Concept Graphs with Cuts as Diagrammatic First Order Logic

Frithjof Dau
Darmstadt Technical University, Darmstadt, Germany
dau@mathematik.tu-darmstadt.de

Abstract

In the research field of diagrammatic reasoning, there are some attempts for providing diagrammatic forms of logic. Probably the most important one is Sowa's system of conceptual graphs from which Sowa claims that they have at least the expressiveness of first order predicate logic (FOPL). But a closer observation shows that their definitions lack (mathematical) preciseness, which yields several ambiguities, gaps and flaws.

In my dissertation [6], I elaborated fragment of conceptual graphs, having the expressiveness of of first order predicate logic (FOPL), in a mathematical manner. The resulting graphs are called concept graphs with cuts (CGwCs). In this paper, a short overview for the crucial definitions and the main results of [6] is provided. This includes the syntax, semantics and a calculus for CGwCs. Moreover, mappings between CGwCs and the linear notion of FOPL are given. It turns out that the calculus is sound and complete, and the mappings can be understood in a precise manner to be translations between CGwCs and the linear form of FOPL.

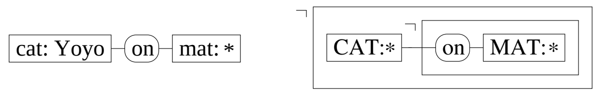
Keywords: Diagrammatic Reasoning, First Order Logic, Concept Graphs with Cuts, Conceptual Graphs

1 Introduction and Overview

In human reasoning, diagrams play an essential role. Complex information or even argumentation can often better be conveyed by diagrams than by any linear notions, like language or formulas. The research field of *diagrammatic reasoning* investigates all forms of reasoning and argumentation wherever diagrams are involved. This research area is constituted from multiple disciplines, including cognitive science and psychology as well as computer science, artificial intelligence, logic and mathematics.

Many aspects of diagrammatic reasoning can –and should– be formalized. This includes particularly cognitive and computational aspects of diagram processing as well as issues in formalizing diagrammatic forms of logic. It should not be overlooked that there has until today been a longstanding prejudice against non-symbolic representation in logic and mathematics. Nonetheless, there are attempts for providing diagrammatic forms of logic.

A very important one is Sowa's system of conceptual graphs, which are based on Peirce's existential graphs and the semantic networks of artificial intelligence. Their purpose is 'to express meaning in a form that is logically precise, humanly readable, and computationally tractable' ([23]). In fact, conceptual graphs yield a powerful diagrammatic system from which Sowa claims that they have at least the expressiveness of first order predicate logic (FOPL). Let us consider two simple examples for concept(ual) graphs (where $\mathfrak{G}_1$ will be used to denote the left and $\mathfrak{G}_2$ to denote the right graph):



$\mathfrak{G}_1$ is a simple conceptual graph. It is composed of so-called *concept boxes* and *relation ovals*. The concept boxes contain a *type* and a *referent* which belongs to the type. The types are usually assumed to be ordered, i.e., we have a so-called *type hierarchy* as underlying alphabet for conceptual graphs. The star '$*$' is a special referent called *generic marker*. It can be understood as an object which is not further specified (similar to a variable in first order logic which is existentially quantified, or to a wild card in computer systems). Besides the generic marker, object names are allowed as referents as well. The relation ovals between concept boxes represent relations between the referents of the concept boxes. Thus, the meaning of the left graph is 'the cat Yoyo is on a mat'. $\mathfrak{G}_2$ is a conceptual graph with negation contexts. They are adopted from Peirce's existential graphs and are used to negate the enclosed subgraph. Thus, the meaning of the right graph is 'it is not true that there is a cat for which is it not true that there is a mat such that that cat is on that mat', i.e., 'every cat is on a mat'.

The system of conceptual graphs has no sharp borders. 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 or linguistic nature. This lead to different modifications and extensions of conceptual graphs, which in turn lead to several difficulties and fallacies, ranging from lacks of preciseness and ambiguities over minor gaps to major mistakes and contradictions, in and between different notations or implementations of conceptual graphs.

Chein/Mugnier et al. fix some of the flaws by providing a *mathematical* theory of some fragment of conceptual graphs. In several works ([2, 3, 4, 5]), they develop a mathematical syntax for conceptual graphs based on mathematical graph theory, provide a semantics by translating these graphs to formulas of FOPL, and they provide an syntactical entailment relation between graphs based on information-preserving mappings between graphs, called *projections*, between graphs. Particularly, they investigate computational aspects for their fragment. But their approach lacks the possibility to express negation. Wermelinger discusses in [26] several flaws and mistakes in Sowa's Φ-operator (he says '*that Sowa's original definition of the mapping (Φ) is incomplete, incorrect, inconsistent, and unintuitive, and the proof system is incomplete, too.*'), which is intended to map, i.e. translate, conceptual graphs into FOPL-formulas. He amends this flaws and provides a mathematical definition of the Φ-operator which maps nested conceptual graphs into formulas of higher-order logic. Gwen Kerdiles investigates in [15] several fragements of conceptual graphs and their expressiveness and their computational aspects, and he combines the projections of Chein/Mugnier and the well-known tableaux methods of logic to obtain a complete calculus for conceputal graphs with negation. Simonet provides in [21] a translation of conceptual graphs, including nestings, but without negation, to FOPL-formulas. Baader et al. elaborate in [1] a fragment of conceptual graphs which corresponds to the guarded fragment of FOPL, and they provide a translation of these graphs to FOPL-formulas. In all these approaches, no direct, extensional semantics for conceptual graphs is provided. Instead, the semantics is indirectly given by translating the graphs to FOPL-formulas. Moreover, in none of these approaches, Peirce's powerful rules for existential graphs are adopted to obtain an adaquate calculus for conceptual graphs. Prediger provides in [19] a direct extensional semantics for conceptual graphs, including nestings, but without negation. She provides a calculus as well, but as she did not include negation into her approach, her rules are much simpler than the rules of Peirce for existential graphs. In [16], Klinger provides an extension of [19], where negation can be expressed for atomar graphs. This fragment of conceptual graphs is striclty weaker than FOPL and still decidable.

The attempt of [6] is to mathematically elaborate a fragment of conceptual graphs which is equivalent to full FOPL. This includes a definition of the syntax (the 'well-formed graphs'), a direct semantics, an adaquate calculus based on Peirce's calculus for existential graphs, and translations in both directions between graphs and formulas of FOPL. Similar to the treatises of Prediger and Klinger, this work is located within Wille's *Contextual Logic*, see [27, 29]). Wille, who is like Sowa strongly influenced by the philosophy of Peirce, introduced in [28] an approach to a mathematical elaboration of traditional, philosophical logic, based on the doctrines of concepts, judgements, and conclusions. In order to do so, he combined Sowa's graphs and his own theory of Formal Concept Analysis (FCA). The resulting graphs are called *concept graphs*. Contextual logic aims to reach 'at least the expressibility of first order predicate logic' (see [29]). To include the full negation of FOPL in the system of concept graphs, in [6], the so-called *cuts* of Peirce's existential graphs are added as new syntactial element to concept graphs. The resulting graphs are thus called *concept graphs with cuts* (CGwCs). CGwCs fulfill Wille's aim that concept graphs can have the expressivness of FOPL.

The aim of this paper is to provide an overview of the main definitions and results of [6]. Due to space limitations, it is impossible to provide all definitions, or even proofs of the following theorems, in this paper. For this reason, the paper is organised as follows: In the first section, in order to provide an idea of the underlying mathematical theory of CGwCs, the necessary mathematical definition for the syntax of CGwCs are given. In the following sections, mathematical notations are avoided as much as possible. Instead of that, I try to provide the main definition in a more informal manner. All formal definitions and proofs can be found in [6].

2 Basic Definitions for Concept Graphs

As discussed in the introduction, a drawback of conceptual graphs is a lack of mathematical preciseness, which leads to ambiguities and flaws. The purpose of CGwCs is to fix fix these flaws by elaborating mathematically a diagrammatic system of FOPL. This elaboration will be done as usual in mathematical logic, that is: We have to provide a syntax for CGwCs, a semantics, and a calculus which is sound and complete. Particularly, syntax, semantics, and the calculus have to be defined mathematically. In this section, we start with the definition of the syntax, i.e., the well-formed graphs.

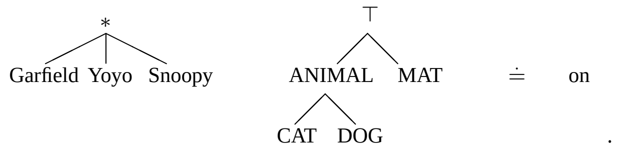
As usual in mathematical logic, we have first to define the underlying alphabet for our graphs. As we have concept boxes with types and referents in as well as relation ovals with names for relations, we distinguish in our aphabet between names for objects, names for types and names for relations. For conceptual graphs, it is usually assumed that the names for types and relations are ordered. We have a special type $\top$, which will be interpreted to have all elements of the the respectice domain of discourse in its extension. Moreover, we assume to have a dyadic relation name $\doteq$ representing identity in our alphabet.

Definition 2.1 (Alphabet) *A triple $\mathcal{A} := (\mathcal{G}, \mathcal{C}, \mathcal{R})$ of finite, disjoint sets $\mathcal{G}$, $\mathcal{C}$, $\mathcal{R}$ is an* alphabet, *if:*

- *$\mathcal{G}$ is a set. The elements of $\mathcal{G}$ are called* object names,[1]
- *$(\mathcal{C}, \leq_{\mathcal{C}})$ is a ordered set with a greatest element $\top$. Its elements are called* concept names.
- *$(\mathcal{R}, \leq_{\mathcal{R}})$ is a family of ordered sets $(\mathcal{R}_k, \leq_{\mathcal{R}_k})$, $k = 1, \ldots, n$ (for an $n \in \mathbb{N}$). Its elements are called* relation names. *Let $\doteq \in \mathcal{R}_2$ be a special name which is called* identity.

On $\mathcal{G} \dot\cup \{*\}$ we define an order $\leq_{\mathcal{G}}$ such that $*$ is the greatest element, but all elements of $\mathcal{G}$ are incomparable.

Below an example for an alphabet is provided, where the set $\mathcal{G}$, $\mathcal{C}$, $\mathcal{R}_2$ are depicted from left to right.



Next, the underlying structures of concept graphs with cuts will be defined. The provided examples of conceptual graphs show that these graphs are "networks" of boxes and relation ovals. It is convenient to formalize the concept boxes as vertices and the relation ovals as (multi- and hyper-) edges of a mathematical graphs. As we want to elaborate a fragment of conceptual graphs with the expressiveness of FOPL, we need a syntactical element which allows to express negation. In conceptual graphs, negation is usually formalized as by negation boxes, which are a special type of so-called *nestings* in conceptual graphs. The approach here is slightly different: As negation is a special logical operator, a unique syntactical element to express negation is added to the syntax. For this purpose, we adopt the approach of Peirce for existenial graphs, that is, we add the Peirce's so-called *cuts* to our formalization. The boxes and relation ovals are "grouped" by cuts, i.e., cuts contain boxes and relation ovals. Mathematically, we will asign to each c a set $area(c)$ contains the vertices (boxes), edges (relation ovals) and cuts which are directly contained by a cut c. We have to add some restrictions to the mapping $area$. For example, we have seen above that cuts must not overlap. Finally, for the further treatment of the graphs, it is convenient to add the so-called *sheet of assertion*, i.e., the plane where the diagram is written on, as a further element. It is mathematically modelled by a single element which is named *sheet of assertion* as well. Now we can define the underlying struture for the graphs as follows:

[1]The letter $\mathcal{G}$ stands for the German word 'Gegenstände', i.e., 'objects'. This letter will recur when we define formal contexts where we have a set G of objects.

Definition 2.2 (Relational Graph with Cuts) *A* relational graph with cuts *is a structure $(V, E, \nu, \top, Cut, area)$ where*

1. *V, E and Cut are pairwise disjoint, finite sets whose elements are called* vertices, edges *and* cuts, *resp.,*
2. *$\nu : E \to \bigcup_{k \in \mathbb{N}} V^k$ is a mapping*[2],
3. *$\top$ is a single element with $\top \notin V \cup E \cup Cut$, called the* sheet of assertion, *and*
4. *$area : Cut \dot\cup \{\top\} \to \mathfrak{P}(V \cup E \cup Cut)$ is a mapping such that*[3] *$area(c_1) \cap area(c_2) = \emptyset$ for $c_1, c_2 \in Cut \dot\cup \{\top\}$ with $c_1 \neq c_2$, $V \cup E \cup Cut = \bigcup_{d \in Cut \cup \{\top\}} area(d)$, and $c \notin area^n(c)$ for each $c \in Cut \dot\cup \{\top\}$ and $n \in \mathbb{N}$ (with $area^0(c) := \{c\}$ and $area^{n+1}(c) := \bigcup\{area(d) \mid d \in area^n(c)\}$).*

For an edge $e \in E$ with $\nu(e) = (v_1, \ldots, v_k)$ we set $|e| := k$ and $\nu(e)|_i := v_i$. Sometimes, we will write $e|_i$ instead of $\nu(e)|_i$, and $e = (v_1, \ldots, v_k)$ instead of $\nu(e) = (v_1, \ldots, v_k)$. We set $E^{(k)} := \{e \in E \mid |e| = k\}$. The elements of $Cut \dot\cup \{\top\}$ are called contexts. *As for every $x \in V \cup E \cup Cut$ we have exactly one context $c \in Cut \dot\cup \{\top\}$ with $x \in area(c)$, we can write $c = cut(x)$ for every $x \in area(c)$.*

Now we can obtain concept graphs from relational graphs by labelling the vertices and edges with names.

Definition 2.3 (Concept Graphs with Cuts) *A structure $\mathfrak{G} := (V, E, \nu, \top, Cut, area, \kappa, \rho)$ is called* concept graph with cuts over *$\mathcal{A}$ if*

- *$(V, E, \nu, \top, Cut, area)$ is a relational graph with cuts,*
- *$\kappa : V \cup E \to \mathcal{C} \cup \mathcal{R}$ is a mapping such that $\kappa(V) \subseteq \mathcal{C}$, $\kappa(E) \subseteq \mathcal{R}$, and all $e \in E$ with $|e| = k$ satisfy $\kappa(e) \in \mathcal{R}_k$, and*
- *$\rho : V \to \mathcal{G} \dot\cup \{*\} \dot\cup \{?i \mid i \in \mathbb{N}\}$ is a mapping.*

For the set E of edges, let $E^{id} := \{e \in E \mid \kappa(e) = \doteq\}$ and $E^{nonid} := \{e \in E \mid \kappa(e) \neq \doteq\}$. The elements of E^{id} are called identity-links.

[2]We set $\mathbb{N} := \{1, 2, 3, \ldots\}$ and $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$.

[3]The sign $\dot\cup$ denotes the *disjoint* union.

In the following, we will usually consider a fixed alphabet, therefore the alphabet will often be not mentioned. The system of all CGwCs (over a given alphabet) will be called CG.

Defs. 2.2 and 2.3 are abstract definitions of graphs which does not capture any graphical properties of the diagrams. Instead of that, the diagrams have to be understood as graphical *representations* of the graphs (a discussion of the distinction between graphs and their representations can be found in [7, 14]). The graphical representation of CGwCs has to be established by drawing conventions. They are provided in [6]. For this paper, the representations of CGwCs can be obtained from the provided examples. For example, we see that the vertices are drawn as rectangles, and inside the rectangle for a vertex v, we write first the concept name $\kappa(v)$ and then the referent $\rho(v)$, separated by a colon. This graphical notation is used in continuous text, too. e.g. we will write 'let $v := \boxed{P : g}$' instead of 'let v be a vertex with $\kappa(v) = P \in \mathcal{C}$ and $\rho(v) = g \in \mathcal{G}$'. Finally, cuts are drawn as bold ovals (instead of boxes with an attached begation sign '$\neg$').

In contrast to usual definitions of the syntax of a formal logic, the definition of well-formed graphs is not carried out inductively. This is mainly done for two reasons: First of all, an inductive definition is the canonical approach for linear notions, but there is in fact no canonical, inductive definition for relational graphs with cuts. To be more precisely: In CGwCs, we can have edges, incident with vertices, which are placed in different cuts, and it is not clear how this should be handled in an inductive definition. Secondly, even if we find an inductive definition of CGwCs, they will have no unique derivational history which is one of the main features of inductive definitions. For these reasons, it it convenient to provide a non-inductive definition for relational graphs (and thus for CGwCs) which is mainly an extension of the definition of mathematical multihypergraphs by adding cuts to these graphs.

Nonetheless, similar to formulas, relational graphs, thus CGwCs, bear a inner structure. A context c of a relational graph with cuts may contain other cuts d in its area (i.e. $d \in area(c)$), which in turn may contain further cuts, etc. It has to be expected that this idea induces an order $\leq$ on the contexts which should be a tree, having the sheet of assertion $\top$ as greatest element. The next definition is the mathematical elaboration of this idea. To ease matters, it is carried out on relational graph with cuts, but as CGwCs are obtained from relational graphs by labelling the vertices and edges, the following definition can be used for CGwCs as well.

Definition 2.4 (Ordering on Contexts) *Let a relational graph with cuts* $(V, E, \nu, \top, Cut, area)$ *be given. We define a mapping* $\beta : V \cup E \cup Cut \dot{\cup} \{\top\} \to Cut \dot{\cup} \{\top\}$ *by* $\beta(x) := x$ *for* $x \in Cut \dot{\cup} \{\top\}$ *and* $\beta(x) := cut(x)$ *for* $x \in V \dot{\cup} E$, *and we set* $x_1 \leq x_2 \;:\Longleftrightarrow\; \exists n \in \mathbb{N}_0 . \beta(x_1) \in area^n(\beta(x_2))$. *In order to avoid misunderstanding, we set* $x < y :\Longleftrightarrow x \leq y \wedge y \not\leq x$ *and* $x \lneq y :\Longleftrightarrow x \leq y \wedge y \neq x$.

Every element x *of* $V \cup E \cup Cut \dot{\cup} \{\top\}$ *with* $x < c$ *is said to be* enclosed by c, *and vice versa:* c *is said to* enclose x. *For every element of* $area(c)$, *we say more specifically that it is* directly enclosed by c. *Let* $n := |\{c \in Cut \mid x \in \leq[c]\}|$. *If* n *is even,* x *is said to be* evenly enclosed, *otherwise* x *is said to be* oddly enclosed.

The sheet of assertion $\top$ *and each oddly enclosed cut is called a* positive context, *and each an evenly enclosed cut is called* negative context.

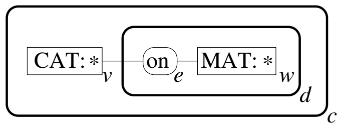
As it has been shown in [6], we get the following lemma:

Theorem 2.5 *For a relational graph with cuts* $(V, E, \nu, \top, Cut, area)$, $\leq$ *is a quasiorder. Furthermore,* $\leq|_{Cut \dot{\cup} \{\top\}}$ *is an order on* $Cut \dot{\cup} \{\top\}$ *which is a tree with the sheet of assertion* $\top$ *as greatest element.*

The ordered set of contexts $(Cut \dot{\cup} \{\top\}, \leq)$ can be considered to be the 'skeleton' of a relational graph. For linear notions of logic, where the well-formed formulas are defined inductively, and many proofs are carried out inductively over the construction of formulas. Although graphs are not defined inductively, Lem. 2.5 now allows us to do inductive definitions and proofs as well.

Below, as an example, you find the formalization of of $\mathfrak{G}_2$: First the mathematical item, then its graphical representation. Furthermore, its its order $\leq$ is depicted.

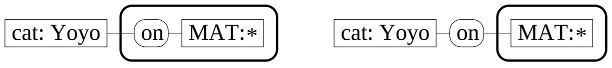
$$\begin{aligned}\mathfrak{G}_2 \;:=\; (&\{v,w\}, \{e\}, \{(e,(v,w))\}, \top,\\ &\{c,d\}, \{(\top,\{c\}), (c,\{v,d\}), (d,\{e,w\})\},\\ &\{(v, CAT), (w, MAT), (e, on)\}, \{(v,*), (w,*)\})\end{aligned}$$



$$\begin{tikzcd} \top \arrow[d, no head] \\ c, v \arrow[d, no head] \\ d, w, e \end{tikzcd}$$

In this example, c is directly enclosed by the sheet of assertion $\top$, d and the vertex v is directly enclosed by the cut c, and the vertex w and the edge e is directly enclosed by the cut d.

In the next section, a formal semantics for CGwCs will be introduced. Nonetheless, we can already discuss informally that graphs containing an edge with a incident and deeper nested vertex cause problems. Consider the following two graphs:



The meaning of the left graph is clear: 'For the cat Yoyo it is not true that there is a mat such that Yoyo is on that mat'. Note that the quantification of the generic marker takes places inside the cut. But how should the right graph be read? If we evaluate the relation oval, we have to know which object is assigned to the concept box *inside* the cut. Thus the so-called 'scope' of the generic marker has to be extended to the sheet of assertion. This would yield the reading 'the cat Yoyo is on something which is not a mat', where the quantification of the generic marker now takes places *outside* the cut. We see that obtaining the meaning of the right graph is possible, but not canonical, which makes the reading of CGwCs of this kind very complicated. For this reason we will forbid graphs having edged incident with deeper nested vertices. That is, from now on, we restrict ourselves to CGwCs which satisfy $cut(e) \leq cut(v)$ for every $e \in E$ and $v \in V_e$. CGwCs satisfying this condition are said to have *dominating nodes*.

3 Semantics

For the most kinds of mathematical logic, the semantics are based on extensional models and the evaluation of formulas in such models. In contrast to that, if mathematical logic is done with diagrams, often no direct extensional semantics is provided, but a translation from the graphs to FOPL is given.[4] In this respect, the models of FOPL serve *indirectly* as models for the graphs as well.

Formulas and graphs are very different 'styles' of logic, thus it is a little bit awkward and unappropiate that the semantics, i.e., meaning, of graphs can only be gained indirectly via FOPL. Hence we will not adopt this approach here, but provide a *direct* semantics for graphs. In [6], the semantics is provided by means of so-called *contextual structures*, based on Wille's Formal Concept Analysis. In contrast to the well-known relational structures, contextual structures contain intensional information as well, thus they can be considered to be an extension of relational structures. Wille argues in [28] for the use of concextual structures instead of relational structures as follows: ' There is a fundamental reason for associating concept(ual) graphs and FCA which lies in their far-back reaching roots in philosophical logic and in their pragmatic orientation; more specifically, both together can play a substantial role in the formalization of (philosophical) logic.' A formalization of this human-oriented understanding of logic is an adequate approach for knowledge representation and processing, which is a main goal of concept(ual) graphs and CGwCs.

Nonetheless, the evaluation of CGwCs in contextual structures is based on the *extension* of concepts. For this reason, it is possible to assign to each contextual structure a corresponding relational structure by, roughly speaking, removing all the intensional information from the contextual structure. This has be elaborated in [6]. To ease matters for readers who are not familiar with Formal Concept Analysis, in this paper, the semantics for CGwCs is provided by means of well-known relational structures.

Definition 3.1 (Relational Structures) *A* relational structure over $\mathcal{A}$ *is a pair* $\mathcal{M} := (U, I)$ *consisting of a* universe $U \neq \emptyset$ *and a function* $I := I_\mathcal{G} \dot\cup I_\mathcal{C} \dot\cup I_\mathcal{R}$ *with* $I_\mathcal{G} : \mathcal{G} \to U$, $I_\mathcal{C} : \mathcal{C} \to \mathfrak{P}(U)$ *and* $I_\mathcal{R} : \mathcal{R}_k \to \mathfrak{P}(U^k)$ *for each* k *such that* $I_\mathcal{C}$ *and* $I_\mathcal{R}$ *are order-preserving,* $I_\mathcal{C}(\top) = U$*, and* $(u_1, u_2) \in I_\mathcal{R}(\doteq) \Leftrightarrow u_1 = u_2$ *for all* $u_1, u_2 \in U$.

Below an example of a relational structure for our alphabet is depicted as crosstables (in the crosstable for $I_\mathcal{R}$, $\mathbb{K}_2$, the objects are abbreviated). The names of the alphabet are mapped to the corresponding elements of the structure. Note that the universe contains a fourth element no name is mapped to, and that the order of the alphabet is respected by the model.

$U, I_\mathcal{G}, I_\mathcal{C}$	cat	dog	animal	mat	$\top$
Yoyo	×		×		×
Garfield	×		×		×
Snoopy		×	×		×
#1				×	×

$I_\mathcal{R}$	on	$\doteq$
(Y,#1)	×	
(S,#1)	×	
(Y,Y)		×
(G,G)		×
(S,S)		×
(#1,#1)		×

When a graph is evaluated in a model, we start on the sheet of assertion $\top$ and proceed inwardly (this is the *endoporeutic method* of Peirce for existential graphs). During this process, we successively assign objects to generic vertices. This shall first be exemplified with the graph $\mathfrak{G}_2$. As only its outermost cut c is directly enclosed by $\top$, we see that $\mathfrak{G}_2$ is true if the subgraph which is enclosed by c is false. This subgraph contains a vertex v and a further cut d. We now have the following: $\mathfrak{G}_2$ is true if it is not true that there exists an object o_v such that o_v is a cat and the proposition which is enclosed by d is false. Now we have to evaluate the area of d. This area contains the edge and the vertex w, which refers to an unknown object o_w. Hence $\mathfrak{G}_2$ is true if there is no cat o_v for which it is not true that there is a mat o_w such that o_v is on o_w. In simpler words: Every cat is on a mat. This proposition is false in the given model.

The next two definitions capture the evaluation of graphs in models in a precise manner.

Definition 3.2 (Partial and Total Valuations) *Let* $\mathfrak{G} := (V, E, \nu, \top, Cut, area, \kappa, \rho)$ *be a CGwCs and let* $\mathcal{M} := (U, I)$ *be a relational structure over* $\mathcal{A}$. *A mapping* $ref : V' \to U$ *with* $V^\mathcal{G} \subseteq V' \subseteq V$ *and* $ref(v) = I_\mathcal{G}(\rho(v))$ *for all* $v \in V^\mathcal{G}$ *is called a* partial valuation of $\mathfrak{G}$. *If* $V' \supseteq \{v \in V^* \mid v > c\}$ *and* $V' \cap \{v \in V^* \mid v \leq c\} = \emptyset$ *then we say that* ref *is a partial valuation* for the context c. *If* $V' = V$ *then* ref *is called* (total) valuation of $\mathfrak{G}$.

[4]This has already been discussed in the introduction for several elaborations of conceptual graphs, but holds for mathematical elaborations of existential graphs as well. See for example [20, 30]).

Definition 3.3 (Endoporeutic Evaluation of Graphs)

Let $\mathfrak{G} := (V, E, \nu, \top, Cut, area, \kappa, \rho)$ be an CGwC and let $\mathcal{M} := (U, I)$ be a relational structure over $\mathcal{A}$. Inductively over the tree $Cut \dot\cup \{\top\}$, we define $\mathcal{M} := (U, I) \models \mathfrak{G}[c, ref]$ for each context $c \in Cut \dot\cup \{\top\}$ and every partial valuation $ref : V' \subseteq V \to U$ for c:

$\mathcal{M} \models \mathfrak{G}[c, ref] :\Longleftrightarrow$

ref can be extended to a partial valuation $\widetilde{ref} : V' \cup (V \cap area(c)) \to U$ (i.e., $\widetilde{ref}(v) = ref(v)$ for all $v \in V'$), such that the following conditions hold:

- *$\widetilde{ref}(v) \in I_{\mathcal{C}}(\kappa(v))$ for each $v \in V \cap area(c)$*
- *$\widetilde{ref}(e) \in I_{\mathcal{R}}(\kappa(e))$ for each $e \in E \cap area(c)$*
- *$\mathcal{M} \not\models \mathfrak{G}[d, \widetilde{ref}]$ for each $d \in Cut \cap area(c)$*

For $\mathcal{M} \models \mathfrak{G}[\top, \emptyset]$ we write $\mathcal{M} \models \mathfrak{G}$. If we have two concept graphs $\mathfrak{G}_1$, $\mathfrak{G}_2$ such that $\mathcal{M} \models \mathfrak{G}_2$ for each relational structure $\mathcal{M}$ with $\mathcal{M} \models \mathfrak{G}_1$, we write $\mathfrak{G}_1 \models \mathfrak{G}_2$.

Note that the second condition for $\widetilde{ref}$ is technical sound as we only consider concept graphs with dominating nodes.

4 Calculus

The calculus for CGwCs is based on Peirce's calculus for existential graphs, which is extended in order to capture the syntactical differences and the higher expressiveness of CGwCs. It is mainly intended to be used by humans. Its rules are much more powerful than rules known from calculi for symbolic notations (like sequencen-calculi or natural deduction), as they allow to change a whole subgraphs of a graph in arbitrary context (for this reason, it is far from beeing trivial to see and prove that the rules are sound). Nonetheless, some authors investigated how automated theorem provers can be implemented by means of Peirce's rules for existential graphs (see [25, 13], which shows that in the area of mechanistic reasoning, existential graphs, thus CGwCs as well, are a promising approach too.

The calculus is provided here in a semi-formal manner. For the mathematical definitions of the rules, see [6].

- **erasure, insertion:** In positive contexts, any directly enclosed edge, isolated vertex, and closed subgraph may be erased, and in negative contexts, any directly enclosed edge, isolated vertex, and closed subgraph may be inserted.
- **iteration, deiteration:** Let $\mathfrak{G}_0$ be a (not necessarily closed) subgraph of $\mathfrak{G}$ and let $c \leq cut(\mathfrak{G}_0)$ be a context such that $c \notin Cut_0$. Then a copy of $\mathfrak{G}_0$ may be inserted into c. For every vertex $v \in V_0^*$ with $cut(v) = cut(\mathfrak{U})$, an identity-link from v to its copy may be inserted.

  If $\mathfrak{G}_0$ is a subgraph of $\mathfrak{G}$ which could have been inserted by rule of iteration, then it may be erased.
- **double cuts** Double cuts (two cuts c_1, c_2 with $area(c_1) = \{c_2\}$) may be inserted or erased.
- **generalization, specialization:** For evenly enclosed vertices and edges, their concept names or object names resp. their relation names may be generalized, and for oddly enclosed vertices and edges, their names may be specialized.
- **isomorphism** A graph may be substituted by an isomorphic copy of itself.
- **exchanging referents** Let $e \in E^{id}$ be an identity link with $\rho(e|_1) = g_1$, $\rho(e|_2) = g_2$, $g_1, g_2 \in \mathcal{G} \cup \{*\}$ and $cut(e) = cut(e|_1) = cut(e|_2)$. Then the referents of v_1 and v_2 may be exchanged, i.e., the following may be done: We can set $\rho(e|_1) = g_2$ and $\rho(e|_2) = g_1$.
- **merging two vertices, splitting a vertex:** Let $e \in E^{id}$ be an identity link with $\nu(e) = (v_1, v_2)$ such that $cut(v_1) \geq cut(e) = cut(v_2)$, $\rho(v_1) = \rho(v_2)$ and $\kappa(v_2) = \top$ hold. Then v_1 may be merged into v_2, i.e., v_1 and e are erased and, for every edge $e \in E$, $e|_i = v_1$ is replaced by $e|_i = v_2$.

  If $\mathfrak{G}_b$ is derived from $\mathfrak{G}_a$ by merging two vertices, then $\mathfrak{G}_a$ may be derived from $\mathfrak{G}_b$ as well.
- **$\top$-erasure, $\top$-insertion:** For $g \in \mathcal{G} \cup \{*\}$, an isolated vertex $\boxed{\top : g}$ may be erased from or inserted into arbitrary contexts.
- **identity-erasure, identity-insertion:** Let v_1, v_2 be two vertices in contexts c_1, c_2, resp. with $v_1 = \boxed{P_1 : g}$ and $v_2 = \boxed{P_2 : g}$ for a $g \in \mathcal{G}$, and let $c \leq c_1, c_2$ be a context. Then any identity-links between v_1 and v_2 may be erased from or inserted into c.

A *proof* for two graphs $\mathfrak{G}_a$, $\mathfrak{G}_b$ is defined as usual in logic, i.e., it is a sequence of graphs, starting with $\mathfrak{G}_a$, ending with $\mathfrak{G}_b$, where each graph of the sequence is derived from its predecessor by one of the rules of the calculus. As usual, this will be denoted $\mathfrak{G}_a \vdash \mathfrak{G}_b$.

5 Syntactical Translations

CGwCs have the expressiveness of FOPL. For example, $\mathfrak{G}_1$ and $\mathfrak{G}_2$ correspond to the formulas $\exists x.cat(Yoyo) \wedge mat(x) \wedge on(Yoyo, x)$ and $\neg\exists y.(cat(y) \wedge \neg\exists x.(mat(x) \wedge on(y, x)))$, resp. In order to elaborate the correspondence between CGwCs and the linear notion of FOPL, we have to provide mappings Ψ : FOPL $\to$ CG and Φ : CG $\to$ FOPL. These mappings can be understood as translations between the two logical systems FOPL and CG.

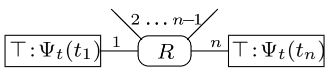
Please note that there are some structural differences between the logical systems FOPL and CG. First of all, in FOPL, we have an infinite set of variables which are used to range over objects. In CG, only the generic marker '$*$' is used for this purpose. Moreover, in CG, we have no syntactical devices which correspond to the free variables of FOPL (below, we consider so-called 'CGwC with variables', but they are a mere helper construction used to define the mapping Ψ). Next, conjunction is an assossiative and commutative operation. For formulas, i.e., the linear notion of FOPL, this is reflected by rules in the calculus. In CG, conjunction is expressed by the juxtaposition of graphs, where we have no order of the juxtaposed graphs. Thus, the assossiativity and commutativity of conjunction is already reflected by the syntax of CGwCs. Finally, in CGwCs it is allowed to have emtpy cuts. In FOPL, there is no corresponding expression.

Due to this reasons, a formula $\Phi(\mathfrak{G})$ will be only given up to the names of the variables and the the order of the subformulas of conjunctions. Particularly, it cannot be expected that $\Phi \circ \Psi$ is the identity mapping.

Now we are prepared to provide the definitions of Ψ and Φ. According to the structures of formulas resp. graphs, they are defined recursively.

Definition of Ψ: Before we provide a translation of formulas to CGwCs, we have to translate the terms. This is done canonically by a mapping Ψ_t (in [6], it is called Ψ_{term}). We set $\Psi_t(g) := g$ for each object name $g \in \mathcal{G}$ and we set $\Psi_t(\alpha) := *_\alpha$ for variables $\alpha \in$ Var. Now we can define Ψ inductively over the composition of formulas For each case, only an informal description of the definition is provided. For the mathematical definition, see [6].

- $C(t)$ for a term t and $C \in \mathcal{C}$: $\Psi(C(t)) := \boxed{C : \Psi_t(t)}$
- $R(t_1, \ldots, t_n)$ for $R \in \mathcal{R}$ with $ar(R) = n$ and terms $t_1, \ldots, t_n$:

  $\Psi(R(t_1, \ldots, t_n)) :=$ 

- $f_1 \wedge f_2$ for formulas f_1 and f_2:

  $\Psi(f_1 \wedge f_2) := \Psi(f_1) \quad \Psi(f_2)$

  (i.e., $\Psi(f_1 \wedge f_2)$ is the juxtaposition of $\Psi(f_1)$ and $\Psi(f_2)$).
- $\neg f$ for a formula f: $\Psi(\neg f) := \boxed{\Psi(f)}$
- $\exists \alpha . f$ for a formula f and a variable α:

  If $\alpha \notin Free(f)$, we set $\Psi(\exists \alpha . f) := \Psi(f)$. For $\alpha \in Free(f)$, the following steps have to be carried out: A new concept box $v_0 := \boxed{\top : *}$ is juxtaposed to $\Psi(f)$. Then, for each edge e, every concept box $\boxed{\top : *_\alpha}$ which is incident with e is substituted by the new concept box v_0. Next, every isolated concept box $\boxed{P : *_\alpha}$ is substituted by a concept box $\boxed{P : *}$, which is linked to the new concept box v_0 with an identity-link: $v_0 — (=) — \boxed{P : *}$. Finally, all concept boxes $\boxed{\top : *_\alpha}$ are erased (as we performed step 2, all these boxes are isolated).

This completes the definition of Ψ. In particular, Ψ translates formulas without free variables to CGwCs. As we have finished the definition of Ψ, we proceed with the definition of $\Phi : \mathrm{CG} \to \mathrm{FOPL}$.

Definition of Φ. Let $\mathfrak{G} := (V, E, \nu, \top, Cut, area, \kappa, \rho)$ be a CGwC with variables. Let $Free(\mathfrak{G}) := \{\alpha \in \mathrm{Var} \mid \exists v \in V. \rho(v) = *_\alpha\}$. We assign to each vertex $v \in V^*$ a fresh variable $\alpha_v \notin Free(\mathfrak{G})$, so that we can define the following mapping Φ_t on V: $\Phi_t(v) := \alpha_v$ for $\rho(v) = *$, $\Phi_t(v) := \alpha$ for $\rho(v) = *_\alpha$ and $\alpha \in$ Var, and $\Phi_t(v) := g$ for $\rho(v) = g$ and $g \in \mathcal{G}$.

Let $\alpha_{empty} \notin Free(\mathfrak{G}) \cup \{\alpha_v \mid v \in V \text{ and } \rho(v) = *\}$ be a further variable. Now, inductively over the tree $Cut \,\dot\cup\, \{\top\}$, we assign to each context $c \in Cut \,\dot\cup\, \{\top\}$ a formula $\Phi(\mathfrak{G}, c)$. So let c be a context such that $\Phi(\mathfrak{G}, d)$ is already defined for each cut $d < c$. First, we define a formula f which encodes all edges and vertices which are directly enclosed by c. Hence, if c does not directly enclose any edges or vertices, simply set $f := (\exists \alpha_{empty} . \top(\alpha_{empty}))$. Otherwise, let f be the conjunction of the following atomic formulas:$\kappa(w)(\Phi_t(w))$ for each $w \in V \cap area(c)$, and $\kappa(e)(\Phi_t(w_1), \ldots, \Phi_t(w_j))$ for each $e \in E \cap area(c)$ and $\nu(e) = (w_1, \ldots, w_j)$.

Let $v_1, \ldots, v_n$ be the vertices of $\mathfrak{G}$ which are enclosed by c and which fulfill $\rho(v_i) = *$, and let $area(c) \cap Cut = \{c_1, \ldots, c_l\}$ (by induction, we already assigned formulas to these cuts). If $l = 0$, set $\Phi(\mathfrak{G}, c) := \exists \alpha_{v_1} . \ldots \exists \alpha_{v_n} . f$, otherwise set $\Phi(\mathfrak{G}, c) := \exists \alpha_{v_1} . \ldots \exists \alpha_{v_n} . (f \wedge \neg \Phi(\mathfrak{G}, c_1) \wedge \ldots \wedge \neg \Phi(\mathfrak{G}, c_l))$. Finally set $\Phi(\mathfrak{G}) := \Phi(\mathfrak{G}, \top)$, and the definition of Φ is finished.

6 Summary and Main Results

We have reached the following situation: We have two logical systems CG and FOPL and two mappings Φ and Ψ between them. Each system has a derivability relation $\vdash$ and an entailment relation $\models$. There are two main results in [6]. First of all, similar to FOPL, the calculus for CGwCs is sound and complete:

Theorem 6.1 (Sound- and Completeness) *Let $\mathfrak{G}_1$ and $\mathfrak{G}_2$ be CGwCs. Then we have $\mathfrak{G}_1 \vdash \mathfrak{G}_2 \Leftrightarrow \mathfrak{G}_1 \models \mathfrak{G}_2$.*

Secondly, we have said that Φ and Ψ can be considered as translations between FOPL and CG. That is, they are not simple mappings between the *sets* of FOPL and CG, but they to preserve the *meaning* of the formulas resp. CGwCs as well, i.e., they respect the semantical entailment relation $\models$. This is captured by the next theorem.

Theorem 6.2 (Main Translation Theorem) *Let $\mathfrak{G}$, $\mathfrak{G}_1$, $\mathfrak{G}_2$ be CGwCs over $\mathcal{A}$ and let f, f_1, f_2 be FOPL-formulas over $\mathcal{A}$. Then we have* $f_1 \models f_2 \;\Leftrightarrow\; \Psi(f_1) \models \Psi(f_2)$ *and* $\mathfrak{G}_1 \models \mathfrak{G}_2 \;\Leftrightarrow\; \Phi(\mathfrak{G}_1) \models \Phi(\mathfrak{G}_2)$.

These theorems yield the full syntactical and semantical equivalence between FOPL and CG.

7 Further Research

In [8] and [9], the system of CGwCs has already be extended. In [8], it has been shown how more complex (compared to the type-hierarchy) background knowledge can be incorporated into the system of CGwCs. In [9], CGwCs are extended to so-called Query Graphs with Cuts, which can be used to describe relations. This yields the possibility to use these graphs as a diagrammatic query language for databases. But this is by now not fully elaborated and should be further investigated.

Another crucial extension is the addition of so-called *nestings*, where whole subgraphs of a graph are enclosed by a vertex. There are different possibilities for interpreting nestings. Nestings are often used to describe specific contexts, e.g., situations. Thus, nestings extend the expressiveness of CGwCs. In [10], nestings are used to describe *nested relations* which occur in form of so-called *set functions* in database systems. The semantics and use of nestings has to be further investigated as well.

References

[1] F. Baader, R. Molitor, S. Tobies: *Tractable and Decidable Fragments of Concpetual Graphs.* In: W. Tepfenhart, W. Cyre (Eds.): Conceptual Structures: Standards and Practices. LNAI 1640, Springer Verlag, Berlin–New York 1999, 480–493.

[2] M. Chein, M.-L. Mugnier: *Conceptual Graphs: Fundamental Notions.* Revue d'Intelligence Artificielle 6, 1992.

[3] M. Chein, M.-L. Mugnier: *Conceptual Graphs are also Graphs.* Rapport de Recherche 95003, LIRMM, Université Montpellier II, 1995.

[4] M. Chein, M.-L. Mugnier: *Positive Nested Conceptual Graphs.* In: D. Lukose et al. (Eds.): Conceptual Structures: Fulfilling Peirce's Dream. LNAI 1257, Springer Verlag, Berlin–New York 1997.

[5] M.-L. Mugnier: *Concept Types and Coreference in Simple Conceptual Graphs.* In: Pfeiffer, H. D.; Wolff, K. E. (Eds): Conceptual Structures at Work, LNAI, Vol. 3127, Springer-Verlag, Berlin – Heidelberg – New York, 2004, p 303–318.

[6] F. Dau: *The Logic System of Concept Graphs with Negation (And Its Relationship to Predicate Logic).* LNAI, Vol. 2892, Springer, Berlin–Heidelberg–New York, 2003.

[7] F. Dau, Types and Tokens for Logic with Diagrams: A Mathematical Approach, In: Pfeiffer, H. D.; Wolff, K. E. (Eds): Conceptual Structures at Work, LNAI, Vol. 3127, Springer-Verlag, Berlin – Heidelberg – New York, 2004, p 62–93.

[8] F. Dau: *Background Knowledge in Concept Graphs.* in P. Eklund (Ed): Concept Lattices, Second International Conference on Formal Concept Analysis. LNAI 2961, Springer Verlag, Berlin–New York 2004, 156–171.

[9] F. Dau: *Query Graphs with Cuts: Mathematical Foundations.* In A. Blackwell, K. Marriott, A. Shimojima (Eds): Diagrammatic Representation and Inference. LNAI 2980, Springer Verlag, Berlin–New York 2004, 32-50.

[10] F. Dau, J. Hereth Correia: *Nested Concept Graphs: Applications for Databases and Mathematical Foundations.* In: Moor, A., Ganter, B. (Eds.): Using Conceptual Structures. Shaker Verlag, Aachen, 2003.

[11] F. Dau, J. Klinger: *From Formal Concept Analysis to Contextual Logic* To appear in the proceedings of the First International Conference on Formal Concept Analysis, 2003,

[12] B. Ganter, R. Wille: *Formal Concept Analysis: Mathematical Foundations.* Springer, Berlin–Heidelberg–New York 1999.

[13] J. E. Heaton: *Goal Driven Theorem Proving using Conceptual Graphs and Peirce Logic* Doctoral Thesis, Loughborough University Library, UK.

[14] J. Howse, F. Molina, S. Shin, J. Taylor: *On Diagram Tokens and Types.* In: Proceedings of Diagrams 2002, LNAI 2317, Springer Verlag, Berlin–New York 2002.

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

[16] J. Klinger: The Logic System of Protoconcept Graphs. PhD thesis, TU Darmstadt, 2005.

[17] C. S. Peirce: *Reasoning and the Logic of Things. The Cambridge Conferences Lectures of 1898.* Ed. by K. L. Ketner, H. Putnam, Harvard Univ. Press, Cambridge 1992.

[18] C. S. Peirce, *MS 478.* Collected Papers, 4.394-417. Harvard University Press, Cambrigde, Massachusetts.

[19] S. Prediger: *Kontextuelle Urteilslogik mit Begriffsgraphen. Ein Beitrag zur Restrukturierung der mathematischen Logik.* Shaker Verlag 1998.

[20] S. J. Shin: *The Iconic Logic of Peirce's Graphs.* Bradford Book, Massachusetts, 2002.

[21] G. Simonet: *Two FOL-Semantics for Simple and Nested Conceptual Graphs.* In: M, -L. Mugnier, M. Chein (Eds.): Conceptual Structures: Theory, Tools and Applications. LNAI 1453, Springer Verlag, Berlin–New York 1998, 240–254.

[22] J. F. Sowa: *Conceptual Structures: Information Processing in Mind and Machine.* Addison Wesley Publishing Company Reading, 1984.

[23] J. F. Sowa: *Conceptual Graphs Summary.* In: T. E. Nagle, J. A. Nagle, L. L. Gerholz, P. W. Eklund (Eds.): Conceptual Structures: current research and practice, Ellis Horwood, 1992.

[24] J. F. Sowa: *Knowledge Representation: Logical, Philosophical, and Computational Foundations.* Brooks Cole Publishing Co., Pacific Grove, CA, 2000.

[25] J. Stewart: *Theorem Proving Using Existential Graphs* Master thesis (advisor: Robert Levinson), 1996. Unpublished.

[26] M. Wermelinger: *Conceptual Graphs and First-Order Logic.* In: G. Ellis, R. Levinson, W. Rich, J. F. Sowa (Eds.): Conceptual Structures: Applications, Implementation, and Theory. LNAI 954, Springer Verlag, Berlin 1995, 323–337.

[27] R. Wille: *Restructuring Mathematical Logic: An Approach Based on Peirce's Pragmatism.* In: A. Ursini, P. Agliano (Eds.): Logic and Algebra. Marcel Dekker, New York 1996, 267–281.

[28] R. Wille: *Conceptual Graphs and Formal Concept Analysis.* In: D. Lukose et al. (Eds.): Conceptual Structures: Fulfilling Peirce's Dream. LNAI 1257, Springer Verlag, Berlin–New York 1997.

[29] R. Wille: *Contextual Logic Summary.* In: G. Stumme (Ed.): Working with Conceptual Structures. Contributions to ICCS 2000. Shaker, Aachen 2000.

[30] J. Zeman: *The Graphical Logic of C. S. Peirce* Ph.D. Diss., University of Chicago, 1964.