Concept Graphs and Predicate Logic

Frithjof Dau

Technische Universität Darmstadt, Fachbereich Mathematik
Schloßgartenstr. 7, D-64289 Darmstadt, dau@mathematik.tu-darmstadt.de

Abstract. In the ICCS 2000 proceedings we introduced negation to simple concept graphs without generic markers by adding *cuts* to their definition. The aim of this paper is to extend this approach of cuts to simple concept graphs *with* generic markers. For these graphs, a set-theoretical semantics is presented. After this a modification of Peirce's beta-calculus is provided, and definitions for mappings Φ and Ψ between concept graps and first order logic are given. If we consider both concept graphs and first order logic formulas, together with their particular derivability relations, as quasiorders, Φ and Ψ are mutually inverse quasiorder isomorphisms between them. The meaning of this fact is elaborated. Finally we provide a result that links the semantics of concept graphs and the semantics of first order logic. This result can be used to show that the calculus for concept graphs is sound and complete.

1 Motivation and Overview

In [Da00], we introduced negation to simple concept graphs without generic markers by adding *cuts* to their definition. These concept graphs are closely related to the α-part of the existential graphs of Charles Sanders Peirce, which consist only of cuts and propositional variables. The aim of this work is to extend the approach of [Da00] to simple concept graphs with generic markers. These graphs correspond to the β-part of existential graphs, namely to existential graphs which are built up of cuts, relation names, and lines of identity. It is accepted that these graphs are equivalent to first order logic. An argumentation to support this (but, in our view, not a strict mathematical proof) can be found in [Ro73]. Therefore it seems to be evident that a class of simple concept graphs with generic markers and negations (i.e. cuts) are equivalent to first order logic, too. Indeed, this equivalence can be described and proven in a mathematically precise way, which will be in [Da01]. In this paper, we want to elaborate some aspects of this work.

To start, we provide the necessary definitions for concept graphs with cuts. A mathematical semantics for these graphs which is based on power context families is presented. After this sematical part, we provide a calculus which is based on the β-calculus for existential graphs, but which captures the specific properties of concept graphs. A mathematical definition for the version of the well-known Φ-operator which maps (in this case) simple concept graphs with cuts to first order logic is given, as well as the definition for a mapping Ψ in the inverse

direction. It turns out that Φ and Ψ are mutually inverse isomorphismns between the quasiordered sets of concept graphs and first order logic. The proof for this is very extensive (and will be given in [Da01]). In this paper, we investigate only the meaning of these isomorphismns. Finally, we give a result that links the semantics for concept graphs to the usual relational semantics for first order logic. This result can be used to show that the calculus for concept graphs is sound and complete.

2 Basic Definitions for Simple Concept Graphs

First, we start with an underlying set of variables and names which are needed in concept graphs as well as in first order logic.

Definition 1.

1. *Let Var be a countably infinite set. The elements of Var are called* variables. *In concept graphs, we need a sign $*$, the* generic marker. *Further we assign a new sign $*_\alpha$ to each variable $\alpha \in Var$.*
2. *An* alphabet *is a triple $\mathcal{A} := (\mathcal{G}, \mathcal{C}, \mathcal{R})$ such that*
   - *$\mathcal{G}$ is a finite set whose elements are called* object names.
   - *$(\mathcal{C}, \leq_\mathcal{C})$ is a finite ordered set with a greatest element $\top$. The elements of this set are called* concept names.
   - *$(\mathcal{R}, \leq_\mathcal{R})$ is a familiy of finite ordered sets $(\mathcal{R}_k, \leq_{\mathcal{R}_k})$, $k = 1, \ldots, n$ (for an $n \in \mathbb{N}$ with $n \geq 1$) whose elements are called* relation names. *Let $id \in \mathcal{R}_2$ be a special name which is called* identity.

Now we can define the underlying structures of concept graphs with cuts. There are two slight changes compared to Definition 2 in [Da00]: First, for purely technical reasons, we add the sheet of assertion to the definition. Second, we change the definition of the mapping $area$ so that the area of a cut c contains vertices, edges, and other cuts which are enclosed by c, *but not* if they are nested deeper inside other cuts. The only reason for this is that this definition reflects the meaning of area in existential graphs better than the definition in [Da00].

Definition 2. *A* relational graph with cuts *is a structure $(V, E, \nu, \top, Cut, area)$ such that*

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

For an edge $e \in E$ with $\nu(e) = (v_1, \ldots, v_k)$ we define $|e| := k$ and $\nu(e)\big|_i := v_i$. For each $v \in V$, let $E_v := \{e \in E \mid \exists\, i\colon \nu(e)\big|_i = v\}$. Analogously, for each $e \in E$, let $V_e := \{v \in V \mid \exists\, i\colon \nu(e)\big|_i = v\}$. If it cannot be misunderstood, we write $e\big|_i$ instead of $\nu(e)\big|_i$.

The empty graph has the form $\mathfrak{G}_\emptyset := (\emptyset, \emptyset, \emptyset, \top, \emptyset, \emptyset)$.

We say that all edges, vertices and cuts in the area of c and all items which are deeper nested are *enclosed by* c:

Definition 3. *For a relational graph with cuts $(V, E, \nu, \top, Cut, area)$ we define $\overline{area} : Cut \cup \{\top\} \to \mathfrak{P}(V \cup E \cup Cut)$, $\overline{area}(c) := \bigcup_{n\in\mathbb{N}} area^n(c)$. Every element k of $\overline{area}(c)$ is said to be* enclosed by c*, and vice versa: c is said to* enclose k*. For every element of $area(c)$, we say more specificly that it is* directly enclosed by c*. Because every $k \in V \cup E \cup Cut$ is directly enclosed by exact one $c \in Cut \cup \{\top\}$, for every $k \in area(c)$ we can write $c = area^{-1}(k)$, or even more simply and suggestive: $c = cut(k)$.*

By $c_1 \leq c_2 :\Longleftrightarrow c_1 \in \overline{area}(c_2)$ a canonical tree ordering on $Cut \cup \{\top\}$ with $\top$ as greatest element is defined.

In this work, we will only consider graphs in which vertices must not be deeper nested than any edge they are incident with. This is captured by the following definition:

Definition 4. *If $cut(e) \leq cut(v)$ holds for every $e \in E$ and $v \in V$ such that e is incident with v, then $\mathfrak{G}$ is said to have* dominating nodes*.*

Now simple concept graphs with cuts are derived from relational graphs with cuts by additionally labeling the vertices and edges with concept names and relation names, respectively, and by assigning a reference to each vertex.

Definition 5. *A* simple concept graph with cuts and variables over the alphabet $\mathcal{A}$ *is a structure $\mathfrak{G} := (V, E, \nu, \top, Cut, area, \kappa, \rho)$ where*

- *$(V, E, \nu, \top, Cut, area)$ 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 $\nu(e) = (v_1, \ldots, v_k)$ satisfy $\kappa(e) \in \mathcal{R}_k$*
- *$\rho : V \to \mathcal{G} \cup \{*\} \cup \{*_\alpha \mid \alpha \in Var\}$ is a mapping.*

If additionally $\rho : V \to \mathcal{G} \cup \{*\}$ holds, then $\mathfrak{G}$ is called simple concept graph with cuts over the alphabet $\mathcal{A}$*. If even $\rho : V \to \mathcal{G}$ holds, then $\mathfrak{G}$ is called* nonexistential simple concept graph with cuts over the alphabet $\mathcal{A}$*. For the set E of edges, let $E_{id} := \{e \in E \mid \kappa(e) = id\}$ and $E_{nonid} := \{e \in E \mid \kappa(e) \neq id\}$. The elements of E_{id} are called* identity-links*.*

In the rest of this work, we will mainly talk about (existential) simple concept graphs with cuts and dominating nodes over the alphabet $\mathcal{A}$, and will call them 'concept graphs' for short. This set of concept graphs is denoted by CG.

The mathematical definitions make up an exact and solid foundation for concept graphs which can serve as a precise reference and a basis for mathematical proofs on concept graphs. But in order to work with concept graphs, their mathematical representations are too clumsy and too difficult to handle. Hence one may prefer the well known graphical representations of conceptual graphs. Because we added cuts as new syntactical elements to the graphs, we have to explain how concept graphs with cuts are drawn. This shall be done now.

Vertices are usually drawn as small rectangles. Inside the rectangle for a vertex v, we write first the concept name $\kappa(v)$ and then the reference $\rho(v)$, seperated by a colon. These rectangels are called *concept boxes*. An edge e is drawn as a small oval with its relation name $\kappa(e)$ in it. The name 'id' for the identity is often replaced by the symbol '='. These ovals are called *relation ovals*. For an edge $e = (v_1, \ldots, v_n)$, each concept box of the incident vertices $v_1, \ldots, v_n$ is connected by a line to the relation oval of e. These lines are numbered $1, \ldots, n$. If it cannot be misunderstood, this numbering is often omitted. There may be graphs such that its lines cannot be drawn without their crossing one another. To distinguish such lines from each other, Peirce introduced a device he called a 'bridge' (see [Ro73], Page 55). But, except for bridges between lines, all the boxes, ovals, and lines of a graph must not intersect. Nearly all graphs which occur in applications do not need bridges. Finally, a cut is drawn as a bold curve (usually an oval) which exactly contains in its inner space all the concept boxes, ovals, and curves of the vertices, edges, and other cuts, resp., which the cut encloses (not necessarily directly). The curve of a cut may not intersect any other curves, ovals or concept boxes, but it may intersect lines which connect relation ovals and concept boxes (this is usually inevitable).

To illustrate these agreements, consider the following graph over the alphabet $\mathcal{A} := (\emptyset, \{\text{CAT, ANIMAL}, \top\}, \{\text{cute, id}\})$ in its mathematical form:

$$\begin{aligned}\mathfrak{G} := (&\{v_1, v_2\}, \{e_1, e_2\}, \{(e_1, (v_1, v_2)), (e_2, v_2)\}, \top, \{c_1, c_2\},\\ &\{(\top, \emptyset), (c_1, \{v_1\}), (c_2, \{v_2, e_1, e_2\})\},\\ &\{(v_1, \text{CAT}), (v_2, \text{ANIMAL}), (e_1, \text{id}), (e_2, \text{cute})\}, \{(v_1, *), (v_2, *)\})\end{aligned}$$

In Figure 1, we give one (possible) diagram for this graph. The indices v_1, v_2, e_1, e_2, c_1, c_2 do not belong to the diagram. They are added to make the translation from $\mathfrak{G}$ to the diagram more transparent. Now one can see that the intuitive meaning of the graph is 'it is not true that there is a cat which is not a cute animal', i.e. 'every cat is a cute animal'. This will be worked out in Section 3.

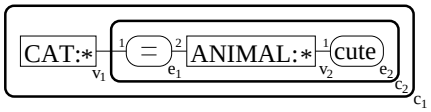
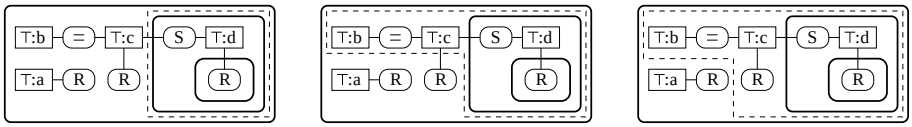


Fig. 1. one example for a simple concept graph with cuts and dominating nodes

For our further work, especially for the calculus, the notion of a *subgraph* is needed. We distinguish beween *subgraphs* and *closed subgraphs*. Informally spoken, a *subgraph* is a part of a graph such that

- if the subgraph contains a cut c, then it contains whatever is scribed inside c, i.e. $\overline{area}(c)$, and
- if the subgraph contains an edge, then it contains all vertices which are incident with the edge.

If it holds furthermore that for each vertex of subgraph all incident edges are part of the subgraph, too, then the subgraph is called a *closed subgraph*. Instead giving a formal definition for this (which will be in [Da01]), we only provide some examples.



The marked area in the left example is no subgraph. The marked area in the middle example is a subgraph which is not closed. The marked area in the right example is a closed subgraph in the outermost cut.

We use the following teminology: For each vertex, edge, cut, or subgraph, we say that it is *evenly enclosed* iff the number of cuts which enclose it is even, otherwise it is *oddly enclosed*. Evenly enclosed cuts are also called *positive cuts*, and oddly enclosed cuts are called *negative cuts*. The formal definitions can be found in [Da00].

3 Semantics for Simple Concept Graphs

As in the papers of Prediger (cf. [Pr98a], [Pr98b]) or as in [Da00], we use power context families as model structures for concept graphs. Their definition can be found in [Wi97], [Pr98b] or in [Da00]. But we want to repeat the definition of a $\vec{\mathbb{K}}$-interpretation which we presented in [Da00], because the identity-relation $\mathrm{id} \in \mathcal{R}_2$ caused a slight change to Prediger's definition of a $\vec{\mathbb{K}}$-interpretation.

Definition 6. *For an alphabet $\mathcal{A} := (\mathcal{G}, \mathcal{C}, \mathcal{R})$ and a power context family $\vec{\mathbb{K}}$, we call the union $\lambda := \lambda_{\mathcal{G}} \dot{\cup} \lambda_{\mathcal{C}} \dot{\cup} \lambda_{\mathcal{R}}$ of the mappings $\lambda_{\mathcal{G}}: \mathcal{G} \to G_0$, $\lambda_{\mathcal{C}}: \mathcal{C} \to \underline{\mathfrak{B}}(\mathbb{K}_0)$ and $\lambda_{\mathcal{R}}: \mathcal{R} \to \mathfrak{R}_{\vec{\mathbb{K}}}$ a* $\vec{\mathbb{K}}$-interpretation *of $\mathcal{A}$ if $\lambda_{\mathcal{C}}$ and $\lambda_{\mathcal{R}}$ are order-preserving, $\lambda_{\mathcal{C}}(\top) = \top$, $\lambda_{\mathcal{R}}(\mathcal{R}_k) \subseteq \underline{\mathfrak{B}}(\mathbb{K}_k)$ for all $k = 1, \ldots, n$, and $(g_1, g_2) \in Ext(\lambda_{\mathcal{R}}(id)) \Leftrightarrow g_1 = g_2$ hold for all $g_1, g_2 \in \mathcal{G}$. The pair $(\vec{\mathbb{K}}, \lambda)$ is called* context-interpretation *of $\mathcal{A}$ or, according to classical logic,* $\mathcal{A}$-structure.

Now we can define whether a concept graph is valid in an $\mathcal{A}$-structure. To do this, we read and evaluate the graph from the outside. Hence we start with the sheet of assertion $\top$, and proceed with the inner cuts. This method (reading

a graph from the outside and proceeding inwardly) was called 'endoporeutic method' by Peirce (see [Ro73]). Before we give a precise definition, we exemplify this method on the graph $\mathfrak{G}$ from Figure 1.

We start the evaluation of this graph on the sheet of assertion $\top$. Because only the cut c_1 is directly enclosed by $\top$, the graph $\mathfrak{G}$ is true if the part of $\mathfrak{G}$ which is enclosed by c_1 is false. Because c_1 contains the vertex v_1 and the cut c_2, we come to the following conclusion: $\mathfrak{G}$ is true iff it is not true that the following two conditions hold: There exists an object such that o_1 is a cat (lets call it o_1) and the information which is enclosed by c_2 is false. Now we have to evaluate the area of c_2. Intuitively spoken, the area of c_2 is true iff there is an object that is a cute animal and identical to o_1. Note that during this evaluation, we refer to the object o_1. That is why the endoporeutic method goes from the outside to the inside: We cannot evaluate the inner cut c_2 unless we know which object is assigned to the generic marker in the vertex v_1. Finally $\mathfrak{G}$ is true if there is no cat such that there is no other object which is identical to the cat (hence the cat itself) and which is a cute animal. In simpler words: $\mathfrak{G}$ is true if there is no cat which is not a cute animal, i.e. if every cat is a cute animal.

Hopefully this example helps the reader to understand the following definitions. The assignment of objects to vertices by valuations:

Definition 7. *Let* $(\vec{\mathbb{K}}, \lambda)$ *be a context-interpretation of the alphabet* $\mathcal{A}$ *and* $\mathfrak{G} = (V, E, \nu, \top, Cut, area, \kappa, \rho)$ *be a graph. A* partial valuation of $\mathfrak{G}$ *is a mapping* $ref : V' \subseteq V \to \mathbb{K}_0$ *such that* $V' \supseteq \{v \in V \mid \rho(v) \in \mathcal{G}\} =: V_{\mathcal{G}}$ *and* $ref(v) = \lambda_{\mathcal{G}}(\rho(v))$ *holds for all* $v \in V_{\mathcal{G}}$. *If* $V' = V$ *holds, then* ref *is called* (total) valuation of $\mathfrak{G}$.

With the endoporeutic method, we can evaluate the area of a cut c in a concept graph $\mathfrak{G}$ if we already have a partial valuation ref which assigns objects to the vertices which are placed outside of c. This is written down $(\vec{\mathbb{K}}, \lambda) \models \mathfrak{G}[c, ref]$ and defined as follows:

Definition 8. *Let* $(\vec{\mathbb{K}}, \lambda)$ *be a context-interpretation of an alphabet* $\mathcal{A}$ *and let* $\mathfrak{G} := (V, E, \nu, \top, Cut, area, \kappa, \rho)$ *be a concept graph. Inductively on the tree* $Cut \cup \{\top\}$, *we define* $(\vec{\mathbb{K}}, \lambda) \models \mathfrak{G}[c, ref]$ *for every cut* $c \in Cut \cup \{\top\}$ *and every partial valuation* $ref : V' \to \mathbb{K}_0$ *with* $V' \supseteq \bigcup\{area(d) \mid d \in Cut \cup \{\top\}, d > c\}$ *and* $V' \cap \overline{area}(c) = \emptyset$:
$(\vec{\mathbb{K}}, \lambda) \models \mathfrak{G}[c, ref] :\Longleftrightarrow$
ref can be extended to a partial valuation $\widetilde{ref} : V' \cup (V \cap area(c)) \to \mathbb{K}_0$ *(i.e.* $\widetilde{ref}(v) = ref(v)$ *for all* $v \in V'$*) such that the following conditions hold:*

- $\widetilde{ref}(v) \in Ext(\lambda_{\mathcal{C}}(\kappa(v)))$ *for each* $v \in V \cap area(c)$ *(vertex condition)*
- $\widetilde{ref}(e) \in Ext(\lambda_{\mathcal{R}}(\kappa(e)))$ *for each* $e \in E \cap area(c)$ *(edge condition)*
- $(\vec{\mathbb{K}}, \lambda) \not\models \mathfrak{G}[c', \widetilde{ref}]$ *for each* $c' \in Cut \cap area(c)$ *(iteration over* $Cut \cup \{\top\}$*)*

For $(\vec{\mathbb{K}}, \lambda) \models \mathfrak{G}[\top, \emptyset]$ *we write* $(\vec{\mathbb{K}}, \lambda) \models \mathfrak{G}$. *If we have two concept graphs* $\mathfrak{G}_1, \mathfrak{G}_2$ *such that* $(\vec{\mathbb{K}}, \lambda) \models \mathfrak{G}_2$ *for each* $\mathcal{A}$*-structure with* $(\vec{\mathbb{K}}, \lambda) \models \mathfrak{G}_1$, *we write* $\mathfrak{G}_1 \models \mathfrak{G}_2$.

Note that this definition (in particular the edge condition) relies on the condition that we consider concept graphs with dominating nodes only.

Now we are prepared to present the calculus for concept graphs.

4 The Calculus for Simple Concept Graphs with Cuts

The following calculus is based on the β-calculus of Peirce for existential graphs with lines of identity. More precisely: The first five rules of the calculus are a concept graph version of Peirce's β-calculus. The rules 'generalization' and 'specialization' encompass the orders on the concept- and relation names. The rules '$\top$- and *id*-Insertion' and '$\top$- and *id*-Erasure' are needed to comprehend the specific properties of the concept name $\top$ and the relation name *id*. For the sake of intelligibility, the whole calculus is described using common language. An appropriate mathematical definition as in [Da00] will be given in [Da01].

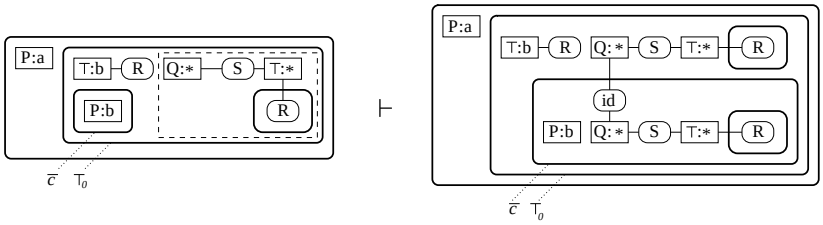
Definition 9. *The calculus for simple concept graphs with cuts over the alphabet $\mathcal{A}$ consists of the following rules:*

- **erasure**
  In positive cuts any directly enclosed edge, isolated vertex and closed subgraph may be erased.
- **insertion**
  In negative cuts any directly enclosed edge, isolated vertex and closed subgraph may be inserted.
- **iteration**
  Let $\mathfrak{G}_0$ be a subgraph of $\mathfrak{G}$ and let $\bar{c} \leq cut(\mathfrak{G}_0)$ be a cut that does not belong to $\mathfrak{G}_0$. Then a copy of $\mathfrak{G}_0$ may be inserted into $\bar{c}$. For every vertex v with $cut(v) = cut(\mathfrak{G}_0)$, an identity-link from v to its copy may be inserted.
- **deiteration**
  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 $cut^{-1}(c_2) = \{c_1\}$) may be inserted or erased.
- **generalization (unrestriction)**
  For evenly enclosed vertices and edges, their concept names or object names resp. their relation names may be generalized.
- **specialization (restriction)**
  For oddly enclosed vertices and edges, their concept names or object names resp. their relation names may be specialized.
- **isomorphism**
  A graph may be substituted by an isomorphic copy of itself.
- **$\top$- and *id*-Insertion**
  1. **$\top$-rule**
     For $g \in \mathcal{G} \cup \{*\}$, an isolated vertex $\boxed{\top : g}$ may be inserted in arbitrary cuts.

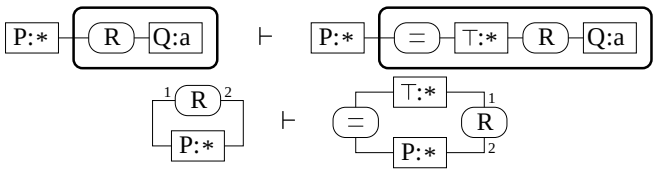
2. **identity**
   Let $g \in \mathcal{G}$, let $\boxed{P_1 : g}$, $\boxed{P_2 : g}$ be two vertices in cuts c_1, c_2, resp., and let $c \leq c_1, c_2$ be a cut. Then an identity-link between $\boxed{P_1 : g}$ and $\boxed{P_2 : g}$ may be inserted in c.
3. **splitting a vertex**
   Let $g \in \mathcal{G} \cup \{*\}$. Let $v = \boxed{P : g}$ be a vertex in cut c_0 and incident with relation edges $R_1, \ldots, R_n$, placed in cuts $c_1, \ldots, c_n$, resp.. Let c be a cut such that $c_1, \ldots, c_n \leq c \leq c_0$. Then the following may be done: In c, a new vertex $v' = \boxed{\top : g}$ and a new identity-link between v and v' is inserted. On $R_1, \ldots, R_n$, arbitrary instances of v are substituted by v'.
4. **congruence**
   Let $g \in \mathcal{G}$ and let $v = \boxed{P : g}$ be a vertex in cut c. Then the following may be done: In c, a new vertex $v' = \boxed{P : *}$ and a new identity link between v and v' is inserted. On every edge, every instance of v is substituted by v'.

– **$\top$- and *id*-Erasure**
The $\top$- and id-Insertion-rule may be reversed.

Example 1. Here is an example for the iteration-rule. Note that for one of the two possible vertices an identity link is inserted to its copy.



Below please find two examples for the $\top$- and *id*-Insertion rule 2 (splitting a vertex).



Here is an example for the $\top$- and *id*-Insertion rule 4 (congruence).

A very common operation for concept graphs is the *juxtaposition*. We ommit the mathematical definition here and describe only its graphical notation. The juxtaposition of a given set $\{\mathfrak{G}_i \mid 1 \leq i \leq n\}$ of graphs is simply writing them side by side: $\mathfrak{G}_1\ \mathfrak{G}_2\ \ldots\ \mathfrak{G}_n$. The juxtaposition of a empty set of graphs is the empty graph. The juxtaposition is needed to define the syntactical entailment relation:

Definition 10. *Let $\mathfrak{G}_a$, $\mathfrak{G}_b$ be two nonexistential concept graphs. Then* $\mathfrak{G}_b$ can be derived from $\mathfrak{G}_a$ *(which is written $\mathfrak{G}_a \vdash \mathfrak{G}_b$) if there is a finite sequence $(\mathfrak{G}_1, \mathfrak{G}_2, \ldots, \mathfrak{G}_n)$ with $\mathfrak{G}_1 = \mathfrak{G}_a$ and $\mathfrak{G}_b = \mathfrak{G}_n$ such that each $\mathfrak{G}_{i+1}$ is derived from $\mathfrak{G}_i$ by applying one of the rules of the calculus. The sequence is called* a proof for $\mathfrak{G}_a \vdash \mathfrak{G}_b$).

If $\{\mathfrak{G}_i \mid i \in I\}$ is a (possibly empty) set of nonexistential concept graphs, then a graph $\mathfrak{G}$ can be derived from $\{\mathfrak{G}_i \mid i \in I\}$ *if there is a finite subset $\{\mathfrak{G}_1, \ldots, \mathfrak{G}_n\} \subseteq \{\mathfrak{G}_i \mid i \in I\}$ with $\mathfrak{G}_1 \ldots \mathfrak{G}_n \vdash \mathfrak{G}$.*

5 The Syntactical Equivalence Between CG and FOL

Another way to understand concept graphs if to translate them to formulas of first order logic. Mappings which translate conceptual graphs to formulas of predicate logic (first or higher order) are usually denoted by Φ. In this work, we give a mathematical definition for an operator Φ which translates concept graphs to formulas of first order logic with equality and with relation names (i.e. the concept and relation names of $\mathcal{A}$) but without function names. The set of these formulas over $\mathcal{A}$ is denoted by $\mathrm{FOL}^{\mathcal{A}}$ or, even simpler, by FOL. For a formula f, the set $\mathrm{Free}(f)$ of the *free variables in* f is defined as usual.

To define the mapping $\Phi : \mathrm{CG} \to \mathrm{FOL}$, let $\mathfrak{G} := (V, E, \nu, \top, Cut, area, \kappa, \rho)$ be a simple concept graph with cuts, variables, and dominating nodes over the alphabet $(\mathcal{G}, \mathcal{C}, \mathcal{R})$. First we define $\mathrm{Free}(\mathfrak{G}) := \{\alpha \in \mathrm{Var} \mid \text{there is a } v \text{ with } \rho(v) = *_\alpha\}$. Then we assign a new variable $\alpha_v \notin \mathrm{Free}(\mathfrak{G})$ to each vertex $v \in V$ with $\rho(v) = *$ so that we can now define the following mapping Φ_t on V:

$$\Phi_t(v) := \begin{cases} \alpha_v \text{ for } \rho(v) = * \\ \alpha \text{ for } \rho(v) = *_\alpha \text{ and } \alpha \in \mathrm{Var} \\ a \text{ for } \rho(v) = a \text{ for } a \in \mathcal{G} \end{cases}$$

Finally we can define the mapping $\Phi : \mathrm{CG} \to \mathrm{FOL}$ inductively on the tree $Cut \cup \{\top\}$. So let $c \in Cut \cup \{\top\}$ be an arbitrary cut. First we define a formula f which encodes all edges and vertices which are directly enclosed by c. If c does not directly enclose any edges or vertices, simply set $f := (\exists x.x = x)$. Otherwise let f be the conjunction of the atomic formulae

$$\begin{aligned} \kappa(w)\underline{(}\Phi_t(w)\underline{)} &\text{ with } w \in V \cap area(c), \\ \Phi_t(w_1)\underline{=}\Phi_t(w_2) &\text{ with } k \in E_{id} \cap area(c) \text{ und } \nu(k) = (w_1, w_2), \quad \text{and} \\ \kappa(e)\underline{(}\Phi_t(w_1)\underline{,}\ldots\underline{,}\Phi_t(w_j)\underline{)} &\text{ with } e \in E_{nonid} \cap area(c) \text{ and } \nu(e) = (w_1, \ldots, w_j). \end{aligned}$$

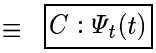
(The signs which have to be understood literally are underlined. For example, the first formula is the sequence of signs which consists of the result of the evaluation of $\kappa(w)$, a left bracket, the result of the evaluation of $\Phi_t(w)$ and a right bracket.)

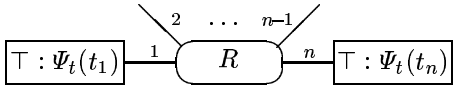
Let $v_1, \ldots, v_n$ be the vertices $\mathfrak{G}$ wich are enclosed by c and which fulfill $\rho(v_i) = *$, and let $area(c) \cap Cut = \{c_1, \ldots, c_l\}$ (by induction we already assigned formulas to these cuts). If l=0, set $\Phi(c) := \underline{\exists}\alpha_{v_1} \ldots \underline{\exists}\alpha_{v_n}.f$,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

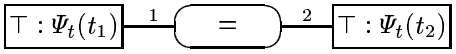
$$\Phi(c) := \underline{\exists}\alpha_{v_1} \ldots \underline{\exists}\alpha_{v_n}.\underline{(}f\underline{\wedge\neg}\Phi(c_1)\underline{\wedge}\ldots\underline{\wedge\neg}\Phi(c_l)\underline{)} \quad ,$$

and the definition of $\Phi : \mathrm{CG} \to \mathrm{FOL}$ is finished.

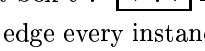
Now we want to give a definition for a mapping $\Psi : \mathrm{FOL} \to \mathrm{CG}$ in the opposite direction. The formulas of FOL are built up inductively, hence the definition of Ψ is done inductively too. To start we set the images of the terms of FOL by a mapping Ψ_t: We set $\Psi_t(C) \equiv C$ for concept names $C \in \mathcal{C}$ and $\Psi_t(\alpha) \equiv *_\alpha$ for variables $\alpha \in \mathrm{Var}$. Now we can define Ψ inductively on the composition of formulas (but we only give graphical descriptions, because the mathematical definition of Ψ is rather technical and therefore omitted here).

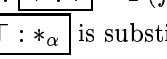
- $g \equiv c(t)$ for a term t and a concept name $C \in \mathcal{C}$: $\Psi(C(t)) \equiv$ 

- $R(t_1, \ldots, t_n)$ for a n-ary relation name R and terms t_1, ..., t_n:
  $\Psi(R(t_1, \ldots, t_n)) \equiv$ 

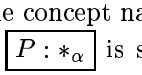
- $t_1 = t_2$ for two terms t_1, t_2:
  $\Psi(t_1 = t_2) \equiv$ 

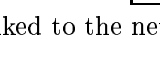
- $f_1 \wedge f_2$ for two formulas f_1 and f_2: $\Psi(f_1 \wedge f_2) \equiv \Psi(f_1) \quad \Psi(f_2)$
  (i.e. the juxtaposition of $\Psi(f_1)$ and $\Psi(f_2)$).
- $\neg f$ for a formula f: $\Psi(\neg f) \equiv$ 

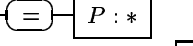
- $\exists \alpha(f)$ for a formula f and a variable α:
  If $\alpha \notin \mathrm{Free}(f)$, let $\Psi(\exists \alpha f) \equiv \Psi(f)$
  For $\alpha \in \mathrm{Free}(f)$, the following steps have to be taken:
  1. A concept box $v :=$ 

 is drawn besides $\Psi(f)$: 

 $\Psi(f)$
  2. On every edge every instance of a concept box 

 is substituted by the new concept box v (note that the concept boxes which are incident with an edge can only bear the concept name $\top$).
  3. Every (isolated) concept box 

 is substituted by a concept box 

, which is linked to the new concept box v with an identity link:
  v — 

  4. All concept boxes

 are erased.

Note that Ψ translates formulas without free variables to simple concept graphs with negations and dominating nodes.

To illustrate the procedures in Φ and Ψ, we give a little example. A well-known example for translating first order logic to existential graphs is the formula which expresses that a binary relation F is a (total) function. This formula is written down only by using the $\exists$-quantor and the junctors $\wedge$ and $\neg$. Please see below:

$$f :\equiv \neg\exists x.\neg\exists y.(xFy \wedge \neg\exists z(xFz \wedge \neg(y = z)))$$

The translations of this formula into a concept graph (by Φ) and into an existential graph are shown in Figure 2. Note the structural similarities of these

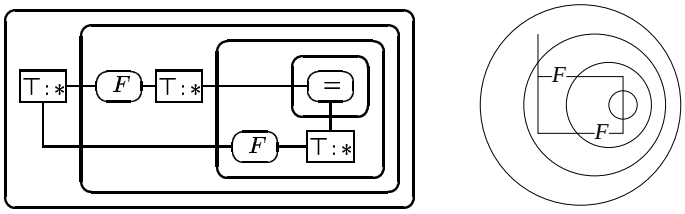


Fig. 2. the concept graph $\Psi(f)$ and the appropriate existential graph for f

two different kinds of graphs. Now the concept graph in Figure 2 can be translated back to a first order logic formula by the mapping Φ. One possible result (depending on the chosen variables and to the order of the subformulas) is:

$$\Phi(\Psi(f)) \equiv \neg\exists x.(\top(x) \wedge \neg\exists y.(\top(y) \wedge xFy \wedge \neg\exists.z(\top(z) \wedge xFz \wedge \neg(\wedge y = z))))$$

If we erase all subformulas $\top(\ldots)$ of this formula, we get f again. In particular, we have $f \vdash \Phi(\Psi(f))$ and $\Phi(\Psi(f)) \vdash f$ (see Theorem 1).

Assume that we have a sound and complete calculus $\vdash$ on FOL. To encompass the orders on the concept- and relation names and the specific properties of the concept name $\top$, we add the following axioms to the calculus:

- $\forall x.\top(x)$
- $\forall x.(C_1(x) \to C_2(x))$ for two concept names $C_1, C_2 \in \mathcal{C}$ with $C_1 \leq_{\mathcal{C}} C_2$
- $\forall x_1 \ldots \forall x_n.(R_1(x_1, \ldots, x_n) \to R_2(x_1, \ldots, x_n))$ for two n-ary relation names R_1, R_2 with $R_1 \leq_{\mathcal{R}} R_2$

Now we have reached the following situation. We have two logical systems with a calculus $\vdash$, and two mappings between them:

$$(\mathrm{CG}^{\mathcal{A}}, \vdash) \quad \underset{\Psi}{\overset{\Phi}{\rightleftarrows}} \quad (\mathrm{FOL}^{\mathcal{A}}, \vdash)$$

It is well accepted that existential graphs and hence simple conceptual graphs with negations are equivalent to first order logic. The meaning of this equivalence shall be elaborated now.

Baader, Molitor and Tobies gave mathematical definitions for Φ and Ψ and a proof for $\Phi(\Psi(f)) \equiv f$ for each formula $f \in$ FOL (see [BMT98]). This is an important part in the equivalence between FOL and CG, but it is in our view not sufficient. To comprehend this, note that *every* injective mapping $\tilde{\Psi} :$ FOL $\to$ CG and its inverse mapping $\tilde{\Phi} := \tilde{\Psi}^{-1} :$ CG $\to$ FOL fulfill $\tilde{\Phi}(\tilde{\Psi}(f)) \equiv f$. So the condition $\Phi(\Psi(f)) \equiv f$ seems not to capture the whole meaning of the statement 'simple conceptual graphs with cuts are equivalent to first order logic'. The crucial point is that FOL and CG are not only sets of graphs resp. formulas but sets which are quasiordered by their particular syntactical entailment relations $\vdash$ (i.e. each relation $\vdash$ is reflexive and transitive). Of course, one expects that Φ and Ψ respect these entailment relations. In fact Φ and Ψ are quasiorder-isomorphisms between $(\mathrm{CG}, \vdash)$ and $(\mathrm{FOL}, \vdash)$ which are mutually inverse. This is captured by the following crucial theorem:

Theorem 1 (Main Syntactical Theorem for the Mappings Φ and Ψ).
Let $\mathfrak{G}$, $\mathfrak{G}_1$ and $\mathfrak{G}_2$ be concept graphs over $\mathcal{A}$ and let f, f_1, f_2 be FOL-formulas over $\mathcal{A}$. Then the following implications hold:

1) $\mathfrak{G}_1 \vdash \mathfrak{G}_2 \quad \Rightarrow \quad \Phi(\mathfrak{G}_1) \vdash \Phi(\mathfrak{G}_2)$
2) $f_1 \vdash f_2 \quad \Rightarrow \quad \Psi(f_1) \vdash \Psi(f_2)$
3) $\mathfrak{G} \vdash \Psi(\Phi(\mathfrak{G}))$ *and* $\Psi(\Phi(\mathfrak{G})) \vdash \mathfrak{G}$
4) $f \vdash \Phi(\Psi(f))$ *and* $\Phi(\Psi(f)) \vdash f$

Instead of giving the proof for this theorem here (the proof for it will be in [Da01]), we want to emphasize that four conditions of the theorem are logically independent, e.g. none of the conditions can be derived from the remaining three (hence all conditions have to be proven seperately). To illustrate this, we provide two extremely simple examples. Each of these examples consists of two very small ordered sets (which represent for the quasiordered sets $(\mathrm{CG}^{\mathcal{A}}, \vdash)$ and $(\mathrm{FOL}^{\mathcal{A}}, \vdash)$) and two mappings between them (which represent the mappings Φ and Ψ).

To see that 1) cannot be derived from 2)-4) have a look at the two ordered sets in the left example of Figure 3, each consiting of two elements and the two mappings between them. Note that the mappings are mutually invers, therefore the conditions 3) and 4) are fulfilled. But only one mapping is order-preserving, hence 2) is fulfilled but 1) is not. Because we have one example which satisfies 2)-4), but not 1), we are done. Analogously it can be shown that 2) cannot be derived from 1), 3) and 4).

To see that 3) cannot be derived from 1),2) and 4), consider the two ordered sets with their mappings between them in the right example of Figure 3. It is easy to see that this example satisfies the conditions 1), 2) and 4), but does not satisfy 3), hence we are done again. In turn again it can be shown analogously that 4) cannot be derived from 1)–3).

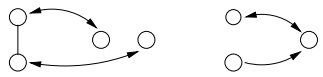


Fig. 3. two examples for Theorem 1

6 The Semantical Equivalence Between FOL and CG

In Section 3 we have introduced a semantics for concept graphs. Now we proceed with the well-known semantics for first order logic. Usually the models for FOL-formulas are not power context families, but relational structures. Relational structures are pairs $\mathcal{M} = (M, I)$, consisting of a *universe* M and a function I with $I : \mathcal{G} \to M$, $I : \mathcal{C} \to \mathfrak{P}(M)$ and $I : \mathcal{R}_k \to \mathfrak{P}(M^k)$ for each k. Relational structures are obviously closely related to power context families. Roughly spoken: If we remove all intensional information from a power context family, we get a relational structure. To put it formally:

Definition 11. *If $(\vec{\mathbb{K}}, \lambda)$ is a $\vec{\mathbb{K}}$-interpretation, define $\mathcal{M}(\vec{\mathbb{K}}, \lambda) := (M, I)$ as follows: $M := G_0$, $I(G) := \lambda_{\mathcal{G}}(G)$ for all $G \in \mathcal{G}$, $I(C) := Ext(\lambda_{\mathcal{G}}(C))$ for all*

$C \in \mathcal{C}$ *and* $I(R) := Ext(\lambda_{\mathcal{G}}(R))$ *for all* $R \in \mathcal{R}$. *The relational structure* $\mathcal{M}(\vec{\mathbb{K}}, \lambda)$ *is called the* relational structure of $(\vec{\mathbb{K}}, \lambda)$.

If we look back to the definition of the relation $\models$ between power context families and concept graphs, in particular to the vertex condition and edge condition of Definition 8, we realize that only the extensions of formal concepts were checked. This yields the following lemma:

Lemma 1. *For a* $\mathcal{A}$*-structure* $(\vec{\mathbb{K}}, \lambda)$ *and a simple concept graph with cuts* $\mathfrak{G}$, *we have* $(\vec{\mathbb{K}}, \lambda) \models \mathfrak{G} \iff \mathcal{M}(\vec{\mathbb{K}}, \lambda) \models \Phi(\mathfrak{G})$

The proof for this lemma will be given in [Da01]. Now we are ready to connect syntax and semantics both for FOL and CG. Remember that we assumed to have a sound and complete calculus $\vdash$ on FOL, i.e. we have

5) $f_1 \vdash f_2 \iff f_1 \models f_2$.

Furthermore Lemma 1 yields an equivalence between the two $\models$-relations on FOL and CG. If we resume these facts, we get the following theorem:

Theorem 2 (Main Semantical Theorem for the Mappings Φ and Ψ). *Let* $\mathfrak{G}_1$ *and* $\mathfrak{G}_2$ *be concept graphs over* $\mathcal{A}$ *and let* f_1 *and* f_2 *be FOL-formulas over* $\mathcal{A}$. *Then the following equivalences hold:*

6) $\mathfrak{G}_1 \models \mathfrak{G}_2 \iff \Phi(\mathfrak{G}_1) \models \Phi(\mathfrak{G}_2)$

The proof for this theorem is by far not as extensive as the proof for the main syntactical theorem. But the proof has to be done, since conditions 5) and 6) are logically independent from the conditions 1)-4) of Theorem 1. This seems to be evident, since in condititions 1)-4) only the relations $\vdash$ on FOL and CG appear. But to show exactly that all conditions 1)-6) are logically independent, we need three little considerations.

1. To see that none of the conditions 1)-4) can be derived from the remaining conditions, simply *define* the relations $\models$ on FOL and CG as follows: For two formulas f_1 and f_2 define $f_1 \models f_2 :\iff f_1 \vdash f_2$. For two concept graphs $\mathfrak{G}_1$ and $\mathfrak{G}_2$, define $\mathfrak{G}_1 \models \mathfrak{G}_2 :\iff \Phi(\mathfrak{G}_1) \vdash (\mathfrak{G}_2)$. Obviously, 5) and 6) hold for arbitrary relations $\vdash$ on FOL and CG. Hence the indepence results for 1)-4) after Theorem 1 yield appropriate independence results for 1)-6). E.g. 1) cannot be derived from 2)-6).
2. To see that 5) cannot be derived from 1)-4) and 6), define $\models$ on FOL and CG as follows: For two formulas f_1 and f_2 always set $f_1 \models f_2$, and for two concept graphs $\mathfrak{G}_1$ and $\mathfrak{G}_2$ always set $\mathfrak{G}_1 \models \mathfrak{G}_2$ too. Since $\models$ does not appear in the condititions 1)-4), these conditions still hold, and it is easy to see that 6) is fulfilled, too, but 5) is not satisfied.
3. To see that 6) cannot be derived from 1)-5), define $\models$ on FOL and CG as follows: For two formulas f_1 and f_2 define $f_1 \models f_2 :\iff f_1 \vdash f_2$. For two concept graphs $\mathfrak{G}_1$ and $\mathfrak{G}_2$ always set $\mathfrak{G}_1 \not\models \mathfrak{G}_2$. Again it is easy to see that 1)-5) hold, but 6) does not.

To summarize the above argument: If we want a full syntactical and semantical equivalence, we have to prove all six conditions 1)-6) from Theorems 2 and 1. But once we have done this, we immediately get the soundness and completeness for the calculus on concept graphs:

Theorem 3 (soundness and completeness for concept graphs).
Let $\mathfrak{G}_1$ and $\mathfrak{G}_2$ be concept graphs over $\mathcal{A}$. Then it holds:

$$\mathfrak{G}_1 \vdash \mathfrak{G}_2 \quad \Longleftrightarrow \quad \mathfrak{G}_1 \models \mathfrak{G}_2$$

Proof: $\mathfrak{G}_1 \vdash \mathfrak{G}_2 \overset{1)-4)}{\Longleftrightarrow} \Phi(\mathfrak{G}_1) \vdash \Phi(\mathfrak{G}_2) \overset{5)}{\Longleftrightarrow} \Phi(\mathfrak{G}_1) \models \Phi(\mathfrak{G}_2) \overset{6)}{\Longleftrightarrow} \mathfrak{G}_1 \models \mathfrak{G}_2$ □

References

[BMT98] F. Baader, R. Molitor, S. Tobies: The Guarded Fragment of Conceptual Graphs. RWTH LTCS-Report. `http://www-lti.informatik.rwth-aachen.de/Forschung/Papers.html`

[Da00] F. Dau: Negations in Simple Concept Graphs, in: B. Ganter, G. W. Mineau (Eds.): Conceptual Structures: Logical, Linguistic, and Computational Issues. Lectures Notes in Artificial Intelligence 1867, Springer Verlag, Berlin–New York 1997, 263–276.

[Da01] F. Dau, Negations in Concept Graphs. PhD-Thesis. To appear.

[GW99a] B. Ganter, R. Wille: Formal Concept Analysis: Mathematical Foundations. Springer, Berlin-Heidelberg-New York 1999.

[Pe98] C. S. Peirce: Reasoning and the Logic of Things. The Cambridge Conferences Lectures of 1898. Ed. by K. L. Kremer, Harvard Univ. Press, Cambridge 1992.

[Pr98a] S. Prediger: Kontextuelle Urteilslogik mit Begriffsgraphen. Ein Beitrag zur Restrukturierung der mathematischen Logik, Shaker Verlag 1998.

[Pr98b] S. Prediger: Simple Concept Graphs: A Logic Approach, in: M, -L. Mugnier, M. Chein (Eds.): Conceptual Structures: Theory, Tools and Applications, Springer Verlag, Berlin–New York 1998, 225–239.

[Ro73] D. D. Roberts: The Existential Graphs of Charles Sanders Peirce, Mouton The Hague – Paris 1973.

[So84] J. F. Sowa: Conceptual Structures: Information Processing in Mind and Machine. Addison Wesley Publishing Company Reading, 1984.

[So99] J. F. Sowa: Conceptual Graphs: Draft Proposed American National Standard, in: W. Tepfenhart, W. Cyre (Eds.): Conceptual Structures: Standards and Practices, Springer Verlag, Berlin–New York 1999, 1-65.

[So00] J. F. Sowa: Knowledge Representation: Logical, Philosophical, and Computational Foundations. Brooks Cole Publishing Co., Pacific Grove, CA, 2000.

[We95] M. Wermelinger: Conceptual Graphs and First-Order Logic, in: G. Ellis et al. (Eds.): Conceptual Structures: Applications, Implementations and Theory, Springer Verlag, Berlin–New York 1995, 323–337.

[Wi97] R. Wille: Conceptual Graphs and Formal Concept Analysis, in: D. Lukose et al. (Hrsg.): Conceptual Structures: Fulfilling Peirce's Dream, Lectures Notes in Artificial Intelligence 1257, Springer Verlag, Berlin–New York 1997, 290–303.