Intension Graphs as Patterns over Power Context Families

Jens Kötters

data2knowledge GmbH, Bremen, Germany

Abstract. Intension graphs are introduced as an intensional variant of Wille's concept graphs. Windowed intension graphs are then introduced as formalizations of conjunctive queries. Realizations describe pattern matching over power context families, which have been introduced with concept graphs as representations of relational data using a sequence of formal contexts. Using windowed intension graphs as patterns within pattern structures, we can define concept lattices, where power context families take the role of formal contexts. Relational Context Families, used in Relational Concept Analysis (RCA), correspond to power context families using sorts and only binary relations, and the lattices generated by the RCA algorithm (using wide scaling) can be represented using rooted trees as intents, which are introduced as a subclass of windowed intension graphs. Consequently, projections of the previously introduced pattern structure can be used as an alternative to the RCA algorithm.

Keywords: Conjunctive Queries, Pattern Structures, Power Context Families, Relational Concept Analysis

1 Introduction

In the terminology of philosophers and linguists, a concept has an *extension* and an *intension*. We may say that "extension" refers to the things belonging to a concept, whereas "intension" refers to the meaning of a concept. Formal Concept Analysis [6] (FCA) provides a mathematical formalization of concepts which represents the extension by a set of formal objects (the *extent*) and the intension by a set of formal attributes (the *intent*). The notion of intension is however a vague one and different representations can be thought of.

Many real-world concepts describe objects in terms of their relations to other objects (e.g. visitor, grandfather, ticket), and this may suggest a different representation of intensions using graphs. It turns out that conjunctive queries offer a rich notational framework to support this kind of representation. In [8], concepts have been defined in terms of family relations, and windowed relational structures have been used to represent conjunctive queries. The qualifier "windowed" is used here to express that a number of designated elements have been chosen from the underlying structure. These are the elements being described. The current paper introduces intension graphs (IGs), which are attribute-labeled graphs, and uses them in place of relational structures. In contrast to relational

ISBN 978-5-600-01454-1, National Research University Higher School of Economics, 2016.

structures, IGs formally represent information in the same way it is drawn (i.e. centered around objects) and are supposed to be more intuitive to work with. Conjunctive queries are accordingly represented by windowed IGs. Section 2 defines IGs, describes pattern matching over power context families [12] (PCFs) and shows that IGs can be represented by PCFs and vice versa.

Sections 3 and 4 define the sum and product of IGs and windowed IGs, respectively. In both cases, sum and product realize the supremum and infimum operations. These operations are called sum and product because they realize certain universal properties (coproduct and product) defined in category theory. Also, Sect. 4 briefly states connections of windowed IGs to primitive positive formulas and relational algebra operations, which are known to exist because windowed IGs model conjunctive queries.

The concept lattice depends not only on the PCF (which plays the role of a formal context), but also depends on the chosen formalization of intension. Section 5.1 states the pattern structure (see [5]) for building the lattice of windowed IGs over a PCF, which contains the formalization of intension in its definition. From there, any general algorithm for pattern structures can be used to build the lattice. Section 5.2 provides an example and illustrations.

Finally, Sect. 6.1 shows that the Relational Concept Analysis (RCA) algorithm, used with the wide scaling operator, generates lattices of *rooted trees*, which form a subclass of conjunctive queries. It is shown that essentially the same lattices can be generated from projections [5] of the pattern structure of Sect. 5.1.

2 Intension Graphs and Power Context Families

2.1 Intension Graphs

A *simple relational graph* is a pair (V, E) consisting of a set V of vertices and a set $E \subseteq \bigcup_{k\geq 1} V^k$ of edges. The edges in $E^{(k)} := E \cap V^k$ are said to have *arity* k ($k \geq 1$).

Definition 1. *An* intension graph *over a family $(M_k)_{k\in\mathbb{N}}$ of attribute sets is a triple (V, E, κ), where (V, E) is a simple relational graph and κ is a map defined on $V \cup E$ with $\kappa(V) \subseteq \mathcal{P}(M_0)$ and $\kappa(E^{(k)}) \subseteq \mathcal{P}(M_k) \setminus \{\emptyset\}$ for $k \geq 1$.*

A *homomorphism* $\varphi : (V_G, E_G, \kappa_G) \to (V_H, E_H, \kappa_H)$ of intension graphs over the same family M of intents is a map $\varphi : V_G \to V_H$, extended to V^k, $k \geq 1$, by setting

$$\varphi((v_1, \ldots, v_k)) := (\varphi(v_1), \ldots, \varphi(v_k)), \tag{1}$$

which preserves edges and intents, i.e.

$$\varphi(e) \in E_H, \tag{2}$$

$$\kappa_G(u) \subseteq \kappa_H(\varphi(u)) \tag{3}$$

must hold for all $e \in E_G$ and $u \in V_G \cup E_G$. We define $\mathbf{IG}_M$ as the category of intension graphs over M.

2.2 Power Context Families

Definition 2. *A* power context family *is a sequence* $(\mathbb{K}_i)_{i\in\mathbb{N}}$ *of formal contexts* $\mathbb{K}_i =: (G_i, M_i, I_i)$ *such that* $G_i \subseteq (G_0)^i$ *for all* $i \geq 1$. *We say that* $(\mathbb{K}_i)_{i\in\mathbb{N}}$ *is a power context family over the family* $(M_i)_{i\in\mathbb{N}}$ *of attribute sets if, in addition,* $g^{I_k} \neq \emptyset$ *for all* $g \in G_k$, $k \geq 1$.

A *homomorphism* $\varphi : ((G_i, M_i, I_i))_{i\in\mathbb{N}} \to ((H_i, M_i, J_i))_{i\in\mathbb{N}}$ of power context families over the same family $(M_i)_{i\in\mathbb{N}}$ of attribute sets is a map $\varphi : G_0 \to H_0$, extended to $(G_0)^k$, $k \geq 1$, by setting

$$\varphi((g_1, \ldots, g_k)) := (\varphi(g_1), \ldots, \varphi(g_k)), \tag{4}$$

which preserves incidences, i.e.

$$g I_k m \Rightarrow \varphi(g) J_k m \tag{5}$$

must hold for all $k \in \mathbb{N}$, $g \in G_k$ and $m \in M_k$. We define $\mathbf{PCF}_M$ as the category of power context families over M.

2.3 Isofunctors

Let $(M_i)_{i\in\mathbb{N}} =: M$ be a family of attribute sets. We may represent a power context family $(\mathbb{K}_i)_{i\in I}$ in $\mathbf{PCF}_M$ by an intension graph

$$\mathrm{ig}_M((\mathbb{K})_{i\in\mathbb{N}}) := (G_0, \bigcup_{k\geq 1} G_k, \{u \mapsto u^{I_k} \mid (k,u) \in \bigcup_{k\in\mathbb{N}} \{k\} \times G_k\}). \tag{6}$$

in $\mathbf{IG}_M$. Conversely, each intension graph G in $\mathbf{IG}_M$ is represented in $\mathbf{PCF}_M$ by the power context family

$$\begin{gathered}\mathrm{pcf}_M(G) := ((E_G^{(k)}, M_k, \ni_G^{(k)}))_{k\in\mathbb{N}}, \\ \text{where } u \ni_{\mathfrak{G}}^{(k)} m :\Leftrightarrow m \in \kappa(u).\end{gathered} \tag{7}$$

It is easy to see that $\mathrm{pcf}_M(\mathrm{ig}_M(\vec{\mathbb{K}})) = \vec{\mathbb{K}}$ and $\mathrm{ig}_M(\mathrm{pcf}_M(G)) = G$ for all $\vec{\mathbb{K}} \in \mathbf{PCF}_M$ and $G \in \mathbf{IG}_M$. Moreover, every homomorphism $\varphi : G \to H$ of intension graphs is also a homomorphism $\varphi : \mathrm{pcf}_M(G) \to \mathrm{pcf}_M(H)$ and vice versa. This means that the categories $\mathbf{IG}_M$ and $\mathbf{PCF}_M$ are essentially the same.

2.4 Interpretations

Power context families can be used to model factual knowledge about objects and their relations to each other. The objects are collected in a set G_0, and the formal contexts (G_0, M_0, I_0) and (G_1, M_1, I_1) describe the objects by attributes. Finally, the contexts (G_k, M_k, I_k), $k \geq 2$, describe how the objects are related to each other.

Intension graphs can be used to model patterns. The nodes describe some unspecified objects, and the map κ describes them in terms of attributes. An edge is used to indicate that the objects involved are related in some way, and the map κ specifies the relation(s) between the objects. A pattern match is formalized by the following definition:

Definition 3. *Let* $G \in \mathbf{IG}_M$ *and* $\vec{\mathbb{K}} \in \mathbf{PCF}_M$*. A* realization $\rho : G \to \vec{\mathbb{K}}$ *is a map* $\rho : V_G \to G_0$ *with* $\rho(u) \in \kappa(u)^{I_k}$ *for all* $u \in E_G^{(k)}$ *and* $k \in \mathbb{N}$.

Since $\mathbf{IG}_M$ and $\mathbf{PCF}_M$ are isomorphic, we may represent patterns and data in the same category. A realization then becomes a homomorphism:

Proposition 1. *Let* $G \in \mathbf{IG}_M$ *and* $\vec{\mathbb{K}} =: (G_i, M_i, I_i)_{i \in I} \in \mathbf{PCF}_M$*. A map* $\varphi : V_G \to G_0$ *is a realization* $\varphi : G \to \vec{\mathbb{K}}$ *iff it is a homomorphism* $\varphi : G \to \mathrm{ig}(\vec{\mathbb{K}})$.

Some remarks are in order why intension graphs and power context families were defined the way they are. First, if edge labels of intension graphs were permitted, we could create more specific patterns by adding edges with empty labels. This could be justified by saying that an edge with an empty label means that the incident vertices are related in some unspecified way. However, it seems better to model this explicitly by adding "is related" attributes. Adding empty rows to a context (G_k, M_k, I_k), $k \geq 1$, of a power context family $\vec{\mathbb{K}}$, on the other hand, results in an equivalent power context family (as per the homomorphism definition). To make $\mathbf{PCF}_M$ and $\mathbf{IG}_M$ isomorphic, empty rows are not permitted in contexts (G_k, M_k, I_k), $k \geq 1$.

3 Graph Operations and Graph Construction

The main result of Sections 3 and 4 is the definition of the product and the sum for IGs and windowed IGs. These operations define infima and suprema in the respective morphism preorders. Moreover, in category theoretical terms, the stated operations realize (categorical) products and coproducts[1]. This means that, given graphs G_1 and G_2, there are morphisms $\pi_1 : G_1 \times G_2 \to G_1$, $\pi_2 : G_1 \times G_2 \to G_2$ such that for any other graph X and morphisms $\varphi_1 : X \to G_1$, $\varphi_2 : X \to G_2$ there is a unique $\varphi : X \to G_1 \times G_2$ with $\varphi_1 = \pi_1 \circ \varphi$ and $\varphi_2 = \pi_2 \circ \varphi$ (Fig. 1), and likewise for the coproduct (Fig. 2). Infinite (co-)products are defined accordingly. Every product is an infimum in the morphism preorder, but the opposite does not hold: products are unique up to isomorphism[1], but infima are only unique up to hom-equivalence (i.e. equivalence in the morphism preorder). The stronger product property is not needed in this paper, but when looking for infima of patterns compared by morphisms, one may check for categorical products as they can often be derived from well-known products. It may seem unfortunate that in Fig. 1 the infimum $G_1 \times G_2$ is drawn *above* G_1 and G_2, but this arrangement seems to be prevalent in drawings of categorical products and is also in line with how patterns are arranged in the concept lattice.

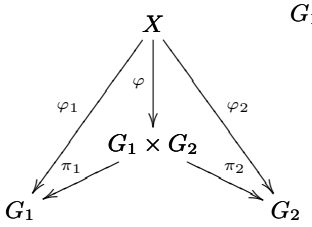


Fig. 1. Product

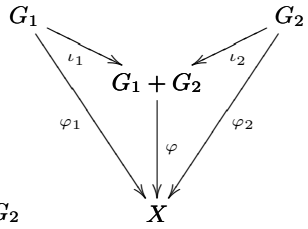


Fig. 2. Coproduct

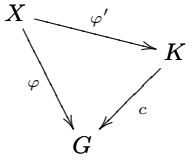


Fig. 3. Co-reflection

3.1 Graph Operations

The *product* of a family $((V_i, E_i, \kappa_i))_{i\in I}$ of intension graphs is the intension graph $\bigtimes_{i\in I}(V_i, E_i, \kappa_i) =: (V, E, \kappa)$ given by

$$V := \bigtimes_{i\in I} V_i, \tag{8}$$

$$\begin{aligned}(v_1, \ldots, v_k) \in E^{(k)} :\Leftrightarrow\ & ((v_1(i), \ldots, v_k(i)))_{i\in I} \in \bigtimes_{i\in I} E_i^{(k)} \\ & \text{and } \bigcap_{i\in I} \kappa_i((v_1(i), \ldots, v_k(i))) \neq \emptyset,\end{aligned} \tag{9}$$

$$\kappa(u) := \bigcap_{i\in I} \kappa_i(u) \tag{10}$$

for $u \in V \cup E$ and $v_1, \ldots, v_k \in V$, $k \geq 1$. Given a set X, an intension graph G and a bijection $\varphi : V_G \to X$, we call the graph

$$\varphi \circ G := (\varphi \circ V_G, \varphi \circ E_G, \kappa_G \circ \varphi^{-1}) \tag{11}$$

a *renaming* of G (cf. (1)). This amounts to a renaming of graph nodes. The *union* of graphs G and H with $V_G \cap V_H \neq \emptyset$ is the graph

$$G_1 \cup G_2 := (V_{G_1} \cup V_{G_2}, E_{G_1} \cup E_{G_2}, \kappa_{G_1} \cup \kappa_{G_2}), \tag{12}$$

and the *disjoint union* or *sum* of two arbitrary graphs G_1 and G_2 is given by

$$G_1 \sqcup G_2 := (\varphi_1 \circ G_1) \cup (\varphi_2 \circ G_2), \tag{13}$$

where $\varphi_i(v) := (i, v)$ for $i \in \{1, 2\}$ and $v \in V_{G_i}$. The disjoint union is the coproduct in $\mathbf{IG}_M$. Now let $G \in \mathbf{IG}_M$ and $\theta \subseteq V_G \times V_G$ an equivalence relation. The *quotient* of G w.r.t. θ is the graph

$$G \setminus \theta := (V_G \setminus \theta, E_G \setminus \theta, \kappa_\theta), \tag{14}$$

$$\text{where } E \setminus \theta := \{([v_1], \ldots, [v_n]) \mid (v_1, \ldots, v_n) \in E_G\}, \tag{15}$$

$$\text{and } \kappa_\theta([u]_\theta) := \bigcup_{x\theta u} \kappa(x) \tag{16}$$

for $u \in V_G$. The operation can be visualized as a merging of nodes within the same graph. For an arbitrary relation $\theta \subseteq V_G \times V_G$, we define $V \backslash \theta := V \backslash \bar{\theta}$, where $\bar{\theta}$ is the smallest equivalence relation with $\theta \subseteq \bar{\theta}$. Finally, for graphs $G, H \in \mathbf{IG}_M$ and $\theta \subseteq V_G \times V_H$, we define the *amalgam*

$$G_1 \underset{\theta}{+} G_2 := (G_1 \sqcup G_2) \setminus \{(\varphi_1(x), \varphi_2(y)) \mid (x, y) \in \theta\}, \tag{17}$$

where φ_1 and φ_2 are given as in (13). The amalgam can be visualized as a merging of two graphs by their nodes.

3.2 Graph Construction

An intension graph $G \in \mathbf{IG}_M$ with $|V_G| < \infty$ is called finite. For attribute sets $B \subseteq M_0$ and $R \subseteq M_k$,$k \geq 1$, we define the following structurally minimal graphs.

$$\mathcal{E}_B(x) := (\{x\}, \emptyset, \{x \mapsto B\}) \tag{18}$$

$$\begin{aligned} \mathcal{S}_R(x_1, \ldots, x_n) := (\{x_1, \ldots, x_n\}, \{(x_1, \ldots, x_n)\}, \kappa_R), \\ \text{where } \kappa_R := \{(x_1, \ldots, x_n) \mapsto R, x_1 \mapsto \emptyset, \ldots, x_n \mapsto \emptyset\}) \end{aligned} \tag{19}$$

Every finite $G \in \mathbf{IG}_M$ can be constructed from these graphs in a finite number of steps, using the amalgam and renaming operations.

4 Windowed Intension Graphs

Definition 4. *A windowed intension graph is a pair (α, G) consisting of an intension graph G and a partial map $\alpha : \mathbb{N} \nrightarrow V_G$.*

A homomorphism $\varphi : (\alpha_1, G_1) \to (\alpha_2, G_2)$ of windowed intension graphs is a homomorphism $\varphi : G_1 \to G_2$ with $\varphi \circ \alpha_1 \leq \alpha_2$.

While a pattern match for an intension graph G in a power context family $\vec{\mathbb{K}}$ has been defined by a realization $\rho : G \to \vec{\mathbb{K}}$, the set

$$(\alpha, G)^\diamond := \{\alpha \circ \varphi \mid \varphi : G \to \vec{\mathbb{K}}\} \tag{20}$$

defines the set of all pattern matches for the windowed intension graph (α, G). A finite windowed intension graph corresponds to a primitive positive formula (pp formula), i.e. a predicate logical formula which is built from atoms using conjunction ($\wedge$) and existence quantification ($\exists$) only (atoms may contain the equals sign). For finite graphs, the $(\cdot)^\diamond$ operation can be inductively defined, starting with

$$(\mathrm{id}_{\{0\}}, \mathcal{E}_B(0))^\diamond = B^{I_0}, \quad B \subseteq M_0, \tag{21}$$

$$(\mathrm{id}_{\{1,\ldots,n\}}, \mathcal{S}_R(1, \ldots, n))^\diamond = R^{I_n}, \quad R \subseteq M_k, k \geq 1, \tag{22}$$

which correspond to the *select* operation on databases, and proceeding with similar rules for the *join* and *project* operations.

When viewing a windowed intension graph (α, G) as a pp formula, the set $\alpha^{-1}(V_G)$ corresponds to the free variables, and the set $V_G \setminus \alpha(\mathbb{N})$ corresponds to the existentially quantified variables.

Let us denote by C_n^S the set of all primitive positive formulas in the free variables $x_0, \ldots, x_{n-1}$ for a given signature S. The lattice of all n-ary relations which can be defined in a given S-structure by formulas in C_n^S can be defined as the concept lattice of the context $((G_0)^n, C_n^S, \models)$, where $\models$ is the satisfaction relation. In Sect. 5.1, an equivalent construction is done using windowed intension graphs as patterns over a power context family.

4.1 Product and Sum

The product of a family of windowed intension graphs is given by

$$\bigtimes_{i\in I} (\alpha_i, G_i) := (\langle\vec{\alpha}\rangle, \bigtimes_{i\in I} G_i), \tag{23}$$

$$\langle\vec{\alpha}\rangle(n) := \begin{cases} (\alpha_i)_{i\in I} & \text{if } \alpha_i(n) \text{ is defined for all } i \in I, \\ \text{undefined} & \text{otherwise} \end{cases}. \tag{24}$$

The sum of windowed intension graphs is given by

$$(\alpha_1, G_1) + (\alpha_2, G_2) := ([\alpha_1, \alpha_2], G_1 \underset{\theta_{(\alpha_1,\alpha_2)}}{+} G_2), \tag{25}$$

where

$$\theta_{(\alpha_1,\alpha_2)} := \{(\alpha_1(k), \alpha_2(k)) \mid k \in \mathbb{N} \wedge \alpha_1(k) \text{ defined } \wedge \alpha_2(k) \text{ defined } \}, \tag{26}$$

$$[\alpha_1, \alpha_2](k) := \begin{cases} [(1, \alpha_1(k))]_{\theta_{(\alpha_1,\alpha_2)}} & \text{if } \alpha_1(k) \text{ is defined,} \\ [(2, \alpha_2(k))]_{\theta_{(\alpha_1,\alpha_2)}} & \text{if } \alpha_2(k) \text{ is defined, .} \\ \text{undefined} & \text{otherwise} \end{cases} \tag{27}$$

The sum can also be defined for arbitrary families $(\alpha_i, G_i)_{i\in I}$, but this is even more tedious and not needed in the following. We denote by $\mathbf{IG}_M^X$ the category of all windowed intension graphs where the first component has domain of definition X.

5 Pattern Concepts

5.1 Concept Lattices of Power Context Families

Let $\vec{\mathbb{K}} \in \mathbf{PCF}_M$ and $n \in \mathbb{N}$. We want to create the lattice which has as its extents all n-ary relations definable by windowed intension graphs (α, G), where α is defined on $n := \{0, \ldots, n-1\}$ (i.e., α is an n-tuple). The most specific description for an n-tuple α is the windowed intension graph

$$\delta_{\vec{\mathbb{K}}}^n(\alpha) := (\alpha, \Delta), \tag{28}$$

where $\Delta := \mathbf{ig}(\vec{\mathbb{K}})$. We state the pattern structure as a triple $((G_0)^n, \mathbf{IG}_M^n, \delta_{\vec{\mathbb{K}}}^n)$, where the second component is a category instead of, as usual, a semilattice. As noted before, the infimum operation in the morphism preorder is realized - up to pattern equivalence - by the categorical product (Sect. 4.1).

The Galois connection which arises from the pattern structure can be stated as follows:

$$A^\diamond := \bigtimes_{\lambda \in A} (\lambda, \Delta), \tag{29}$$

$$(\alpha, G)^\diamond := \{\lambda \in (G_0)^n \mid \exists \varphi \; \varphi : (\alpha, G) \to (\lambda, \Delta)\}. \tag{30}$$

The definitions in (20) and (30) coincide. The pattern concepts are the pairs $((\alpha, G)^\diamond, (\alpha, G)^{\diamond\diamond})$ for $G \in \mathbf{IG}_M$ and $\alpha \in (G_0)^n$. The same concepts arise as the pairs $(A^{\diamond\diamond}, A^{\diamond\diamond\diamond})$ for $A \subseteq (G_0)^n$; the patterns $A^\diamond$ and $A^{\diamond\diamond\diamond}$ are hom-equivalent, but generally not identical.

5.2 Example

We define a family $M := (\{a, b\}, \emptyset, \{r, s\}, \emptyset, \dots)$ of attribute sets. Figure 4 shows a power context family $\vec{\mathbb{K}}$ over M. The intension graph $\mathbf{ig}(\vec{\mathbb{K}})$ is shown in Fig. 5. It has three components, which are individually listed in Fig. 6 as components C_1, C_2 and C_3.

Let us construct the concept lattice for patterns in $\mathbf{IG}_M^1$ (Fig. 7). First of all, a pattern in $\mathbf{IG}_M^1$ can be stated as (x, G) with $x \in V_G$ (by writing $(\alpha(0), G)$ instead of (α, G)), and we may alternatively state this as (x, C), where C is the component of x. The object intents, given by $\delta_{\vec{\mathbb{K}}}^1$, are the patterns $(1, C_1)$,$(2, C_3)$,$(3, C_3)$,$(4, C_3)$,$(5, C_2)$ and $(6, C_2)$. In Fig. 7, they can be found directly on top of the pattern $((), C_0)$ for the bottom concept, which is generated by the empty product. The product $C_1 \times C_3$ has a single component, which is denoted C_5 (Fig. 6). This yields $(1, C_1) \times (j, C_3) = ((1, j), C_5)$ for $j = 2, 3, 4$. As we can see, when we multiply intension graphs, several products of windowed intension graphs are obtained at once. In this case, all three products are hom-equivalent, and yield the topmost circle pattern in Fig.7.

Let us generate all 2-generated concepts in lectic order. The next concept would be $(\{1, 5\}^{\diamond\diamond}, \{1, 5\}^\diamond)$ up to hom-equivalence. Formally, Sect. 5.1 states the intent as $\{1, 5\}^{\diamond\diamond\diamond}$, but this is not practically relevant. The product $C_1 \times C_2$ provides $\{1, 5\}^\diamond$ and $\{1, 6\}^\diamond$, which are different patterns (the co-atoms in Fig. 7) with the same underlying component C_7. To obtain $\{2, 3\}^\diamond$, we compute $C_3 \times C_3$, which is disconnected ($C_3 \times C_3 \cong C_3 \sqcup C_4 \sqcup C_4$). The new component C_4 yields the three remaining circle patterns in Fig. 7. Computing $\{2, 5\}^\diamond$ leads to $C_3 \times C_2 = C_6 \sqcup C_8$, which gives six new pattern concepts, and finally $\{5, 6\}^\diamond$ (the only missing combination) yields the top concept. There are two more patterns, which are 3-generated and have undelying component C_{10}.

All patterns produced were minimal (i.e. they have no proper hom-equivalent subpattern). The reason for this is that the nodes in the generating patterns

$\mathbb{K}_0$:

	a	b
1		
2		×
3	×	
4	×	
5		
6	×	×

$\mathbb{K}_2$:

	r	s
(1,1)	×	
(2,3)	×	×
(3,4)	×	
(4,2)	×	
(5,6)		×
(6,5)	×	

Fig. 4. Power context family

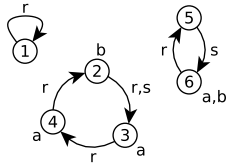


Fig. 5. Intension graph

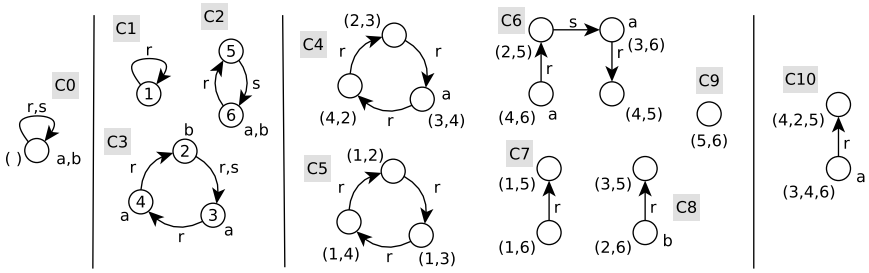


Fig. 6. Components

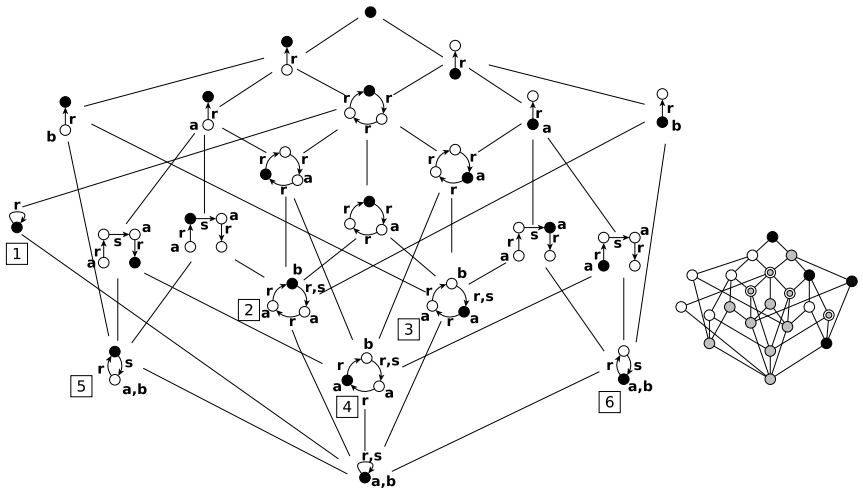


Fig. 7. Concept lattice (left), tree $\bigwedge$-sublattices (right)

have indegree and outdegree bounded by 1. Otherwise it may happen that a component contains two concepts (i.e. nodes), say X and Y, such that X is necessary to describe Y, but a minimal description of Y does not contain X, which has to be taken care of during lattice construction. Another point is that hom-equivalent patterns need to be discovered, which generally requires homomorphism checks. The inherent complexity can be avoided if patterns are restricted to trees (see Sect.6). An implementation of lattice construction, which currently uses a variant of Ganter's NextConcept algorithm [6], is available at `https://github.com/koetters/cgnav`.

6 Relational Concept Analysis and Tree Patterns

6.1 Relational Concept Analysis

Relational Concept Analysis uses relational scaling to express relations between objects by means of formal attributes. A number of scaling operators are defined, but only the wide scaling operator is covered here. The RCA algorithm builds a lattice from a Relational Context Family (RCF), which can be seen as a many-sorted PCF with binary relations only. We only deal with the one-sorted case, because the general case is implied. In this case, an RCF can be likened to a PCF with two contexts $\mathbb{K}_0$ and $\mathbb{K}_2$. The *RCA algorithm* definies an iterative procedure which incrementally adds new attributes to $\mathbb{K}_0$. The sequence of contexts can be described as follows:

$$\mathbb{K}^{(0)} := \mathbb{K}_0, \tag{31}$$

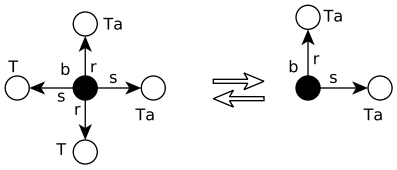
$$\mathbb{K}^{(i+1)} := \mathbb{K}_0 \mid (G_0, M_2 \times \underline{\mathcal{B}}(\mathbb{K}^{(i)}), J^{(i+1)}), \tag{32}$$

$$\text{where } gJ^{(i+1)}(r, C) :\Leftrightarrow \exists h : h \in \text{ext}(C) \wedge (g, h) \in r. \tag{33}$$

Consider the PCF from Fig. 2. To obtain $\mathbb{K}^{(1)}$, we first have to generate $\mathcal{B}(\mathbb{K}^{(0)})$. The concept lattice consists of the four black nodes shown in the miniature lattice in Fig. 7. Relational scaling produces eight new attributes. Fig.8 shows the context $\mathbb{K}^{(1)}$. The left tree in Fig. 9 represents the intent of the object 2 in $\mathbb{K}^{(1)}$ by a tree pattern. The neighbors of the black node are supposed to represent concepts of $\mathcal{B}(\mathbb{K}^{(0)})$, and the full tree pattern is obtained by substituting these with their pattern intents (this adds two occurrences of a). The right tree in Fig. 9 represents a minimal hom-equivalent subpattern. The lattice $\mathcal{B}(\mathbb{K}^{(1)})$ can be generated by intersecting all object intents (as attribute sets) or alternatively, by computing the tree products. The lattice $\mathcal{B}(\mathbb{K}^{(2)})$ consists of the gray nodes in addition to the black nodes (Fig. 7). The context $\mathbb{K}^{(3)}$ is a fixed point of the RCA algorithm, the final lattice additionally contains the dotted nodes. The white nodes are not discovered by the RCA algorithm (although five of them can be discovered by adding r^{-1} and s^{-1} to $\mathbb{K}_2$). The tree intents of the objects in $\mathbb{K}^{(i)}$ can be obtained directly from $\text{ig}_M(\vec{\kappa})$ using the $\text{splice}^{(i)}$ operation from Sect. 6.3. It is also shown that splice^i is a pattern projection(cf.[5]), which enables the use of pattern structure algorithms for RCA. The rest of the section proves the relevant claims.

	a	b	$\exists r:\top$	$\exists r:T_a$	$\exists r:T_b$	$\exists r:\bot$	$\exists s:\top$	$\exists s:T_a$	$\exists s:T_b$	$\exists s:\bot$
1			×							
2		×	×	×			×	×		
3	×		×	×						
4	×		×		×					
5							×	×	×	×
6	×	×	×							

Fig. 8. Scaled Context



$\top := ((5,6), C_9)$ $T_a := ((3,4,6), C_{10})$
$T_b := ((2,6), C_8)$ $\bot := (6, C_2)$

Fig. 9. Equivalent Object Patterns

6.2 Rooted Trees

Definition 5. *A* rooted tree *is a windowed intension graph which can be constructed by the following rules:*

(RT1) *For a given set $B \subseteq M_0$ of attributes, the windowed intension graph $(0, \mathcal{E}_B(0))$ is a rooted tree with*

$$\operatorname{depth}((0, \mathcal{E}_B(0))) := 0. \tag{34}$$

(RT2) *For a given set $B \subseteq M_0$ of attributes, an index set $I \neq \emptyset$, a family $(R_i)_{i\in I}$ of attribute sets $R_i \subseteq M_2$, $R_i \neq \emptyset$, and a family $(x_i, T_i)_{i\in I}$ of rooted trees such that $\sup_{i\in I} \operatorname{depth}((x_i, T_i)) < \infty$, the windowed intension graph*

$$(x, T) := (0, \mathcal{E}_B(0)) + \sum_{i\in I} (0, \mathcal{S}_{R_i}(0,1) \underset{\{(1,x_i)\}}{+} T_i) \tag{35}$$

is a rooted tree with

$$\operatorname{depth}((x, T)) := 1 + \max_{i\in I} \operatorname{depth}((x_i, T_i)). \tag{36}$$

A rooted tree is called thin *if it can be constructed by rules* (**RT1**) *and* (**RT2′**)*, where* (**RT2′**) *is obtained from* (**RT2**) *by adding the additional requirement that $|R_i| = 1$ for all $i \in I$.*

We denote by $\mathbf{It}_{M,n}$ the subcategory of $\mathbf{IG}_M$ that consists of the thin rooted trees of depth at most n.

Proposition 2. *Let $n \in \mathbb{N}$. The concept extents of $\underline{\mathcal{B}}(\mathbb{K}^{(n)})$ are precisely the sets $(x, T)^\diamond$ described by thin rooted trees (x, T) of depth $\leq n$.*

Proof. This is proved by induction over $n \in \mathbb{N}$. For $n = 0$, the claim follows from (21). If T is a thin rooted tree with $\operatorname{depth}(T) = n + 1$, there is a family $(x_i, T_i)_{i\in I}$ of thin rooted trees of depth $\leq n$ and a family $(r_i)_{i\in I}$ of attributes in M_2, such that $(x, T) = (0, \mathcal{E}_B(0)) + \sum_{i\in I} (0, \mathcal{S}_{\{r_i\}}(0,1) \underset{\{(1,x_i)\}}{+} T_i)$. By the

induction hypothesis, there exists a family $(C_i)_{i\in I}$ of concepts $C_i \in \underline{\mathcal{B}}(\mathbb{K}^{(n)})$ with $(x_i, T_i)^\diamond = \mathrm{ext}(C_i)$. Then

$$\begin{aligned}(x,T)^\diamond =&((0,\mathcal{E}_B(0)) + \sum_{i\in I}(0,\mathcal{S}_{\{r_i\}}(0,1) \underset{\{(1,x_i)\}}{+} T_i))^\diamond && (37)\\ =&B' \cap \bigcap_{i\in I} r_i^{-1}((x_i,T_i)^\diamond) && (38)\\ =&B' \cap \bigcap_{i\in I} r_i^{-1}(\mathrm{ext}(C_i)) && (39)\\ =&(B \cup \{(r_i, C_i) \mid i \in I\})'. && (40)\end{aligned}$$

This shows that $(x, T^\diamond)$ is a concept extent in $\underline{\mathcal{B}}(\mathbb{K}^{(n+1)})$. Conversely, a concept extent in $\underline{\mathcal{B}}(\mathbb{K}^{(n+1)})$ is defined by an attribute set as in (40), and the induction hypothesis is used in (39) to obtain the family $(x_i, T_i)_{i\in I}$ for given $(C_i)_{i\in I}$. □

6.3 Graph Splicing

A graph $G \in \mathbf{IG}_M$ can be unfolded into a (possibly infinite) tree, starting at any given vertex which becomes the root of the tree. It is easy to see that, among all trees more general than G, the unfolding is the most specific one. In other words, the unfolding is a kernel operation (the dual of a closure operation). This implies that an $\bigwedge$-sublattice is obtained if patterns are restricted to trees. A similar operation constructs a thin rooted tree from a graph: In addition to unfolding, every edge carrying multiple relation attributes is spliced into several edges, each carrying exactly one of the relation attributes. The operation is formalized by the following inductive definition, where $G \in \mathbf{IG}_M$ and $x \in V_G$:

$$\mathrm{splice}^{(0)}(x,G) := (x, \mathcal{E}_{\kappa(x)}(x)), \tag{41}$$

$$\begin{aligned}\mathrm{splice}^{(i+1)}(x,G) :=& (x,\mathcal{E}_{\kappa(x)}(x)) + \sum_{(x,y)I_2 r} (x, \mathcal{S}_{\{r\}}(x,y) \underset{\{(y,\tilde{y})\}}{+} T_y^{(i)}),\\ \text{where } (\tilde{y}, T_y^{(i)}) :=& \mathrm{splice}^{(i)}(y,G)\end{aligned} \tag{42}$$

The following proposition states, in category theoretical terms, that the splice operation maps each $(x, G) \in \mathbf{IG}_M^1$ to its coreflection in $\mathbf{It_M}$ (cf. Fig. 3). As can be seem from Fig. 3, this implies that splicing is a kernel operation (or pattern projection). This means that the pattern structure $(G_0, \mathbf{It}_M, \mathrm{splice} \circ \delta^1_{\mathbb{K}})$ creates the concepts of the RCA algorithm.

Proposition 3. *For each $(x, G) \in \mathbf{IG}_M^1$, there exists a morphism $\varphi_{(x,G)} : \mathrm{splice}((x, G)) \to (x, G)$ such that for every $(y, T) \in \mathbf{It_M}$ and $\varphi : (y, T) \to (x, G)$ there exists a unique morphism $\psi : (y, T) \to \mathrm{splice}((x, G))$ such that $\varphi = \varphi_{(x,G)} \circ \psi$.*

Proof sketch: We inductively prove a unique morphism $\psi^{(i)} : \mathrm{splice}^{(i)}((y, T)) \to \mathrm{splice}((x, G))$. In the induction step, the image of each neighbor of the root node

is uniquely determined. The union of the $\psi := \psi^{(i)}$ is well-defined (because of the uniqueness). Since $\mathrm{splice}((x,T)) = (x,T)$ holds, φ is the required morphism. □

From a given graph $G \in \mathbf{IG}_M^1$, we can determine for each $x \in V_G$ the extent $\mathrm{ext}_\Delta(x)$ in $\mathrm{splice}((x,G))$ without actually splicing the graph. Let us denote this as the tree extent $\mathrm{tex}_\Delta(x)$ of x in G. The tree extent can be computed as follows:

$$\mathrm{tex}_\Delta^{(0)}(x) := \kappa_G(x)^{I_k} \text{ for } x \in E_G^{(k)}, \tag{43}$$

$$\mathrm{tex}_\Delta^{(i+1)}(x) := \mathrm{tex}_\Delta^{(0)}(x) \cap \bigcap_{(x,y)I_2 r} r^{-1}(\mathrm{tex}_\Delta^{(i)}(y)). \tag{44}$$

This can be proven by inductively showing $\mathrm{splice}^{(i)}((x,G))^\diamond = \mathrm{tex}_\Delta^{(i)}(x)$.

7 Related Work

Power context families and concept graphs have been introduced by Rudolf Wille in [11]. Concept graphs have been presented as a mathematical formalization of Conceptual Graphs [10]. Different kinds of concept graphs are presented in [12] but, to the knowledge of the author, abstract concept graphs mentioned in introductory paper [11] are the only kind of concept graphs defined without a realization. Abstract concept graphs use symbols as node labels rather than sets of attributes.

In [8], windowed structures have been introduced as triples $(X, \nu, \mathcal{G})$, and a Galois connection into a complete lattice of data tables (where the infimum is realized by the *join* for database tables) has been presented. A follow-up paper [9] addresses the connection to logic, features sorts, uses a "relational structure with concept labels" hybrid and shows the connection to pattern structures by representing extensions as sets of partial interpretations. Pattern structures were introduced in [5], and the use of Conceptual Graphs as patterns is suggested in there. The representation of conjunctive queries (and thus pp formulas) by graphs, and of entailment by graph homomorphism, is credited to [2]. In [3], these relationships are stated for λ-BGs, which are Basic Conceptual Graphs with distinguished concepts given by a mapping λ, and this representation directly corresponds to the windowed abstract concept graphs (and their homomorphisms) in the paper at hand. Moreover, in [3, Chapter 8], the categorical product is used to describe the least generalization of Conceptual Graphs. The Projected Graph Patterns (PGPs) in [4], their inclusion and intersection, corresponds to λ-BGs and windowed abstract concept graphs and their respective notions of homomorphism and product. In [4], as in [8], concept lattices are generated, with intents realized using the respective formalizations.

Relational Context Families and the construction algorithm are described in [7], and the RCA algorithm has been described for different kinds of inter-object relations which are not covered here.

8 Conclusion

The paper has introduced windowed intension graphs as a formalization of conjunctive queries. Intension graphs correspond to concept graphs without the realization component. Some notation has been introduced which establishes connections to logic and database theory. The lattices generated by the RCA algorithm have been characterized as $\bigwedge$-sublattices of conjunctive queries. The results concerning rooted trees still have to be implemented and compared with the RCA algorithm. While a bound for the maximum number of iterations of the RCA algorithm can be given by $|G|$, the pattern structures algorithms might benefit from a better bound computed in advance from the context family.

References

1. Adámek, J., Herrlich, H., Strecker, G.E.: Abstract and concrete categories: the joy of cats. Pure and applied mathematics, Wiley, New York (1990)
2. Chandra, A.K., Merlin, P.M.: Optimal implementation of conjunctive queries in relational databases. In: Proceedings of the ninth annual ACM symposium on Theory of computing. pp. 77–90. STOC '77, ACM, New York, NY, USA (1977)
3. Chein, M., Mugnier, M.L.: Graph-based Knowledge Representation: Computational Foundations of Conceptual Graphs. Advanced Information and Knowledge Processing, Springer, London (2009)
4. Ferré, S.: A proposal for extending formal concept analysis to knowledge graphs. In: Baixeries, J., Sacarea, C., Ojeda-Aciego, M. (eds.) Formal Concept Analysis - 13th International Conference, ICFCA 2015, Nerja, Spain, June 23-26, 2015, Proceedings. LNAI, vol. 9113, pp. 271–286. Springer (2015)
5. Ganter, B., Kuznetsov, S.O.: Pattern structures and their projections. In: Delugach, H.S., Stumme, G. (eds.) Proceedings of ICCS 2001. LNCS, vol. 2120, pp. 129–142. Springer (2001)
6. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer, Berlin (1999)
7. Huchard, M., Hacene, M.R., Roume, C., Valtchev, P.: Relational concept discovery in structured datasets. Annals of Mathematics and Artificial Intelligence 49(1-4), 39–76 (2007)
8. Kötters, J.: Concept lattices of a relational structure. In: Pfeiffer, H.D., Ignatov, D.I., Poelmans, J., Gadiraju, N. (eds.) Proceedings of ICCS 2013. LNCS, vol. 7735, pp. 301–310. Springer (2013)
9. Kötters, J., Schmidt, H.W.: A database browser based on pattern concepts. In: Carpineto, C., Kuznetsov, S.O., Napoli, A. (eds.) Proceedings of FCAIR 2013. CEUR Workshop Proceedings, vol. 977. CEUR-WS.org (2013), `http://ceur-ws.org/Vol-977`
10. Sowa, J.F.: Conceptual Structures: Information Processing in Mind and Machine. Addison-Wesley (1984)
11. Wille, R.: Conceptual Graphs and Formal Concept Analysis. In: Lukose, D., Delugach, H.S., Keeler, M., Searle, L., Sowa, J.F. (eds.) Proceedings of ICCS 1997. LNCS, vol. 1257, pp. 290–303. Springer, Heidelberg (1997)
12. Wille, R.: Formal Concept Analysis and Contextual Logic. In: Hitzler, P., Schärfe, H. (eds.) Conceptual Structures in Practice, pp. 137–173. Chapman & Hall/CRC (2009)