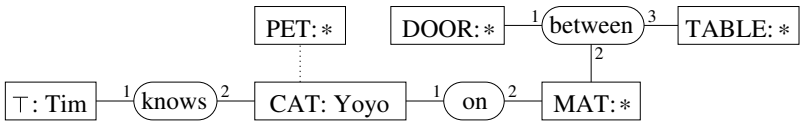
Sowa uses various kinds of referents, for example

- names: PERSON: John , DOG: Lucky, Matula ,
- quantifiers: MEAT: $*$ (some meat), DOG: $\forall$ (all dogs),
- measure specifications: TIME-PERIOD: @5 seconds , Money:@$5
- control marks ?, !, #: DOG: ? (which dog?), Dog: Lucky ! (emphasis on Lucky), MEATBALL: # (*the* meatball),
- generic plurals: BONE: {$*$} (some bones) or BONE: {$*$}4 (four bones),
- prefixes: PERSON: Dist{Bill, Mary}@5 (five persons distributively including Bill, Mary, and others), or LADY: Cum{$*$} (a set of Ladies).

(all examples taken from [Sow92]).

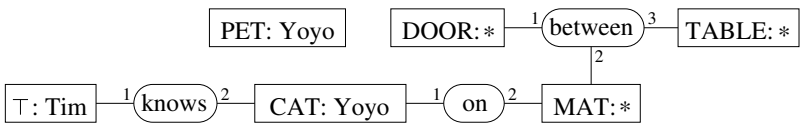
These various kinds of referents verify the comprehensive knowledge representation claim of CG. On the other hand, it is clear that most of them go beyond the expressiveness of first order logic. For the elaboration of CGs in term of formal logic, we will restrict the referents to object names or the generic marker.

Below, another example is depicted, which covers more of the constituent elements of CG.



First of all, please note that concept boxes can be linked to more than one relation oval, and that we have a triadic relation in this graph. The type label '⊤' in the leftmost concept box is the UNIVERSAL TYPE, which contains every object (of the respective universe of discourse) in its extension. As the type labels are usually ordered, ⊤ is the greatest element in this order. Finally note the dotted line between the concept boxes with the type labels Cat and Pet, respectively. This dotted line is a COREFERENCE LINK, expressing the identity of the referents in the linked concept boxes. Concept boxes which are linked with a coreference link are said to be COREFERENT. A COREFERENCE SET is a set of (pairwise) coreferent concept boxes. The two boxes with the type labels Cat and Pet are coreferent, and the subgraph consisting of these two boxes and the the coreference link expresses that Yoyo is a cat and there is a pet which is the same as Yoyo, or for short: Yoyo is both a cat and a pet. The meaning of the whole graph is therefore Tim knows Yoyo, Yoyo is both a cat and a pet, and Yoyo is on a mat which is between a door and a table.

Although most practical examples are, CGs do not need to be connected. In the last example, we used two concept boxes, one of them being generic, and a coreference link to express that Yoyo is both a cat and a pet. Below, another graph is depicted which has exactly the same meaning as the last one. Now we use two non-connected, non-generic concept boxes to express that that Yoyo is both a cat and a mat.

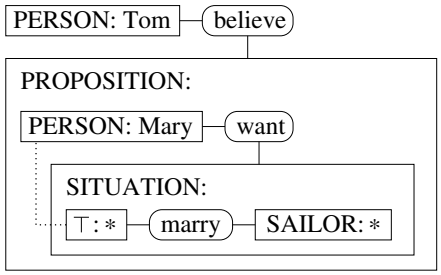


The CGs which can be constructed by means of concept boxes, relation ovals and coreference links, where the referents can be names for objects or the generic marker, are called SIMPLE CONCEPTUAL GRAPHS (SCGs). The class of SCGs has the expressiveness of the ∃, ∧-fragment of first order logic (including identity). It is the most important and prominent fragment of CGs.

There are two kinds of SCGs which cannot be further decomposed. A SINGLETON GRAPH is a CG which consists only of a single concept box. A STAR GRAPH is a CG which contains a single relation oval, and concept boxes which are linked to that oval. Our first example is a star graph. Finally, even having no concept box or relation oval at all is a CG, the so-called BLANK GRAPH.[1]

1.2.2 Nested Conceptual Graphs

We have already seen a range of different referents in CGs, but one important referent, which vastly extends the expressiveness of CGs , is still missing. Sowa allows for whole CGs as referents in concept boxes. This construction is called NESTING. When CGs are nested, some CGs make statements and assertions about other CGs, i.e., the system of CGs offers the possibility of meta-level statements. Concept boxes whose referents are CGs are called *contexts*.[2] Besides concept boxes which contain CGs as referents, the complete area of the CG is a context as well, which is called the OUTERMOST CONTEXT. Below we present a well-known example of a nested CG.



Besides the outermost context, this graph contains two further contexts, namely the concept boxes of type PROPOSITION and SITUATION (which are common types of contexts). The graph can be read as follows: The person Tom believes a proposition, which is described by a graph itself. The proposition states that the person Mary wants a situation, which again is described by a graph. In this situation we have a concept box $\boxed{\top : *}$ which is connected with a coreference link to the concept box $\boxed{\text{PERSON:Mary}}$ in the context above. So the situation is that Mary marries a sailor. The formal understanding of the whole graph is now: The person Tom believes the proposition that the person Mary wants the situation that Mary marries a sailor. In short: The person Tom believes that the person Mary wants to marry a sailor.

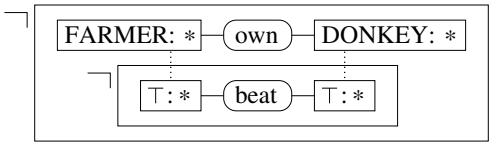
[1] The blank graph corresponds to the empty *sheet of assertion* in Peirce's existential graphs.

[2] The term 'context' occurs in several meanings and implementations in logics, linguistics or artificial intelligence. An introduction into the various ideas of contexts is beyond the scope of this chapter. We refer to [Sowb] or Chap. 5 of [Sow00] for an introduction and discussion of contexts in conceptual graphs.

1.2.3 Negation in Conceptual Graphs

For the discussion of CGs in terms of mathematical logic, it is important to note that Sowa expresses negation with contexts, too. As Sowa states in [Sow97]: 'The EG [Existential Graph] negative contexts are a special case of the CG contexts. They are represented by a context of type NEGATION whose referent field contains a CG that states the proposition which is negated.' Concept boxes of type NEGATION are introduced by Sowa as abbreviations for contexts of type proposition with an unary relation 'NEG' attached to it (see [Sow92], where he says that 'Negation (NEG) is one of the most common relations attached to contexts'), and Sowa often abbreviates these contexts by drawing a simple rectangle with the mathematical negation symbol ¬ (e.g. in [Sow99]).

A well known example for a CG with negations is presented below. The device of two nested negation contexts can be understood as an implication. So the meaning of the graph is 'if a farmer owns a donkey, then he beats it'.



1.2.4 Conceptual Graphs and First Order Logic

Sowa states in [Sow84] that 'any formula in first-order logic can be expressed with simply nested contexts and lines of identity.' (the term 'line of identity' is adopted from Peirce's existential graphs and refers in this quotation to the coreference links of CGs). In particular CGs are designed to have at least the full power of first order logic, that is first order logic with identity, object names and predicate names, but without function names. In the following, we will use the abbreviation FOL for this style of logic. Anyhow, as we have seen, the system of CGs is richer than FOL.

To show how CGs and FOL are related, Sowa defines a formal operator Φ which maps CGs to FOL. The definition of Φ can be found in [Sow84], and we find various comments and examples among the chapters of Sowa. The Φ-operator is useful for making the structural differences between CGs and the symbolic elaborations of FOL explicit, and it serves as a semantics for CGs as well. An example for the Φ-operator is the translation of the CG of Section 1.2.3 to

$$\neg\exists x.\exists y.(FARMER(x) \wedge DONKEY(y) \wedge owns(x,y) \wedge \neg beat(x,y)) \quad .$$

The last formula can is logically equivalentbe to the simplified formula

$$\forall x.\forall y.((FARMER(x) \wedge DONKEY(y) \wedge owns(x,y)) \rightarrow beat(x,y)) \quad .$$

1.2.5 Reasoning with Conceptual Graphs

When he developed the system of CGs, Sowa did not only adopt the iconicity and many syntactical and semantical elements of existential graphs. He also adopted the calculus of existential graphs. First of all, Sowa distinguishes between *equivalence rules*, which do not change the meaning of a CG, *specialization rules*, which (usually) specialize the meaning of a CG, and finally *generalization rules*, which in turn generalize the meaning of a CG. All rules transform a graph into a new graph. Equivalence rules can be carried out in both directions, whereas both specialization rules and generalization rules can be carried out only in one direction, and the specialization rules and generalization rules are mutually inverse. A simple example of a generalization rule is the generalization of a type or a referent in a concept box.

In [Sow84] Sowa provides a calculus which is a one-to-one translation of Peirce's rules for existential graphs into the system of CGs. In [Sow97] or in [Sowa], this system is refined. Below, Sowa's rules from [Sow97] are given:

Erasure: In a positive context, any graph u may be replaced by a generalization of u; in particular, u may be erased (i.e. replaced by the blank, which is a generalization of every CG).

Insertion: In a negative context, any graph u may be replaced by a specialization of u; in particular, any graph may be inserted (i.e. it may replace the blank).

Iteration: If a graph u occurs in a context c, another copy of u may be drawn in the same context c or in any context nested in c.

Deiteration: Any graph u that could have been derived by iteration may be erased.

Equivalence: Any equivalence rule (copy, simplify, or double negation) may be performed on any graph or subgraph in any context.

1.3 Conceptual Graphs from a Formal Point of View

In some sense the system of CGs is not fixed, but open-minded. It is designed to be used in fields like software specification and modelling, knowledge representation, natural language generation and information extraction, and these fields have to cope with problems of implementational, mathematical, linguistic and even philosophical nature. Sowa addresses many of these problems in his landmark work [Sow84]. Due to the complexity of the system of CGs, it is nearly impossible and perhaps not even desirable to consider the overall system as an approach to formal logic. Consequently, all works –including Sowa's– which link CGs and formal logic only cover specific fragments of CGs. Anyhow, we have seen that one goal of CGs is to provide a humanly readable form of FOL.

In this section we scrutinize Sowa's CGs from the the more viewpoint of formal logic. It has to be acknowledged that in his works, Sowa provides core ideas for an formal elaboration of CGs. In the strict and narrow framework of mathematical logic, these core ideas are still ambigious and lack mathematical preciseness. This is the main reasons why several authors elaborated fragments of CGs in minute mathematical detail. Before we come to an overview of these elaborations in the next section, this section first describes in which respects Sowa's ideas have to be refined from a mathematical point of view.

1.3.1 Sowa's Syntax

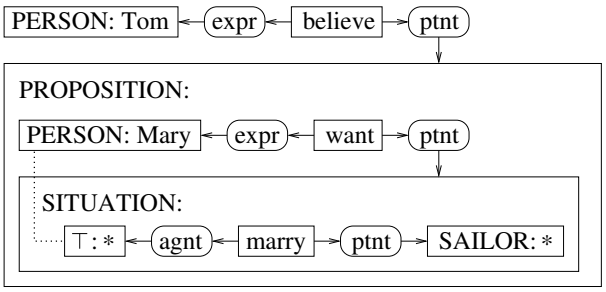
First of all, Sowa wanted to provide at least for a core of CGs a precise definition, termed *abstract syntax*, which was intended to result in an ISO-standard (see [Sow99, Sowa]). In the working draft [Sowa] for the definition he writes: 'Informally, a CG is a structure of concepts and conceptual relations where every arc links a concept node [a concept box] and a conceptual relation node [a relation oval]. Formally, the abstract syntax specifies CGs as mathematical structures without making any commitments to any concrete notation or implementation.' The definition of the abstract syntax is not a mathematical definition, but it is written in common English. Furthermore, the definitions are incomplete, and many aspects of CGs which should be covered by the definition are only explained in the comments. So the first step for a formal elaboration of CGs has to provide mathematical definition for fragments of CGs. This is usually done by means of mathematical graph theory, as it is described in Sec. 1.4.1.

1.3.2 Sowa's Definition of Φ

Next we consider Sowa's definition of the Φ-operator, which is the 'classical' approach to equip CGs with a semantics. The fallacies in Sowa's definition of Φ have been addressed by several authors, for example by Chein and Mugnier [CM92], Wermelinger [Wer95], and Dau (the author of this chapter) [Dau03b].

Wermelinger writes in [Wer95]) that 'Sowa's original definition of the mapping (Φ) is incomplete, incorrect, inconsistent, and unintuitive, and the proof system is incomplete, too.' Although this judgment is formulated very harshly, it has somewhat to be acknowledged from a mathematical point of view. First of all, Wermelinger critizises that the universal type $\top$, although it has a special meaning, is not treated in a special way. This can be considered a minor gap, which can be easily fixed. Another minor gap is that Sowa translates the blank graph into (), which is not a well-formed formula. More importantly, Wermelinger reveals that Sowa's algorithm to translate CGs into FOL formulas is not sufficiently specified. Depending on the the order in which graphs in a given context are translated, the algorithm yields different formulas which are not semantically equivalent, but the algorithm does not impose such an order. This is a surely a more serious gap.

Finally we consider how Sowa translates contexts into formulas. As CGs go beyond FOL, Sowa writes that the Φ-operator 'is only defined for features [of CGs] that can be represented in predicate calculus' [Sow92]. As Sowa does not clearly specify what features he talks about, different formal elaborations of CGs in fact cover different features. For example, different kinds of quantifiers are addressed. More importantly is the handling of nestings on formal elaborations. In [Sow92], Sowa provides the following graph for the proposition 'Tom believes that Mary wants to marry a sailor' (this graph is more complicated than the first graph we provided for the same proposition, because Sowa explicates in this graph the linguistic roles of the verbs):



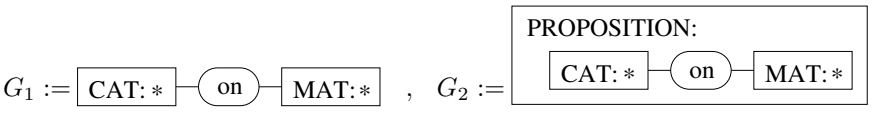
Afterwards, he translates this graph to the following formula:

$$\begin{array}{l}\exists x.Person(Tom) \wedge believe(x) \wedge expr(x, Tom) \wedge ptnt(x,\\ \quad \exists y.\exists z.PERSON(Mary) \wedge want(y) \wedge SITUATION(z) \wedge\\ \quad expr(y, Mary) \wedge ptnt(y, z) \wedge descr(z,\\ \quad\quad \exists u.\exists v.(marry(u) \wedge SAILOR(v) \wedge agnt(u, Mary) \wedge ptnt(u, v))))))\end{array}$$

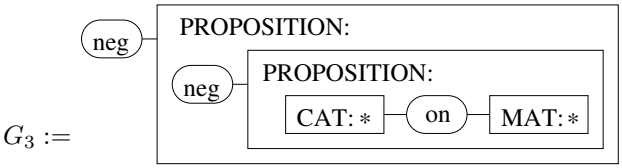
In this formula, whole formulas occur as arguments of relations, namely of *ptnt* (patient) and *descr* (description), which Sowa uses to express nestings in FOL. But using formulas as arguments in relations is a feature beyond FOL, so this translation cannot be considered as translation to FOL. This gap in handling nestings is fixed by different formal elaborations in different ways.

1.3.3 Sowa's Understanding of Negation

We have seen that in CGs, negations are implemented as special contexts, namely as contexts of type 'Proposition', linked to a relation oval labelled with 'neg'. Recall that contexts are used to draw meta-level propositions on CGs. Usually, negation is understood as a *logical* operator, and its properties have to be captured by the semantics and by any calculus. On the other hand, in CGs negation changes to a *meta-level* operator. This is stated quite explicit by Sowa in [Sowb]: 'To support FOL, the only necessary meta-level relation is *negation*.' Expressing negation as a meta-level operator might leads to difficulties in the formal treatment. To discuss this, we consider the following two CGs G_1 and G_2.



The meaning of the first graph is well known: It is 'a cat is on a mat'. The meaning of the second graph is, strictly speaking, 'there exists a proposition, which states that a cat is on a mat'. Quoting a proposition changes its character: There is a crucial difference between asserting and quoting a proposition, between using and mentioning a linguistic item. In particular these two graphs have different meanings. Now we consider the graph below:



G_3 contains a double negation. In the usual understanding of negation as a logical operator, G_1 and G_3 are equivalent. On the other hand, in the CG understanding, G_3 formalizes a statement on a statment (first nesting) on a statement (second nesting), i.e. we have a meta-meta-statement. These points of view are conflicting. For this reason, formal elaborations deviate from Sowa's meta-level understanding of negation and implement negation on the logical level.

1.3.4 Sowa's Calculus

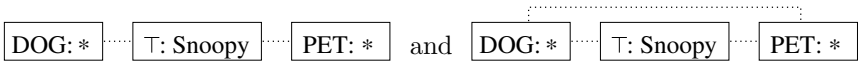
An example of Sowa's rules for reasoning with CGs has been given in Section 1.2.5. We have already seen that in different writings of Sowa, the set of rules is varied. Moreover, the rules are only informally described in common English. To obtain a precise and non-ambibous understanding on how the rules act on CGs, a formal definition for the rules is needed.

Sowa's rules are basically Peirce's rules for existential graphs, being adapted to the notations of CGs. Anyhow, first there are some crucial syntactical differences between existential graphs and CGs. In existential graphs, both existential quantification and identity are expressed by the same syntactical device, the so-called *line of identity*. In CGs, we have separate devices for these two logical functions: Existential quantification is expressed by the generic marker, and identity is expressed by coreference links. Moreover, in CGs, we have names for constants, a syntactical device which is missing in existential graphs. This leads to a higher expressiveness of CGs. The syntactical differences and the higher expressiveness have to be reflected by the calculus. To some extent, Sowa provides rules which cover these differences, but it is not clear whether these rules are sufficient. Sowa does neither prove the soundness nor the completeness of his set of rules for CGs.

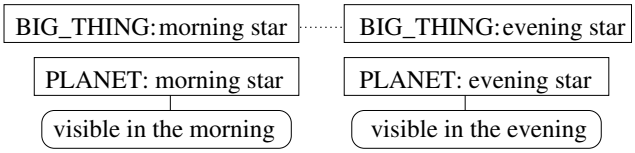
In fact, it may be doubted that Sowa's rules are complete. A proof that Sowa's rules are not complete cannot be provided due to missing formal definitions of the syntax, semantics and calculus for CGs. Anyhow, we provide a few examples of entailments between graphs which are unlikely to be mirrored by proofs with Sowa's rules. First of all, the calculus has to cover the special meaning of the universal type $\top$. For example, it should be allowed to derive $\boxed{\top : \text{Tom}}$, if Tom is a valid object name, and it should be possible to derive $\boxed{\top : *}$, too. This seems to be impossible applying the calculus of Sowa.

Next, it seems that coreference links are not adequately treated by the calculus. For example, if the following graph on the left is a well-formed CG (which is not clear from the definition of CGs), it should be provably equivalent to the CG on the right:

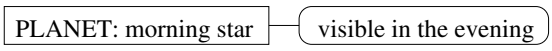
As identity is a transitive relation, the next two graphs should be provably equivalent, too:



Finally, given the graph



it should be possible to derive



It is difficult to decide from Sowa's definition of the calculus whether these derivations are possible. A formal elaboration of the calculus allows to address this question, and together with a formal elaboration of the syntax and the calculus, it is then possible to mathematically prove its soundness and completeness.

1.4 The General Approach for Formal Logic with CGs

Before we discuss different classes of CGs in more detail, this section provides a general overview how formal logic can be carried out by means of CGs, and introduces some core notations.

Roughly speaking, the scrutiny of this chapter follows the common (model-theoretic) layered approach of formal logic, which can be outlined by the terms syntax, semantics, and calculus. That is, first CGs are introduced as purely syntactical structures. Particularly, the concept boxes and relation ovals in graphs are labelled with *names* for concepts and relations. These syntactical structures carry per se no meaning: They gain their meaning when some sort of formal semantics is defined. Thus after defining the syntax, the semantics is formally elaborated as well. Finally reasoning facilities, which are sound and complete w.r.t. the semantics, are provided.

Following this approach is not the sole possibility to formalize CGs. Particularly Wille and some scholars are interested in CGs as an extension of FCA for allowing judgments and conclusions in a *semantical manner*. That is, he does not strictly separate syntax and semantics of CGs. In his approach, he does not assign names to the concept boxes and relation ovals of CGs, but already (formal) objects, (formal) concepts and relations of a given power context family instead. That is, each CGs is closely linked to a given power context family. Conclusions in Wille's approach are not drawn with the hep of inference rules of some adequate calculus, but via an algebraic investigation of the graphs. The approach following symbolic logic is called SYNTACTICAL or LOGICAL approach, the approach of Wille is called SEMANTICAL or ALGEBRAIC APPROACH. To some extent, the syntactical approach corresponds to the open world paradigm, whereas the semantical approach correspond to the closed world paradigm. This chapter focuses on the syntactical approach. For the semantical approach, please see Chapter **??**.

Even for the syntactical approach to CGs, unfortunately there exists a variety of different formalizations with different notations. This chapter provides some references to these different notations, but due to space limitations, it does not provide the complete definitions. Instead, only the most convenient among the different formalizations is given. A more comprehensive comparison of the different formal approaches to SCGs can be found in [JPA06].

1.4.1 Syntax

The starting point in fixing the syntax is defining a set of names. In literature, there are different terms used for the set of names. The CG standard [Sow99] refers to this set as a *type hierarchy*. Authors like Prediger, Klinger and Dau use the term *alphabet*. Other authors, eg Chein, Mugnier, Baget, Kerdiles, Simonet use *vocabulary* or *support* as well. Following the common notations of symbolic logic and knowledge representation, we will use the term VOCABULARY. Basically, a vocabulary consists of three ordered sets of the names for the objects, concepts (the types), and relations. Again, different authors use different notations here. The core notation of a vocabulary if fixed in the following definition.

DEFINITION 1.1 Vocabulary *A* VOCABULARY *is a triple* $\mathcal{V} := (\mathcal{O}, \mathcal{C}, \mathcal{R})$ *of finite and pairwise disjoint sets* $\mathcal{O}$*,* $\mathcal{C}$ *and* $\mathcal{R}$*. The elements of* $\mathcal{O}$ *are called* OBJECT NAMES *or* INDIVIDUAL MARKERS*. The elements of* $\mathcal{C}$ *are called* CONCEPT NAMES*. The elements of* $\mathcal{R}$ *are called* RELATION NAMES*.*

Besides the object names, we assume to have a further sign '$*$*', called the* GENERIC MARKER*. We assume to have a special concept name* $\top$*, called the* UNIVERSAL CONCEPT*. To each relation name* $R \in \mathcal{R}$*, we assign its* ARITY *ar*(R) *(*$ar(R) \in \{1, 2, 3, \ldots\}$*). We set* $\mathcal{R}_n := \{R \in \mathcal{R} \mid ar(R) = n\}$*.*

Finally, the sets of object, concept and relation names are ordered:

- *We have an order* $\leq_{\mathcal{O}}$ *on the set* $\mathcal{O} \cup \{*\}$*, with* $O \leq_{\mathcal{O}} *$ *for each* $O \in \mathcal{O}$*, and different elements of* $\mathcal{O}$ *are incomparable,*
- *We have an order* $\leq_{\mathcal{C}}$ *on the set* $\mathcal{C}$*, where* $\top$ *is the greatest element, and*
- *We have an order* $\leq_{\mathcal{R}}$ *on the set* $\mathcal{R}$*, where relation names with different arities are incomparable.*

'Conceptual graphs are also graphs' is the title of a 1995 report of Chein and Mugnier [CM95], nicely describing the fundamental approach to a formalization of CGs by means of mathematical graph theory. This approach has already been addressed by Sowa, who suggests to formalize CGs as labelled bipartite graphs (for this reason, Sowa sometimes speaks of concept *nodes* and relation *nodes*). The first formal definitions and theoretical results for SCGs have been provided in the landmark work [CM92]. In this approach, the underlying structure of a CG is a multi-bipartite graph (C, R, E), where C and R are disjoint sets of vertices, and E is a multiset, where each element $e \in E$ is a pair (c, r) with $c \in C$ and $r \in R$. Concept boxes of CGs correspond to concept vertices $c \in C$, relation ovals correspond to relation vertices $r \in R$, and the arcs of CGs correspond to edges $e \in E$. So each edge links a concept vertex (aka node) and a relation vertex (aka node). The bipartite graph (C, R, E) is augmented with additional mappings, which capture the full syntax of CGs. Labelling functions are provided which assign to each concept vertex a concept name (the type of the concept box) and an object name (the referent of the concept box), and to each relation vertex an relation name. The n edges incident to a given relation vertex r are labelled with $1, \ldots, n$ to indicate the order of the argument of the relation of r. Coreference links are modelled as a equivalence relation on C.

Another possibility is to formalize CGs as directed multi-hypergraphs. This approach has been first suggested by Wille in [Wil97]. In this approach, the underlying structure of CGs is a directed multi-hypergraph (V, E, ν), where V is a set of vertices, E is a set of edges, and $\nu : E \to \bigcup_{k \in \mathbb{N}_0} V^k$ is a mapping. Now concept boxes correspond to vertices $v \in V$, and relation ovals correspond to edges $e \in E$. Note that, in contrast to the multi-bipartite approach, in this approach the order of the arguments of a relation is already been formalized. To ease the notation, in later works the term *directed multi-hypergraph* has

been replaced by RELATIONAL GRAPH. Again, relational graphs have to be augmented with labelling functions to capture the object, concept and relation names concept boxes resp. relation ovals are labelled with.

Roughly speaking, the 'Montpellier-school', i.e. Chein and Mugnier and the scholars, follow the approach on bipartite graphs, and the 'Darmstadt-school', i.e. Wille and his scholars, follow the relational graph approach. As multi-bipartite graphs, including the labelling function which captures the order of the arguments for the relations, and relational graphs can be mutually transformed into each other, it is a mere matter of taste which of these two approaches is taken. Anyhow, Baget (from the Montpellier-school) slightly advocates in [Bag03] the relational graph approach for computational reasons. This chapter uses the relational graph approach.

It should be noted that the diagrams we presented in Section 1.2 are diagrammatic representations of an underlying abstract syntax. Already Sowa emphasizes in several places that the notation of CGs has to be independent from their diagrammatic representation. So, in contrast to symbolic notations of formal logic, we have two layers: A precisely defined *abstract syntax*, which is independent of any diagrammatic or topological properties of CGs, and the –informally given– diagrammatic representations of the abstract syntax. This might seem obvious, but is has to be acknowledged that some authors working on other versions of diagrammatic logic miss this important distinction. A thorough discussion of this issue can be found in [HMST02, Dau04].

To distinguish the more open computer-science based theory of CGs to the formal mathematical theory, Wille coined the term *concept graph* for the mathematizations of CGs. Although this distinction is sensible, the use of these two different terms might lead to confusion, and it is not adopted by the whole CG community. For this reason, the term *concept graph* is, despite for referencing, not used in this chapter.

Unfortunately, even if we consider only SCGs, nearly all works differ in significant details. Care has to be taken on how in different works the labelling functions for object names and concept names is modelled, and how and coreference links are implemented. Sometimes the universal type $\top$ is required in the vocabulary, sometimes it is not. Most works assign *one* object name or the generic marker as referent to concept vertices, but some authors (for example Klinger and Prediger) allow for *sets* of object names instead. Most works assign a *single* concept name to a concept vertex, but some works allow for a *set* of concept names, which is interpreted as the conjunction of the concept names (e.g, [Bag03, CM04]). Coreference links are by some authors modelled as equivalence relations (e.g. Chein and Mugnier, or Prediger), sometimes as special edges, labelled with a name for identity (e.g. [Dau03b, Dau03a, Bag99]). Often coreference is allowed only between *generic* concept vertices (e.g., Chein/Mugnier, Prediger), sometimes it is allowed between arbitrary concept vertices (e.g. Dau). Some of these differences do not change the expressiveness of the system which is considered (for example, for SCGs, it does not make a difference whether we allow only single object names

or sets of object names as referents), some of these difference can (slightly) change the expressiveness (for example, whether we allow coreference only between generic concept boxes or between arbitrary concept boxes).

1.4.2 Semantics

In literature, we basically find three kinds of semantics:

- The translation of CGs to FOL formulas by means of Φ.
- The evaluation of CGs in Tarski-style interpretations.
- The evaluation of CGs in power context families.

As mentioned in Section 1.2.4, Φ is the 'classical' semantics for CGs. To some extent, it is not a 'real' semantics, but a translation from some syntactical structures –i.e., graphs– to other syntactical structures –i.e., formulas–. As formulas of FOL have in turn a precisely defined semantics based on Tarski-style interpretations, Φ can be understood to provide an 'indirect' evaluation of CGs in Tarski-style interpretations. The semantics via Φ is sometimes called LOGICAL SEMANTICS [JPA06]. Unsurprisingly, some authors provide a *direct* evaluation of CG in Tarski-style interpretations, i.e., an EXTENSIONAL SEMANTICS. Finally, some authors (the authors of the Darmstadt-school) provide a called CONTEXTUAL SEMANTICS, where CGs are evaluated in power context families, which are contextual interpretation based on Formal Concept Analysis. For an introduction into FCA, please see Chapter **??**.

All three types of semantics have to reflect the orders of concept names and relations names of the given vocabulary, and the special meaning of the universal concept as well. In the following three paragraphs, let $C_1, C_2 \in \mathcal{C}$ be arbitary concept names with $C_1 \leq_{\mathcal{C}} C_2$ and $R_1, R_2 \in \mathcal{R}_n$ be arbitary relation names with $R_1 \leq_{\mathcal{R}} R_2$.

For Φ, Sowa proposed to translate the orders into FOL-formulas. For two concepts C_1, C_2, the corresponding formula is $\forall x : C_1(x) \rightarrow C_2(x)$. Similarly for the relation names R_1, R_2, the corresponding formula is $\forall x_1, \ldots \forall x_n : R_1(x_1, \ldots, x_n) \rightarrow R_2(x_1, \ldots, x_n)$. Finally, the meaning of $\top$ is reflected by $\forall x : \top(x)$. The (finite) set of these formulas is denoted $\Phi(\mathcal{V})$.

An EXTENSIONAL INTERPRETATION is a pair (D, I), where D is a non-empty set, the UNIVERSE (OF DISCOURSE), and I is an INTERPRETATION FUNCTION, which maps object names to elements of D, concept names to subsets of D, and relation names of arity n to subsets of D^n. The special properties of the vocabulary are covered by additionally requiring that I is order-preserving, i.e. for two C_1, C_2 it holds $I(C_1) \subseteq I(C_2)$, and similarly for R_1, R_2 it holds $I(R_1) \subseteq I(R_2)$. Moreover, we require that $I(\top) = D$ holds.

Analogously to extensional interpretations, a CONTEXTUAL INTERPRETATION is a pair $(\vec{\mathbb{K}}, \lambda)$, where now $\vec{\mathbb{K}} = (\mathbb{K}_0, \mathbb{K}_1, \ldots)$ is a power context family and λ is a mapping which maps object names to elements of G_0, concept

names to formal concepts of $\mathbb{K}_0$ and n-ary relation names to formal concepts of $\mathbb{K}_n$. Clearly, λ is the contextual counterpart of the extensional interpretation function I. Again, λ has to be order preserving, i.e. for C_1, C_2 it holds $\lambda(C_1) \leq \lambda(C_2)$, and for R_1, R_2 it holds $\lambda(R_1) \leq \lambda(R_2)$. Moreover, we require that $\top$ is mapped to the top-concept of $\mathbb{K}_0$.

Given a CG G and an extensional or contextual interpretation, G is then EVALUATED in the interpretation to true or false. A precise definition of the evaluation depends of the kind and definition of the graph and the kind of the interpretation and is omitted here due to space limitations. If G evaluates to true in the interpretation, the interpretation is called an (extensional resp. contextual) MODEL of G, and we write $(D, I) \models G$ resp. $(\vec{K}, \lambda) \models G$. Finally, if we have a set of CGs $\mathcal{G}$ and a CG G such that whenever we have an interpretation which is a model for all CGs of $\mathcal{G}$, then it is a model for G as well, we say that $\mathcal{G}$ (SEMANTICALLY) ENTAILS G, and we write $\mathcal{G} \models G$. If $\mathcal{G}$ consists only of a single graph H, we write $H \models G$.

Contextual interpretations take benefit only of the order of the formal concepts of the power context family, its lattice structure is not used. The order of formal concepts in turn is based on the set-theoretical inclusion of the *extents* of formal concepts. For this reason, it is possible to assign to each contextual interpretation $(\vec{K}, \lambda)$ an extensional interpretation (D, I) such that we have $(\vec{K}, \lambda) \models G \iff (D, I) \models G$. Vice versa, we can assign to each extensional interpretation (D, I) a contextual interpretation $(\vec{K}, \lambda)$ such that we again have $(D, I) \models G \iff (\vec{K}, \lambda) \models G$. In the light of this observation, extensional and contextual interpretations can be considered equivalent. The details of this transformation can be found in [Dau03b] or [JPA06].

1.4.3 Reasoning Facilities

In literature, we find different kinds of reasoning facilities for CGs. The most important ones are projections, and sets of rules based on graph transformations. A thorough discussion of these kinds of reasoning facilities goes beyond the scope of this chapter; only an overview will be provided. Anyhow, some more details will be given in the next section. For all the following reasoning facilities, in the papers or treatises where they are described it has been proven that they are sound and complete.

Projections are graph-homomorphisms which respect the orderings of the names the vertices (and edges, if the relational graph approach is taken) are labelled with.[3] They are applied to SCGs. Roughly speaking, the information in a SCG G is the collection of the atomar information in the singleton graphs and star graphs the graph G is composed of, thus if we have a projection $\Pi : G_1 \to G_2$, then each piece of information in G_1 can be found in G_2 as

[3] The entailment relation between RDF-graphs is similar to projections. See [Bag04, Bag05, Dau06] for papers where CG-results are transferred to RDF.

well. I.e. G_2 contains all the information of G_1 (maybe more), which in turn is equivalent to $G_2 \models G_1$. Anyhow, projections have to cope with syntactically different, but semantically equivalent graphs, and with coreference links. More information on projections can be found in Section 1.5.1.

A set of graph transformations is the form of reasoning which is closest to the calculi for symbolic formalizations of logic. A graph transformation is a rule which transforms a graph G_1 into a graph G_2, and a proof is simply a sequence of CGs, where each CG in the sequence is obtained from its predecessor by one of the transformations. We can roughly distinguish between *simple* and *complex* graph transformations. A simple transformation transforms G_1 into G_2 by modifying small parts of G_1. Examples are the erasure of an relation edge, the merging of two concept vertices which are coreferent (with some constraints for the concept names of these vertices), or the generalization of a object or concept name in a concept vertex resp. the generalization of a relation name in a relation edge. Complex graph transformations in turn modify a whole subgraph of a given CG. Examples are the erasure of a subgraph or the iteration (making a copy) of a subgraph. Complex transformations are in most cases inspired by Peirce's rules for existential graphs.

Roughly speaking, calculi for CGs which include full negation are based on the complete set of Peirce's rules, augmented with some simple graph transformation rules which are needed to cover the syntactical differences between existential graphs and CGs. The informally given calculus of Sowa is an example for such a calculus. A formally elaborated example is provided in [Dau03b]. Calculi for CGs without full negation mostly or even totally consist of simple graph transformations. Examples for SCGs can be found in [CM92, CM95, Pre98a, Pre98b, Mug00, Dau03a], for CGs which allow for negation of concepts and relations in [Kli05], and for nested CGs without negation in [Pre98a, Pre00].

Besides projections and graph transformations, other forms of reasoning facilities have been explored as well. First the notion of STANDARD MODELS has to be mentioned. The basic idea of standard models is to assign to a CG G a corresponding model $\mathcal{M}^G$ in which exactly the information represented by G is encoded. Once this is done, there are two possibilities to check whether a CG G_1 entails a CG G_2. First, we can check whether G_2 evaluates to true in the standard model $\mathcal{M}^{G_1}$ of G_1. The second is to check whether the information encoded in the standard model $\mathcal{M}^{G_1}$ of G_1 can be found in the standard model $\mathcal{M}^{G_2}$ of G_2. This can be verified by a meaning-preserving mapping from $\mathcal{M}^{G_2}$ to $\mathcal{M}^{G_1}$. This mapping is, roughly speaking, the semantical counterpart of the syntactical projection between G_2 and G_1. Standard models cannot be constructed for CGs which allow to express disjunction or full negation. They have been used for SCGs [Pre98a, Pre98b, Dau03a] and CGs which allow for negation of concepts and relations [Kli01, Kli02, Kli05].

Other approaches to employ reasoning facilities can be based on tableaux algorithms (see [Ker01] for an tableaux algorithm for CGs with full negation), or resolution-like calculi combined with projections [MS96, BS06].

1.5 Different Forms of Conceptual Graphs

This section provides an overview over different formalizations of CGs which encompass some sound and complete reasoning facilities. This overview is not intended to be comprehensive. Instead, a (subjective) selection of important works is provided.

1.5.1 Simple Conceptual Graphs

.

Probably the most prominent and best investigated fragment of CGs is the class of simple conceptual graphs (SCGs). Nonetheless, as have already been mentioned in Section 1.4.1, there does exist a variety of different formalizations and even different understandings of SCGs. Basically, a SCG is a CG without contexts, which – due to Sowa's modelling of negation as a special context– excludes (full) negation. Anyhow, there are at least two finer subdivisions of SCGs: We will distinguish between NON-EXISTENTIAL and EXISTENTIAL SCGs, that is SCGs without or with generic markers, and between SCGs without or with coreference links.

First we provide a possible formalization of SCGs by means of directed multi-hypergraphs, close to the definition in [Dau03a].

DEFINITION 1.2 Simple Conceptual Graphs *A* SIMPLE CONCEPTUAL GRAPH OVER $\mathcal{V}$ (WITHOUT COREFERENCE LINKS) *is a structure* $G := (V, E, \nu, \kappa, \rho)$ *where*

- V *and* E *are finite sets of* (CONCEPT) VERTICES *and* (RELATION) EDGES,
- $\nu : E \to \bigcup_{k\in\mathbb{N}} V^k$ *is a mapping (we write* $|e| = k$ *for* $\nu(e) \in V^k$*),*
- $\kappa : V \cup E \to \mathcal{C} \cup \mathcal{R}$ *is a mapping such that* $\kappa(V) \subseteq \mathcal{C}$*,* $\kappa(E) \subseteq \mathcal{R}$*, and all* $e \in E$ *with* $|e| = n$ *satisfy* $\kappa(e) \in \mathcal{R}_n$*, and*
- $\rho : V \to \mathcal{O} \cup \{*\}$ *is a mapping.*[4]

The vertices v *with* $\rho(v) = *$ *are called* GENERIC VERTICES*, the vertices* v *with* $\rho(v) \in \mathcal{C}$ *are called* OBJECT VERTICES*. We set* $V^* := \{v \in V \mid \rho(v) = *\}$.

A SIMPLE CONCEPTUAL GRAPH OVER $\mathcal{V}$ WITH COREFERENCE LINKS *is a structure* $G := (V, E, \nu, \kappa, \rho, \Theta)$*, where* $(V, E, \nu, \kappa, \rho)$ *is a simple conceptual graph over* $\mathcal{V}$ *and* Θ *is an equivalence relation on* V^**. Two vertices* $v, w \in V^*$ *with* $v\Theta w$ *are said to be* COREFERENT.

[4]The letter ρ is is intended to remind that ρ (rho) maps vertices to their referents.

Of course, the equivalence relation Θ can be canonically extended to V by additionally setting $v\Theta w$ for two non-generic vertices v, w with $\rho(v) = \rho(w)$ (in some works, i.e. [Mug00], the authors speak then of *coidentical* instead of coreferent vertices). Note that with this understanding, even in SCGs without coreference links, we can have different coreferent vertices, i.e. two object vertices labelled with the same object name.

Projections The notion of projections as a reasoning facility has already been proposed by Sowa in [Sow84]. In [CM92], it has been pointed out that projections are indeed graph-homomorphisms. They respect the orders of the names, as well as coreference links. For the graphs of Definition 1.2, they can be formally be defined as follows:

DEFINITION 1.3 Projection *Let $G_i := (V_i, E_i, \nu_i, \kappa_i, \rho_i, \Theta_i)$ $(i = 1, 2)$ be two SCGs. A* PROJECTION *from G_1 to G_2 is a mapping $\Pi : V_1 \cup E_1 \to V_2 \cup E_2$ which maps vertices to vertices and edges to edges, such that*

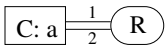
- $\nu_2(\Pi(e)) = (\Pi(v_1), \ldots, \Pi(v_n))$ *for each edge $e \in E_1$ with $\nu_1(e) = (v_1, \ldots, n_n)$,*
- $\kappa_2(\Pi(x)) \leq \kappa_1(x)$ *for each $x \in V_1 \cup E_1$,*
- $\rho_2(\Pi(v)) \leq \rho_1(v)$ *for each $x \in V_1$, and*
- $v\Theta_1 w \Rightarrow \Pi(v)\Theta_2\Pi(w)$ *for all $v, w \in V_1$*

Sowa argues that projection is sound, but does not prove its completeness. The soundness and completeness has been proven for the first time in [CM92]. But for the completeness, an additional constraint is needed. A SCG (without or with coreference links) is said to be in normal form, if no different concept vertices are coreferent. We then have for a vocabulary $\mathcal{V}$ and two SCGs G, H, where H is in normal form:

$$\Phi(\mathcal{V}), \Phi(H) \models \Phi(G) \quad \Longleftrightarrow \quad \text{there exists a projection } \Pi : G \to H \qquad (1.1)$$

So projection relies on the target graph being normalized (this restriction has not been pointed out in [CM92]). A simple example for normal forms is given below. H is the normal form of G. Both G and H have the same meaning, but there exists only a projection from G to H, but not vice versa.

$G :=$ 

$H :=$ 

A normal form of a graph can be easily computed by merging concept vertices which are coreferent. But this depends on whether it is possible to compute a concept name corresponding to the conjunction of the concept names of the vertices which are merged. This is not always possible. For this reason, in some works it is required that coreferent vertices are labelled with the same concept name, and other works consider SCG where vertices can be labelled with *sets* of concept names.

In [CM04], Chein and Mugnier consider existential SCGs with coreference links, and they overcome the need for normalforms in projections. To do so, the firstly allow concept vertices to be labelled with sets of concept names, which are interpreted as their conjunction. More importantly, they introduce the notion of COREF-PROJECTIONS. In their approach, for each vertex v they consider its COREFERENCE CLASS, that is set of all vertices which are coreferent to v. Coref-projections then map coreference classes to coreference classes instead of vertices to vertices.

Simonet investigates a different approach to eliminate the need for normalforms. In [Sim98, CMS98], she provides a new translation Ψ from CGs to FOL, which covers not only the meaning, but also the syntactical structures of CGs. With this new semantics, projection is sound and complete without requiring that the target graph is normalized.

Calculi Sowa provided in [Sow84] a set of simple transformation rules for SCGs, but this set is not complete [Mug00]. Several works have proposed different set of rules for SCGs. Prediger considers in [Pre98a, Pre98b] SCGs with a contextual semantics (in [Pre98b], she considers so-called *relational* power context families, the full contextual approach is provided in [Pre98a]). Prediger generally allows references with more than one object name. In [Pre98b], she considers non-existential SCGs without coreference links. For these graphs, a sound and complete calculus composed of ten simple graph transformations (except one rule which allows to make a copy of the whole graph) is provided. In her (German) PhD-thesis [Pre98a], this calculus is extended to existential SCGs with coreference links between generic concept vertices, which then has eleven rules.

Chein and Mugnier provide for existential SCGs, both for graphs without and with coreference links, in [CM95, Mug00] sets of sound and complete graph transformation rules. Similar to Prediger, coreference links only act on generic concept vertices. The calculus for SCGs without coreference links consists of five rules, called 'relation duplicate', 'unrestrict', 'detach', 'substract' (which corresponds to Peirce's erasure-rule), and 'join'. When coreference links are added, detach is replaced by a rule called 'co-identical split', and a new rule 'coreference deletion', which allows the erasure of coreference links, is added.

In [Dau03a] existential SCGs are considered, where coreference links are modelled by special relation edges labelled with '=' (a similar approach can be found in [Bag99], where the properties of the identity relation are modelled by special *rule graphs*, which are described in the next paragraph). In contrast to other approaches, coreference links between arbitrary concept vertices are allowed, even if both are non-generic. The set of rules he considers are exactly thse rules of his calculus for concept graphs with negation (see Section 1.5.4) that can be applied to CGs without negation. They are: Erasure, iteration, deiteration, generalization, isomorphism, exchanging references, merging two vertices, splitting a vertex, $\top$-erasure, $\top$-insertion, identity-erasure and identity-insertion. This calculus could probably be simplified.

An "if-then"-statement can not ad hoc be expressed with *one* SCG. Anyhow, Baget invented in [Bag99] SCGs where the set of vertices is additionally divided into HYPOTHESIS VERTICES and CONCLUSION VERTICES. Moreover, FRONTIER-VERTICES are those hypothesis vertices having a conclusion-vertex as neighbour. Now, if such a rule-graph G_R is given, and an graph G such that the hypothesis-part of G_R projects to G. then the conclusion-part of G_R can be added to G, where the frontier-vertices of the conclusion are merged with the corresponding vertices in the projection of the hypothesis. With this idea, a knowledge base of SCGs can contain not only facts stated by graphs, but if-then-rules as well (but as projection is involved, this approach still relies on the normalization of some graphs). Baget's idea can even be extended. For example, positive and negative constraints can be expressed with similarly augmented SCGs. This leads to a framework called *Constrained Derivation Model* [Mug00, BM02]. Within this model, already SCGs become a quite powerful knowledge representation language.[5]

Standard Models Standard models as a means for reasoning with CGs have been introduced by Prediger in [Pre98a, Pre98b]. In [Pre98b], she assigns to each non-existential SCG G without coreference links a standard model $\mathcal{M}^G$, which is a formal context (O^G, A^G, I^G) with an additional mapping λ^G which maps n-ary relation names to n-ary relations over O, and she proves

$$G_1 \models G_2 \text{ iff } G_2 \text{ is valid in } \mathcal{M}^{G_1} \tag{1.2}$$

$$G_1 \models G_2 \text{ iff } I^{G_1} \supseteq I^{G_2} \text{ and } \lambda^{G_1}_{\mathcal{R}}(R) \supseteq \lambda^{G_2}_{\mathcal{R}}(R) \text{ for all } R \in \mathcal{R} \tag{1.3}$$

The so-called RELATIONAL CONTEXTS of [Pre98b] are in [Pre98a] replaced by power context families. For existential SCGs with coreference links, Prediger proves in [Pre98a] the direction "$\Rightarrow$" of (1.2), but does not provide a counterpart of (1.3).

In [Dau03a] a translation into standard models is provided as well. It is proven that $G_1 \models G_2$ if and only if $\mathcal{M}^{G_1}$ entails $\mathcal{M}^{G_2}$. The entailment between $\mathcal{M}^{G_1}$ and $\mathcal{M}^{G_2}$ is expressed via the existence of a meaning-preserving mapping from $\mathcal{M}^{G_2}$ to $\mathcal{M}^{G_1}$, which is a sort of projection between models. Now, even in the existential case (with a slightly higher expressiveness of [Dau03a] compared to [Pre98a]), we have in [Dau03a] results which correspond to (1.2) and (1.3). So reasoning with SCGs fully carries over to reasoning with models. Consequently, Dau provides moreover a sound and complete set of four rules (removing an element, doubling an element, exchanging attributes, restricting the incidence relation) for entailment between models.

[5] Besides the already mentioned contributions, the papers show that finding a projection between SCGs is equivalent to the problem to the conjunctive-query containment known from relational databases, and the constraint-satisfaction-problem, thus linking reasoning with SCGs to other formalisms, and it is argued for CGs instead of formulas both from a knowledge modelling and a computational point of view.

Other approaches Based on the PhD-thesis of Salvat, approaches with resolution-style calculi (with implication as sole logical connective) are presented in [MS96, BS06]. Tappe investigates in [Tap00a, Tap00b] SCGs where the generic marker –which represents existential quantification– is replaced by the universal quantifier ∀. Tappe provides a contextual semantics and a sound and complete calculus.

1.5.2 Nested Conceptual Graphs

In contrast to SCGs, there does not exist a generally agreed upon semantics for nested CGs. Instead, different approaches exist. Some of them are presented in this section.

One critics given in Section 1.3 for Sowa's handling of nesting in his definition of the Φ-operator was his use of formulas as arguments in a predicate, as this goes beyond FOL. Preller et al [PMC98] have used this approach to define a non-classical logics with 'nested' formulas, which is similar to the logic of contexts of [McC93]. Then they define a Gentzen-style sequent formal system with nested formulas and prove the soundness and completeness of this system with respect to the projection in simple and nested graph models.

A different approach has been taken by Simonet in [Sim98, CMS98]. She inductively defines a nested CG by assigning to each concept vertex v an additional argument $Desc(v)$, where $Desc(v)$ is either the blank graph or a nested SCG. Then she provides a new Φ-operator for nested CGs by adding to each predicate a further argument called CONTEXT ARGUMENT, which denotes the context. So, n-ary relation names in nested CGs are translated to $n+1$-ary relation names in the corresponding FOL-formulas. Finally, she extends the notion of projection to these graphs. To obtain soundness and completeness of projection, the target graph has to be in normalform again, but the notion of normalforms has extended to cover nestings. First of all, each nesting of the target graph, understood as a SCG, has to be in normalform. Moreover, if v, v' are coreferent concept vertices, then $Desc(v')$ must be an *exact* copy $Desc(v)$, i.e., $Desc(v')$ is a copy $Desc(v)$ such that each generic vertex appearing in $Desc(v)$ is coreferent to its copy in $Desc(v')$. This is called STRONGLY NORMAL. Assumed that we two graphs G, H where H is a strongly normal nested CG, we have a corresponding result to (1.1). The notion of being strongly normal can be weakened to a notion of inductively defined k-normality, which is, roughly speaking, the strong normality condition up the a level k of the depth of nestings. We omit the details due to space limitations. Finally, similar to her new semantics Ψ for SCGs, she provides a corresponding semantics Ψ for nested CGs, where projection is sound and complete without any need for normalizing the target graph.

In [Pre00], which is basically an excerpt of her PhD-thesis [Pre98a], Prediger elaborates nested CGs similar to her approach to SCG. Her syntax is based on directed multi-hypergraphs, augmented with a mapping $\rho : V \to \mathfrak{P}(V)$ which models the nesting of graphs. First of all, she considers only non-

existential nested CGs. In contrast to her approach to SCGs, she now assigns a single object name as referent to each vertex. She equips these graphs with a situation-based contextual semantics, which is now based on *triadic* power context families. Again, similar as for SCGs, she is able to assign to each nested CG a corresponding standard model. Her result (1.2) for SCGs can be lifted to the nested case, but she does not provide a counterpart for (1.3). Finally, she provides a set of eight inference rules (double a vertex, delete an isolated, non-complex vertex, double an edge, delete an edge, exchange a concept name, exchange a relation name, join vertices with equal references in the same nesting, copy a sub concept graph into an equally referenced nesting) that is sound and complete.

1.5.3 Conceptual Graphs with Atomic Negation

Negation in CGs can be expressed on two levels. We can restrict negation to concept names and relation names only, or where can allow to negate whole subgraphs. The first kind of negation is called ATOMIC NEGATION and covered in this section, the second kind of negation is called FULL NEGATION and will be covered in the next section. These two levels lead to a significant difference in the expressiveness of the systems. When atomic negation is added to SCGs, the resulting system is still decidable, whereas full negation leads to the equivalence to full FOL, thus undecidability. Anyhow, if negation on the atomic level is implemented, to some extent it is possible to express disjunctive information. To provide an example from [Ker01]: Take the formulas $f_1 := P(a) \wedge R(a,b) \wedge R(b,c) \wedge \neg P(c)$ and $f_2 := \exists x \exists y : P(x) \wedge R(x,y) \wedge \neg P(y)$. Then f_1 entails f_2. To see this, we have to consider two cases: If $P(b)$ holds in an interpretation, then f_2 holds for $x := b$ and $y = c$, and if $\neg P(b)$ holds in an interpretation, then f_2 holds for $x := a$ and $y := b$. The conceptual graph formalism does not allow to explicitely express this kind of disjunctive information. The two approaches we present in this section have to cope with this problem, caused by the law of the excluded middle.

Kerdiles considers in [Ker01] the negation of relation names only. Syntactically, there are two approaches to implement atomic negation. It is either possible to augment a given vocabulary $\mathcal{V}$ by adding for each relation name $R \in \mathcal{R}$ a new relation name R^- to the vocabulary (which of course denotes the negation of R), or we can augment a SCG $G := (V, E, \nu, \kappa, \rho)$ with an additional function $sign : E \to \{+, -\}$, which divides the set of edges into positive and negative edges. These graphs are called POLARIZED. Obviously, both approaches work equally well and can mutually transformed into each other, which is even formally proven by Kerdiles in [Ker01]. Kerdiles extends the projections of SCGs to SCGs with atomic negation: In addition to the usual constraints, projection has to respect *sign* as well. As usual, the target graph must be normalized. Besides this requirement, an additional constraint is put on the source graph: It has to be DISCRIMINATED. This basically means that the graph is the juxtaposition of two graphs, where one graphs contains

only positive, and the other graph only negative relation edges. Not every SCG with atomic negation can be converted into an equivalent graph which is discriminated, thus Kerdiles considers only the fragment of graphs where this is possible. He then shows that for two SCGs with atomic negation G, H where G is in normal form and H is discriminated, that $G \models H$ holds iff there exists a projection from H to G.

The approach of Kerdiles has be extended in [LM06, ML07]. Similar to Kerdiles, Mugnier and Leclère consider polarized CGs, where only relation names can be negated (but as they argue, concept names can be replaced by unary relation names, so this does not lead to a loss of expressiveness). They consider three different logical frameworks for interpreting atomic negation: The framework where they rely on the closed world assumption CWA, the framework of classical logic (particularly, here the open world assumption OWA is employed), and the framework of intuitionistic logic (again based on OWA). For the framework of CWA, to check whether a polarized CG H projects to a polarized CG G can be reduced to check whether the positive part of H projects (in the classical understanding) to some sort of completition of G. For the framework of intuitionistic logic, they argue that projection precisely captures the underlying intuition of projection (recall that the law of excluded middle, which causes the problem we mentioned at the beginning of this section, does not hold in intuitionistic logic). Finally, for checking whether a polarized CG H projects to a polarized CG G in the classical logic setting, on has to check whether H projects to –maybe exponentially many– completitions of G. The first part of [LM06] elaborates this in detail for polarized CGs without coreference links. In the second part, coreference is added to polarized CGs by means of special edges. As identity is a relation which can be negated, they add a second kind of special edges which are used to express non-identity as well. Assuming that the unique name assumption hold, they then show that their results can be lifted to polarized graphs with identity and non-identity.

Klinger takes in [Kli01, Kli02, Kli05] a different approach. First of all, on the syntactical level, she assigns to each concept vertex two kinds of referents: a positive and a negative one (similar to Prediger, she allows sets of object names instead of single object names). To illustrate this, a simple example is provided below.

As it can be seen, Klinger uses variables instead of the generic marker for existential quantification. The referents on the lefthand side of the bar are the positive referents, the referents on the righthand side of the bar are the negative referents. The intuitive meaning of this graph is 'there exists a dog (namely x), Yoyo is not a dog, Tim is a man, there exists something (namely y) which is not a man, x likes Tim, and Yoyo does not like y.

First of all, Klinger equips her graphs with a contextual semantics. If one demands that for a formal concept (A, B), the extent of the negation of (A, B) is the set-theoretic complement $G \backslash A$ of A, then this set does not have to be the extent of a concept itself. For this reason, Klinger does not assign only formal concepts to concepts and relation names. Instead, in her papers [Kli01, Kli02], she considers contextual interpretations where *semi*-concepts of the contextual interpretations are assigned to the names of the vocabulary. In her PhD-thesis [Kli05], she assigns the more general *proto*-concepts to the names (each semiconcept is a protoconcept; see Chapter **??**).

In [Kli01], Klinger introduces so-called simple semiconcept graphs, which provide no means of quantification. Nonetheless, coreference can be expressed by relation edges labelled with the name '='. She provides the syntax and semantics of simple semiconcept graphs, and she assigns a standard model to each of them. [Kli02] is the extension of [Kli01] to the existential case (where existential quantification is expressed with variables, not with the generic marker). Besides the definitions and their discussion, no results are presented in [Kli02] and [Kli01].

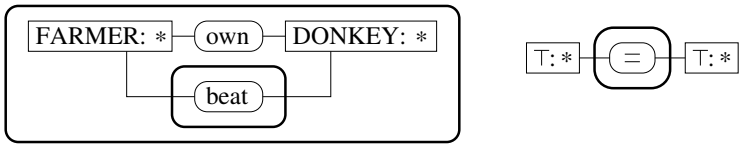
Here full elaboration of CGs with atomic negation can be found in her PhD-thesis [Kli05], where so-called (non-existential and existential) protoconcept graphs are fully elaborated. A first main concern of her is the satisfiability of graphs. As it is possible to express self-contradictory proposition with atomic negation, Klinger provides criteria when a protoconcept is satisfiable. For non-existential protoconcept graphs, standard models are introduced, and a result similar to (1.2) for these graphs is proven. The notion of standard models is dismissed for the existential case. Both for non-existential and existential protoconcept graphs, a sound and complete calculus is provided. For the non-existential case, it consists of 14 rules, that are besides one rule (iterating a subgraph) simple transformation rules. For the existential case, exactly these rules are adopted, and an additional rule, the "existential rule", is added. Unfortunately, this rule is quite complex: Its definition takes nearly three pages, and it is therefore hard to understand. As Klinger writes, the rule is needed in order to obtain the completeness for the existential case, but it 'certainly is not a very practical one'.

1.5.4 Conceptual Graphs with Full Negation

After discussing CGs with atomic negation, Kerdiles considers in [Ker01] full negation as well. He inductively defines CGs with negation, where each negated subgraph is a simple graph in normal form. coreference is still employed as an equivalence relation on the generic vertices, with some syntactical restrictions. Kerdiles did not want to use coreference edges as representation of identity, but simply to mark the multiple occurrences of a variable in traditional textual languages. So if one wants to express identity (or its negation), this should be in his system done by using additional relation edges labelled with '=' (personal communication). For this system of CGs with (a slightly

restricted) full negation, Kerdiles provides an extensional semantics, a translation of the graphs to FOL, and a sound and complete tableau algorithm.

A comprehensive approach to add full negation to CGs is provided by Dau in [Dau00, Dau01, Dau03b]. In [Dau00], Dau argues that to express full negation to CGs, a new syntactical entity to express negation has to be added, and he suggests to use the negation ovals of Peirce's existential graphs (which are called 'cuts' by Peirce for this purpose (recall the discussion of Section 1.3.3 – which is in fact taken from [Dau03b]– that negation should not be implemented as meta-level operation, which is syntactically expressed by means of special contexts). Moreover, he argues that for CGs with full negation, it is convenient to express identity by relation edges labelled with '=' (between arbitrary concept vertices). The resulting graphs are called *concept graph with cuts* (CGwCs). In the diagrammatic representation of CGwCs, to distinguish negation ovals from relation ovals, they are drawn bold. In contrast to the use of contexts, there is no reason to draw negation ovals only around 'complete' subgraphs. Below, two CGwCs are depicted. The left one is the CGwC-counterpart of Sowa's CG provided in Section 1.2.3, the right one is a CGwC expressing 'there are at least two things'.



Dau's approach is in [Dau00] carried out for non-existential CGwCs. A contextual semantics for CGwCs is provided, and a sound and complete calculus with twelve rules, based on Peirce's calculus for existential graphs, is given.

The full approach to both non-existential and existential CGwCs can be found in [Dau03b], [Dau01] is a rough overview over [Dau03b] without proofs. For CGwCs, [Dau03b] provides the syntax, both an extensional and contextual semantics, including mappings between these semantics, a sound and complete calculus, and translations from CGwCs to FOL (the Φ-operator) and vice versa, from FOL to CGwCs (called Ψ). The rules of the calculus are erasure, insertion, iteration, deiteration, double cuts, generalization, specialization, isomorphism, exchanging references, merging two vertices, splitting a vertex, $\top$-erasure, $\top$-insertion, identity-erasure, identity-insertion. The first five rules correspond to Peirce's rules for existential graphs. The rules generalization and specialization capture the order of the names. The rules $\top$-erasure and $\top$-insertion capture the special properties of the concept name $\top$. The rules exchanging references, identity-erasure, and identity-insertion capture some special properties of the relation name $=$, and the rules merging two vertices, splitting a vertex rely on both the properties of $\top$ and $=$.

1.6 Works not Cited

Conceptual graphs are a still a field of active research. This can be easily seen from this chapter, which basically cites research papers and several PhD-thesises. This section mentions some more important research works.

Both the Darmstadt-school and the Montpellier-school have generated a number of important PhD-thesises. The latter have not been cited yet, as they are due to the regulations in France published in french. Nonetheless, this section will give an rough overview over the (concerning the aim of this chapter) the most important ones. First of all, the Montpellier-school developped a CG-based reasoner called 'CoGITo'. The first version has been designed and implemented in C++ by Ollivier Haemmerle in his thesis 'Une plate-forme de développement de logiciels sur les graphes conceptuels' (1995).[6] He studied also the relationships between CGs and relational databases and developed a question/answering system. The thesis of Michel Leclère 'Les connaissances du niveau terminologique du modéle des graphes conceptuels: construction et exploitation' (1995) focuses on type definitions, i.e., contraction, expansion and classification of simple conceptual graphs with defined types. The thesis of Eric Salvat 'Raisonner avec des opérations de graphes: graphes conceptuels et règles de d'inférence' (1997) is devoted to the processing of 'if-then' rules. He defined and implemented in CoGITo sound and complete graph-based forward and backward chaining mechanisms for 'if SCG then SCG' rules. Finally, the thesis of Jean-Francois Baget 'Représenter des connaissanceset raisonner avec des hypergraphes: de la projection à la dérivation sous constraintes' (2001) contains a lot of results. For example, he defined algorithms for projection based on constraint processing techniques, and he defined a family of formalisms based on SCGs, rules and constraints and studied their complexity.

We already mentioned a pioneering paper of Wermelinger. To the best of my knowledge, he dedicated his MSc thesis 'Teoria Básica das Estruturas Conceptuais' ('Basic Conceptual Structures Theory', New University of Lisbon, Portugal, 1995) to a formalization of conceptual graphs. But unfortunately, the thesis is published in Portugese.

Heaton's english thesis 'Goal Driven Theorem Proving Using Conceptual Graphs and Peirce Logic' (Loughborough University of Technology, UK, 1994) focuses on conceptual graphs and existential graphs. He augments CG with constructs of Peirce's existential graphs, including Peirce's negation ovals. See http://myweb.tiscali.co.uk/openworld/index.html for an treatise summarizing Heaton's research on conceptual graphs so far.

Finally, I'd like to mention that in [BMT98, BMT99, Ker01], the decidable, so-called 'guarded fragment' of conceptual graphs is investigated.

[6] CoGITo has later been extended to typed nested CGs and be renamed CogiTaNT. See `http://cogitant.sourceforge.net/`

References

[Bag99] J.-F. Baget. A simulation of co-identity with rules in simple and nested graphs. In Tepfenhart and Cyre [TC99], pages 442–455.

[Bag03] J.-F. Baget. Hypergraphs and conjunctive types for efficient projection algorithms. In de Moor et al. [dMLG03], pages 229–241.

[Bag04] Jean-François Baget. Homomorphismes d'hypergraphes pour la subsumption en rdf/rdfs. *Langages et modèles à objets 2004 (actes 10e conférence), RSTI - L'objet (numéro spécial)*, 10(2-3):203–216, 2004.

[Bag05] Jean-François Baget. Rdf entailment as a graph homomorphism. In Yolanda Gil, Enrico Motta, V. Richard Benjamins, and Mark A. Musen, editors, *International Semantic Web Conference*, volume 3729 of *Lecture Notes in Computer Science*, pages 82–96. Springer, Berlin – Heidelberg – New York, 2005.

[BM02] J.-F. Baget and M.-L. Mugnier. Extensions of Simple Conceptual Graphs: The Complexity of Rules and Constraints. *JAIR*, 16:425–465, 2002.

[BMT98] F. Baader, R. Molitor, and S. Tobies. The guarded fragment of conceptual graphs. RWTH LTCS-Report. See also [BMT99]. Available at: `http://lat.inf.tu-dresden.de`, 1998.

[BMT99] F. Baader, R. Molitor, and S. Tobies. Tractable and decidable fragments of conceptual graphs. In Tepfenhart and Cyre [TC99], pages 480–493. Excerpt of [BMT98].

[BS06] J.-F. Baget and E. Salvat. Rule dependencies in backward chaining of conceptual graph rules. In Øhrstrøm et al. [ØSH06], pages 102–116.

[CM92] M. Chein and M.-L. Mugnier. Conceptual Graphs: Fundamental Notions. *Revue d'Intelligence Artificielle*, 6(4):365–406, 1992.

[CM95] M. Chein and M.-L. Mugnier. Conceptual graphs are also graphs. Technical report, LIRMM, Université Montpellier II, 1995. Rapport de Recherche 95003.

[CM04] Michel Chein and Marie-Laure Mugnier. Concept types and coreference in simple conceptual graphs. In Wolff et al. [WPD04], pages 303–318.

0-8493-0052-5/00/$0.00+$.50

[CMS98] M. Chein, M.-L. Mugnier, and G. Simonet. Nested Graphs: A Graph-based Knowledge Representation Model with FOL Semantics. In *Proceedings of the Sixth International Conference on Principles of Knowledge Representation and Reasoning (KR'98)*. Morgan Kaufmann, 1998. Revised version available at http://www.lirmm.fr/~mugnier/.

[Dau00] Frithjof Dau. Negations in simple concept graphs. In Ganter and Mineau [GM00], pages 263–276.

[Dau01] Frithjof Dau. Concept graphs and predicate logic. In Delugach and Stumme [DS01], pages 72–86.

[Dau03a] Frithjof Dau. Concept graphs without negations: Standardmodels and standardgraphs. In de Moor et al. [dMLG03], pages 243–256. This paper is a part of [Dau03b] as well.

[Dau03b] Frithjof Dau. *The Logic System of Concept Graphs with Negations and its Relationship to Predicate Logic*, volume 2892 of *LNAI*. Springer, Berlin – Heidelberg – New York, November 2003.

[Dau04] Frithjof Dau. Types and tokens for logic with diagrams: A mathematical approach. In Wolff et al. [WPD04], pages 62–93.

[Dau06] Frithjof Dau. Rdf as graph-based, diagrammatic logic: Syntax, semantics, calculus, normalforms. In F. Esposito, Z.W. Ras, D. Malerba, and G Semeraro, editors, *Foundations of Intelligent Systems*, volume 4203 of *Lecture Notes in Computer Science*. Springer, Berlin – Heidelberg – New York, 2006.

[dMLG03] Aldo de Moor, Wilfried Lex, and Berhard Ganter, editors. *Conceptual Structures for Knowledge Creation and Communication*, volume 2746 of *LNAI*. Springer, Berlin – Heidelberg – New York, 2003.

[DS01] Harry S Delugach and Gerd Stumme, editors. *Conceptual Structures: Broadening the Base*, volume 2120 of *LNAI*, Stanford, USA, July,, 2001. Springer, Berlin – Heidelberg – New York.

[GM00] Bernhard Ganter and Guy W. Mineau, editors. *Conceptual Structures: Logical, Linguistic and Computational Issues*, volume 1867 of *LNAI*, Darmstadt, Germany, 2000. Springer, Berlin – Heidelberg – New York.

[HMST02] J. Howse, F. Molina, Sun-Joo Shin, and J. Taylor. On diagram tokens and types. In Mary Hegarty, Bernd Meyer, and N. Hari Narayanan, editors, *Diagrams*, volume 2317 of *Lecture Notes in Computer Science*, pages 146–160. Springer, Berlin – Heidelberg – New York, 2002.

[JPA06] Michel Chein Jean-Pierre Aubert, Jean-Francois Baget. Simple conceptual graphs and simple concept graphs. In Øhrstrøm et al. [ØSH06], pages 87–101.

[Ker99] G. N. Kerdiles. Dynamic semantics for conceptual graphs. In Tepfenhart and Cyre [TC99], pages 494–507.

[Ker01] G. N. Kerdiles. *Saying it with Pictures: A Logical Landscape of Conceptual Graphs*, volume DS 2001-09. ILLC Dissertation Series, 2001.

[Kli01] J. Klinger. Simple semiconcept graphs: A boolean logic approach. In Delugach and Stumme [DS01].

[Kli02] J. Klinger. Semiconcept graphs with variables. In Uta Priss, Dan Corbett, and Galia Angelova, editors, *Conceptual Structures: Integration and Interfaces*, volume 2393 of *LNAI*, Borovets, Bulgaria, July, 15–19, 2002. Springer, Berlin – Heidelberg – New York.

[Kli05] J. Klinger. *The Logic System of Protoconcept Graphs*. Shaker Verlag, Aachen, 2005. Dissertation, Darmstadt University of Technology.

[LM06] Michel Leclère and Marie-Laure Mugnier. Simple conceptual graphs with atomic negation and difference. In Øhrstrøm et al. [ØSH06], pages 331–345.

[MC98] Marie-Laure Mugnier and Michel Chein, editors. *Conceptual Structures: Theory, Tools and Applications, 6th International Conference on Conceptual Structures, ICCS '98, Montpellier, France, August 10-12, 1998, Proceedings*, volume 1453 of *LNAI*. Springer, Berlin – Heidelberg – New York, 1998.

[McC93] J. McCarthy. Notes on formalizing context. In *IJCAI-93: 13th International Joint Conference on Artificial Intelligence*, pages 555–560. Morgan Kaufmann, 1993.

[ML07] Marie-Laure Mugnier and Michel Leclère. On querying simple conceptual graphs with negation. *Data Knowl. Eng.*, 60(3):468–493, 2007.

[MS96] M.-L. Mugnier and E. Salvat. Sound and complete forward and backward chaining of conceptual graph rules. In P. W. Eklund, G. Ellis, and G. Mann, editors, *Conceptual Structures: Knowledge Representation as Interlingua*, volume 1115 of *LNAI*, pages 248–262. Springer, Berlin – Heidelberg – New York, 1996.

[Mug00] M.-L. Mugnier. Knowledge representation and reasonings based on graph homomophism. In Ganter and Mineau [GM00], pages 172–192.

[ØSH06] Peter Øhrstrøm, Henrik Schärfe, and Pascal Hitzler, editors. *Conceptual Structures: Inspiration and Application*, volume 4068 of *Lecture Notes in Computer Science*. Springer, Berlin – Heidelberg – New York, 2006.

[PMC98] A. Preller, M.-L. Mugnier, and M. Chein. Logic for Nested Graphs. *Computational Intelligence: An International Journal*, 14-3:335–357, 1998.

[Pre98a] Susanne Prediger. *Kontextuelle Urteilslogik mit Begriffsgraphen – Ein Beitrag zur Restrukturierung der Mathematischen Logik*. Shaker Verlag, Aachen, 1998. Dissertation, Darmstadt University of Technology.

[Pre98b] Susanne Prediger. Simple concept graphs: A logic approach. In Mugnier and Chein [MC98], pages 225–239.

[Pre00] Susanne Prediger. Nested concept graphs and triadic power context families: A situation–based contextual approach. In Ganter and Mineau [GM00], pages 263–276.

[Rob73] Don D. Roberts. *The Existential Graphs of Charles S. Peirce*. Mouton, The Hague, Paris, 1973.

[Shi02] Sun-Joo Shin. *The Iconic Logic of Peirce's Graphs*. Bradford Book, Massachusetts, 2002.

[Sim98] G. Simonet. Two fol-semantics for simple and nested conceptual graphs. In Mugnier and Chein [MC98], pages 240–254.

[Sowa] John F. Sowa. Conceptual graphs: Draft proposed american national standard. Old version: `http://www.jfsowa.com/cg/cgdpansw.htm`, New version: `http://www.jfsowa.com/cg/cgstandw.htm`. See also [Sow99].

[Sowb] John F. Sowa. Semantic foundations of contexts. This paper is a revised merger of [**?**] and [**?**]. Available at: `http://www.jfsowa.com/ontology/contexts.htm`.

[Sow84] John F. Sowa. *Conceptual structures: information processing in mind and machine*. Addison-Wesley, Reading, Mass., 1984.

[Sow92] John F. Sowa. Conceptual graphs summary. In T. E. Nagle, J. A. Nagle, L. L. Gerholz, and P. W. Eklund, editors, *Conceptual Structures: current research and practice*, pages 3–51. Ellis Horwood, 1992.

[Sow97] John F. Sowa. Logic: Graphical and algebraic. manuscript, Croton-on-Hudson, 1997.

[Sow99] John F. Sowa. Conceptual graphs: Draft proposed american national standard. In Tepfenhart and Cyre [TC99], pages 1–65. See also [Sowa].

[Sow00] John F. Sowa. *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Brooks Cole, Pacific Grove, CA, 2000.

[Tap00a] J. Tappe. Simple concept graphs with universal quantifiers. Master's thesis, Darmstadt University of Technology, 2000.

[Tap00b] J. Tappe. Simple concept graphs with universal quantifiers. In Gerd Stumme, editor, *Working with Conceptual Structures. Contributions to ICCS 2000*, pages 95–104. Shaker Verlag, Aachen, 2000.

[TC99] Wiliam Tepfenhart and W. Cyre, editors. *Conceptual Structures: Standards and Practices*, volume 1640 of *LNAI*. Springer, Berlin – Heidelberg – New York, 1999.

[Wer95] M. Wermelinger. Conceptual graphs and first-order logic. In G. Ellis, R. Levinson, W. Rich, and J. F. Sowa, editors, *Conceptual Structures: Applications, Implementation, and Theory*, volume 954 of *LNAI*, pages 323–337. Springer, Berlin – Heidelberg – New York, 1995.

[Wil97] Rudolf Wille. Conceptual graphs and formal concept analysis. In D. Lukose, editor, *Conceptual Structures: Fulfilling Peirce's Dream*, volume 1257 of *LNAI*, pages 290–303. Springer, Berlin – Heidelberg – New York, 1997.

[WPD04] Karl Erich Wolff, Heather D. Pfeiffer, and Harry S. Delugach, editors. *Conceptual Structures at Work: 12th International Conference on Conceptual Structures, ICCS 2004, Huntsville, AL, USA, July 19-23, 2004. Proceedings*, volume 3127 of *Lecture Notes in Computer Science*. Springer, Berlin – Heidelberg – New York, 2004.

[Zem64] Jay J Zeman. *The Graphical Logic of C. S. Peirce*. PhD thesis, University of Chicago, 1964. Available at: `http://www.clas.ufl.edu/users/jzeman/`.