TECHNISCHE UNIVERSITÄT DRESDEN

Faculty of Computer Science • Institute of Theoretical Computer Science • Chair of Automata Theory

How KR benefits from Formal Concept Analysis

Session 1/2

Francesco Kriegel*, Barış Sertkaya (Frankfurt UAS)

*Funded by DFG in Project 430150274 (Repairing Description Logic Ontologies) and partially in Project 389792660 (TRR 248: Foundations of Perspicuous Software Systems).

ReDLO Repairing Description Logic Ontologies

CPEC CENTER FOR PERSPICUOUS COMPUTING

20th International Conference on Principles of Knowledge Representation and Reasoning (KR 2023), 2–8 September 2023

Tutorial Outline

Session 1:

1. Conceptual clustering with FCA (Francesco)
2. Extracting dependencies with FCA (Barış)

Session 2:

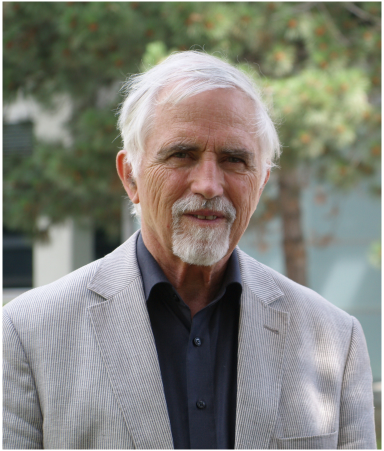
3. Acquiring complete knowledge about an application domain, enriching OWL ontologies (Barış)
4. Mining axioms from interpretations and knowledge graphs (Francesco)
5. Computing Stable Extensions of Argumentation Frameworks using FCA (Barış)

Early Days of Formal Concept Analysis

Early Days of Formal Concept Analysis

Rudolf Wille: **Restructuring Lattice Theory: An Approach Based on Hierarchies of Concepts.** Symposium on Ordered Sets, 1981

"Lattice theory today reflects the general status of current mathematics: there is a rich production of theoretical concepts, results, and developments, many of which are reached by elaborate mental gymnastics; on the other hand, the connections of the theory to its surroundings are getting weaker and weaker, with the result that the theory and even many of its parts become more isolated. Restructuring lattice theory is an attempt to reinvigorate connections with our general culture by interpreting the theory as concretely as possible, and in this way to promote better communication between lattice theorists and potential users of lattice theory."



Rudolf Wille (∗ 1937, † 2017)

Early Days of Formal Concept Analysis

In 1983 Rudolf Wille established the research group on FCA. One of its first members was Bernhard Ganter.

They published the first and only textbook on FCA:

Bernhard Ganter, Rudolf Wille: **Formal Concept Analysis - Mathematical Foundations.** Springer, 1996 (in German) resp. 1999 (in English)

International community on FCA:

- International Conference on Conceptual Structures (ICCS), since 1993
- International Conference on Concept Lattices and their Applications (CLA), since 2002
- International Conference on Formal Concept Analysis (ICFCA), since 2004

Bernhard Ganter ($*$ 1949)

Basics of Formal Concept Analysis

Representing Data in FCA

Definition. A formal context (G, M, I) consists of a set G of objects, a set M of attributes, and an incidence relation $I \subseteq G \times M$ where $(g, m) \in I$ indicates that object g has attribute m.

Example. We consider the formal context $\mathbb{K}_{\text{Cities}} := (G, M, I)$ about cities and waterbodies.

- It has become a tradition in FCA literature to represent formal contexts as "cross tables."
- The rows show the objects in G, which are here five cities.
- The columns show the attributes in M, which here indicate presence of waterbodies.
- A cross indicates a pair in the incidence relation I, e.g., $(\text{Dresden}, \text{River}) \in I$ but $(\text{Rhodes}, \text{Lake}) \notin I$.

$\mathbb{K}_{\text{Cities}}$	River	Lake	Sea
Dresden	×	·	·
Rhodes	·	·	×
Madrid	·	·	·
Lisbon	×	·	×
Kyiv	×	×	·

The Commonality Operators of a Formal Context

Definition. Given a formal context (G, M, I), we define two mappings:

1. for each object subset $A \subseteq G$, let $A^I := \{ m \mid m \in M \text{ and } (g, m) \in I \text{ for each } g \in A \}$,
2. for each attribute subset $B \subseteq M$, let $B^I := \{ g \mid g \in G \text{ and } (g, m) \in I \text{ for each } m \in B \}$.

Example. We reconsider $\mathbb{K}_{\text{Cities}}$.

- In the cross table, $\{g\}^I$ is the row of g and $\{m\}^I$ is the column of m, e.g., $\{\text{Kyiv}\}^I = \{\text{River}, \text{Lake}\}$.
- For sets with more than one object we intersect the rows: $\{g_1, \ldots, g_n\}^I = \{g_1\}^I \cap \cdots \cap \{g_n\}^I$. Likewise for attribute sets. For instance, $\{\text{Lisbon}, \text{Kyiv}\}^I = \{\text{River}\}$.
- For empty sets we have $\emptyset^I = M$ resp. $\emptyset^I = G$.

$\mathbb{K}_{\text{Cities}}$	River	Lake	Sea
Dresden	×	·	·
Rhodes	·	·	×
Madrid	·	·	·
Lisbon	×	·	×
Kyiv	×	×	·

Formal Concepts

Definition. A formal concept is a pair (A, B) consisting of subsets $A \subseteq G$ and $B \subseteq M$ with $A^I = B$ and $B^I = A$, where A is the extent and B is the intent.

Example. We reconsider $\mathbb{K}_{\text{Cities}}$.

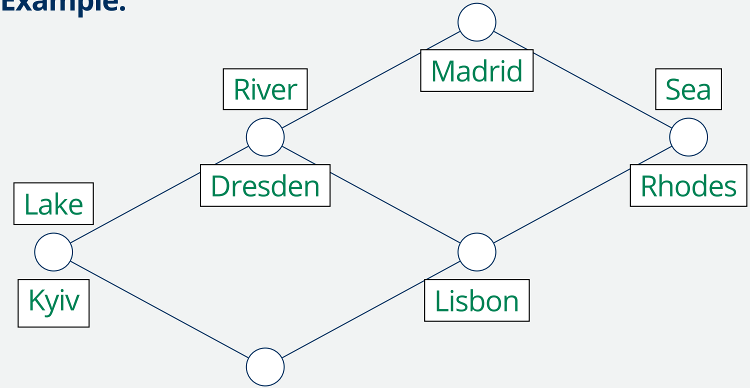
- $(\{\text{Dresden}\}, \{\text{River}\})$ is no formal concept, since $\{\text{River}\}^I = \{\text{Dresden}, \text{Lisbon}, \text{Kyiv}\} \neq \{\text{Dresden}\}$.
- $(\{\text{Dresden}, \text{Lisbon}, \text{Kyiv}\}, \{\text{River}\})$ is a formal concept, since $\{\text{Dresden}, \text{Lisbon}, \text{Kyiv}\}^I = \{\text{River}\}$ and $\{\text{River}\}^I = \{\text{Dresden}, \text{Lisbon}, \text{Kyiv}\}$.
- Similarly, $(\{\text{Madrid}\}, \emptyset)$ is no formal concept, but $(\{\text{Dresden}, \text{Rhodes}, \text{Madrid}, \text{Lisbon}, \text{Kyiv}\}, \emptyset)$ is a concept.

$\mathbb{K}_{\text{Cities}}$	River	Lake	Sea
Dresden	×	·	·
Rhodes	·	·	×
Madrid	·	·	·
Lisbon	×	·	×
Kyiv	×	×	·

The Concept Lattice

Definition. The concept lattice of (G, M, I) is the set of all formal concepts, ordered by $(A, B) \leq (C, D)$ iff $A \subseteq C$ (equivalently: iff $D \subseteq B$).

Example.



$\mathbb{K}_{\text{Cities}}$	River	Lake	Sea
Dresden	×	·	·
Rhodes	·	·	×
Madrid	·	·	·
Lisbon	×	·	×
Kyiv	×	×	·

Live Demo

Concept Explorer FX

Source Code: `https://github.com/francesco-kriegel/conexp-fx`

Live Demo

Technical Details

The Concept Lattice is Complete

Theorem. For each formal context (G,M,I), its concept lattice is a complete lattice:

1. For all concepts $(A_1,B_1),\ldots,(A_n,B_n)$, the set
$$\{\,(C,D)\mid(A_1,B_1)\leq(C,D),\ \ldots,\ (A_n,B_n)\leq(C,D)\,\}$$
has a smallest element. It is called the supremum of these concepts, is denoted as $(A_1,B_1)\vee\cdots\vee(A_n,B_n)$, and satisfies the following equation:
$$(A_1,B_1)\vee\cdots\vee(A_n,B_n)=((B_1\cap\cdots\cap B_n)^I,\ B_1\cap\cdots\cap B_n).$$

2. For all concepts $(A_1,B_1),\ldots,(A_n,B_n)$, the set
$$\{\,(C,D)\mid(C,D)\leq(A_1,B_1),\ \ldots,\ (C,D)\leq(A_n,B_n)\,\}$$
has a greatest element. It is called the infimum of these concepts, is denoted as $(A_1,B_1)\wedge\cdots\wedge(A_n,B_n)$, and satisfies the following equation:
$$(A_1,B_1)\wedge\cdots\wedge(A_n,B_n)=(A_1\cap\cdots\cap A_n,\ (A_1\cap\cdots\cap A_n)^I).$$

The Galois Connection of a Formal Context

We often apply the commonality operators one after the other and simply write A^{II} for $(A^I)^I$.

Lemma. The two mappings $A \mapsto A^I$ and $B \mapsto B^I$ form a Galois connection:

1. $A \subseteq B^I$ iff $B \subseteq A^I$ iff $A \times B \subseteq I$
2. If $A \subseteq C$, then $C^I \subseteq A^I$
3. $A \subseteq A^{II}$
4. $A^I = A^{III}$
5. If $B \subseteq D$, then $D^I \subseteq B^I$
6. $B \subseteq B^{II}$
7. $B^I = B^{III}$

The Galois Connection of a Formal Context

We often apply the commonality operators one after the other and simply write A^{II} for $(A^I)^I$.

Lemma. The two mappings $A \mapsto A^I$ and $B \mapsto B^I$ form a Galois connection:

1. $A \subseteq B^I$ iff $B \subseteq A^I$ iff $A \times B \subseteq I$
2. If $A \subseteq C$, then $C^I \subseteq A^I$
3. $A \subseteq A^{II}$
4. $A^I = A^{III}$
5. If $B \subseteq D$, then $D^I \subseteq B^I$
6. $B \subseteq B^{II}$
7. $B^I = B^{III}$

It follows that the mappings $A \mapsto A^{II}$ and $B \mapsto B^{II}$ are closure operators on G resp. M.

Definition. A closure operator on M is a mapping $\varphi\colon M \to M$ that is

1. extensive: $B \subseteq B^\varphi$
2. monotonic: $B \subseteq D$ implies $B^\varphi \subseteq D^\varphi$
3. idempotent: $B^{\varphi\varphi} = B^\varphi$.

Each subset of the form B^φ is called a closure of φ.

Computing the Concept Lattice

Computing all Formal Concepts

There are several algorithms for computing all formal concepts:

- *NextClosure*
- *Close-by-One (CbO)*
- *Fast Close-by-One (FCbO)*
- *In-Close2, In-Close3, In-Close4, In-Close5*
- *LCM*

At the core, all compute the concepts in a similar way — but they are optimized differently.

Bernhard Ganter: **Two Basic Algorithms in Concept Analysis.** FB4-Preprint 831, Technische Hochschule Darmstadt, 1984.
Sergei O. Kuznetsov: **A fast algorithm for computing all intersections of objects from an arbitrary semilattice.** 1993.
Petr Krajča, Jan Outrata, Vilém Vychodil: **Advances in Algorithms Based on CbO.** CLA 2010.
Simon Andrews: **In-Close2, a High Performance Formal Concept Miner.** ICCS 2011.
Simon Andrews: **A Partial-Closure Canonicity Test to Increase the Efficiency of CbO-Type Algorithms.** ICCS 2014.
Simon Andrews: **Making Use of Empty Intersections to Improve the Performance of CbO-Type Algorithms.** ICFCA 2017.
Simon Andrews: **A New Method for Inheriting Canonicity Test Failures in Close-by-One Type Algorithms.** CLA 2018.
Radek Janoštík, Jan Konečný, Petr Krajča: **LCM from FCA point of view: A CbO-style algorithm with speed-up features.** Int. J. Approx. Reason. 142, 2022.

Sergei O. Kuznetsov, Sergei A. Obiedkov: **Comparing performance of algorithms for generating concept lattices.** J. Exp. Theor. Artif. Intell. 14.2-3, 2002.
Jan Konečný, Petr Krajča: **Systematic categorization and evaluation of CbO-based algorithms in FCA.** Inf. Sci. 575, 2021.

Computing all Formal Concepts

Proposition. Each concept of a context (G, M, I) has the form (B^I, B) for some closure B of the closure operator $B \mapsto B^{II}$. Conversely, all such pairs are concepts.

In order to compute all concepts, it is thus sufficient to compute all closures of $B \mapsto B^{II}$.

Bernhard Ganter, Rudolf Wille: **Formal Concept Analysis - Mathematical Foundations.** 1996 resp. 1999.

Computing all Formal Concepts

Proposition. Each concept of a context (G, M, I) has the form (B^I, B) for some closure B of the closure operator $B \mapsto B^{II}$. Conversely, all such pairs are concepts.

In order to compute all concepts, it is thus sufficient to compute all closures of $B \mapsto B^{II}$.

A simple approach to computing all closures of a closure operator φ on M is as follows.

1. The smallest closure is $\emptyset^\varphi$.
2. When we have found a closure B, then all next closures above must be of the form $(B \cup \{m\})^\varphi$ for some attribute $m \in M \setminus B$.

   If there is a closure D with $B \cup \{m\} \subseteq D \subseteq (B \cup \{m\})^\varphi$, then $(B \cup \{m\})^\varphi \subseteq D \subseteq (B \cup \{m\})^\varphi$ since φ is monotonic and idempotent, i.e., $D = (B \cup \{m\})^\varphi$. We thus do not miss closures in between.

$\mathfrak{C} := \emptyset$
$\texttt{Recurse}(\emptyset^\varphi)$
def $\texttt{Recurse}(B)$
  $\mathfrak{C} := \mathfrak{C} \cup \{B\}$
  foreach $m \in M \setminus B$
    $\texttt{Recurse}((B \cup \{m\})^\varphi)$
return $\mathfrak{C}$

Bernhard Ganter, Rudolf Wille: **Formal Concept Analysis - Mathematical Foundations.** 1996 resp. 1999.

Close-by-One (CbO)

Although it works, this simple approach should not be used in implementations since it computes a lot of duplicates.

Close-by-One (CbO) tries to avoid unnecessary recursive calls by means of a special ordering of the subsets of M.

Definition. Let $\leq$ be a linear order on M. The lexicographic tree order $\sqsubseteq$ on subsets of M is defined by $B \sqsubseteq D$ if $B \subseteq D$ and $m \leq n$ for each $m \in B$ and each $n \in D \setminus B$.

A linear order $\leq$ on M can be given by means of an enumeration $M = \{m_1, \ldots, m_\ell\}$ with $m_i \leq m_j$ if $i \leq j$.

Sergei O. Kuznetsov: **A fast algorithm for computing all intersections of objects from an arbitrary semilattice.** 1993.

Close-by-One (CbO)

Although it works, this simple approach should not be used in implementations since it computes a lot of duplicates.

Close-by-One (CbO) tries to avoid unnecessary recursive calls by means of a special ordering of the subsets of M.

Definition. Let $\leq$ be a linear order on M. The lexicographic tree order $\sqsubseteq$ on subsets of M is defined by $B \sqsubseteq D$ if $B \subseteq D$ and $m \leq n$ for each $m \in B$ and each $n \in D \setminus B$.

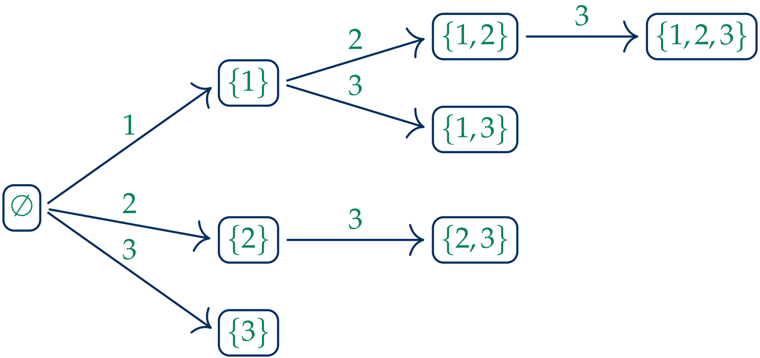
A linear order $\leq$ on M can be given by means of an enumeration $M = \{m_1, \ldots, m_\ell\}$ with $m_i \leq m_j$ if $i \leq j$.

1. The first if-condition ensures that $B \sqsubseteq B \cup \{n\}$.
2. The second if-condition is called canonicity test and ensures $B \sqsubseteq (B \cup \{n\})^\varphi$.

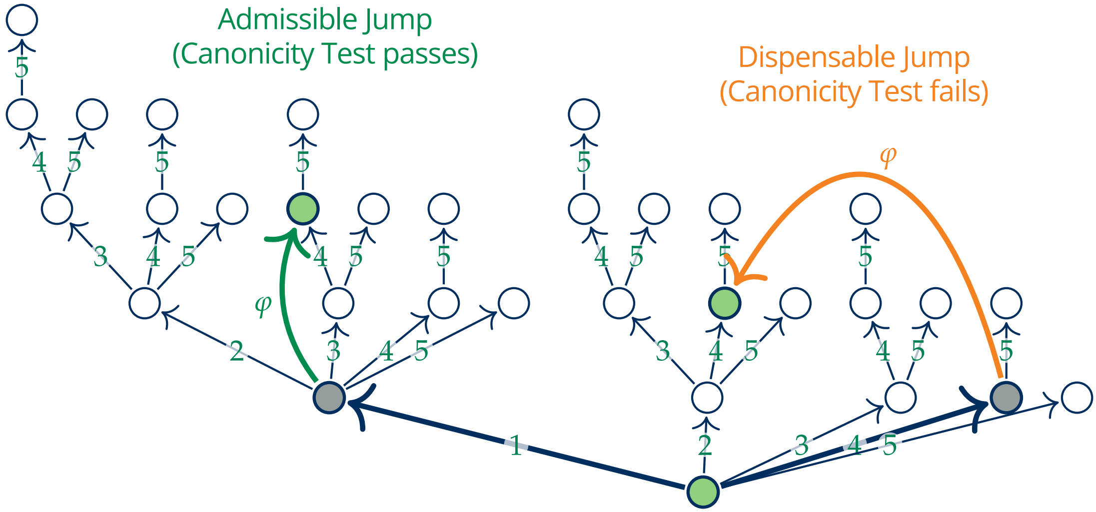
```
𝔈 := ∅
Recurse(∅^φ, ·)
def Recurse(B, m)
    𝔈 := 𝔈 ∪ {B}
    foreach n ∈ M \ B
        if m ≤ n
            D := (B ∪ {n})^φ
            if B ⊑ D
                Recurse(D, n)
return 𝔈
```

Sergei O. Kuznetsov: **A fast algorithm for computing all intersections of objects from an arbitrary semilattice.** 1993.

The Lexicographic Tree Order



The Canonicity Test



Computing the Hierarchical Order

When we want to navigate through all formal concepts, we also need to compute the hierarchical order $\leq$ on them. Recall that $(A,B) \leq (C,D)$ iff $A \subseteq C$ (equivalently: iff $D \subseteq B$).

The naïve, inefficient way is to consider all pairs of concepts, but this is infeasible for larger contexts.

Instead, we can efficiently compute the neighborhood relation $\prec$ between concepts, where $(A,B) \prec (C,D)$ iff $(A,B) < (C,D)$ and there is no concept (E,F) with $(A,B) < (E,F) < (C,D)$. The hierarchical order $\leq$ is then the reflexive, transitive closure of $\prec$.

- The line diagram can be easily drawn from $\prec$ (but not from $\leq$).
- In applications where the concept lattice is not shown as a line diagram but can rather be browsed concept by concept, the immediate sub-concepts and super-concepts of the current concept can be read off from $\prec$.

Petko Valtchev, Rokia Missaoui, Pierre Lebrun: **A Fast Algorithm for Building the Hasse Diagram of a Galois Lattice.** Colloque LaCIM 2000.
Ben Martin, Peter Eklund: From Concepts to Concept Lattice: **A Border Algorithm for Making Covers Explicit.** ICFCA 2008.
Jaume Baixeries, Laszlo Szathmary, Petko Valtchev, Robert Godin: **Yet a Faster Algorithm for Building the Hasse Diagram of a Concept Lattice.** ICFCA 2009.

Do you have questions or comments?