Algorithmic Problems of FCA
Sergei O. Kuznetsov
FCA
Computing Concept Lattices
Intractability Results
Algorithms and Their Complexity
Computing Implication Bases
Intractability of pseudo-intents recognition
Corollaries
Conclusion

Algorithmic Problems of FCA

Sergei O. Kuznetsov

School for Applied Mathematics and Information Science,
National Research University Higher School of Economics,
Moscow, Russia

european social fund in the czech republic
EUROPEAN UNION
MINISTRY OF EDUCATION, YOUTH AND SPORTS
OP Education for Competitiveness
INVESTMENTS IN EDUCATION DEVELOPMENT

University of Olomouc

January 12, 2012

Outline

1. Formal Concept Analysis
2. Computing Concept Lattices
   - Intractability Results
   - Algorithms and Their Complexity
3. Computing Implication Bases
   - Intractability of pseudo-intents recognition
   - Corollaries
4. Conclusion

Basic notions of FCA

[Wille 1982], [Ganter, Wille 1996]

- M, a set of **attributes**
- G, a set of **objects**
- relation $I \subseteq G \times M$ such that $(g, m) \in I$ if and only if the object g has the attribute m.
- $\mathbb{K} := (G, M, I)$ is a **formal context**.

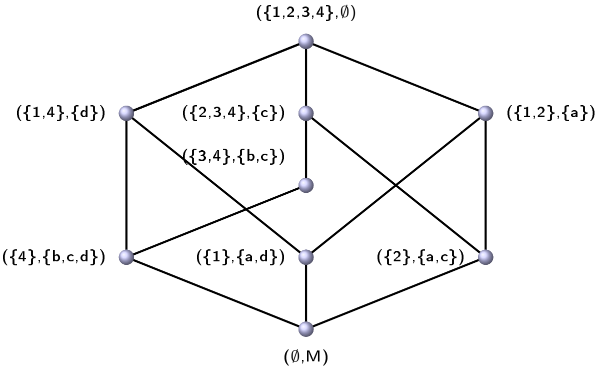
Derivation operators: $A \subseteq G$, $B \subseteq M$

$$A' \stackrel{\text{def}}{=} \{m \in M \mid gIm \text{ for all } g \in A\}, \quad B' \stackrel{\text{def}}{=} \{g \in G \mid gIm \text{ for all } m \in B\}$$

A **formal concept** is a pair (A, B): $A \subseteq G,\ B \subseteq M,\ A' = B, \text{ and } B' = A$.

- A is the **extent** and B is the **intent** of the concept (A, B).
- The concepts, ordered by $(A_1, B_1) \geq (A_2, B_2) \iff A_1 \supseteq A_2$ form a complete lattice, called **the concept lattice** $\mathfrak{B}(G, M, I)$.

Example: A formal context and its concept lattice



	G \ M	a	b	c	d
1		×			×
2		×		×	
3			×	×	
4			×	×	×

a – has exactly 3 vertices,

b – has exactly 4 vertices,

c – has a right angle,

d – is equilateral

Implications

- **Implication** $A \to B$ for $A, B \subseteq M$ holds if $A' \subseteq B'$, i.e., every object that has all attributes from A also has all attributes from B.
- **Armstrong rules**:

$$\frac{A \to B}{A \cup C \to B}, \quad \frac{A \to B, A \to C}{A \to B \cup C}, \quad \frac{A \to B, B \to C}{A \to C}$$

- **A Minimal implication base**:
  A base with the minimum number of implications [Duquenne, Guigues 1986] or
  the **stem base**, its premises can be given (Ganter 1987) by pseudointents:
  - A set $P \subseteq M$ is a **pseudointent** if
    $P \neq P''$ and
    $Q'' \subset P$ for every pseudointent $Q \subset P$.

Implications and functional dependencies. 1

Functional dependency, viz. [D. Meier 1983], in terms of FCA:
$X \to Y$ is a **functional dependency** in a complete many-valued context (G, M, W, I) if the following holds for every pair of objects $g, h \in G$:

$$(\forall m \in X \;\; m(g) = m(h)) \Rightarrow (\forall n \in Y \;\; n(g) = n(h)).$$

The reduction of functional dependencies to implications [Ganter, Wille 1996]:
Proposition A. For a many-valued context (G, M, W, I), one defines the context $K_N := (\mathcal{P}_2(G), M, I_N)$, where $\mathcal{P}_2(G)$ is the set of all pairs of different objects from G and I_N is defined by

$$\{g, h\} I_N m :\Leftrightarrow m(g) = m(h).$$

Then a set $Y \subseteq M$ is functionally dependent on the set $X \subseteq M$ if and only if the implication $X \to Y$ holds in the context K_N.

Implications and functional dependencies. 2

An inverse reduction [Kuznetsov 2001]:

Proposition B. For a context $K = (G, M, I)$ one can construct a many-valued context K_W such that an implication $X \rightarrow Y$ holds if and only if Y is functionally dependent on X in K_W.

Intractability of computing all concepts.

- The number of concepts can be exponential in the context size: $(A, A, \neq)$.
- The problem of determining the number of all concepts of a context is $\#P$-complete: Reduction from monotone 2-CNF problem [Kuznetsov 1989, 2001].

  Recall: #P is the class of counting problems associated with the decision problems in NP: A problem is in #P if there is a polynomial NDTM that, for each instance I of the problem, has a number of accepting computations equal to the number of distinct solutions for instance I [Valiant 1979].

  A problem is #P-**complete** if it is in #P and it is #P-**hard**, i.e., any problem in #P can be reduced by Turing to it (or a #P-complete problem can be reduced to it) in polynomial time. Obviously, $\#P = P \Longrightarrow NP = P$.

  Examples of #P-complete problems: Computing permanent of a matrix, the number of perfect matchings of a bipartite graph, the number of satisfying assignments of a CNF, the number of vertex covers of a graph, etc.

Approximating the number of concepts

A **randomized approximation scheme** for a counting problem $f : \Sigma^* \to \mathbb{N}$ (e.g., the number of formal concepts of a context) is a randomized algorithm that takes as input an instance $x \in \Sigma^*$ (e.g. a formal context $\mathbb{K} = (G, M, I)$) and error tolerance $\varepsilon > 0$, and outputs a number $N \in \mathbb{N}$ such that, for every input instance x,

$$Pr[(1-\varepsilon)f(x) \leq N \leq (1+\varepsilon)f(x)] \geq \frac{3}{4}$$

If the time of randomized approximation scheme is polynomial in $|x|$ and ε^{-1}, then this algorithm is called **fully polynomial randomized approximation scheme**, or FPRAS.

In [M. Boley et al. SDM'2010] it was proved that if FPRAS for computing the number of all concepts exists, then P=NP.

Decision problems about concepts

INSTANCE A context (G, M, I), a constant $k \leq max\{|G|, |M|\}$
QUESTION Is there a concept (A, B) with extent/intent smaller/equal/larger than k?

	$\leq$	$=$	$\geq$
$\lvert A\rvert$	P	NP	P
$\lvert B\rvert$	P	NP	P
$\lvert A\rvert + \lvert B\rvert$	NP	NP	P

NP: problem is NP-complete (reduction from 3-Matching [Kuznetsov 1989, 2001])
P: there is a polynomial algorithm for the problem

The size of "maximal concept"

INSTANCE A context (G, M, I), a constant $k \leq max\{|G|, |M|\}$
QUESTION Is there a concept (A, B) with $|A| + |B| \geq k$?
Consider the context $\overline{K} = (G, M, \overline{I})$ with $\overline{I} = G \times M \backslash I$, and the corresponding bipartite graph $\overline{\Gamma}_B$.

- A concept (A, B) of $\mathfrak{B}(K)$ corresponds to an inclusion-maximal independent set of vertices of $\overline{\Gamma}_B$.
- The number of vertices in the largest independent set is $|V(\overline{\Gamma}_B)| - |M|$, where $|V(\overline{\Gamma}_B)|$ is the number of vertices in $\overline{\Gamma}_B$ and $|M|$ is the number of edges in the maximal matching of $\overline{\Gamma}_B$ [König, 1930].
- The size of a maximal matching can is found by a polynomial-time algorithm [Karp, Hopcroft 1973].

.

Algorithms for computing concepts: A historical survey

Algorithmic Problems of FCA

Sergei O. Kuznetsov

FCA

Computing Concept Lattices

Intractability Results

Algorithms and Their Complexity

Computing Implication Bases

Intractability of pseudo-intents recognition

Corollaries

Conclusion

1. First naive algorithms:
   - Y. Malgrange, 1962: Recherche des sous-matrices premieres d'une matrice a coefficients binaire
   - M. Chein, 1969: Algorithme de recherche des sous-matrices premieres d'une matrice.
   - G. Fay, 1975: Algorithm for finite Galois connections
2. Algorithms with efficient testing of concept duplication
   - E. Norris, 1978
   - B. Ganter, 1984
   - J.-P. Bordat, 1986 (also constructed the covering relation)
   - S. Kuznetsov, 1993 (Close-by-One)
   - R. Godin, P. Valtchev, 1995
   - L. Nourine, O. Raynaud, 1999
   - D. van der Merwe, D. Kourie, S. Obiedkov, 2004 (Addintent)
   - J. Outrata, V. Vychodil, 2010 (Fast CbO)

Common feature of the latter: they use some efficient methods for testing whether a concept has been already generated

Classification of algorithms. 1

- Top-down vs. bottom-up
- Depth-first vs. breadth-first (for algorithms using graph structure)
- Incremental vs. batch

  Incremental algorithms: Norris 1978, Dowling 1993, Godin et al. 1995, Valtchev et al. 2002
  Batch algorithms: Chein 1969, Ganter 1984, Bordat 1986, Zabezhailo et al. 1987, Kuznetsov 1993,

- Usage vs. nonusage of the diagram graph for computing concepts.

  For example, the algorithms of Carpineto and Romano (1996) and that of Valtchev et al. (2001) use the diagram graph to compute concepts and those of Norris (1978) and Ganter (1984) do not use it.

Classification of algorithms. 2

- The use of lexicographical order on concept generations (attributes) (Norris 1978, Ganter 1984, Kuznetsov 1993)

  This assumes that there is a linear order on the set of objects (attributes). A generation of a concept is considered canonical if at each step i of applying a closure operator there are no objects in $(A \cup \{g_i\})'' \setminus A$ that have smaller number than i.
- Some algorithms divide the set of all concepts into disjoint sets, which allows narrowing down the search space. For example, the algorithm of M. Chein stores concepts in layers, each layer corresponding to some step of the algorithm.

Classification of algorithms. 3

- Computing intersections:
  - intersecting a generated intent with some object intent
  - intersecting already obtained intents (Chein 1969, Zabezhailo et al. 1987)
  - intersecting all object intents of the corresponding extent (Ganter 1984)
  - not using intersections at all (Stumme et al. 2000) (attributes are added to already generated intents and some condition on the number of objects with a chosen set of attributes is checked)
- The use of hash functions (algorithm of Godin et al. 1995), which makes it possible to distribute concepts among "buckets" and to reduce the test for uniqueness.

Data structures

Two major steps of algorithms:

- generating a concept candidate
- "testing uniqueness" (to avoid repetitive generation of the same concept)

Tests of this kind require efficient data structures, such as

- Spanning trees (Bordat 1986, Ganter and Reuter 1991),
- Ordered lists (e.g., Ganter 1984),
- CbO trees (consistent with order on concepts, but not with the covering relation, Kuznetsov 1993),
- Tries (Nourine and Raynaud 1999)

Possible alternatives:

- 2-3 balanced trees (Aho et al. 1983)

Delay of batch algorithms

Definition

An algorithm for listing a family of combinatorial structures is said to have ***delay*** *d [Johnson et al. 1988] if it executes at most d many computation steps before either outputting each next structure or halting.*

Definition

Polynomial delay *– d is bounded from above by a polynomial from input size.*

Many batch algorithms have polynomial delay, among them, e.g., those of

- B. Ganter 1984
- J.-P. Bordat 1986
- S. Kuznetsov 1993,
- D. van der Merwe et al. 2004
- J. Outrata and V. Vychodil, 2010

and others with polynomial delays $O(|G|^2 \cdot |M|)$ and $O(|G|^3 \cdot |M|)$.

All these algorithms have $O(|G|^2 \cdot |M| \cdot |L|)$ total runtime complexity.

Cumulative delay of incremental algorithms

An algorithm is said to have a **cumulative delay** d [Johnson et al. 1988, Goldberg 1993] if at any point of time in any execution of the algorithm with any input i the total number of computation steps that have been executed is at most $d(i)$ plus $K \cdot d(i)$, where K is the number of structures that have been output so far. If $d(i)$ can be bounded by a polynomial of i, the algorithm is said to have a **polynomial cumulative delay**.

- Most of the well-known incremental algorithms have polynomial cumulative delays, such as those of Dowling (1993), Carpineto and Romano (1996), The algorithm of Godin et al. (1995) does not have a polynomial cumulative delay, however is very fast for real sparse contexts.
- The algorithm with the best known theoretical worst-case complexity bound is that of L. Nourine and O. Raynaud (1999): $O((|G| + |M|) \cdot |G| \cdot |L|)$ total runtime. This algorithm, which is a realization of the idea of E.Norris that uses trie (suffix tree), still waits for efficient practical implementation.

(Difficulties of) experimental comparison

1. Algorithms, as described by their authors, often allow for different interpretation of crucial details, such as the test of uniqueness of a generated concept.
2. Authors seldom describe exactly data structures and their realizations.
3. The choice of a programming language and platform strongly affects the performance of an algorithm.
4. The best approach to comparison seems the one based on author implementations run on one computer, using standard datasets and randomly generated datasets.

Duquenne-Guigues implication base

Implications satisfy **Armstrong rules**:

$$\frac{}{A \to A}\ ,\quad \frac{A \to B}{A \cup C \to B}\ ,\quad \frac{A \to B, B \cup C \to D}{A \cup C \to D}.$$

Duquenne-Guigues base (also called **canonical basis** and **stembase**: A cardinality-minimal subset of implications, from which all other implications can be deduced by means of Armstrong rules; characterized in terms of **noeuds minimaux de non-redondance (NMNR)** in [J.-L. Guigues, V. Duquenne 1984].

NMNR can be equivalently described as pseudo-intents [Ganter 1984]: a set $P \subseteq M$ is a **pseudo-intent** if $P \neq P''$ and $Q'' \subset P$ for every pseudo-intent $Q \subset P$.

Computing implications: First observations

- The decision problem about the existence of a nontrivial implication (whose premise does not contain the conclusion) is solved in polynomial time.
- Existing algorithms for generating pseudointents, e.g., [Ganter 1984] can terminate in exponential time without outputting any pseudointent.

Problem: What is the largest size of the stem base (wrt. the context size) and the computational complexity of its generation?

D.-G. base of exponential size: the context $K_{\exp,n}$

[S.Kuznetsov, 2004]

$G \setminus M$	m_0	$m_1 \ldots m_n$	$m_{n+1} \ldots m_{2n}$
g_1			
$\vdots$		$\neq$	$\neq$
g_n			
g_{n+1}	$\times$		
$\vdots$	$\vdots$		
$\vdots$	$\vdots$	$\neq$	
$\vdots$	$\vdots$		
g_{3n}	$\times$		

The set $\{m_1, \ldots, m_n\}$ is a pseudo-intent. Replacing m_i with m_{n+i} independently for each i, one obtains all 2^n pseudo-intents.

An example: $K_{\exp,3}$

Algorithmic Problems of FCA

Sergei O. Kuznetsov

FCA

Computing Concept Lattices

Intractability Results

Algorithms and Their Complexity

Computing Implication Bases

Intractability of pseudo-intents recognition

Corollaries

Conclusion

$G \setminus M$	m_0	m_1	m_2	m_3	m_4	m_5	m_6
g_1			×	×		×	×
g_2		×		×	×		×
g_3		×	×		×	×	
g_4	×		×	×	×	×	×
g_5	×	×		×	×	×	×
g_6	×	×	×		×	×	×
g_7	×	×	×	×		×	×
g_8	×	×	×	×	×		×
g_9	×	×	×	×	×	×	

Here, we have $2^3 = 8$ pseudo-intents: $\{m_1, m_2, m_3\}$, $\{m_1, m_2, m_6\}$, $\{m_1, m_5, m_3\}$, $\{m_1, m_5, m_6\}$, $\{m_4, m_2, m_3\}$, $\{m_4, m_2, m_6\}$, $\{m_4, m_5, m_3\}$, $\{m_4, m_5, m_6\}$.

#P-hardness of counting pseudo-intents

Proposition The following problem is $\#P$-hard. [S.Kuznetsov, 2004]
INPUT A formal context $K = (G, M, I)$
OUTPUT The number of pseudo-intents of K

Proof: by reduction from the problem of counting all (inclusion) minimal covers proved to be $\#P$-complete in

L. G. Valiant, The Complexity of Enumeration and Reliability Problems, SIAM J. Comput. 8, 3 (1979), 410–421.

For a graph (V, E) a subset $W \subseteq V$ is a **vertex cover** if every edge $e \in E$ is incident to some $w \in W$.

The reduction context

$\bar{\mathcal{I}}$ is the complement of the edge-vertex graph incidence matrix

$G \setminus M$	m_0	$m_1, \ldots\ldots\ldots, m_{\vert V\vert}$
g_1		
$\vdots$		
$\vdots$		$\bar{\mathcal{I}}$
$\vdots$		
$g_{\vert E\vert}$		
$g_{\vert E\vert+1}$	$\times$	
$\vdots$	$\vdots$	
$\vdots$	$\vdots$	$\neq$
$\vdots$	$\vdots$	
$g_{\vert E\vert+\vert V\vert}$	$\times$	

Quasi-closed

Let M be a set and let $'' : \mathcal{P}(M) \to \mathcal{P}(M)$ be a closure operator.
Definition [Ganter 1984] A set $Q \subseteq M$ is *quasi-closed* if for any $R \subseteq Q$ one has $R'' \subset Q$ or $R'' = Q''$.

For example, closed sets are quasi-closed. A quasi-closed set $Q \subseteq M$ of a context $\mathbb{K} = (G, M, I)$ $''$ is called *quasi-intent*.

Proposition. [Ganter 1984] A set $Q \subseteq M$ is a quasi-intent iff $Q \cap C$ is closed for every closed set C with $Q \not\subseteq C$.

Equivalently, $Q \subseteq M$ is a quasi-intent iff for every $R \subseteq M$, the set $Q \cap R''$ is closed or $Q \cap R'' = Q$.

Proposition. [Ganter 1984] Quasi-intents of a context make a closure system. Intersection of quasi-intents is quasi-intent.

Quasi-intents and pseudo-intents

Actually, we can weaken the condition over all subsets of the set Q to a condition over all quasi-closed subsets of Q.

Proposition. Let $K = (G, M, I)$ be a context and $Q \subseteq M$. Then the following two statements are equivalent:
1. Q is quasi-intent;
2. For any quasi-intent $R \subseteq Q$ one has $R'' \subset Q$ or $R'' = Q''$.

Proposition. [Ganter 1984] A quasi-intent P is *pseudo-intent* if $P'' \neq P$ and for any quasi-intent $Q \subset P$ one has $Q'' \subset P$.

So, a pseudo-intent is a minimal quasi-intent in its closure class, i.e., among quasi-intents with the same closure.

Testing quasi-closedness in polynomial time

From

Proposition. [Ganter 1984] A set $Q \subseteq M$ is a quasi-intent iff $Q \cap C$ is closed for every closed set C with $Q \not\subseteq C$.

we have

Proposition. The set S is a quasi-intent iff for any object $g \in G$ either $S \cap g'$ is closed or $S \cap g' = S$.

Corollary. Testing whether $S \subseteq M$ is quasi-intent in the context (G, M, I) may be performed in $O(|G|^2 \cdot |M|)$ time.

Recognizing pseudo-intents is in coNP

Algorithmic Problems of FCA
Sergei O. Kuznetsov
FCA
Computing Concept Lattices
Intractability Results
Algorithms and Their Complexity
Computing Implication Bases
Intractability of pseudo-intents recognition
Corollaries
Conclusion

[S.Kuznetsov, S.Obiedkov 2006, 2008]
Proposition. The problem
INSTANCE Given a context $K = (G, M, I)$ and a set $S \subseteq M$.
QUESTION Is S a nonpseudo-intent of $K = (G, M, I)$?
is in NP.

Corollary. The problem
INSTANCE Given a context $K = (G, M, I)$ and a set $S \subseteq M$.
QUESTION Is S a pseudo-intent of $K = (G, M, I)$?
is in coNP.

Related decision problems

Note that the decision problem

INSTANCE A context $K = (G, M, I)$, a natural number $k \leq |M|$.
QUESTION Is there a pseudo-intent of the context K of size not greater than k?

is NP-hard by the same reduction from the vertex covering problem.

At the same time the problem

INSTANCE A context $K = (G, M, I)$
QUESTION Is there a pseudo-intent of the context K?

is solvable in polynomial time: Test whether the reduced context of K, after respective permutations of objects and attributes, coincides with $(A, A, \neq)$.

Pseudo-intents recognition

Problem: Pseudo-intent recognition (PI)
INPUT: A context $\mathbb{K} = (G, M, I)$ and a set $P \subseteq M$.
QUESTION: Is P a pseudo-intent of $\mathbb{K}$?

PI belongs to coNP [S.Kuznetsov and S.Obiedkov, Discrete Applied Mathematics (2008)]

PI is coNP-hard [M.Babin and S. Kuznetsov, CLA 2010].

Draft of the reduction

We reduce CNF satisfiability to PI: A given CNF instance is satisfiable iff the corresponding P is not a pseudo-intent of the corresponding context $\mathbb{K}$

Draft of the reduction

We reduce CNF satisfiability to PI: A given CNF instance is satisfiable iff the corresponding P is not a pseudo-intent of the corresponding context $\mathbb{K}$

For any $A \subseteq \{x_1, \neg x_1, \ldots, x_n, \neg x_n\}$ that satisfies $A \cap \{x_i, \neg x_i\} \neq \emptyset$ for $1 \leq i \leq n$, we define truth assignment ϕ_A:

$$\phi_A(x_i) = \begin{cases} true, & \text{if } x_i \notin A \text{ and } \neg x_i \in A; \\ false, & \text{if } \neg x_i \notin A \text{ and } x_i \in A; \\ false, & \text{otherwise } (x_i \in A \text{ and } \neg x_i \in A); \end{cases}$$

Draft of the reduction

$$P = M \setminus \{e\}$$

$$x \mathcal{C} C_i \text{ iff } x \notin C$$

	p	$C_1 \ C_2 \ \cdots \ C_k$	$x_1 \ \neg x_1 \ \cdots \ x_n \ \neg x_n$	e
g_{x_1}				
$g_{\neg x_1}$				
$\vdots$		$\mathcal{C}$	$\mathcal{I}_{\neq}$	
g_{x_n}				
$g_{\neg x_n}$				
g_{CX}		$\times \cdots\cdots \times$	$\times \cdots\cdots\cdots\cdots \times$	
g_C	$\times$	$\times \cdots\cdots \times$		
g_{l_1}	$\times$		L_1	
$g_{l_1}^{x_1}$	$\times$		$L_1^{x_1}$	
$g_{l_1}^{\neg x_1}$	$\times$		$L_1^{\neg x_1}$	
$\vdots$	$\vdots$		$\vdots$	
$g_{l_n}^{\neg x_n}$	$\times$		$L_n^{\neg x_n}$	

Draft of the reduction

Proposition

A set $Q \subseteq \{x_1, \ldots, \neg x_n\}$ such that ϕ_Q is defined is closed iff ϕ_Q satisfies CNF instance.

- A truth assignment ϕ_Q satisfies CNF $(Q \subseteq \{x_1, \ldots, \neg x_n\})$ iff $\{p\} \cup Q$ is a pseudo-intent and $(\{p\} \cup Q)'' = M$.
- There is no other pseudo-intents $R \subset P$ and $R'' \not\subseteq P$.

Example

Consider an example of our reduction with the following CNF:

$$(x \vee y) \wedge (\neg x)$$

Example

Consider an example of our reduction with the following CNF:

$$(x \vee y) \wedge (\neg x)$$

First add auxiliary clauses $(x \vee \neg x)$ and $(y \vee \neg y)$. Thus we transform our CNF to

$$(x \vee y) \wedge (\neg x) \wedge (x \vee \neg x) \wedge (y \vee \neg y)$$

This transformation does not affect satisfiability of the initial CNF.

Example

Consider an example of our reduction with the following CNF:

$$(x \vee y) \wedge (\neg x)$$

First add auxiliary clauses $(x \vee \neg x)$ and $(y \vee \neg y)$. Thus we transform our CNF to

$$(x \vee y) \wedge (\neg x) \wedge (x \vee \neg x) \wedge (y \vee \neg y)$$

This transformation does not affect satisfiability of the initial CNF.

Now construct the corresponding context $\mathbb{K}$.

Example

CNF: $(x \vee y) \wedge (\neg x) \wedge (x \vee \neg x) \wedge (y \vee \neg y)$

	p	$x \vee y$	$\neg x$	$x \vee \neg x$	$y \vee \neg y$	x	$\neg x$	y	$\neg y$	e
g_x			×		×		×	×	×	
$g_{\neg x}$		×			×	×		×	×	
g_y			×	×		×	×		×	
$g_{\neg y}$		×	×	×		×	×	×		
g_{CX}		×	×	×	×	×	×	×	×	
g_C	×	×	×	×	×					
g_x	×							×	×	
g_x^y	×								×	
$g_x^{\neg y}$	×							×		
g_y	×					×	×			
g_y^x	×						×			
$g_y^{\neg x}$	×					×				

$$Q = \{p\} \cup \{\neg x, y\},\ \phi_Q = (x = false,\ y = true)$$

Recognizing essential intents

Definition

An intent B is essential intent if there is a pseudo-intent P such that $P'' = B$.

Recognizing essential intents

Definition

An intent B is essential intent if there is a pseudo-intent P such that $P'' = B$.

Problem: Essential intents recognition(EI)
INPUT: A context $\mathbb{K} = (G, M, I)$ and a set $B \subseteq M$.
QUESTION: Is B an essential intent of $\mathbb{K}$?

Recognizing essential intents

Definition

An intent B is essential intent if there is a pseudo-intent P such that $P'' = B$.

Problem: Essential intents recognition(EI)
INPUT: A context $\mathbb{K} = (G, M, I)$ and a set $B \subseteq M$.
QUESTION: Is B an essential intent of $\mathbb{K}$?

Proposition

EI is NP-complete.

Lectically largest pseudo intent and enumeration of pseudo-intents in reversed lectic order

$M = \{m_1 <, \ldots, < m_n\}$ is a finite set with linear order on it.
$A \subseteq M$ *lectically smaller* than $B \subseteq M$ ($A < B$, B is lectically larger than A) if $\exists m_i \in B \setminus A : A \cap \{m_j \in M \mid j < i\} = B \cap \{m_j \in M \mid j < i\}$.

Lectically largest pseudo intent and enumeration of pseudo-intents in reversed lectic order

FCA

Computing Concept Lattices

Intractability Results

Algorithms and Their Complexity

Computing Implication Bases

Intractability of pseudo-intents recognition

Corollaries

Conclusion

$M = \{m_1 <, \ldots, < m_n\}$ is a finite set with linear order on it.
$A \subseteq M$ *lectically smaller* than $B \subseteq M$ ($A < B$, B is lectically larger than A) if $\exists m_i \in B \setminus A : A \cap \{m_j \in M \mid j < i\} = B \cap \{m_j \in M \mid j < i\}$.

Problem: The lectically largest pseudo-intent (LLPI)
INPUT: A context $\mathbb{K} = (G, M, I)$ with linear order on M and a set $P \subseteq M$.
QUESTION: Is P the lectically largest pseudo-intent of $\mathbb{K}$?

Lectically largest pseudo intent and enumeration of pseudo-intents in reversed lectic order

$M = \{m_1 <, \ldots, < m_n\}$ is a finite set with linear order on it.
$A \subseteq M$ *lectically smaller* than $B \subseteq M$ ($A < B$, B is lectically larger than A) if $\exists m_i \in B \setminus A : A \cap \{m_j \in M \mid j < i\} = B \cap \{m_j \in M \mid j < i\}$.

Problem: The lectically largest pseudo-intent (LLPI)
INPUT: A context $\mathbb{K} = (G, M, I)$ with linear order on M and a set $P \subseteq M$.
QUESTION: Is P the lectically largest pseudo-intent of $\mathbb{K}$?

Proposition

LLPI is coNP-hard [M.Babin, S.Kuznetsov, 2010]

Lectically largest pseudo intent and enumeration of pseudo-intents in reversed lectic order

FCA

Computing Concept Lattices

Intractability Results

Algorithms and Their Complexity

Computing Implication Bases

Intractability of pseudo-intents recognition

Corollaries

Conclusion

$M = \{m_1 <, \ldots, < m_n\}$ is a finite set with linear order on it.
$A \subseteq M$ *lectically smaller* than $B \subseteq M$ ($A < B$, B is lectically larger than A) if $\exists m_i \in B \setminus A : A \cap \{m_j \in M \mid j < i\} = B \cap \{m_j \in M \mid j < i\}$.

Problem: The lectically largest pseudo-intent (LLPI)
INPUT: A context $\mathbb{K} = (G, M, I)$ with linear order on M and a set $P \subseteq M$.
QUESTION: Is P the lectically largest pseudo-intent of $\mathbb{K}$?

Proposition

LLPI is coNP-hard [M.Babin, S.Kuznetsov, 2010]

Proposition

It is impossible to enumerate pseudo-intents in reverse lectic order with polynomial delay unless P=NP [M.Babin, S.Kuznetsov, 2010]

Another intractability result [F.Distel, B.Sertkaya, 2009]:

Proposition

Enumerating pseudo-intents in arbitrary order is TransEnum-hard

Alternatives to Duquenne-Guigues base: Proper premises

In [B.Ganter, R.Wille 1999] **proper premise** was defined as a subset of attributes $B \subseteq M$ such that

$$B'' \setminus (B \cup \bigcup_{S \subset B} S'') \neq \emptyset.$$

The set $\{B \to B'' \mid B \text{ is a proper premise } \}$
is a complete and irredundant subset of implications.

Computing proper premises

- A proper premise can be recognized in polynomial time
- Computing all proper premises is reduced to enumerating minimal hypergraph transversals (TransEnum), for which a quasipolynomial algorithm is known [L.Khachiyan et al. 1996]
- An algorithm using TransEnum [F.Distel et al. CLA'2011] performs in practice much (sometimes orders of magnitudes) faster than algorithms computing pseudo-intents

Conclusion

- Concepts can be generated with polynomial delay. The problem of recognizing a concept is solved in polynomial time Computing the number of concepts is #P-complete.
- Pseudo-intents cannot be generated with polynomial delay in lectic order or reverse lectic order unless P=NP The problem of recognizing a pseudo-intent is coNP-complete. Computing the number of pseudo-intents is #P-hard.
- Other implication bases, which may have less practical complexity, should be studied: those based on proper premises, minimal generators, quasi-intents, etc.

Algorithmic Problems of FCA

Sergei O. Kuznetsov

FCA

Computing Concept Lattices

Intractability Results

Algorithms and Their Complexity

Computing Implication Bases

Intractability of pseudo-intents recognition

Corollaries

Conclusion

Conclusion

Thank you!