Query Graphs with Cuts: Mathematical Foundations

Frithjof Dau

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

Abstract. Query graphs with cuts are inspired by Sowa's conceptual graphs, which are in turn based on Peirce's existential graphs. In my thesis 'The Logic System of Concept Graphs with Negations', conceptual graphs are elaborated mathematically, and the cuts of existential graphs are added to them. This yields the system of concept graphs with cuts. These graphs correspond to the closed formulas of first order predicate logic. Particularly, concept graphs are propositions which are evaluated to truth-values. In this paper, concept graphs are extended to so-called query graphs, which are evaluated to relations instead. As the truth-values TRUE and FALSE can be understood as the two 0-ary relations, query graphs extend the expressiveness of concept graphs.
Query graphs can be used to elaborate the logic of relations. In this sense, they bridge the gap between concept graphs and the Peircean Algebraic Logic, as it is described in Burch's book 'A Peircean Reduction Thesis'. But in this paper, we focus on deduction procedures on query graphs, instead of operations on relations, which is the focus in PAL. Particularly, it is investigated how the adequate calculus of concept graphs can be transferred to query graphs.

1 Introduction and Overview

At the dawn of modern logic, two important diagrammatic systems for mathematical logic have been developed. One of them is Frege's Begriffsschrift. The ideas behind the Begriffsschrift had an influence on mathematics which can hardly be underestimated, but the system itself had never been used in practice. The other diagrammatic system are Peirce's existential graphs, which are unfortunately not known by many mathematicians. Nonetheless, a lot of research has been done on existential graphs, and they have influenced other diagrammatic systems as well. Among these, Sowa's system of conceptual graphs, which are based on Peirce's existential graphs and the semantic networks of artificial intelligence. is the most important. Their purpose is 'to express meaning in a form that is logically precise, humanly readable, and computationally tractable' (see [30]). In fact, conceptual graphs yield a powerful diagrammatic system with a higher expressiveness than existential graphs. But a closer observation shows that their definitions lack (mathematical) preciseness, which leads to several ambiguities, gaps and flaws (see [7]).

In order to fix these gaps and flaws, a mathematical elaboration of conceptual graphs is appropriate. Wille, who is like Sowa strongly influenced by the philosophy of Peirce, introduced in [37] an approach for such an elaboration, combining Sowa's graphs and his theory of Formal Concept Analysis. The resulting graphs are called concept graphs (and they are a crucial part of Wille's Contextual Logic, see [36, 38]). Until today, several systems of concept graphs with different kinds of negations, quantifiers etc. have been developed (an overview of these systems can be found in [10]). The system which will be used in this paper are the concept graphs with cuts (CGwCs), which have the expressiveness of first order predicate logic and are studied in detail by the author in [7].

Let us consider an example for existential graphs and concept graphs with cuts:

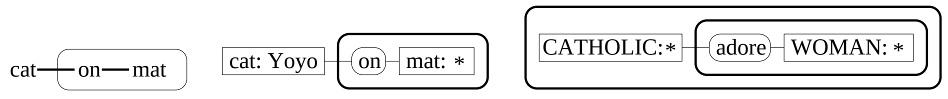


Fig. 1. An existential graph, a similar concept graph and a second concept graph

The leftmost graph is an existential graph with the meaning ‘there is a cat which is not on any mat’. It is composed of predicates of different arities (cat, on, mat), so-called *lines of identity* which stand for objects and which are drawn bold, and finally of closed curves, so-called *cuts*, which are used to negate the enclosed subgraph.

The graph in the middle is a concept graph with cuts. Instead of lines of identity, concept boxes are used. These boxes contain a concept name and a referent. 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 similar to a wildcard in computer systems). Besides the generic marker, object names are allowed as referents as well. The ovals between concept boxes represent relations between the referents of the concept boxes. The cuts of existential graphs appear in concept graphs with cuts as well (as the name says). But now, as they should not be confused with relation ovals, they are drawn bold. The meaning of this graph is ”Yoyo is a cat and it is not true that there is a mat such that Yoyo is on this mat”, or ‘the cat Yoyo is not on any mat’ for short.

On the right, we have a slightly more complex concept graph with cuts. As in existential graphs, it is allowed to iterate or nest the cuts (but cuts may not overlap). The meaning of this graph is ”it is not true that there is a catholic, but there is no woman this catholic adores”, or ”every catholic adores some woman” for short.[1]

Existential graphs and concept graphs are a diagrammatic form of propositions.[2] It is well known that Peirce developed a *logic of relations* as well, and the graphical notation of existential graphs can be used for describing relations as well. Burch elaborated in his book ‘A Peircean Reduction Thesis’ ([2]) Peirce’s algebra of relations, the so-called *Peircean Algebraic Logic (PAL)*. But, although the development of PAL is driven by the diagrammatic representation of relations, Burch developed a *linear* notion for PAL and explains not until the last chapter of his book how this linear notion is related to its diagrammatic representation. For the framework of Contextual Logic and inspired by the work of Burch, Pollandt and Wille invented and investigated the so-called *relation graphs* which represent relations (see [20, 21, 39]). The left graph of Fig. 2 is a relation graph describing the relation `is_stepmother_of`.

The free (or, in other words, unsaturated) valences of the relation correspond to so-called *pending edges* of the relation graphs, which are drawn as labelled lines of identity (see [21]). For concept graph with cuts, a small syntactical extension allows us to represent free valences of a relation: In addition to object names and the generic

[1] This example is adopted from Peirce.

[2] More precisely: Of *judgements*, which are *asserted* propositions. But this distinction shall not be discussed here.

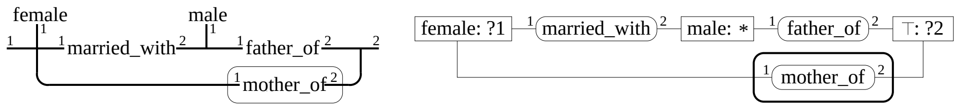


Fig. 2. A relation graph and the corresponding query graph

marker, numbered question marks called *query markers* are allowed to be referents of concept boxes. The resulting graphs are termed *query graph with cuts (QGwCs)*. The right graph of Fig. 2 is therefore a QGwC. It describes the relation of all pairs of objects (o_1, o_2), which can replace the query markers ?1 and ?2, respectively, such that we obtain a valid concept graph (the concept name $\top$ denotes the universal concept which contains every object –of the respective universe of discourse– in its extension).

Pollandt and Wille focus on *operations* on relation graphs, that is, they are interested in the *algebra* of relations. In contrast to that, we consider *derivations* on graphs, i.e., our focus is the *logic* of relations. This logic will be elaborated in the following sections. In the first section, the basic definitions for query graphs with cuts are provided. In the next section, we describe a direct extensional semantics for query graphs. In Sec. 4, it is investigated how the calculus for concept graph with cuts can be extended for query graphs. In Sec. 5, the class of query graphs is restricted so that they better fit to the relation graphs of Burch, Pollandt and Wille. This requires further investigations on the calculus. Finally, an outlook for further research is given.

2 Basic Definitions for Query Graphs

As discussed in the introduction, a drawback of conceptual graphs is a lack of mathematical preciseness, which leads to ambiguities and flaws in the system of conceptual graphs. The purpose of query graphs with cuts is to elaborate a diagrammatic system for the mathematical logic of relations. This elaboration is done as usual in mathematical logic, that is: We have to provide a syntax for the graphs, a semantics, and a calculus which is sound and complete. Particularly, syntax, semantics, and the calculus have to be defined mathematically.

Not every reader will be familiar with the use of mathematical notions. Moreover, 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 structured as follows: In this section, the necessary mathematical definitions for the syntax of QGwCs are given, and it is explained why these definitions capture the intuition behind QGwCs, so that readers who are not trained in reading mathematical definitions hopefully get an idea how these definitions work. In the following sections, mathematical notations are avoided as much as possible. For those readers who are interested in the mathematical theory behind this paper, an extended version of it is provided at the homepage of the author (see the remarker after the bibliography) which contains all further definitions and proofs.

We start with the definition of the underlying structures of concept graphs with cuts and query graphs with cuts. The examples in Fig. 1 and Fig. 2 show that these graphs are "networks" of boxes, relation ovals, and cuts. We see that relation ovals "connect" the boxes (but we have no direct connection of boxes). The boxes and relation ovals are "grouped" by cuts, i.e., cuts contain boxes and relation ovals. Cuts may even contain other cuts, as the last example of Fig. 1 shows, but tey may not intersect. Besides the boxes, relation ovals, and cuts, 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 (e.g., this gives us the possibility to say that each box, relation oval, or cut is contained by exactly one cut or the sheet of assertion). These conditions will be captured mathematically by the following definition.

Definition 1 (Relational Graphs with Cuts).
A structure $(V, E, \nu, \top, Cut, area)$ *is called a* relational graph with cuts[3] *if*

1. V*,* E *and* Cut *are pairwise disjoint, finite sets whose elements are called* vertices, edges *and* cuts*, respectively,*
2. $\nu : E \to \bigcup_{k\in\mathbb{N}} V^k$ *is a mapping*[4],
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*[5]
   a) $c_1 \neq c_2 \Rightarrow area(c_1) \cap area(c_2) = \emptyset$,
   b) $V \cup E \cup Cut = \bigcup_{d\in Cut\cup\{\top\}} area(d)$,
   c) $c \notin area^n(c)$ *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* $\nu(e)\big|_i := v_i$*. Sometimes, we also write* $e\big|_i$ *instead of* $\nu(e)\big|_i$*, and* $e = (v_1, \ldots, v_k)$ *instead of* $\nu(e) = (v_1, \ldots, v_k)$*. We set* $E^{(k)} := \{e \in E \mid |e| = k\}$*.*

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* $c = area^{-1}(x)$ *for every* $x \in area(c)$*, or even more simple and suggestive:* $c = cut(x)$*.*

In particular the empty graph, i.e. the empty sheet of assertion, exists. Its form is $\mathfrak{G}_\emptyset := (\emptyset, \emptyset, \emptyset, \top, \emptyset, \emptyset)$.

Def. 1 is an abstract definition of graphs which does not try to capture any graphical properties of the diagrams. Instead, 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 [8] and [14]). An example for a relational graph with cuts and its representation will be provided after the next definition.

In contrast to *linear* notations of logic, there is no need to define the graphs *inductively.* Nonetheless, similar to formulas, relational graphs 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 implementation of this idea.

Definition 2 (Ordering on Contexts and Enclosing Relation).
Let $(V, E, \nu, \top, Cut, area)$ *be a relational graph with cuts. We define a mapping* $\beta : V \cup E \cup Cut \,\dot\cup\, \{\top\} \to Cut \,\dot\cup\, \{\top\}$ *by*

$$\beta(x) := \begin{cases} x \text{ for } x \in Cut \,\dot\cup\, \{\top\} \\ cut(x) \text{ for } x \in V \cup E \end{cases} ,$$

[3] Please do not mistake relation graphs and relational graphs.
[4] We set $\mathbb{N} := \{1, 2, 3, \ldots\}$ and $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$.
[5] The sign $\dot\cup$ denotes the *disjoint* union.

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 [7], we get the following lemma:

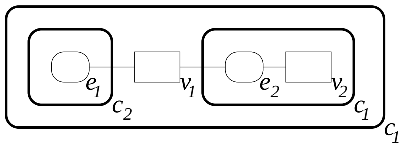
Lemma 1. *For a relational graph with cuts* $(V, E, \nu, \top, Cut, area)$*,* $\leq$ *is a quasiorder. Furthermore,* $\leq|_{Cut \dot\cup \{\top\}}$ *is an order on* $Cut \,\dot\cup\, \{\top\}$ *which is a tree with the sheet of assertion* $\top$ *as greatest element.*

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, many proofs are carried out inductively over the construction of formulas. Although graphs are not defined inductively, Lem. 1 now allows us to do inductive definitions and proofs as well.

Of course the preceding lemma is not surprising: It had to be expected. But as the results, which are clear from a naive understanding of concept graphs, can be *proven*, the lemma indicates that Def. 1 and Def. 2 are a 'correct' mathematization of the underlying structure of query graphs with cuts.

The following figure provides a simple example for the last definitions. In the first two lines, a relational graph with cuts $\mathfrak{G}$ is defined (of course, $\{v_1, v_2\}$, $\{e_, e_2\}$,$\{c_1, c_2\}$, $\{\top\}$ are disjoint sets with $v_1 \neq v_2$, $e_1 \neq e_2$ and $c_1 \neq c_2$)). Below, a graphical representation of $\mathfrak{G}$ and of its order $\leq$ is depicted.

$$\begin{aligned}\mathfrak{G} := (&\{v_1, v_2\}, \{e_1, e_2\}, \{(e_1, (v_1)), (e_2, (v_1, v_2))\}, \top, \{c_1, c_2, c_3\},\\ &\{(\top, \{c_1\}), (c_1, \{v_1, c_2, c_3\}), (c_2, \{e_1\}), (c_3, \{v_2, e_2\})\})\end{aligned}$$

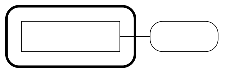


$$\begin{tikzcd}
 & \top \arrow[d, dash] & \\
 & c_1, v_1 \arrow[dl, dash] \arrow[dr, dash] & \\
c_2, e_1 & & c_3, v_2, e_2
\end{tikzcd}$$

As this example shows, the vertices are usually drawn as boxes, and edges are drawn as 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. In order to distinguish such lines from each other, Peirce introduced a device he called a 'bridge' or 'frog' (see [24], p. 55). But, except for bridges between lines, all the boxes, ovals, and lines of a graph must not intersect. Finally, a cut is drawn as a closed 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). In order to distinguish the curves of cuts from relation ovals, they are drawn bold.

In our example, the cut c_1 is directly enclosed by the sheet of assertion $\top$, the cuts c_2, c_3 and the vertex v_1 are directly enclosed by the cut c_1, the edge e_1 is directly enclosed by the cut c_2, and v_2 and e_2 are directly enclosed by the cut c_3.

The following graph represents another relational graph with cuts:



In this graph, we have an edge with a incident and deeper nested vertex. In the semantics for QGwCs, it will turn out that graphs with this property may cause troubles (we will come back to this point in Sec. 3). Thus, we have to forbid graphs of this kind. This is captured by the following definition:

Definition 3 (Dominating Nodes).
If $cut(e) \leq cut(v)$ $(\Leftrightarrow e \leq v)$ *for every* $e \in E$ *and* $v \in V_e$*, then* $\mathfrak{G}$ *is said to have* dominating nodes*.*

Now QGwCc are be obtained from relational graphs by additionally labelling the vertices and edges with names for objects, concepts, and relations. We first define the underlying alphabet for our graphs, then QGwCs are defined.

Definition 4 (Alphabet).
An alphabet *is a triple* $\mathcal{A} := (\mathcal{G}, \mathcal{C}, \mathcal{R})$ *of disjoint sets* $\mathcal{G}$, $\mathcal{C}$, $\mathcal{R}$ *such that*

- $\mathcal{G}$ *is a finite set whose elements are called* object names,[6]
- $(\mathcal{C}, \leq_{\mathcal{C}})$ *is a finite ordered set with a greatest element* $\top$ *whose elements are called* concept names*, and*
- $(\mathcal{R}, \leq_{\mathcal{R}})$ *is a family of finite ordered sets* $(\mathcal{R}_k, \leq_{\mathcal{R}_k})$, $k = 1, \ldots, n$ *(for an* $n \in \mathbb{N}$*) whose 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* $\mathcal{G} \,\dot{\cup}\, \{*\}$*, but all elements of* $\mathcal{G}$ *are incomparable.*

Definition 5 (Query Graphs with Cuts).
A structure $\mathfrak{G} := (V, E, \nu, \top, Cut, area, \kappa, \rho)$ *is called* query graph with cuts over the alphabet $\mathcal{A}$*, when*

- $(V, E, \nu, \top, Cut, area)$ *is a relational graph with cuts that has dominating nodes,*
- $\kappa : V \cup E \to \mathcal{C} \cup \mathcal{R}$ *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*
- $\rho : V \to \mathcal{G} \,\dot{\cup}\, \{*\} \,\dot{\cup}\, \{?i \,|\, i \in \mathbb{N}\}$ *is a mapping such that there exists a natural number* $ar(\mathfrak{G}) \in \mathbb{N}_0$ *with* $\{i \,|\, \exists v \in V \text{ with } \rho(v) = ?i\} = \{1, \ldots, ar(\mathfrak{G})\}$*. The number* $ar(\mathfrak{G})$ *is called the* arity of $\mathfrak{G}$ *.*

If $ar(\mathfrak{G}) = 0$*, then* $\mathfrak{G}$ *is called* concept graph with cuts over the alphabet $\mathcal{A}$*.*

For the set E *of edges, let* $E^{id} := \{e \in E \,|\, \kappa(e) = \doteq\}$ *and* $E^{nonid} := \{e \in E \,|\, \kappa(e) \neq \doteq\}$*. The elements of* E^{id} *are called* identity-links*. Moreover we set* $V^* := \{v \in V \,|\, \rho(v) = *\}$, $V^? := \{v \in V \,|\, \rho(v) = ?i,\ i \in \mathbb{N}\}$*, and* $V^{\mathcal{G}} := \{v \in V \,|\, \rho(v) \in \mathcal{N}\}$*. The nodes in* V^* *are called* generic nodes*, the nodes in* $V^?$ *are called* query nodes*, and the nodes in* $V^{\mathcal{G}}$ *are called* object nodes*.*

[6] 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.

In the following, the alphabet is considered to be fixed, thus we simply speak of 'query graphs with cuts'. As already done, the terms 'concepts graph with cuts' and 'query graph with cuts' will be abbreviated by CGwC and QGwC, respectively.

For the graphical representation of QGwCs, the underlying relational graph is drawn as explained above. Now, 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. As already done, these rectangles are called *concept boxes* (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 $\rho(v) = g \in \mathcal{G}$'). Analogously, for an edge e, we write its relation name $\kappa(e)$ into the representing oval. These ovals are called *relation ovals*.

3 Contextual Semantics

For the most kinds of mathematical logic, a semantics in form of extensional models is provided. Particularly, for first order logic, the extensional models are relational structures $\mathcal{M} := (U, I)$, consisting of a universe (of discourse) U and a function I, which assigns objects, relations and functions in U to the object-, relation- or function-names of the alphabet. If mathematical logic is done with diagrams, there is often no direct extensional semantics provided (see for example [28, 41]). Instead, a translation from the graphs to first order logic is given, so that the models of first order logic serve indirectly as models for the graphs as well.

Formulas and graphs are very different 'styles' of logic, thus it seems a little bit awkward and unappropiate that the semantics, i.e., meaning, of graphs can only be gained indirectly via first order logic. Therefore, this approach is not adopted here, but a direct semantics for graphs is provided. This will be done in the following subsections.

3.1 Contextual Models

The semantics used here is a so-called *contextual semantics*, which is based on Formal Concept Analysis (FCA). This semantics was introduced by Wille in [37], and a comprehensive mathematical elaboration of FCA be found in [11]. The basic structure of FCA are *formal contexts*. Roughly speaking, a formal context $\mathbb{K}$ is a cross-table, fixing a set of objects G and a set of attributes M, and an incidence-relation I between these sets, indicating that an object g has an attribute m. In order to describe relations between objects, so-called *power context families (PCFs)* are introduced. A PCF is a family $(\mathbb{K}_0, \mathbb{K}_1, \mathbb{K}_2, \ldots, \mathbb{K}_n)$ of formal contexts such that the objects in the context $\mathbb{K}_i$, $i \geq 1$ are tuples of the objects of $\mathbb{K}_0$.

In Fig. 3, an example of a PCF is depicted. It describes the working group of the author. The objects are the members of the working group (e.g., 'RW' stands for 'Rudolf Wille', the inventor of FCA and the advisor of the author, and 'FD' stands for the author himself). The meaning of the attributes is obvious (the attribute $\top$ in $\mathbb{K}_0$ is used for the universal concept, which has already been mentioned in the introduction, the attribute $\doteq$ in $\mathbb{K}_2$ is the identity).

For this paper, only the basic notions of FCA and of contextual structures are provided.

Definition 6 (Formal Contexts and Power Context Families).

A formal context *is a triple $\mathbb{K} := (G, M, I)$, where G is a set of (formal) objects, M is a set of (formal) attributes and $I \subseteq G \times M$ is an incidence-relation. A pair*

$\mathbb{K}_0$	male	female	Diploma	PhD	Prof.
RW	×		×	×	×
PB	×		×	×	×
RH	×		×	×	
FD	×		×	×	
JK		×	×		
BV	×		×		
JHC	×		×		
LS	×		×		
TK	×		×		
BW	×				

$\mathbb{K}_2$	AdvisorOf	CoAuthorOf	$\doteq$
(RW,FD)	×	×	
(RW,JK)	×		
(RW,BV)	×	×	
(RW,JHC)	×	×	
(RW,TK)	×		
(RW,BW)	×		
(RW,LS)	×	×	
(PB,RH)	×	×	
(FD,RW)		×	
(FD,JK)		×	
(JK,FD)		×	
(FD,JHC)		×	
(JHC,FD)		×	
⋮	⋮	⋮	⋮
(BV,RW)		×	
(JHC,RW)		×	
(JHC,JK)		×	
(JK,JHC)		×	
(BV,JK)		×	
(JK,BV)		×	
(LS,RW)		×	
(RW,RW)			×
(PB,PB)			×
(RH,RH)			×
(FD,FD)			×
(JK,JK)			×
(BV,BV)			×
(JHC,JHC)			×
(TK,TK)			×
(BW,BW)			×

Fig. 3. An example of a power context family

(A, B) with $A \subseteq G$ and $B \subseteq M$ is called a formal concept of $\mathbb{K}$, *if and only if $A = \{g \in G \mid gIn$ for all $b \in B\}$ and $B = \{m \in M \mid aIm$ for all $a \in A\}$. $Ext(A, B) := A$ is called* extension of the concept (A, B), *and $Int(A, B) := B$ is called* intension of the concept (A, B). *The set of all formal concepts of a formal context $\mathbb{K}$ is denoted by $\mathfrak{B}(\mathbb{K})$.*

A family $\vec{\mathbb{K}} := (\mathbb{K}_0, \mathbb{K}_1, \mathbb{K}_2, \dots)$ of contexts $\mathbb{K}_k := (G_k, M_k, I_k)$ that satisfies $G_0 \neq \emptyset$ and $G_k \subseteq (G_0)^k$ for each $k \in \mathbb{N}$ is called power context family. *The elements of G_0 are the* objects *of $\vec{\mathbb{K}}$. All elements of $\bigcup_{k \in \mathbb{N}_0} \mathfrak{B}(\mathbb{K}_k)$ are called* concepts. *We set furthermore $\mathfrak{R}_{\vec{\mathbb{K}}} := \bigcup_{k \in \mathbb{N}} \mathfrak{B}(\mathbb{K}_k)$, and the elements of $\mathfrak{R}_{\vec{\mathbb{K}}}$ are called* relation-concepts.

When interpreting a concept graph in a power context family, the object names of our alphabet will be interpreted by objects, the concept names of our alphabet will be interpreted by formal concepts of the context $\mathbb{K}_0$, the relation names of arity k will be interpreted by relation-concepts of $\mathbb{K}_k$, and this interpretation has to respect the orders on the names. This motivates the following definition:

Definition 7 (Contextual Models).

Let $\mathcal{A} := (\mathcal{G}, \mathcal{C}, \mathcal{R})$ be an alphabet and $\vec{\mathbb{K}}$ be a power context family. Then we call the disjoint 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, and $\lambda_\mathcal{C}, \lambda_\mathcal{R}$ satisfy $\lambda_\mathcal{C}(\top) = \top$, $\lambda_\mathcal{R}(\mathcal{R}_k) \subseteq \underline{\mathfrak{B}}(\mathbb{K}_k)$ for all $k = 1, \dots, n$, and $(g_1, g_2) \in Ext(\lambda_\mathcal{R}(\doteq)) \Leftrightarrow g_1 = g_2$ for all $g_1, g_2 \in G_0$. The pair $(\vec{\mathbb{K}}, \lambda)$ is called* contextual model *over $\mathcal{A}$ or* contextual structure over $\mathcal{A}$.[7]

The mappings on V can naturally be extended to mappings on E. Moreover, as $\lambda_\mathcal{G}$ is a mapping on the set $\mathcal{G}$ of object names, it can be naturally extended to tuples of object names. In particular we set $\lambda_\mathcal{G}(\rho(e)) := (\lambda_\mathcal{G}(\rho(v_1)), \dots, \lambda_\mathcal{G}(\rho(v_n)))$.

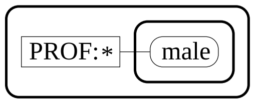
[7] The name 'contextual structure' is chosen according to the term 'relational structure'.

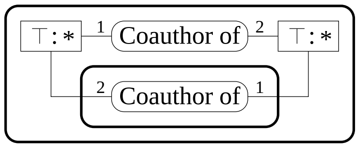
Contextual structures can be considered to be an extension of relational structures. Roughly speaking: If we remove all the intensional information from a contextual structure, we obtain a relational structure. More precisely: Let $(\vec{\mathbb{K}}, \lambda)$ be a contextual structure. Then, for $U := G_0$, $I_{\mathcal{G}}(g) := \lambda_{\mathcal{G}}(g)$ for all $g \in \mathcal{G}$, $I_{\mathcal{C}}(C) := Ext(\lambda_{\mathcal{C}}(C))$ for all $C \in \mathcal{C}$ and $I_{\mathcal{R}}(R) := Ext(\lambda_{\mathcal{R}}(R))$ for all $R \in \mathcal{R}$, it is easy to see that the relational structure $(U, I_{\mathcal{G}} \dot\cup I_{\mathcal{C}} \dot\cup I_{\mathcal{R}})$ corresponds to $(\vec{\mathbb{K}}, \lambda)$. Hence, for readers who are not familiar with Formal Concept Analysis, the results of this paper can easily be transferred to relational structures.

As contextual structures bear a richer structure than relational structures, it has to be argued why we do not simply use the relational structures in our semantics. As Wille says in [37]: ‘ There is a fundamental reason for associating concept(ual) graphs and Formal Concept Analysis 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 (elementary) logic. [...] Elementary logic was understood and taught by the traditional paradigm of philosophical logic based on the three essential main functions of thinking – *concepts*, *judgements* and *conclusions*.’ A formalization of this understanding of logic is an adequate approach for knowledge representation and processing, which is a main goal of concept(ual) graphs and query graphs. The formalization of concepts, particularly the understanding of a concept as a unit of thought constituted by its extension and intension, has been successfully established by Formal Concept Analysis. A crucial point is that the extension and intension of a concept can only be grasped in a fixed universe of discourse, which is formalized as a so-called *formal context*. Due to this contextual view, the semantics which is presented here is called *contextual semantics*. In the line of the philosophical understanding of logic, concept graphs can be understood as a formalization of judgements, and the deductive procedures on concept graphs can be understood as a formalization of conclusions. For a deeper discussion on this topic, we refer to [37].

3.2 Evaluation of Graphs in Contextual Models

The evaluation of a graph in a contextual structure shall be explained with some examples. Consider $\mathfrak{G}_1$ of the next two graphs:

$\mathfrak{G}_1 :=$ 

$\mathfrak{G}_2 :=$ 

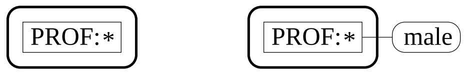
The evaluation of $\mathfrak{G}_1$ starts on the sheet of assertion $\top$ and proceeds inwardly (this is the so-called *endoporeutic method* of Peirce for existential graphs). As only the outermost cut (let us call it c_1) is directly enclosed by $\top$, we know that $\mathfrak{G}$ is true if the subgraph which is enclosed by c_1 is false. This subgraph contains a vertex v and a further cut c_2. We now have: $\mathfrak{G}$ is true if it is not true that there exists an object o such that o is a professor and the proposition enclosed by c_2 is false. Now we have to evaluate the area of c_2. This area contains only one edge, and the unary relation of this edge refers to the object o. Hence $\mathfrak{G}$ is true if there is no professor such that this professor is not male. In simpler words: Every professor is male. This proposition is true in our given contextual structure.

Similarly, the meaning of the right graph is ‘it is not true that there are two objects o_1, o_2 such that o_1 is the[8] advisor of of o_2, but o_2 is not a co-author of o_1’. In other

[8] More precisely, we should write ‘an advisor’ instead of ‘the advisor’

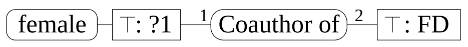
words: Each advisor of a person is a co-author of that person as well. In our given contextual structure, this proposition is false.

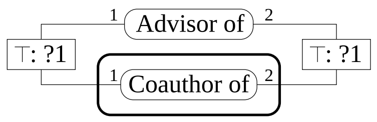
Now it can be explained why we forced the graphs to have dominating nodes. Consider the next two graphs, where the right graph has no dominating nodes:



The meaning of the left graph is clear: 'It is not true that there is a professor'. Particularly, the generic marker is existentially quantified ('there is'), and this quantification takes places inside the cut. But now, if we try to read the right graph inwardly, we have to evaluate the edge labelled with 'male', which refers to the object of the concept box *inside* the cut. Therefore, the 'place' of the existential quantification moves outside the cut, i.e., scope of the generic marker has to be extended to the sheet of assertion. This is possible, but it makes the reading of CGwCs very complicated. Therefore, graphs like this are not allowed. (A more comprehensive discussion of dominating nodes can be found in [7].)

The semantics which has been exemplified so far is the semantics for CGwCs, as it is described in [7]. This semantics can naturally be extended for QGwCs. Consider the following two QGwCs:





In contrast to CGwCs, QGwCs are not a formalization of propositions, but their evaluation in a contextual structure model yield relations. The idea is simple: For a given contextual structure, a QGwC of arity n (i.e., it contains query markers $?1, \ldots, ?n$) describes the relation of all n-tuples $(o_1, \ldots, o_n)$ of objects which can replace the query markers $?1, \ldots, ?n$, respectively, such that we obtain a valid concept graph for $\mathcal{M}$.

The first graph can be understood to be the following query: 'Give me all persons[9] which are female and which are a co-author of FD', or 'give me all female co-authors of FD' for short. In our example, we obtain only one person, namely JK.

The second graph can be understood to be the following query: 'Give me all pairs of persons such that the first person is the advisor, but not a co-author, of the second person.' In our example, we obtain the following set of pairs: $\{(RW, JK)$, (RW, TK), $(RW, BW)\}$.

Now we are prepared for the mathematical definitions of the evaluation QGwCs. When an QGwC is evaluated in a contextual structure $(\vec{\mathbb{K}}, \lambda)$, we have to assign objects of our universe of discourse G_0 to its generic markers. These assignments are done by valuations. Valuations do not need to be total, i.e. we will consider valuations where we assign values only to a subset of V. Nevertheless, it is clear that we will assign to each vertex $v \in V^{\mathcal{G}}$ the object $\lambda(\rho(v))$. Moreover, for a fixed $i \in \mathbb{N}$, all vertices v with $\rho(v) = ?i$ shall of course denote the same object.

Furthermore, we want to define specific partial valuations for contexts. Assume we want know which relations can be assigned to the query vertices in a context such that the context evaluates to true or false in a given model. Then we need to know which objects are assigned to vertices above the context, but, on the other hand, we should not have assigned objects yet to generic vertices which are enclosed by the

[9] More formally, we should write 'object' instead of 'person'.

context. This will be done by *partial valuations*. Finally, we will distinguish between *open* partial valuations, which do not assign object to any query vertices, and *closed* partial valuations, which assign objects to all query vertices.

Definition 8 (Partial and Total Valuations).

Let $\mathfrak{G} := (V, E, \nu, \top, Cut, area, \kappa, \rho)$ be a QGwC and let $\mathcal{M} := (\vec{\mathbb{K}}, \lambda)$ be a model. A mapping $ref : V' \to G_0$ is called partial valuation of $\mathfrak{G}$, *iff $V^{\mathcal{G}} \subseteq V' \subseteq V$, $ref(v) = \lambda(\rho(v))$ for all $v \in V^{\mathcal{G}}$, and if $v_1, v_2 \in V^?$ with $\rho(v_1) = \rho(v_2) = ?i$ for an $i \in \mathbb{N}$, then $ref(v_1) = ref(v_2)$. If furthermore $V' \cap V^? = \emptyset$, then ref is called* open, *and if $V' \supseteq V^?$, then ref is called* closed.

If c is a context such that $V' \supseteq \{v \in V^* \mid v > c\}$ 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 $\mathfrak{G}$.

Now let ref be an open partial valuation, and for $1 \leq i \leq ar(\mathfrak{G})$, let g_i be an element of G_0. Then let $ref[g_1, \ldots, g_{ar(\mathfrak{G})}] : dom(ref) \cup V^? \to G_0$ be the partial valuation which extends ref and which satisfies $ref[g_1, \ldots, g_{ar(\mathfrak{G})}](v) = g_i$ for each $v \in \cap V^?$ with $\rho(v) = ?i$.

Now we can define how a QGwC is evaluated in a contextual model. As mentioned above, the provided evaluation method is adopted from the endoporeuteic method of Peirce to evaluate existential graphs. Its mathematical implementation for QGwCs is as follows:

Definition 9 (Endoporeutic Evaluation of Graphs).

Let $\mathfrak{G} := (V, E, \nu, \top, Cut, area, \kappa, \rho)$ be an QGwC and $(\vec{\mathbb{K}}, \lambda)$ be a contextual model over $\mathcal{A}$. Inductively over the tree $Cut \,\dot\cup\, \{\top\}$, we define $(\vec{\mathbb{K}}, \lambda) \models \mathfrak{G}[c, ref]$ for each context $c \in Cut \,\dot\cup\, \{\top\}$ and every closed partial valuation $ref : V' \subseteq V \to G_0$ for c:

$(\vec{\mathbb{K}}, \lambda) \models \mathfrak{G}[c, ref] :\Longleftrightarrow$

ref can be extended to a partial valuation $\widetilde{ref} : V' \cup (V \cap area(c)) \to G_0$, 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)*
- *$(\vec{\mathbb{K}}, \lambda) \not\models \mathfrak{G}[d, \widetilde{ref}]$ for each $d \in Cut \cap area(c)$ (cut condition and iteration over $Cut \,\dot\cup\, \{\top\}$)*

Now let $ref_\emptyset$ be the empty valuation. We set:

$$\mathfrak{R}_{\mathcal{M},\mathfrak{G}} := \{(g_1, \ldots, g_{ar(\mathfrak{G})}) \mid \mathcal{M} \models \mathfrak{G}[ref_\emptyset[g_1, \ldots, g_{ar(\mathfrak{G})}]]\}$$

Finally, let $\mathfrak{G}_a, \mathfrak{G}_b$ be to QGwCs with arity n. We set

$$\mathfrak{G}_a \models_n \mathfrak{G}_b :\Longleftrightarrow \textit{ for all contextual models } \mathcal{M} \textit{ we have } \mathfrak{R}_{\mathcal{M},\mathfrak{G}_a} \subseteq \mathfrak{R}_{\mathcal{M},\mathfrak{G}_b}$$

According to the discussion above, this definition (namely the edge condition) relies on the condition that we consider concept graphs with dominating nodes.

It should be noted that this definition extends the definition given in [7] for concept graphs with cuts, which are evaluated to one of the truth-values TRUE and FALSE in models: If $\mathfrak{G}$ is a concept graph with cuts, then $\mathfrak{R}_{\mathcal{M},\mathfrak{G}}$ is one of the 0-ary relations $\{\}$ and $\{\{\}\}$. The relation $\{\}$ can be interpreted as FALSE, the relation $\{\{\}\}$ can be interpreted as TRUE. Then

$$\mathcal{M} \models \mathfrak{G} \Longleftrightarrow \mathfrak{R}_{\mathcal{M},\mathfrak{G}} = \{\{\}\} \text{ and } \mathfrak{G}_a \models \mathfrak{G}_b \Longleftrightarrow \mathfrak{G}_a \models_0 \mathfrak{G}_b$$

for concept graphs $\mathfrak{G}, \mathfrak{G}_a, \mathfrak{G}_b$ and contextual models $\mathcal{M}$.

4 Calculus

In [7], a sound and complete calculus for concept graphs with cuts is provided. This calculus is a based on Peirce's calculus for the beta part of existential graphs, which is here extended in order to capture the syntactical differences and the higher expressiveness of concept graph with cuts. As we can nearly adopt this calculus for QGwCs, we repeat it here, using common spoken language. For the mathematical definitions of the rules, we refer to [7].

Definition 10 (Calculus for Concept Graphs with Cuts). .

The calculus for concept graphs with cuts over the alphabet $\mathcal{A} := (\mathcal{G}, \mathcal{C}, \mathcal{R})$ consists of the following rules:

- **erasure**
  In positive contexts, any directly enclosed edge, isolated vertex, and closed subgraph may be erased.
- **insertion**
  In negative contexts, any directly enclosed edge, isolated vertex, and closed subgraph may be inserted.
- **iteration**
  Let $\mathfrak{G}_0 := (V_0, E_0, \nu_0, \top_0, Cut_0, area_0, \kappa_0, \rho_0)$ 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 $cut(v) = cut(\mathfrak{G}_0)$, 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 $area(c_1) = \{c_2\}$) may be inserted or erased.
- **generalization**
  For evenly enclosed vertices and edges, their concept names or object names resp. their relation names may be generalized.
- **specialization**
  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.
- **exchanging referents**
  Let $e \in E^{id}$ be an identity link with $\rho(e\big|_1) = g_1$, $\rho(e\big|_2) = g_2$, $g_1, g_2 \in \mathcal{G} \cup \{*\}$ and $cut(e) = cut(e\big|_1) = cut(e\big|_2)$. Then the referents of v_1 and v_2 may be exchanged, i.e., the following may be done: We can set $\rho(e\big|_1) = g_2$ and $\rho(e\big|_2) = g_1$.
- **merging two vertices**
  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 $e \in E$, $e\big|_i = v_1$ is replaced by $e\big|_i = v_2$.
- **splitting a vertex**
  Let $g \in \mathcal{G} \cup \{*\}$. Let $v = \boxed{P : g}$ be a vertex in the context c_0 and incident with relation edges $R_1, \ldots, R_n$, placed in contexts $c_1, \ldots, c_n$, respectively. Let c be a context 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 occurences of v are substituted by v'.

– **$\top$-erasure**
For $g \in \mathcal{G} \cup \{*\}$, an isolated vertex $\boxed{\top : g}$ may be erased from arbitrary contexts.
– **$\top$-insertion**
For $g \in \mathcal{G} \cup \{*\}$, an isolated vertex $\boxed{\top : g}$ may be inserted in arbitrary contexts.
– **identity-erasure**
Let $g \in \mathcal{G}$, let $v_1 = \boxed{P_1 : g}$ and $v_2 = \boxed{P_2 : g}$ be two vertices. Then any identity-link between v_1 and v_2 may be erased.
– **identity-insertion**
Let $g \in \mathcal{G}$, let $v_1 = \boxed{P_1 : g}$, $v_2 = \boxed{P_2 : g}$ be two vertices in contexts c_1, c_2, resp. and let $c \leq c_1, c_2$ be a context. Then an identity-link between v_1 and v_2 may be inserted into c.

A *proof* in the system of graphs is defined as usual in logic.

Definition 11 (Proof).

Let $\mathfrak{G}_a$, $\mathfrak{G}_b$ be two concept graphs with cuts. 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 *$\mathfrak{G}_a \vdash \mathfrak{G}_b$. Two graphs $\mathfrak{G}_a, \mathfrak{G}_b$ with $\mathfrak{G}_a \vdash \mathfrak{G}_b$ and $\mathfrak{G}_b \vdash \mathfrak{G}_a$ are said to be* provably equivalent.

The question arises how the calculus for CGwCs can be extended to a calculus for QGwCs. The basic idea is that query markers can be interpreted as 'generic object names'. Thus it has to be expected that, if we treat the query markers like object names, we get an adequate calculus for QGwCs. The definition of the calculus is as follows:

Definition 12 (Calculus for Query Graphs with Cuts).

The calculus for QGwCs consists of the rules of the calculus for concept graph with cuts (Def. 10), where

1. *the query markers $?i$ are treated like object names, and*
2. *an application of a rule to a QGwC $\mathfrak{G}_a := (V_a, E_a, \nu_a, \top_a, Cut_a, area_a, \kappa_a, \rho_a)$ with arity n is only allowed if it preserves the arity, i.e., for the derived graph $\mathfrak{G}_b := (V_b, E_b, \nu_b, \top_b, Cut_b, area_b, \kappa_b, \rho_b)$ we have*

$$\{i \mid \exists v \in V_a \text{ with } \rho(v) = ?i\} = \{i \mid \exists v \in V_b \text{ with } \rho(v) = ?i\} = \{1, \ldots, n\} \quad .$$

If $\mathfrak{G}_a, \mathfrak{G}_b$ are two QGwCs with arity n such that $\mathfrak{G}_b$ is derived from $\mathfrak{G}_a$ with this calculus, we write $\mathfrak{G}_a \vdash_n \mathfrak{G}_b$.

In order to show that this calculus is complete (the soundness is obvious), we extend the alphabet with the query markers and use the completeness of the calculus for concept graphs with cuts. The extension of the alphabet is defined canonically:

Definition 13 (Extensions of Alphabet and Models).

Let $\mathcal{A} := (\mathcal{G}, \mathcal{C}, \mathcal{R})$ be an alphabet. Let $\mathcal{A}^n := (\mathcal{G}', \mathcal{C}, \mathcal{R})$ with $\mathcal{G}' := \mathcal{G} \dot\cup \{\mathbf{?1}, \ldots, \mathbf{?n}\}$.[10] *$\mathcal{A}^n$ is called the* query-marker-extension *of $\mathcal{A}$. Let $\mathcal{M} := (\vec{\mathbb{K}}, \lambda)$ be a model for $\mathcal{A}$, let $g_1, \ldots, g_n \in G_0$. Then let $\mathcal{M}[g_1, \ldots, g_n] := (\vec{\mathbb{K}}, \lambda')$ be the model for $\mathcal{A}^n$ with:*

$$\lambda' := (\lambda'_{\mathcal{G}}, \lambda_{\mathcal{C}}, \lambda_{\mathcal{R}}) \text{ , where } \lambda'_{\mathcal{G}}(c) := \begin{cases} \lambda_{\mathcal{G}}(c) \text{ for } c \in \mathcal{G} \\ g_i \text{ for } c = \mathbf{?i} \text{ , } i \in \{1, \ldots, n\} \end{cases}$$

[10] In order to distinguish the object names $\mathcal{A}^n$ from the query markers, they are bold.

If $\mathcal{M}$ is a model for $\mathcal{A}$, then $\mathcal{M}'$ is obviously an model for $\mathcal{A}^n$. Moreover, each model for $\mathcal{A}^n$ is the query-marker-extension of a model for $\mathcal{A}^n$. Obviously, the QGwCs of arity n over the alphabet $\mathcal{A}$ correspond to the the concept graphs with cuts over the alphabet $\mathcal{A}^n$, in which each object name **?n** occurs at least once (see the condition for ρ in Def. 5).

Finally, if $ref_\emptyset$ denotes the empty valuation again, it is easy to see that we have

$$\mathcal{M} \models \mathfrak{G}[ref_\emptyset[g_1, \dots, g_{ar(\mathfrak{G})}]] \iff \mathcal{M}[g_1, \dots, g_n] \models \mathfrak{G} \tag{1}$$

Now we obtain the completeness of $\vdash_n$, which is the first main result of this paper.

Theorem 1 (The Calculus for QGwCs is Complete).
The calculus $\vdash_n$ is complete.

Proof: Let $\mathfrak{G}_a, \mathfrak{G}_b$ be two QGwCs with $ar(\mathfrak{G}_a) = ar(\mathfrak{G}_b) = n$. Then we have:

$$\begin{aligned}
\mathfrak{G}_a \models_n \mathfrak{G}_b &\iff \text{for all models } \mathcal{M} := (\vec{\mathbb{K}}, \lambda) \text{ over } \mathcal{A} : \mathfrak{R}_{\mathcal{M},\mathfrak{G}_a} \subseteq \mathfrak{R}_{\mathcal{M},\mathfrak{G}_b} \\
&\iff \text{for all models } \mathcal{M} := (\vec{\mathbb{K}}, \lambda) \text{ over } \mathcal{A} : \forall g_1, \dots, g_n \in G_0 : \\
&\qquad \mathcal{M} \models \mathfrak{G}_a[ref_\emptyset[g_1, \dots, g_{ar(\mathfrak{G})}]] \Rightarrow \mathcal{M} \models \mathfrak{G}_b[ref_\emptyset[g_1, \dots, g_{ar(\mathfrak{G})}]] \\
&\overset{(1)}{\iff} \text{for all models } \mathcal{M} := (\vec{\mathbb{K}}, \lambda) \text{ over } \mathcal{A} : \forall g_1, \dots, g_n \in G_0 : \\
&\qquad \mathcal{M}[g_1, \dots, g_n] \models \mathfrak{G}_a \Rightarrow \mathcal{M}[g_1, \dots, g_n] \models \mathfrak{G}_b \\
&\overset{\text{s.a.}}{\iff} \text{for all models } \mathcal{M}' \text{ over } \mathcal{A}^n : \mathcal{M}' \models \mathfrak{G}_a \Rightarrow \mathcal{M}' \models \mathfrak{G}_b \\
&\iff \mathfrak{G}_a \models \mathfrak{G}_b \quad (\text{over } \mathcal{A}^n)
\end{aligned}$$

As the calculus for concept graphs with cuts is sound and complete, we have a proof, i.e., a sequence $(\mathfrak{G}_1, \mathfrak{G}_2, \dots, \mathfrak{G}_k)$ with $\mathfrak{G}_1 = \mathfrak{G}_a$ and $\mathfrak{G}_b = \mathfrak{G}_k$, in the system of concept graphs with cuts over $\mathcal{A}^n$. In this sequence, there may be graphs in which not every object name **?i** occurs. But the proof can be transformed into a proof consisting solely of QGwCs with arity n as follows: For each $j = 1, \dots, k$, let $\mathfrak{G}'_j$ be the graph obtained from $\mathfrak{G}_j$ by juxtaposing n vertices $\boxed{\top : ?i}$, $i = 1, \dots, n$. Then $(\mathfrak{G}'_1, \mathfrak{G}'_2, \dots, \mathfrak{G}'_k)$ is again a proof. Finally, $\mathfrak{G}'_1$ can be derived from $\mathfrak{G}_1$ by adding the n vertices $\boxed{\top : ?i}$ with the $\top$-insertion-rule, and $\mathfrak{G}_k$ can be derived from $\mathfrak{G}'_k$ by erasing the additional n vertices $\boxed{\top : ?i}$ with the $\top$-erasure-rule. This procedure yields a proof for $\mathfrak{G}_a \vdash_n \mathfrak{G}_b$ with $k + 2n$ QGwCs of arity n. □

5 Normed Query Graphs

The relation graphs, as they have been described in the introduction, have some simple syntactical restrictions which are not adopted for query graphs. If a relation graph describes a relation of arity n, it has exactly n pending edges (one edge for each unsaturated valence of the relation), and these pending edges end on the sheet of assertion. In contrast to that, in a QGwC, a query marker $?i$ may occur in an arbitrary number of concept boxes, each of them placed in an arbitrary context. In the following, we restrict the system of QGwCs in order to get a class of graphs which corresponds more closely to relation graphs. That is, we consider QGwCs where each query marker $?i$ appears only once, namely in a concept box $\boxed{\top : ?i}$ placed directly on the sheet of assertion. These graphs are called *normed QGwCs*.

They are formally defined as follows:

Definition 14 (Normed Query Graphs).
Let $\mathfrak{G} := (V, E, \nu, \top, Cut, area, \kappa, \rho)$ be a QGwC. If we have moreover

1. $|\{v \in V \mid \rho(v) = ?i\}| = 1$ *for each* $i = 1, \ldots, ar(\mathfrak{G})$,
2. $ctx(v) = \top$ *for each* $v \in V^?$, *and*
3. $\rho(v) = \top$ *for each* $v \in V^?$,

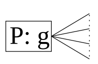
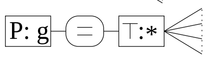
then $\mathfrak{G}$ is called normed.

We have provided a sound and complete calculus for –not necessarily normed– QGwCs. It is not obvious whether this calculus is still complete if we restrict it to the class of normed query graphs. In fact, the rules 'splitting a vertex' and 'merging two vertices' have to be slightly extended. Usually, if a vertex $v_1 := \boxed{P : g}$ is split, a new vertex $v_2 := \boxed{\top : g}$ is inserted. This is captured by the condition $\rho(v_1) = \rho(v_2)$ in Def. 10. Note that if we split a query vertex $\boxed{P : ?i}$ with this form of the rule 'splitting a vertex', the derived graph contains (at least) two vertices with the referent $?i$, hence the derived graph is not normed. Thus, in the class of normed QGwCs, this rule can never be applied. In order to make this rule and the rule 'merging two vertices' usable for the class of normed QGwCs, the condition $\rho(v_1) = \rho(v_2)$ is weakened in both rules to $\rho(v_1) \leq \rho(v_2)$. That is, we set:

- **merging two vertices (extended version)**
  Let $e \in E^{id}$ be an identity link with $\nu(e) = (v_1, v_2)$ such that $cut(v_1) \geq cut(e) = cut(v_2)$, $\rho(v_1) \leq \rho(v_2)$ 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 $e \in E$, $e\big|_i = v_1$ is replaced by $e\big|_i = v_2$.
- **splitting a vertex (extended version)**
  Let $g \in \mathcal{G} \cup \{*\}$. Let $v = \boxed{P : g}$ be a vertex in the context c_0 and incident with relation edges $R_1, \ldots, R_n$, placed in contexts $c_1, \ldots, c_n$, respectively. Let c be a context such that $c_1, \ldots, c_n \leq c \leq c_0$. Then the following may be done: In c, a new vertex $v' = \boxed{\top : g'}$ with $g' \geq g$ and a new identity-link between v and v' is inserted. On $R_1, \ldots, R_n$, arbitrary occurences of v are substituted by v'.

The calculus obtained from $\vdash_n$ (see Def. 12) with the old rules 'merging two vertices/splitting a vertex' replaced by its extended versions shall be denoted by $\Vdash_n$.

First we will show that $\Vdash_n$ is still sound. This will be done be deriving the generalized rules from the calculus $\vdash_n$. First, we need a simple congruence rule, which can be stated as follows:

Let $\mathfrak{G}$ be a QGwC, and let $v :=$  be a vertex of $\mathfrak{G}$. Let $\mathfrak{G}'$ be the graph we get when we replace v by  in $\mathfrak{G}$. Then $\mathfrak{G}'$ and $\mathfrak{G}'$ are equivalent.

Note that in this description of the rule, we have only drawn the crucial graph elements which are needed for the rule. Moreover, in order to indicate that a vertex of the graph incident with several edges which are not relevant in the proof, we attach ' ' to the its concept box. Formally, this rule is defined as follows:

Definition 15 (Congruence Rule).
Let $\mathfrak{G} := (V, E, \nu, \top, Cut, area, \kappa, \rho)$ be a QGwC, and let v be a vertex of $\mathfrak{G}$. Let $\mathfrak{G}' := (V', E', \nu', \top', Cut', area', \kappa', \rho')$ be the following graph:

- $V' := V \dot\cup \{v'\}$,
- $E' := E \dot\cup \{e'\}$,

– $\nu' := \nu \,\dot\cup\, \{e', (v, v')\}$,
– $\top' := \top$,
– $Cut' := Cut$,
– *For* $c \in Cut$ *with* $c \neq cut(v)$*, we set* $area'(c) := area(c)$*, and we set* $area'(cut(v)) := area(cut(v)) \,\dot\cup\, \{e', v'\}$,
– $\rho' := \rho \backslash \{(v, \rho(v))\} \,\dot\cup\, \{(v', \rho(v)), (v, *)\}$*, and*
– $\kappa' := \kappa \backslash \{(v, \kappa(v))\} \,\dot\cup\, \{(v', \kappa(v)), (v, \top)\}$.

We say that $\mathfrak{G}'$ *is derived from* $\mathfrak{G}$ *by* applying the congruence rule to the vertex v.

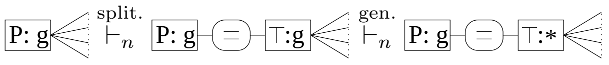
This rule can be derived from $\vdash_n$, as the following lemma shows.

Lemma 2 (The Congruence Rule can be Derived with $\vdash_n$).

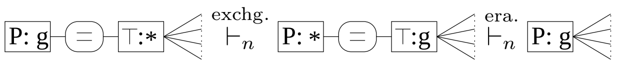
Let $\mathfrak{G} := (V, E, \nu, \top, Cut, area, \kappa, \rho)$ *be a QGwC, let* $v \in V$*,* $v := \boxed{P : g}$ *be a vertex, and let* $\mathfrak{G}'$ *be derived from* $\mathfrak{G}$ *by applying the congruence rule to* v*. Then we have* $\mathfrak{G} \vdash_n \mathfrak{G}'$ *and* $\mathfrak{G}' \vdash_n \mathfrak{G}$.

Proof: We distinguish whether $ctx(v)$ is a positive or negative context. Like in the description of the congruence rule, we sketch only the crucial graph elements.

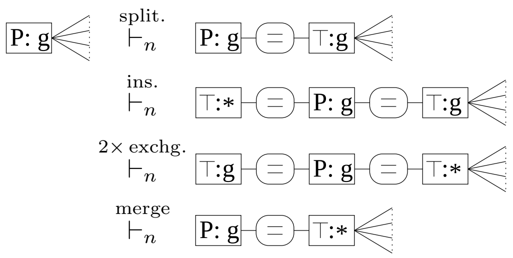
We start with the positive case, with the the direction '$\mathfrak{G} \vdash_n \mathfrak{G}'$'. The proof is as follows:



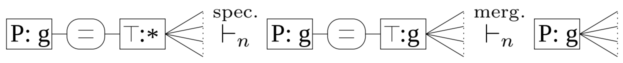
The direction '$\mathfrak{G}' \vdash_n \mathfrak{G}$' is done as follows:



Now let $ctx(v)$ be negative. The direction '$\mathfrak{G} \vdash_n \mathfrak{G}'$' of this lemma is done as follows:



Finally, the direction '$\mathfrak{G}' \vdash_n \mathfrak{G}$' is done as follows:



$\square$

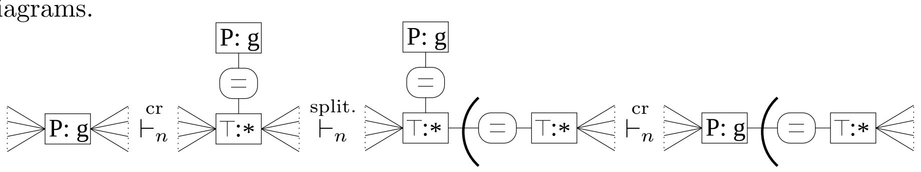
Please note the following: As in some graphs in the proofs the object name g appears twice, this proof cannot be applied for normed query graphs, if $g = ?i$ for an $i \in \mathbb{N}$.

With this congruence rule, we can now prove the generalized versions of the rules 'merging two vertices' and 'splitting a vertex'.

Lemma 3 (Generalization: Merging Two Vertices, Splitting a Vertex).

In the system of query graphs with cuts, the generalized rules 'merging two vertices' and 'splitting a vertex' can be derived from the calculus $\vdash_n$ *of Definition 12. Particularly, they are sound.*

Proof: Again, we sketch only the crucial graph elements. Moreover, in order to indicate that, when the rule 'splitting a vertex is applied to a vertex v, the copy of v may be placed in a deeper nested cut, we have added a cut segment to the diagrams.



Note that the proof can be carried out in both directions and in arbitrary contexts, thus we are done. □

It remains to show that $\Vdash_n$ is a complete calculus for normed QGwCs. In order to show this, we first assign to each QGwC $\mathfrak{G}$ a normed QGwC $norm(\mathfrak{G})$ as follows:

1. For each $i = 1, \ldots, ar(\mathfrak{G})$, we add a new vertex $v_{?i} := \boxed{\top :?i}$ to the sheet of assertion.
2. Then, for each vertex $v \neq v_{?i}$ with $\rho(v) =?i$, an identity link between $v_{?i}$ and v is added.
3. Finally, for each vertex $v \neq v_{?i}$ with $\rho(v) =?i$, its reference $?i$ is replaced by the generic marker $*$.

The formal definition is as follows:

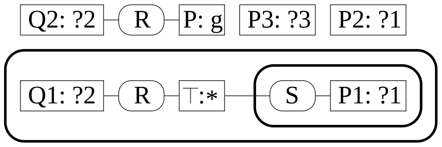
Definition 16 ($norm(\mathfrak{G})$).
Let $\mathfrak{G} := (V, E, \nu, \top, Cut, area, \kappa, \rho)$ be a QGwC with arity n. We assign to $\mathfrak{G}$ a normed QGwC $norm(\mathfrak{G}) := (V_n, E_n, \nu_n, \top_n, Cut_n, area_n, \kappa_n, \rho_n)$ as follows: Let $v_{?1}, \ldots, v_{?n}$ be new vertices. To each $v \in V^?$ let e_v be a new edge. Now we set

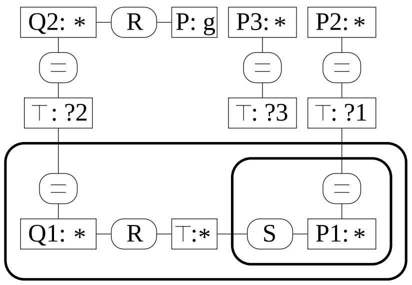
- $V_n := V \dot\cup \{v_{?1}, \ldots, v_{?n}\}$,
- $E_n := E \dot\cup \{e_v \mid v \in V^?\}$,
- $\nu_n := \nu \dot\cup \{(e_v, (v_{?i}, v)) \mid \rho(v) =?i$ *for an* $i \in \{1, \ldots, n\}\}$,
- $\top_n := \top$,
- $Cut_n := Cut$,
- *For* $c \in Cut$ *we set* $area_n(c) := area(c) \dot\cup \{e_v \mid v \in V^? \cap area(c)\}$ *and we set* $area_n(\top_n) := area(\top) \dot\cup \{e_v \mid v \in V^? \cap area(\top)\} \dot\cup \{v_{?1}, \ldots, v_{?n}\}$,
- $\rho_n := \rho \backslash \{(v, \rho(v)) \mid v \in V^?\} \dot\cup \{(v, *) \mid v \in V^?\} \dot\cup \{(v_{?i}, ?i) \mid i = 1, \ldots, n\}$, *and*
- $\kappa_n := \kappa \dot\cup \{(v_{?i}, \top) \mid i = 1, \ldots, n\} \dot\cup \{(e_v, \doteq) \mid v \in V^?\}$.

$norm(\mathfrak{G})$ is called the normalization of $\mathfrak{G}$.

Example

In the example below, the right graph is the normalization of the left graph.





The next lemma shows that $\mathfrak{G}$ and $norm(\mathfrak{G})$ are provably equivalent. But even if $\mathfrak{G}$ is a normed QGwC, we do not have $\mathfrak{G} = norm(\mathfrak{G})$. Nonetheless, is is easy to see that $norm(\mathfrak{G})$ can be derived from $\mathfrak{G}$ by a simple application of the extended 'splitting a vertex' rule. Thus, for a normed QGwC $\mathfrak{G}$ with $ar(\mathfrak{G}) = n$, we have $\mathfrak{G} \Vdash_n norm(\mathfrak{G})$ and $norm(\mathfrak{G}) \Vdash_n \mathfrak{G}$.

Lemma 4 ($\mathfrak{G}$ and $norm(\mathfrak{G})$ are Equivalent).

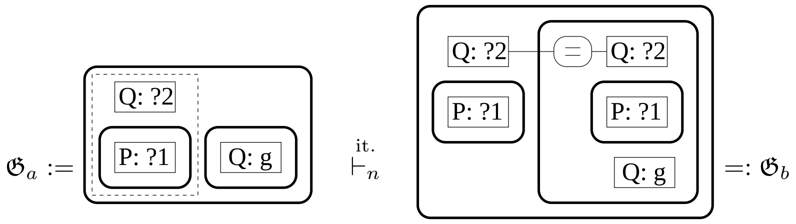
Let $\mathfrak{G}$ be a QGwC. Then $norm(\mathfrak{G})$ is a normed QGwC which is syntactically equivalent (with $\vdash_n$ or $\Vdash_n$) to $\mathfrak{G}$.

Proof: It is obvious that $norm(\mathfrak{G})$ is a normed QGwC. Let $n := ar(\mathfrak{G})$. Now $norm(\mathfrak{G})$ can be derived from $\mathfrak{G}$ as follows:

1. First, for each $i = 1, \ldots, n$, a new vertex $w_i := \boxed{\top : ?i}$ is inserted on the sheet of assertion with the $\top$-insertion-rule.
2. Then, for each vertex $v := \boxed{P : ?i}$ ($i \in \{1, \ldots, n\}$ and $v \notin \{w_1, \ldots, w_n\}$), an identity-link between v and w_i is inserted with the identity-insertion-rule (the identity link is of course placed in the same context as v).
3. For each vertex $v := \boxed{P : ?i}$ ($i \in \{1, \ldots, n\}$ and $v \notin \{w_1, \ldots, w_n\}$), we have finally to change its referent $?i$ to the generic marker $*$. This is done by first merging v into w_i, and then this operation is reversed by splitting w_i, but when w_i is split, the new vertex has to be $\boxed{P : *}$ instead of $\boxed{P : ?i}$. □

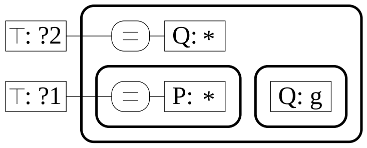
As the graphs $\mathfrak{G}$ and $norm(\mathfrak{G})$ are syntactically, and hence, semantically, equivalent, the class of normed QGwCs has the same expressiveness as the class of QGwCs.

Now let $\mathfrak{G}_a$ and $\mathfrak{G}_b$ be two normed QGwCs of arity n with $\mathfrak{G}_a \models \mathfrak{G}_b$. As $\vdash_n$ is complete, we have $\mathfrak{G}_a \vdash_n \mathfrak{G}_b$ as well. The proof for $\mathfrak{G}_a \vdash_n \mathfrak{G}_b$ is a sequence of graphs, but this sequence is a sequence of QGwCs which are not necessarily normed. But the proof can be transformed into a proof in the class of normed QGwCs, together with the calculus $\Vdash_n$. Let us consider an example, where the iteration-rule of $\vdash_n$ is applied to a QGwC (the iterated subgraph $\mathfrak{G}_0$ is marked by the dashed line):

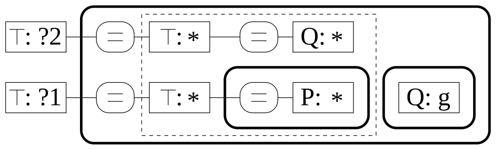


We cannot derive $norm(\mathfrak{G}_b)$ from $norm(\mathfrak{G}_a)$ by a simple application of the iteration-rule, but in the class of normed QGwCs, we can construct a proof for $norm(\mathfrak{G}_a) \vdash \vdash_n norm(\mathfrak{G}_b)$, which is as follows:

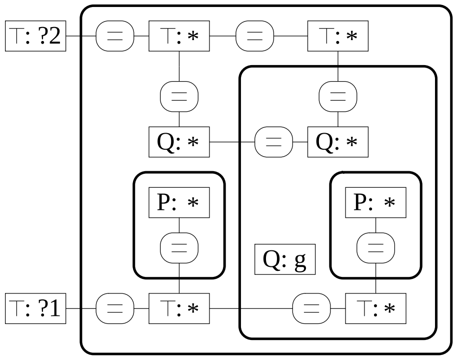
We start with $norm(\mathfrak{G}_a)$:



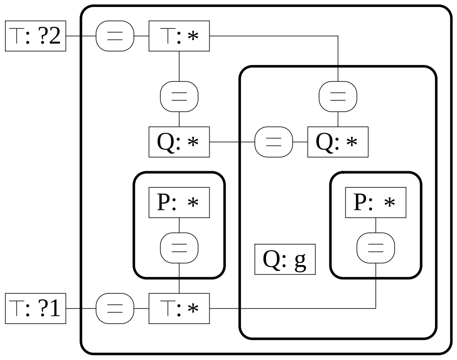
The query vertices are split such that their copies –we will call them w_i– are placed in the context where $\mathfrak{G}_0$ is iterated into.



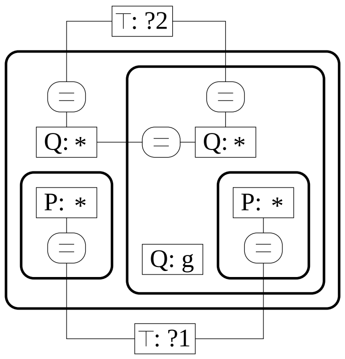
In the derived graph, we have a subgraph which corresponds $\mathfrak{G}_0$ (it is marked with dashed lines in the diagram above). Particularly, this subgraph contains the vertices w_i. Now this subgraph is iterated, and we insert an identity link from each each w_i to its copy.



Now the copies of the vertices w_i are merged back into their origins w_i.



Finally, the vertices w_i are merged back into the query vertices. This step yields $norm(\mathfrak{G}_b)$.



The underlying idea of this proof can be carried over to all rules in the calculus $\vdash_n$. This yields the following theorem:

Theorem 2 (Transformation of $\vdash_n$ into $\Vdash_n$).

Let $\mathfrak{G}_a, \mathfrak{G}_b$ be two QGwCs such that $\mathfrak{G}_b$ is derived from $\mathfrak{G}_a$ by applying one of the rules of the calculus $\vdash_n$. Then we have $norm(\mathfrak{G}_a) \Vdash_n norm(\mathfrak{G}_b)$, where the proof contains only normed QGwCs.

Proof: We set

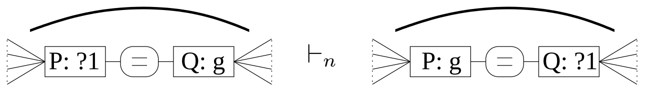
$$\begin{aligned}
\mathfrak{G}_a &:= (V_a, E_a, \nu_a, \top_a, Cut_a, area_a, \kappa_a, \rho_a) \\
\mathfrak{G}_b &:= (V_b, E_b, \nu_b, \top_b, Cut_b, area_b, \kappa_b, \rho_b) \\
norm(\mathfrak{G}_a) &:= (V_{na}, E_{na}, \nu_{na}, \top_{na}, Cut_{na}, area_{na}, \kappa_{na}, \rho_{na}) \\
norm(\mathfrak{G}_b) &:= (V_{nb}, E_{nb}, \nu_{nb}, \top_{nb}, Cut_{nb}, area_{nb}, \kappa_{nb}, \rho_{nb})
\end{aligned}$$

Let n be the arity of the graphs. The proof is carried out for each rule of the calculus separately. Due to the symmetry of the calculus, it is sufficient to prove the lemma only for one rule of the pairs erasure/insertion, iteration/deiteration, generalization/specialization, etc.

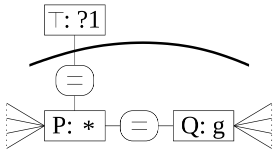
- **erasure and insertion**
  We only consider the erasure of an closed subgraph, all other cases can be done analogously.
  Let $\mathfrak{G}_0 := (V_0, E_0, \nu_0, \top_0, Cut_0, area_0, \kappa_0, \rho_0)$ be a closed subgraph of $\mathfrak{G}_a$ which is erased. Let $V' := V_0 \cap V_a^?$. Note that we have in $norm(\mathfrak{G}_a)$ a subgraph which corresponds to $\mathfrak{G}_0$, i.e., it has exactly the same vertices, edges and cuts, only the referents of the vertices $v \in V'$ have changed from $?i$ to $*$. This subgraph will be denoted $\mathfrak{G}_0$ as well. Now $norm(\mathfrak{G}_b)$ can be derived from $norm(\mathfrak{G}_a)$ as follows: First, erase in $norm(\mathfrak{G}_a)$ all edges e_v with $v \in V'$, then erase $\mathfrak{G}_0$ from this graph.
- **iteration and deiteration**
  The procedure herein described can be best understood with the example above. We will carry out the proof the the iteration-rule.
  Let $\mathfrak{G}_0 := (V_0, E_0, \nu_0, \top_0, Cut_0, area_0, \kappa_0, \rho_0)$ be a (not necessarily closed) subgraph of $\mathfrak{G}$ which is iterated into the context $c \leq cut(\mathfrak{G}_0)$.
  For $i = 1, \ldots, n$, let $v_{?i} \in V_{na}$ be the vertex with $\rho_{na}(v) = ?i$. For each $i = 1, \ldots, n$, we apply the rule 'splitting a vertex' to $v_{?i}$. The copy of $v_{?i}$ shall be denoted with w_i and be placed in $area(\top_0)$. Each occurence of a vertex $v \in V_0^?$ (i.e., $\rho_a(v) = ?i$, thus $\rho_{na}(v) = *$), and $v \in V_0$), shall be replaced by w_i. In our example, this yields the second diagram.
  Let $\mathfrak{G}_{n0}$ be the subgraph of the derived graph which contains all vertices, edges and cuts of $\mathfrak{G}_0$ and additionally all vertices w_i and all identity links between an vertex w_i and an vertex $v \in V_0^?$. This subgraph is iterated into c. During this iteration, for each $i = 1, \ldots, n$, an identity link is added between w_i and its copy. In our example, this yields the third diagram.
  After the iteration, the copies of the vertices w_i are merged into w_i. In our example, this yields the fourth diagram.
  Finally, the vertices w_i are merged back into $v_{?i}$. The graph we have finally reached is $norm\mathfrak{G}_b$. In our example, this yields the fifth diagram.
- **double cuts**
  It is easy to see that we can derive $norm(\mathfrak{G}_b)$ from $norm(\mathfrak{G}_a)$ with an application of the double cut rule as well.
- **generalization and specialization**
  If a vertex $v \in V^{\mathcal{G}}$ of $\mathfrak{G}$ is generalized, we have $norm(\mathfrak{G}_a) \overset{\text{gen.}}{\Vdash_n} norm(\mathfrak{G}_b)$ by generalizing the vertex $\mathfrak{G}$ as well.
  If a vertex $v \in V^?$ of $\mathfrak{G}$ is generalized, then $norm(\mathfrak{G}_b)$ is derived from $norm(\mathfrak{G}_b)$ by generalizing the vertex $\mathfrak{G}$ and erasing the vertex e_v.

– **exchanging referents**
Let v_1, v_2 be two vertices where the referents are exchanged. There are three cases to distinguish:

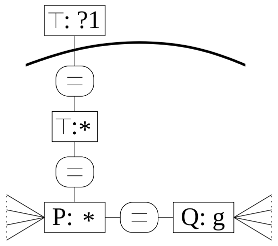
If v_1, v_2 are no query vertices, we have $norm(\mathfrak{G}_a) \overset{\text{exchg.}}{\Vdash_n} norm(\mathfrak{G}_b)$ by exchanging the referents of v_1, v_2 as well.
If $v_1 \in V^?$ and $v_2 \notin V^?$, we sketch the relevant parts of $\mathfrak{G}_a$ and $\mathfrak{G}_b$ as follows (w.l.o.g. we assume $\rho_a(v_1) = ?1$):



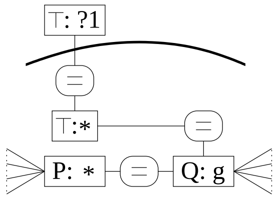
In order to derive $norm(\mathfrak{G}_b)$ from $norm(\mathfrak{G}_a)$, we do the following:
We start with $norm(\mathfrak{G}_a)$:



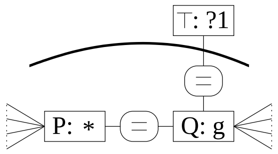
We split $v_{?1}$ as follows (the new vertex shall be denoted by w_1):



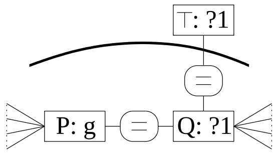
Lemma 11.5 from [7] allows the reflexive, symmetric and transitive reconfigurations of identity links in a context. For our graph, this lemma yields:



Now w_1 is merged back into $v_{?1}$:



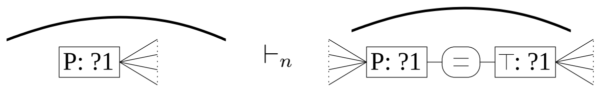
Finally, exchanging referents yields $norm(\mathfrak{G}_b)$:



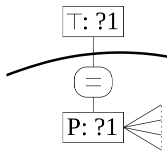
The last case, i.e., if $v_1, v_2 \in V^?$, can be handled analogously (in fact, it is even simpler, as no final exchanging of referents is needed).

– **splitting a vertex and merging two vertices**
We provide a proof for the rule 'splitting a vertex'.

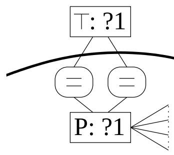
If in $\mathfrak{G}_a$ a vertex $v \notin V_a^?$ is split, we have $norm(\mathfrak{G}_a) \overset{\text{split.}}{\Vdash_n} norm(\mathfrak{G}_b)$ as well.
If $v \in V_a^?$ (w.l.o.g. we assume $\rho_a(v_1) = ?1$) is split, we sketch the relevant parts of $\mathfrak{G}_a$ and $\mathfrak{G}_b$ as follows:



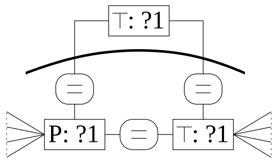
In order to derives $norm(\mathfrak{G}_b)$ from $norm(\mathfrak{G}_a)$, we do the following:
We start with $norm(\mathfrak{G}_a)$:



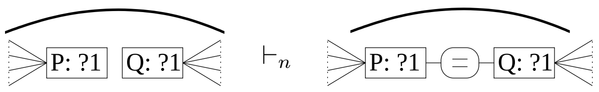
The identity-link is doubled (doubling edges is a rule in the calculus of Prediger which can easily be proven in $\vdash_n$):



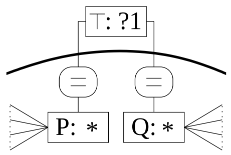
Finally, splitting $v_{?1}$ yields $norm(\mathfrak{G}_b)$:



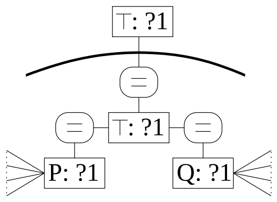
– **⊤-erasure and ⊤-insertion**
We provide a proof for the rule '⊤-erasure'.
If in $\mathfrak{G}_a$ a vertex $v \notin V_a^?$ is erased, we have $norm(\mathfrak{G}_a) \overset{\top-\text{era.}}{\Vdash_n} norm(\mathfrak{G}_b)$ as well.
If in $\mathfrak{G}_a$ an isolated vertex $v := \boxed{\top : ?i}$ is erased, we can get derive $norm(\mathfrak{G}_b)$ from $norm(\mathfrak{G}_a)$ by merging the corresponding vertex $w_i := \boxed{\top : *}$ into $v_{?i}$.
– **identity-erasure and identity-insertion**
We provide a proof for the rule 'identity-insertion. The relevant parts of $\mathfrak{G}_a$ and $\mathfrak{G}_b$ can be sketched as follows:



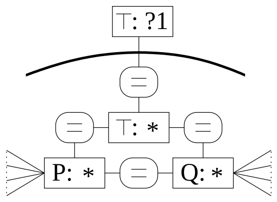
In order to derive $norm(\mathfrak{G}_b)$ from $norm(\mathfrak{G}_a)$, we do the following:
We start with $norm(\mathfrak{G}_a)$:



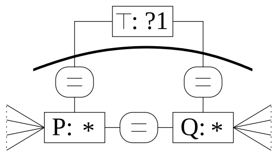
We split $v_{?1}$ as follows (the new vertex shall be denoted by w_1):



Lemma 11.5 from [7] yields:



Now w_1 is merged back into $v_{?1}$:



As we have investigated all rules of the calculus $\vdash_n$, we are done. □

With the preceding theorem, it follows immediately that $\Vdash_n$ is complete. This is the second main result of this paper.

Corollary 1 (Completeness of $\Vdash_n$ for Normed QGwCs).
Let $\mathfrak{G}_a, \mathfrak{G}_b$ be two QGwCs with $\mathfrak{G}_a \vdash_n \mathfrak{G}_b$. Then we have $norm(\mathfrak{G}_a) \Vdash_n norm(\mathfrak{G}_b)$, where the proof contains only normed QGwCs. Particularly, the calculus $\Vdash_n$ is complete for normed query graphs.

6 Conclusion and Further Research

There are two viewpoints for relation graphs: From an algebraic point of view, *operations* on graphs, corresponding to operations on relations, have to be investigated. This has been done by Burch, Pollandt and Wille. From a logical point of view, *inference rules* have to be investigated. This is the purpose of this paper. Of course these viewpoints are not competing, but complementing. To make the results of this paper fruitful for the theory of existential graphs and relation graphs, the relationships between existential graphs and concept graphs resp. between relation graphs and query graphs have to be further elaborated. A first approach for existential graphs can be found in [6].

Relation graphs can easily be defined inductively. So it seems appropriate to provide an inductive definition for query graphs as well and to discuss the advantages and disadvantages of non-inductive and inductive definitions. This is particularly important for logicians who are much more familiar with inductive definitions and their use in many proofs, e.g., for formulas.

The system of query graphs can be syntactically extended. A crucial extension is the addition of so-called *nestings*, where whole subgraphs of a graph are enclosed in a vertex. There are different possibilities for interpreting nestings. They are often used to describe specific contexts, e.g., situations. In [9], nestings are used to describe *nested relations* which occur in form of so-called *set functions* in database systems. The implementation of nestings has to be further investigated.

References

[1] John Barwise: *Heterogenous reasoning.* In G. W. Mineau, B. Moulin, J. F. Sowa (Eds.): Conceptual Graphs for Knowledge Representation. LNAI 699, Springer Verlag, Berlin–New York 2000, 64–74.

[2] R. W. Burch: *A Peircean Reduction Thesis: The Foundations of Topological Logic.* Texas Tech University Press, 1991.

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

[4] F. Dau: *Negations in Simple Concept Graphs.* In: B. Ganter, G. W. Mineau (Eds.): Conceptual Structures: Logical, Linguistic, and Computational Issues. LNAI 1867, Springer Verlag, Berlin–New York 2000, 263–276.

[5] F. Dau: *Concept Graphs and Predicate Logic.* In: H. S. Delugach, G. Stumme (Eds.): Conceptual Structures: Broadening the Base. LNAI 2120, Springer Verlag, Berlin–New York 2001, 72–86.

[6] F. Dau: *An Embedding of Existential Graphs into Concept Graphs with Negations.* In G. Angelova, D. Corbett, U: Priss (Eds.): Conceptual Structures: Integration and Interfaces. LNAI 2393, Springer Verlag, Berlin–Heidelberg 2002.

[7] F. Dau: *The Logic System of Concept Graphs with Negations (And its Relationship to Predicate Logic).* LNAI 2892, Springer Verlag, Berlin–Heidelberg 2003.

[8] F. Dau: *Types and Tokens for Logic with Diagrams: A Mathematical Approach.* Submitted to Diagrams (Journal).

[9] F. Dau, J. Hereth Correia: *Nested Concept Graphs: Applications for Databases and Mathematical Foundations.* In: Moor, A., Ganter, B. (Eds.): Using Conceptual Structures. 11th International Conference on Conceptual Structures, ICCS 2003, Dresden, July 21-July 26, 2003, Contributions to ICCS 2003.
Skaker Verlag, Aachen, 2003.

[10] 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,

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

[12] E. M. Hammer: *Logic and Visual Information.* CSLI Publications, Stanford, California, 1995.

[13] E. M. Hammer: *Semantics for Existential Graphs.* Journal Philosohpical Logic, Vol. 27, 1998, 489–503.

[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, 146–160.

[15] K. Oehler: *Charles Sanders Peirce.* Beck'sche Reihe, 1993.

[16] H. Pape: *Charles S. Peirce: Phänomen und Logik der Zeichen.* German translation of Peirce's *Syllabus of Certain Topics of Logic.*. Suhrkamp Verlag Wissenschaft, 1983.

[17] C. S. Peirce: *Reasoning and the Logic of Things. The Cambridge Conferences Lectures of 1898.* Cambridge, Massachusetts, London, England, 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] C. S. Peirce, J. F. Sowa: *Existential Graphs: MS 514 by Charles Sanders Peirce with Commentary by John F. Sowa.*
`http://www.jfsowa.com/peirce/ms514.htm`

[20] S. Pollandt: *Relational Constructions on Semiconcept Graphs.* In: B. Ganter, G. Mineau (Eds.): Conceptual Structures: Extracting and Representing Semantics. Contributions to ICCS 2001, Stanford.

[21] S. Pollandt: *Relation Graphs: A Structure for Representing Relations in Contextual Logic of Relations.* In G. Angelova, D. Corbett, U: Priss (Eds.): Conceptual Structures: Integration and Interfaces. LNAI 2393, Springer Verlag, Berlin–Heidelberg 2002.

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

[23] S. Prediger: *Simple Concept Graphs: A Logic Approach.* In: M, -L. Mugnier, M. Chein (Eds.): Conceptual Structures: Theory, Tools and Applications. LNAI 1453, Springer Verlag, Berlin–New York 1998, 225–239.

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

[25] D. D. Roberts: *The Existential Graphs.* Computers Math. Appl.., Vol. 23, No. 6–9, 1992, 639–63.

[26] S. Shin: *The Logical Status of Diagrams.* Cambridge, 1994.

[27] S. Shin: *Reconstituting Beta Graphs into an Efficacious System.* Journal of Logic, Language and Information, Vol. 8, No. 3, July 1999.

[28] S. Shin: *The Iconic Logic of Peirce's Graphs.* MIT, Bradford, 2002.

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

[30] 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, 3–51.

[31] J. F. Sowa: *Logic: Graphical and Algebraic.* Manuskript, Croton-on-Hudson 1997.

[32] J. F. Sowa: *Conceptual Graphs: Draft Proposed American National Standard.* In: W. Tepfenhart, W. Cyre (Eds.): Conceptual Structures: Standards and Practices. LNAI 1640, Springer Verlag, Berlin–New York 1999, 1–65.
See also [34]

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

[34] J. F. Sowa: *Conceptual Graphs: Draft Proposed American National Standard.*
Old version: `http://www.jfsowa.com/cg/cgdpansw.htm`
New version: `http://www.jfsowa.com/cg/cgstandw.htm`
See also [32]

[35] R. Wille: *Plädoyer für eine Philosophische Grundlegung der Begrifflichen Wissensverarbeitung.* In: R. Wille, M. Zickwolff (Eds.): Begriffliche Wissensverarbeitung: Grundfragen und Aufgaben. B.I.–Wissenschaftsverlag, Mannheim, 1994, 11–25.

[36] 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.

[37] R. Wille: *Conceptual Graphs and Formal Concept Analysis.* In: D. Lukose et al. (Hrsg.): Conceptual Structures: Fulfilling Peirce's Dream. LNAI 1257, Springer Verlag, Berlin–New York 1997, 290–303. y

[38] R. Wille: *Contextual Logic Summary.* In: G. Stumme (Ed.): Working with Conceptual Structures. Contributions to ICCS 2000. Shaker, Aachen 2000, 265–276.

[39] R. Wille: *Lecture Notes on Contextual Logic of Relations.* FB4-Preprint, TU-Darmstadt, 2000.

[40] R. Wille: *Existential Concept Graphs of Power Context Families.* In G. Angelova, D. Corbett, U: Priss (Eds.): Conceptual Structures: Integration and Interfaces. LNAI 2393, Springer Verlag, Berlin–Heidelberg 2002.

[41] J. Zeman: *The Graphical Logic of C. S. Peirce* Ph.D. Diss., University of Chicago, 1964. Published 2002 in WWW at:
`http://www.clas.ufl.edu/users/jzeman/graphicallogic/`