Workshop Notes

9th International Workshop

"What can FCA do for Artificial Intelligence?"

FCA4AI 2021

30th International Joint Conference on Artificial Intelligence

IJCAI 2021

August 21 2021

Montréal, Québec, Canada

Editors

Sergei O. Kuznetsov (NRU HSE Moscow)

Amedeo Napoli (LORIA Nancy)

Sebastian Rudolph (TU Dresden)

http://fca4ai.hse.ru/2021/

IJCAI 2021 MONTREAL

The eight editions of the FCA4AI Workshop showed that many researchers working in Artificial Intelligence are deeply interested by a well-founded method for classification and data mining such as Formal Concept Analysis (see `https://conceptanalysis.wordpress.com/fca/`).

FCA4AI started with ECAI 2012 (Montpellier) and the last edition was co-located with ECAI 2020 (Santiago de Compostela, virtual conference). The FCA4AI workshop has now a quite long history and all the proceedings are available as CEUR proceedings (see `http://ceur-ws.org/`, volumes 939, 1058, 1257, 1430, 1703, 2149, 2529, and 2729). This year, the workshop has again attracted researchers from many different countries working on actual and important topics related to FCA, showing the diversity and the richness of the relations between FCA and AI.

Formal Concept Analysis (FCA) is a mathematically well-founded theory aimed at data analysis and classification. FCA allows one to build a concept lattice and a system of dependencies (implications and association rules) which can be used for many AI needs, e.g. knowledge discovery, machine learning, knowledge representation, reasoning, ontology engineering, as well as information retrieval and text processing. Recent years have been witnessing increased scientific activity around FCA, in particular a strand of work emerged that is aimed at extending the possibilities of FCA w.r.t. knowledge processing. These extensions are aimed at allowing FCA to deal with more complex data, both from the data analysis and knowledge discovery points of view. Actually these investigations provide new possibilities for AI practitioners within the framework of FCA. Accordingly, we are interested in the following issues:

- How can FCA support AI activities such as knowledge processing, i.e. knowledge discovery, knowledge representation and reasoning, machine learning (clustering, pattern and data mining), natural language processing, information retrieval...
- How can FCA be extended in order to help AI researchers to solve new and complex problems in their domains, in particular how to combine FCA with neural classifiers for improving interpretability of the output and producing valuable explanations...

The workshop is dedicated to discussion of such issues. First of all we would like to thank all the authors for their contributions and all the PC members for their reviews and precious collaboration. This year, 24 papers were submitted and 14 were accepted for presentation at the workshop, out of which 6 short papers. The papers submitted to the workshop were carefully peer-reviewed by three members of the program committee. Finally, the order of the papers in the proceedings (see page 5) follows the program order (see `http://fca4ai.hse.ru/2021/`).

The Workshop Chairs

Sergei O. Kuznetsov
National Research University Higher School of Economics, Moscow, Russia

Amedeo Napoli
Université de Lorraine, CNRS, Inria, LORIA, 54000 Nancy, France

Sebastian Rudolph
Technische Universität Dresden, Germany

Program Committee

Mehwish Alam (AIFB Institute, FIZ KIT Karlsruhe, Germany)

Jaume Baixeries (UPC Barcelona, Catalunya)

Karell Bertet (L3I, Université de La Rochelle, France)

Aleksey Buzmakov (National Research University HSE Perm, Russia)

MiguelCouceiro (LORIA, Nancy France)

Diana Cristea (Babes-Bolyai University, Cluj-Napoca, Romania)

Mathieu D'Aquin (National University of Ireland Galway, Ireland)

Florent Domenach (Akita International University, Japan)

Elizaveta Goncharova (NRU Higher School of Economics, Moscow, Russia)

Tom Hanika (University of Kassel, Germany)

Marianne Huchard (LIRMM/Université de Montpellier, France)

Dmitry I. Ignatov (National Research University HSE Moscow, Russia)

Dmitry Ilvovsky (NRU Higher School of Economics, Moscow, Russia)

Mehdi Kaytoue (Infologic, Lyon, France)

Jan Konecny (Palacky University, Olomouc, Czech Republic)

Francesco Kriegel (Technische Universität Dresden, Germany)

Leonard Kwuida (Bern University of Applied Sciences, Switzerland)

Florence Le Ber (ENGEES/Université de Strasbourg, France)

Tatiana Makhalova (National Research University HSE Moscow, Russia, and Inria LORIA, Nancy, France)

Nizar Messai (Université François Rabelais Tours, France)

Rokia Missaoui (UQO Ottawa, Canada)

Sergei A. Obiedkov (NRU Higher School of Economics, Moscow, Russia)

Uta Priss (Ostfalia University, Wolfenbüttel, Germany)

Christian Sacarea (Babes-Bolyai University, Cluj-Napoca, Romania)

Henry Soldano (Laboratoire d'Informatique de Paris Nord, Paris, France)

Francisco José Valverde Albacete (Universidad Carlos III de Madrid, Spain)

Renato Vimieiro (Universidade Federal de Minas Gerais, Belo Horizonte, Brazil)

Contents

1 *Modelling Conceptual Schemata with Formal Concept Analysis*
Uta Priss 7

2 *Data Overview by means of delta-classes of equivalence. The case of the Titanic dataset*
Aleksey Buzmakov, Sergei O. Kuznetsov, Tatiana Makhalova, and Amedeo Napoli . . 19

3 *FCA Went (Multi-)Relational, But Does It Make Any Difference?*
Mickaël Wajnberg, Petko Valtchev, Mario Lezoche, Alexandre Blondin-Massé, and Hervé Panetto 27

4 *Likely-occurring itemsets for pattern mining*
Tatiana Makhalova, Sergei O. Kuznetsov, and Amedeo Napoli 39

5 *Concept-based Chatbot for Interactive Query Refinement in Product Search*
Elizaveta Goncharova, Dmitry Ilvovsky, and Boris Galitsky 51

6 *Variability Extraction from Simulator I/O Data Schemata in Agriculture Decision-Support Software*
Thomas Georges, Marianne Huchard, Mélanie König, Clémentine Nebut, and Chouki Tibermacine 59

7 *Multimodal Clustering with Evolutionary Algorithms*
Mikhail Bogatyrev, Dmitry Orlov and Tatiana Shestaka 71

8 *On Suboptimality of GreConD for Boolean Matrix Factorisation of Contranominal Scales*
Dmitry Ignatov and Alexandra Yakovleva 87

9 *Summation of Decision Trees*
Egor Dudyrev and Sergei O. Kuznetsov 99

10 *Ensemble Techniques for Lazy Classication Based on Pattern Structures*
Ilya Semenkov and Sergei Kuznetsov 105

11 *A concept of self-supervised logical rule inference in symbolic classifications*
Xenia Naidenova and Vladimir Parkhomenko 113

12 *Non-Redundant Link Keys in RDF Data: Preliminary Steps*
Nacira Abbas, Alexandre Bazin, Jérôme David, and Amedeo Napoli 125

13 *Formal Concept Analysis for Semantic Compression of Knowledge Graph Versions*
Damien Graux, Diego Collarana, and Fabrizio Orlandi 131

Modelling Conceptual Schemata with Formal Concept Analysis

Uta Priss

Ostfalia University, Wolfenbüttel, Germany

Abstract. This paper discusses how to construct conceptual schemata (which are meant to provide conceptual information in a manner close to natural language, easy to memorise and mentally parse) from concept lattices which tend to present a more computational view of conceptual information. Different methods for constructing schemata from concept lattices (such as OR-definitions for reducing the number of attributes and implications of a concept lattice) are considered.

1 Introduction

As the name states, Formal Concept Analysis (FCA) provides a means for analysing concepts. While there are many applications for FCA, it is often easier to employ FCA for *computational* problems than to actually analyse concepts in a manner similar to how *natural language* is processed by humans. For the purpose of structuring and developing teaching materials it would be desirable if FCA could serve as a means for representing concepts in a manner that is close to how students learn domain knowledge. A representation would be desirable that is similar to relations amongst natural language words, for example hyponyms such as "poodle" and "dog". Lattices that are automatically generated from natural language databases, however, such as WordNet or Roget's Thesaurus tend to be more computational because their relations are not sufficiently precisely defined (Priss & Old 2010).

In this paper, we are introducing an approach for shifting between word-based natural language representations and more formal representations with FCA. For that purpose we are distinguishing *(conceptual) schemata* which utilise natural language words from *(conceptual) classes* which contain a set of formal contexts. Investigating the connections between schemata and classes is a *semiotic* task because it considers a relationship between words as representations of signs and concepts as meanings of signs. It is of interest to determine how well the information of a class is retained in a schema, how efficiently it is represented and how well a schema covers a class. The semiotic perspective and the relationship to educational research have been discussed elsewhere (Priss 2021a and 2021b) and are not further elaborated in this paper.

The notion of "schema" is influenced by Lakoff's (1987) "image schema". But the focus of schemata in this paper is on verbal description, not on images. The words or phrases that are defined by a schema and relate one schema to other schemata are called *head representamens* in this paper in analogy to the headwords of dictionary entries which are also called catchwords, keywords, subject headings, index terms or

descriptors in other disciplines. From a semiotic view, head representamens are representamens of signs (Priss 2017). From a computational view, head representamens are just strings that are elements of a set. Head representamens are to be distinguished from other representamens which have an auxiliary function.

Head representamens can serve as building blocks for constructing compound representamens using the operations AND, OR and NOT. For example, head representamens for poodles might be "poodle" and "miniature poodle" whereas a compound representamen might be "poodle AND cute". Such operations are syntactically defined in schemata and semantically defined in classes with *interpretations* mapping schemata into classes[1]. The operations AND and OR for head representamens are similar but not identical to natural language "and" and "or" because, in natural language, "and" is sometimes used for an intersection (such as "dog and cute"), sometimes for a union (such as "dogs and cats") and "or" can be exclusive or inclusive. An interpretation should map an AND-operation amongst head representamens into a meet of concepts in a class, an OR-operation into a join of concepts or a construction involving a union of extensions and a NOT-operation into an extensional set difference.

The basics of FCA can be found in the textbook by Ganter & Wille (1999) and are not repeated in this paper. But it should be mentioned that a concept (a', a'') is called an *attribute concept of* a and a concept (o'', o') an *object concept*. The ordering amongst object concepts is called *object order*. Concepts that are not object concepts are called *supplemental concepts* in this paper. The extension of a supplemental concept equals the union of the extensions of its proper subconcepts. In this paper supplemental concepts are drawn as empty nodes in the Hasse diagrams. Each supplemental concept corresponds to a clause because for such a concept c with extension $ext(c)$ and intension $int(c)$ and the condition $\forall o_i \in ext(c) : \exists c_i < c : o_i \in ext(c_i)$ it follows that $\bigwedge(a_i \in int(c)) \Rightarrow \bigvee(a_i \mid \exists c_i : c_i < c, a_i \in int(c_i), a_i \notin int(c))$ is a clause. It is particularly interesting to consider whether some concepts always have to be supplemental with respect to background knowledge even if more objects are added to a context. An example for this feature is provided in the next section.

Some aspects presented in Section 2 which discusses a certain type of reduction of concept lattices have already been covered elsewhere, for example, by Ganter & Obiedkov (2016). Ganter (2019) discusses how to render an implication basis of a formal context more human readable by changing and grouping some of the implications and Lopez-Rodriguez et al. (2021) provide a means for determining core implications from a basis. In this paper, the focus is on reducing implications combined with representing some of the information by other means (as subconcept hierarchies or prototypical examples) if that renders the information more human readable. OR-reductions are also relevant for reducing a concept lattice to its AOC-poset (Osswald & Petersen, 2002) which consists only of the attribute and object concepts and possibly for feature models of Product Line Representations (Carbonnel et al. 2016).

The definitions of conceptual schemata, classes and interpretations in Section 3 are similar to a standard modelling with formal semantics, for example, Prediger's (1998) K-interpretations which map ordered sets of concept and relation names into power

[1] Contrary to standard formal semantics where interpretations map strings into sets, in this paper interpretations map head representamens into concepts.

context families. The aim of Prediger's work and others who extended it was to establish a connection between FCA and Conceptual Graphs and focused on logical properties. The focus of this paper is on the relationship between representamens and concepts in a more closed world setting. Most established FCA exploration and reduction methods tend to focus on reducing the lower parts of a lattice whereas in this paper mainly supplemental concepts in the upper part of a concept lattice are reduced. Thus, this paper draws on existing research but from a somewhat different perspective.

2 OR-Reduction

This section uses an example of a formal context and lattice from Ganter & Wille (1999) consisting of seven prototypical types of triangles and their defining properties (Fig. 1, left). In this example, the supplemental concepts (represented as empty nodes) must always be supplemental because every triangle must have exactly one of the attributes "acute", "obtuse" or "right" and either be equilateral or not or isosceles or not. Thus according to background knowledge about triangles, the object concepts describe actual examples of triangles whereas the extensions of the supplemental concepts must always be unions of the extensions of their subconcepts even if more triangles are added to the context. The lattice displays subconcept relationships for the types of triangles. If a student wants to learn about triangles, their types and their definitions, it would not be efficient to memorise all of the concepts of the lattice on the left side of Fig. 1. because it displays more a computational view than a natural language view.

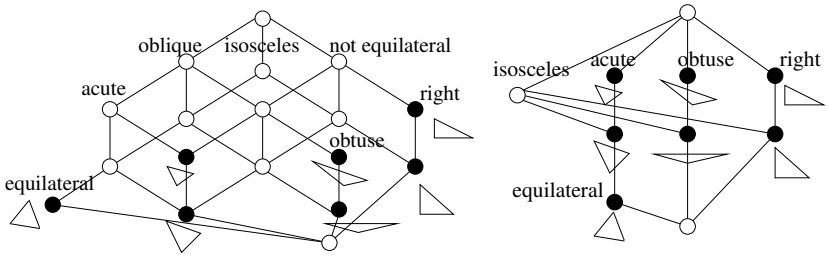


Fig. 1. A lattice of triangles (cf. Ganter & Wille (1999)) and its reduced form

The right side of Fig. 2 shows a reduced version of the lattice on the left. The attributes "oblique" and "not equilateral" have been removed because oblique represents "acute OR obtuse" and "not equilateral" is the negation of "equilateral". Normally in FCA *reducing* means to remove all attributes and objects from a context which are at attribute or object reducible concepts. Another form of reduction is to calculate an AOC-poset which only keeps object and attribute concepts and their ordering (Osswald & Petersen 2002). AOC-posets are compact and can be algorithmically produced (Berry et al. 2014). A lattice can be reconstructed from its AOC-poset if a clause is added for each concept that is neither an attribute nor an object concept. A disadvantage of AOC-posets is that conjunctions of attributes and therefore implications need not correspond

to a single node and cannot easily be read from Hasse diagrams. This disadvantage is avoided by the reduction methods in this paper. Other means for reducing the size of concept lattices discussed in the literature tend to rely on statistical or probabilistic methods which cannot easily be reversed (cf. Priss & Old (2011) for an overview).

In this paper, only reducing attributes is of interest. Reducing attributes in the standard manner (called *AND-reduction* in this paper) changes the labelling of a concept lattice but not its structure. Removing attributes that are OR combinations (called *OR-reduction* in this paper) or NOT combinations (*NOT-reduction*) may change the concept lattice itself and reduce its size as demonstrated in Fig. 1. All of the following definitions focus on attributes and assume that the contexts are finite and clarified (or purified) which means that for any two attributes $a \neq b \Longrightarrow a' \neq b'$.

Definition 1. *An attribute a of a formal context (O, A, J) is called* OR-reducible *if a set $A^\star := \{a_1, ..., a_n\} \subseteq A$ exists with $a' = a'_1 \cup ... \cup a'_n$ and $a_i \in A^\star \iff a'_i \subset a'$ and $\neg\exists b \in A : a'_i \subset b' \subset a'$.*

Definition 2. *For a formal context (O, A, J): For an OR-reducible attribute a, its* OR-definition *is provided by $a := a_1$ OR ... OR a_n for $a_i \in A^\star$. An attribute a with $\exists\{a_1, ..., a_n\} \subseteq A : a' = a'_1 \cap ... \cap a'_n$ is called* (AND-)reducible *with an* AND-definition *provided by $a := a_1$ AND ... AND a_n. An attribute a with $\exists b \in A : a' = O \backslash b'$ is called* NOT-reducible *with its* NOT-definition *provided by $a :=$ NOT b.*

Lemma 1. *If a is OR-reducible, then its set $A^\star := \{a_1, ..., a_n\}$ for its representation as a_1 OR ... OR a_n is uniquely determined. If a is NOT-reducible then its NOT-definition is uniquely determined.*

Proof: For $b \in A$ with $b' \subset a'$: if $b' \setminus \bigcup\{a'_i : a_i \in A^\star, a_i \neq b\} \neq \emptyset$, then $b \in A^\star$. Else $b' \subseteq \bigcup\{a'_i : a_i \in A^\star, a_i \neq b\} = a'$ and either $\exists a_i \in A^\star : b' \subset a'_i$ (thus $b \notin A^\star$) or $\neg\exists a_i \in A^\star : b' \subset a'_i$ (thus $b \in A^\star$). Thus $b \in A^\star$ or $b \notin A^\star$ is uniquely determined. NOT-definitions are unique because the context is clarified.

An attribute b that was removed during clarification can be considered a strong synonym or *SYN-definition* in the form of $a := b$. As mentioned above, AND-reduction corresponds to standard FCA $\wedge$-reduction. AND-definitions are not unique because often several possibilities exist to represent a $\wedge$-reducible concept as a meet of other concepts. OR-reduction focuses on attributes whereas standard FCA $\vee$-reduction focuses on objects. Thus, these two notions are different. An OR-reducible attribute must belong to a $\vee$-reducible concept, but not every $\vee$-reducible concept has an OR-reducible attribute. In fact OR- and NOT-reducible attributes need not exist at all in a lattice. In a similar manner, XOR-definitions could be declared as OR-definitions where the a'_i are pairwise disjoint.

Lemma 2. *i) The AND-definition of an attribute concept (a', a'') is a.*
ii) An object concept (o'', o') cannot be OR-reducible.
iii) A $\vee$-reducible concept that is an attribute concept (a', a'') and not an object concept can always be made OR-reducible by adding further attributes to the formal context.

Proof: i) Trivial. ii) Because o cannot be in the extension of proper subconcepts of (o'', o'). iii) Attributes $a_1, ..., a_n$ can be added so that each lower neighbour of (a', a'')

is an attribute concept (a_i', a_i''). Because the concept is not an object concept, Def. 1 is then fulfilled with $A^\star = \{a_1, ..., a_n\}$.

Thus Lemma 1 only states that an OR-definition is unique with respect to a fixed formal context. Turning each lower neighbour into an attribute concept is always possible but may not be the best strategy. For example in Fig. 1, oblique is definable as "acute OR obtuse" even though only one of its lower neighbours is an attribute concept.

Standard FCA implications only use logical AND. Implications that are formed with combinations of AND and OR are called (cumulated) clauses and are more complicated than standard implications. For example, there is no equivalent to the Duquenne-Guigues basis for clauses (Ganter & Obiedkov 2016). Because the requirements for an OR-definition are more specific than just a logical OR, the implications discussed in the next lemma are not standard FCA clauses. The lemma shows that if attributes are removed from a context as OR-definitions, some of the implications of the original context can be directly reconstructed from the OR-definitions (as background knowledge) and the implications of the reduced context.

Lemma 3. *For implications involving OR-definitions with* $a := a_1$ *OR ... OR* a_n*:*
i) $\forall a_i : a_i \to a$
ii) $a \to x \iff (a_1$ *OR ... OR* $a_n) \to x \iff (a_1 \to x)$ *and ... and* $(a_n \to x)$
iii) $x \to a \iff x \to (a_1$ *OR ... OR* $a_n) \iff (x \to a_1)$ *or ... or* $(x \to a_n)$
iv) $a_1 ... \ a_n \to x \implies a \to x$
v) $\forall a_i : (x \to a_i y \implies x \to ay)$

Proof: i) Because $a_i' \subset a'$. ii) $a_1' \cup ... \cup a_n' \subseteq x' \iff a_1' \subseteq x'$ and ... and $a_n' \subseteq x'$. iii) With $A^\star = \{a_1, ..., a_n\}$ it follows that $x' \subseteq a_1' \cup ... \cup a_n' \iff \exists a_i \in A^\star : x' \subseteq a_i'$ because otherwise $x \in A^\star$. iv) follows from ii) and the Armstrong rule of composition. v) because of transitivity of "$\to$".

Removing an OR-reducible attribute changes a lattice unless the attribute is also AND-reducible. It would be desirable to develop an efficient algorithm for reconstructing the implications of an original non-reduced context from the implications of a reduced context together with the OR-definitions. Lemma 3 contains some rules for such an algorithm, but the list is not complete and it is not clear whether it can be completed. A challenge for such an algorithm is that if several OR-definitions exist, they can mutually affect each other and thus cannot be processed in a linear sequence. It would be even more desirable if such an algorithm were to convert basis implications into basis implications. While all implications of an OR-reduced context are implications of its non-reduced context, applying Lemma 3 to an implication that belongs to a basis does not guarantee that it results in a basis implication of the non-reduced context. If efficiency is not an issue, then the implications of the non-reduced context can always be calculated by adding OR-definitions to a formal context as columns (for attributes) that are unions of other columns. For the purposes of developing schemata as discussed in the next section, it is sufficient to store those implications that cannot be easily reconstructed with Lemma 3 in a separate list in addition to the basis implications.

Presumably reconstructing the implications after NOT-reduction is even more complicated. OR-reduction does not change the object order of a lattice. But adding or deleting NOT-reducible attributes does change the object order as shown in Fig. 1. Therefore

removal of NOT-reducible attributes may not in general be advisable. For the same reason, combining AND, OR and NOT in definitions is not even discussed in this paper. A further reason for removing OR-reducible attributes is because their existence is somewhat arbitrary. In the example in Fig. 1, oblique is OR-definable as "acute OR obtuse". There are no similar attributes for "acute OR right" and "obtuse OR right", but there could be. It is arbitrary which OR-definitions happen to exist as an attribute and which do not. Successive OR-reduction might reduce a concept lattice to its object ordering. Presumably, implications involving object concepts are particularly important whereas all other implications somewhat depend on how upper level concepts are labelled.

3 Conceptual Schemata and Classes

Learning is a complicated task that consists of memorising information but also acquiring skills and modes of thinking. With respect to conceptual knowledge different modes of thinking correspond to structuring content in a variety of manners: some information as concepts, some as implications, clauses or examples and some by techniques for deducing further information from the memorised information. The idea for conceptual schemata is that they present information in a format that is closer to how information would be structured for learning purposes. The role of conceptual classes is then to ensure that the information that is behind a schema is consistent and as complete as possible. The definitions in this section only provide a general framework and will need to be specified with further details for actual applications.

Fig. 2 shows the conceptual schema (on the left) for the concept lattice (on the right) of the example of Fig. 1. In this case the reduced lattice is already quite close to a schema, except that the top and bottom node are not necessary because they can be deduced. Further details about the schema are explained below. Evidence for the adequacy of the diagram on the left of Fig. 2 is provided by the fact that the German Wikipedia page about triangles contains basically the same image[2] for the "hierarchy of triangles". Thus, the Wikipedia authors appear to consider it a suitable summary of knowledge about basic triangles. Students can memorise that diagram together with the definition of "oblique" and the fact that acute, right and obtuse are mutually exclusive. Students can then deduce further implications (such as "obtuse $\rightarrow$ oblique not_equilateral" and "equilateral $\rightarrow$ isosceles acute") from the memorised information.

The following definitions specify the relationship between conceptual schemata and classes more precisely. The definitions are similar to standard definitions of formal semantics except that interpretations result in concepts instead of sets.

Definition 3. *A* (conceptual) class (O, A_L, J, N) *consists of a set* O *of formal objects, a set* A_L *of predicates (or "attributes", formed according to some language* L*), a relation* $J \subseteq O \times A_L$ *with* $oJa \iff (a(o)$ *is true) and a set* N *of formal contexts with* $(O_i, A_i, J_i) \in N$ *for* $O_i \subseteq O, A_i \subseteq A_L, J_i \subseteq J$ *and* $J_i \subseteq O_i \times A_i$*. The set of all concepts that can be derived from any of the contexts is denoted by* $\mathcal{C}(O, A_L, J, N)$*, the set of all true statements that can be derived from any of the contexts by* $\mathcal{T}(O, A_L, J, N)$*.*

[2] https://de.wikipedia.org/wiki/Datei:Hierarchie.Dreiecke.png

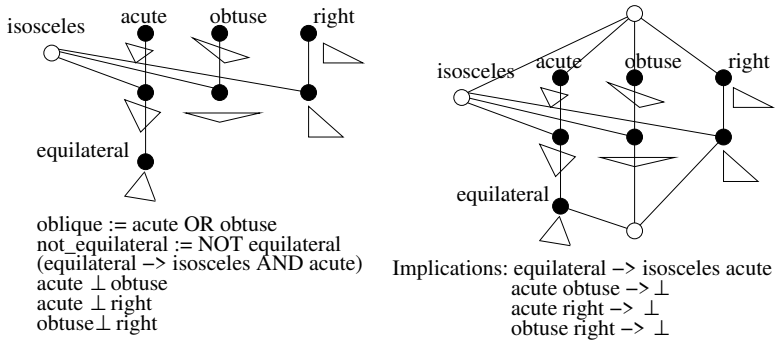


Fig. 2. A conceptual schema (left) and a NOT-OR-reduced concept lattice (right)

A conceptual class is a set of formal contexts which are defined with respect to a common set of objects and attributes. While it would be possible to consider (O, A_L, J) a formal context itself, it may be too big to compute anything useful for it. Therefore concepts and implications are only computed for the contexts in N. It is possible for implications from one $n_1 \in N$ to contradict implications from another $n_2 \in N$, but that can be avoided by renaming attributes and is a matter of how the data of an application is modelled. In this paper, the language L contains expressions formed from unary predicates and the symbols AND, OR, NOT, $\rightarrow$ and $:=$, although the symbol "AND" is usually omitted as the default operation. In general, L can be more complex. The implications of a context are considered true for all objects of the context. In this paper, the examples of classes only consist of a single context where the predicates are unary attributes. But in general, Def. 3 encompasses a wide variety of possibilities. For example, a class can be a computer program of a declarative programming language or a relational database where the elements of O are tuples and a single predicate for each table determines whether or not a tuple exists in the table.

Definition 4. *A (conceptual) schema $(R_H, R_L, \mathcal{B})$ consists of a set R of head representamens, a set R_L of (representamen) expressions that are formed using head representamens and elements of a language L with $R_H \subseteq R_L$ and a set $\mathcal{B}$ of binary (representamen) relations $B \subseteq R_L \times R_L$.*

Further, non-mathematical conditions of conceptual schemata could be formulated, for example, that a schema should be coherent, focused on a single topic and have a certain minimal and maximal size. In this paper, the vocabulary of L is AND, OR and NOT and $\mathcal{B} := \{\rightarrow, \hookrightarrow, =, :=, \perp\}$ with ":=" $\subseteq R_H \times R_L$, $r_1 = r_2 \iff (r_1 \rightarrow r_2, r_2 \rightarrow r_1)$, $(r_1 := r_2 \implies r_1 = r_2)$ and $(r_1 \hookrightarrow r_2 \implies r_1 \rightarrow r_2)$. The relations are *definition* $(:=)$, *strong synonymy* $(=)$, *hyponymy* ($\rightarrow$ or edge in a Hasse diagram), *distant hyponymy* ($\hookrightarrow$ or arrow in a Hasse diagram) and *mutual exclusivity* ($\perp$). Two expressions are in a distant hyponymy relation if the exact hyponymy chain from one to the other is not specified. Further syntactic conditions need to be provided for actual applications. The Hasse diagram on the left of Fig. 2 is an abbreviation for some of the expressions and the hyponymy relation. Each node corresponds to an expression, either

by itself ("acute") or as an AND-definition ("isosceles AND acute"), but only involving hyponyms, not distant hyponyms. The hyponymy relation is the transitive closure of the edges in the diagram. The hyponymy instance "equilateral $\rightarrow$ isosceles AND acute" is in brackets because it is redundant and can be read from the Hasse diagram.

Expressions and relations in a schema are meaningless strings that are manipulated according to the rules of a language. In order to evaluate whether expressions and relations are meaningful or true, they need to be mapped into classes using interpretations. The following definition specifies that head representamens and expressions are mapped onto concepts and relations onto true statements.

Definition 5. *A schema* $(R_H, R_L, \mathcal{B})$ *is* interpretable over *a class* (O, A_L, J, N) *if a set* $\mathcal{I}$ *of partial functions (called* interpretations*) can be defined so that* $\forall r \in R_L\ \exists i \in \mathcal{I} : i(r) \in \mathcal{C}(O, A_L, J, N)$ *and* $\forall B \in \mathcal{B}\ \forall b \in B\ \exists i \in \mathcal{I} : i(b) \in \mathcal{T}(O, A_L, J, N)$.

The definition does not provide any details with respect to how the interpretations are constructed. Further conditions must be supplied for specific applications. For example, if r_1 is a hyponym of r_2, it should be required that $i(r_1) <_n i(r_2)$ in some context n. Ideally, there should be exactly one interpretation for each formal context so that a head representamen can be assigned different concepts for different contexts but only at most one concept within a single context. Because different relation instances in a schema can utilise different interpretations, a certain amount of flexibility, ambiguity or fuzziness is possible. For example, a tomato can be a fruit in one context and a vegetable in another context. A schema should not just be interpretable, but also provide sufficient information about its underlying class as specified in the next definition. All examples of schemata in this paper fulfil Def. 6.

Definition 6. *A schema* $(R_H, R_L, \mathcal{B})$ covers *a class* (O, A_L, J, N) *under a set* $\mathcal{I}$ *of interpretations if each object concept is an interpretation of at least one representamen expression, if the object order and the relationship between an object concept and its attribute concepts is an interpretation of some instances of "*$\rightarrow$*" and* $\mathcal{T}(O, A_L, J, N)$ *can be logically derived from interpretations of representamen relations.*

4 Two Further Examples of Schemata and Classes

This section provides two further examples for developing conceptual schemata. The example in Fig. 3 is based on Ganter & Obiedkov (2016) where it is utilised for a discussion of clauses. According to the example, a driving license is passed exactly if both the theoretical and the driving part are passed and failed if one of them is failed. Ganter & Obiedkov argue that the 8 implications of the lattice do not represent the information in a natural manner. Instead they are suggesting to use 6 clauses and 2 implications. A difference between the clauses of Ganter & Obiedkov and the OR-definitions in this paper is that OR-defined attributes can be removed from the set of attributes before calculating the remaining implications. Thus the set of implications becomes smaller. The example in Fig. 3 shows that after defining the attribute "license fail" as "driving fail OR theory fail" and then removing it from the formal context, only four relation instances corresponding to implications are left. The first two correspond to both directions of

an AND-definition (“license pass” as “driving pass AND theory pass”). The other two state that passing and failing each part of a driving test is mutually exclusive. Thus the schema on the left of Fig. 3 presents all relevant information and covers the class on the right in a succinct manner.

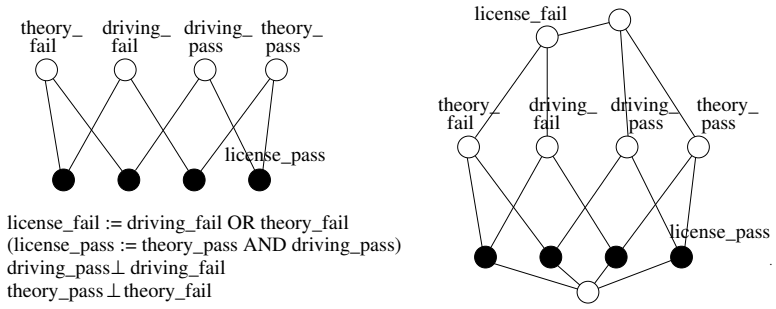


Fig. 3. A conceptual schema (left) for the driving license example (right)

The final example of this paper is based on Ganter & Wille (1999) and consists of properties of binary relations as defined in the following table[3].

property	definition
reflexive	$\forall a \in A : aRa$
irreflexive	$\forall a \in A : \neg aRa$
symmetric	$\forall a, b \in A : aRb \rightarrow bRa$
asymmetric	$\forall a, b \in A : aRb \rightarrow \neg(bRa)$
antisymmetric	$\forall a, b \in A : aRb$ and $bRa \rightarrow a = b$
transitive	$\forall a, b, c \in A : aRb$ and $bRc \rightarrow aRc$
semiconnex	$\forall a \neq b \in A : aRb$ or bRa
connex	$\forall a, b \in A : aRb$ or bRa

The concept lattice in Fig. 4 follows Ganter & Wille (1999). But it contains additional AND-defined attributes, such as “preorder := reflexive AND transitive”. It also contains some attributes about extreme cases. The example assumes that the relations R are defined as $R \subseteq S \times S$ for a non-empty set S. It may seem counter-intuitive that a relation can be both an order relation and an equivalence relation or symmetric and antisymmetric at the same time because that is only possible for extreme cases with attributes such as $R = S \times S$, $R = \{(i, j) \mid i = j\}$, $|S| = 1$ or $R = \{\}$. These attributes are included in the lattice.

Supplemental concepts are identified in Fig. 4 using background knowledge about binary relations. OR-definitions are only applicable to supplemental concepts. In the

[3] It should be remarked that the notions “semiconnex” and “connex” are used ambiguously in the literature. Sometimes “connex” is used instead of “semiconnex” and “strong connex” instead of “connex”.

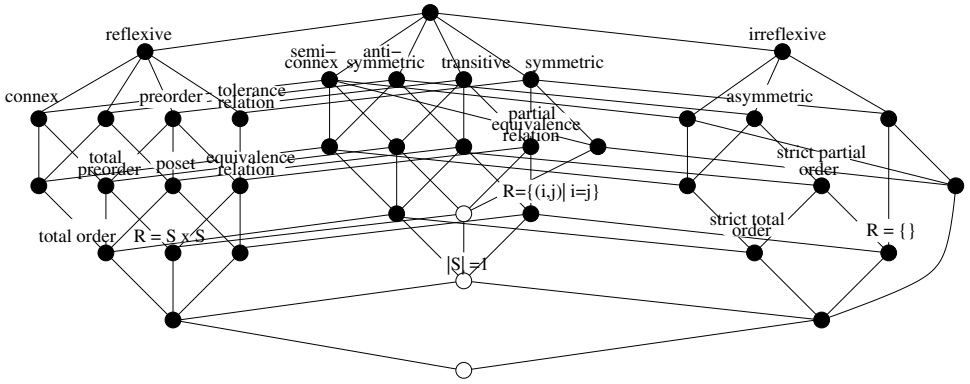


Fig. 4. A conceptual class of types of binary relations

previous examples, the concept lattices had supplemental concepts higher up in the lattice which could be removed using OR-definitions. This example only has very few supplemental concepts which are at the bottom of the lattice. These could be removed by introducing more attributes according to Lemma 2, but that does not reduce the complexity of the lattice significantly and increases the set of implications. Instead, the suggestion for developing a conceptual schema in this example is to extract meaningful parts of the lattice. Fig. 4 indicates a subdivision according to whether a relation is reflexive, irreflexive or neither. But such a division groups equivalence relations closely with order relations which are separated from strict orders. Thus it seems more natural to consider symmetric and NOT-symmetric as the main dividing factor for types of binary relations. Therefore, Fig. 5 and Fig. 6 divide the conceptual schema of binary relations into 3 parts: those that are not symmetric and tend to be orders, those that are symmetric and closely related to equivalence relations and the extreme cases at the bottom of the lattice which are antisymmetric and symmetric at the same time.

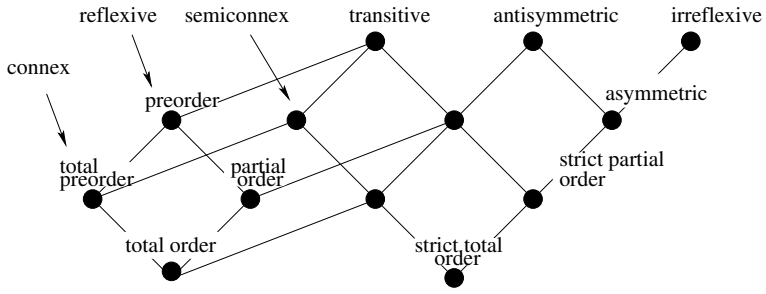


Fig. 5. Conceptual schema for types of binary relations: part 1

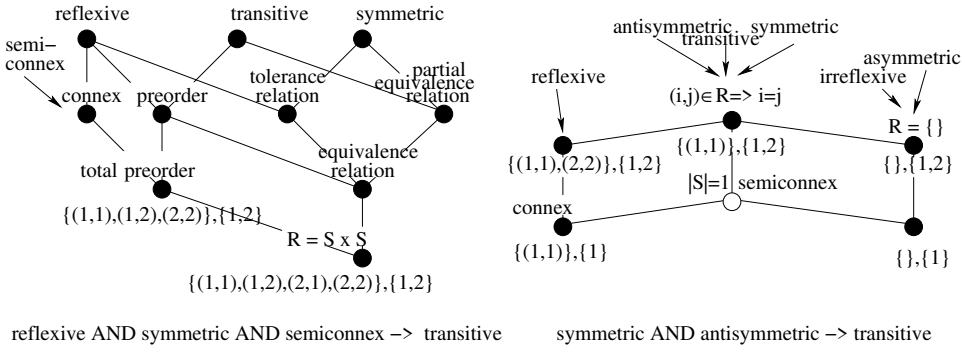


Fig. 6. Conceptual schema for types of binary relations: parts 2 and 3

The three parts of the schema in Fig. 5 and Fig. 6 are derived from the lattice of the class by deleting and restricting some attributes. Restricting means in this case that an attribute is replaced by its meet with another attribute. For example, in Fig. 5, the attribute "symmetric" is deleted and the attributes "reflexive", "semiconnex" and "connex" are replaced by their meet with "transitive" which results in distant hyponyms in the schemata. AND-definitions involving distant hyponyms cannot be read from the Hasse diagrams of the schemata. The left schema in Fig. 6 is derived by deleting the attributes "antisymmetric", "asymmetric", "irreflexive" and "semiconnex". The right schema is derived by restricting all attributes to their meet with "antisymmetric", "symmetric" and "transitive". For the non-restricted attributes, the hyponymy relation of the schema corresponds to the subconcept relation of the class and results in the same implications. The restricted attributes are considered distant hyponyms because their implications are not completely contained in the parts of the schema. All phrases in Fig. 5 are head representamens. In Fig. 6, the phrases and formulas that are written above the nodes are head representamens. The strings below the nodes represent prototypical examples and are not head representamens. The conceptual class contains four implications which are not just AND-definitions. Each of these four implications is included in the part of the schema where it is visible. In this case, even the schemata are still quite complex. But that is due to the subject manner. Learning all the relevant information about the head representamens in Fig. 5 and Fig. 6 will require a significant amount of time.

5 Conclusion

In summary, conceptual classes and schemata mutually influence each other. In some cases, it might be more suitable to extract a class from a schema using some form of conceptual exploration. In other cases, a schema can be constructed after reducing a class. The following strategies can be employed:

- Possibly splitting the context into smaller coherent subcontexts
- Conceptual exploration (for completing the set of objects and attributes)

- Purifying the lattice, adding SYN-definitions
- AND-reduction
- OR-reduction
- Further OR-reduction after adding attributes according to Lemma 2
- NOT-reduction

The motivation behind this strategy is that with respect to relationships between conceptual schemata and classes, the core content of a class is retained in its object concepts, their ordering and the AND-definitions of object concepts. The concepts that are above the object order may be less important, in particular if they are supplemental concepts, because they tend to represent attributes that may be expressible as OR-definitions.

References

1. Berry, A.; Gutierrez, A.; Huchard, M.; Napoli, A.; Sigayret, A. (2014). Hermes: a simple and efficient algorithm for building the AOC-poset of a binary relation. Annals of Mathematics and Artificial Intelligence, 72, 1, p. 45-71.
2. Carbonnel, J.; Bertet, K.; Huchard, M.; Nebut, C. (2016). FCA for software product lines representation: Mixing configuration and feature relationships in a unique canonical representation. In: Concept Lattices and their Applications (CLA'16), CEUR, p. 109-122.
3. Ganter, B.; Wille, R. (1999). Formal Concept Analysis. Mathematical Foundations. Springer.
4. Ganter, B.; Obiedkov, S. (2016). Conceptual Exploration. Springer.
5. Ganter, B. (2019). "Properties of Finite Lattices" by S. Reeg and W. Weiß, Revisited. In: Cristea et al. (eds) Formal Concept Analysis. ICFCA 2019. LNCS 11511, Springer, p. 99-109.
6. Lakoff, G. (1987). Women, Fire, and Dangerous Things. What Categories Reveal about the Mind. The University of Chicago Press.
7. Lopez-Rodriguez D., Cordero P., Enciso M., Mora A. (2021). Clustering and Identification of Core Implications. In: Braud et al. (eds.) Formal Concept Analysis. ICFCA 2021. LNAI 12733, p. 138-154.
8. Osswald, R.; Petersen, W. (2002). Induction of classifications from linguistic data. In: Proc. of the ECAI-Workshop on Advances in Formal Concept Analysis for Knowledge Discovery in Databases.
9. Prediger, S. (1998). Simple concept graphs: A logic approach. In: Mugnier et al. (eds.) Conceptual Structures: Theory, Tools and Applications. ICCS 1998. LNCS 1453, Springer, p. 225-239.
10. Priss, U.; Old, L. J. (2010). Concept Neighbourhoods in Lexical Databases. In: Kwuida; Sertkaya (eds.), Formal Concept Analysis. ICFCA 2010. LNCS 5986, Springer, p. 283-295.
11. Priss, U.; Old, L. J. (2011). Data Weeding Techniques Applied to Roget's Thesaurus. In: Wolff et al. (eds.) Knowledge Processing and Data Analysis. KPP 2007. LNAI 6581, Springer, p. 150-163.
12. Priss, U. (2017). Semiotic-Conceptual Analysis: A Proposal. International Journal of General Systems, 46, 5, p. 569-585.
13. Priss, U. (2021a). Diagrammatic Representation of Conceptual Structures. In: Braud et al. (eds.) Formal Concept Analysis. ICFCA 2021. LNAI 12733, p. 281-289.
14. Priss, U. (2021b). Conceptual Schemata as a Means for Structuring Teaching Materials. In: Concepts in Action: Representation, Learning, and Application (CARLA'21). Available at: https://www.conceptuccino.uni-osnabrueck.de/carla_workshop/carla_2021.html

Exploring the dataset structure by means of delta-classes of equivalence. *The case of the Titanic dataset*[⋆]

Aleksey Buzmakov[1][0000−0002−9317−8785], Sergei O. Kuznetsov[1][1111−2222−3333−4444], Tatyana Makhalova[2][2222−−3333−4444−5555], and Amedeo Napoli[1,2][2222−−3333−4444−5555]

[1] National Research University Higher School of Economics, Russia
[2] LORIA (CNRS – Inria NGE – University of Lorraine), Vandœuvre-lès-Nancy, France
{avbuzmakov,skuznetsov}@hse.ru,
tatiana.makhalova@inria.fr, amedeo.napoli@loria.fr

Abstract. Being able to have a quick look at a dataset is essential in many applications. One way is to summarize the dataset by means of a small set of patterns. In this paper we suggest defining such set of patterns as the closed elements of delta-classes of equivalence. This approach allow us to propose an overview of a dataset and then, if necessary, any delta-class of equivalence can be expanded to provide more detailed information about a certain part of the dataset.
To demonstrate our proposal we deeply studied the Titanic dataset about survival of passengers and showed the connections between the passenger attributes and a possible dataset summary in terms of patterns.

Keywords: FCA · Δ-closure · Δ-concepts · use case · Δ-implications.

1 Introduction

In this paper, we are interested in pattern or itemset mining in tabular data. There is a considerable amount of work on many aspects of this subject, especially regarding algorithms and search for interesting patterns [1]. One recurrent problem in itemset mining is the exponential number of resulting itemsets. Focusing on closed itemsets allows a significant reduction of this number by replacing a whole class of itemsets with the largest one, i.e., the closed itemset, which has the same support [7]. Nowadays, there are very efficient algorithms for computing frequent closed itemsets [6], even for low frequency thresholds. However, the efficient generation of closed itemsets only partially solves the problem of the exponential explosion of itemsets, since the main difficulties appear afterwards, when the generated itemsets are processed.

An alternative to the exhaustive enumeration of itemsets is based on "sampling" [4] and on a gradual search for itemsets according to an interestingness

⋆ The reported study was funded by RFBR, project number 20-31-70047

measure or a set of constraints [8]. Such algorithms usually result in a rather small set of itemsets while they may provide only an approximate solution. Although both approaches use quite different techniques, they rely on the same assumption, namely that the "internal or intrinsic structure" of the dataset under study can be understood by means of subset of selected itemsets.

Then, in each approach a particular set of itemsets is returned, which provides a "multifaceted view" of the intrinsic structure underlying the data.

Our approach is based on Δ-classes of equivalence, the generalization of standard classes of equivalence based on closure operator. A user-set parameter Δ measures how much a closed set can differ from its upper neighbors in the partial order of closed sets.

A Δ-class of equivalence allows one to characterize the distribution underlying the data, i.e., when Δ is large, there are only a few Δ-classes of equivalence whose elements are very stable, while when Δ is small, the number of Δ-classes increases and the related information becomes less stable. Moreover, the Δ-classes of equivalence are very stable for large Δ and do not significantly depend on the data sampling used for the analysis.

In this paper we study Δ-classes for Titanic dataset. In particular, we show what kind of conclusions w.r.t. the passengers of Titanic can be made, starting from Δ-classes of equivalence for large Δ and then gradually "diving" into the most interesting Δ-classes of equivalence by decreasing the value of Δ.

The paper has the following structure. First we introduce basic definitions related to generalized closure operator. Then in Section 3 we evaluate this closure operator on Titanic dataset.

2 Δ-classes of equivalence

The proposed approach is introduced in terms of Formal Concept Analysis (FCA) [5] and the following notation. A formal context is a triple (G, M, I), where G and M are sets of objects and attributes correspondingly and $I \subseteq G \times M$ is a relation between them. Derivation operator is denoted with arrows in order to clearly show the range and domain of the corresponding mappings:

$$A^{\uparrow} = \{m \in M \mid (\forall g \in A)(g, m) \in I\}, A \subseteq G \quad (1)$$

$$B^{\downarrow} = \{g \in G \mid (\forall m \in B)(g, m) \in I\}, B \subseteq M \quad (2)$$

We should note that operators $(\cdot)^{\uparrow\downarrow}$ and $(\cdot)^{\downarrow\uparrow}$ form closure operators on 2^G and 2^M, respectively. Hereafter, we focus on the operator $(\cdot)^{\downarrow\uparrow}$ and its generalizations. Let us reformulate the definition in terms of "instance counting". Let B be any set of attributes.

Definition 1. *An attribute set B is closed iff* $(\forall m \in M \setminus B)|(B \cup \{m\})^{\downarrow}| \neq |B^{\downarrow}|$.

It can be seen that $(\cdot)^{\downarrow\uparrow}$ maps any set of attributes to a closed set of attributes. Let us reformulate Definition 1 in the following equivalent way. B is closed iff

$$(\forall m \in M \setminus B)|B^{\downarrow}| - |(B \cup \{m\})^{\downarrow}| \geq 1.$$

This form allows changing the threshold 1 at the end of the formula to any other positive value. The larger this value, the less attribute sets would pass a test similar to the one from Definition (1). This leads to the definition of Δ-closedness of an attribute set [2].

Definition 2. *A set of attributes B is called Δ-closed if for any $m \in M$:*

$$|B^{\downarrow}| - |(B \cup \{m\})^{\downarrow}| \geq \Delta \geq 1. \tag{3}$$

In [2] it was shown that Δ-closedness is a closure operator. In particular it means that from a computational point of view, given a non Δ-closed set of attributes B, i.e., $\exists m \in M(|B^{\downarrow}| - |(B \cup \{m\})^{\downarrow} < \Delta)$, it can be closed by iteratively changing B to $B \cup \{m\}$, for any m violating (3) until such attribute is not found. The corresponding closure operator is denoted by $(\cdot)^{\Delta}$. Since it is a closure operator, it divides the all sets of attributes 2^M into classes of equivalences having the same closure.

Definition 3. *Given an attribute set B, its equivalence class $Equiv_{\Delta}(B)$ is the set of all attribute sets with the closure equal to the closure of B, i.e.,*

$$Equiv_{\Delta}(B) = \{X \subseteq M \mid (X)^{\Delta} = (B)^{\Delta}\}. \tag{4}$$

Moreover, since according to Definitions 1 and 2 if a set of attributes is Δ-closed than it is necessary closed. Thus, these Δ-classes of equivalence are joins of several closure-based classes of equivalence. Such closure operators allows introducing a new derivation operator related to Δ-closure.

$$A^{\uparrow_{\Delta}} = (A^{\uparrow})^{\Delta}, A \subseteq G \tag{5}$$

$$B^{\downarrow_{\Delta}} = (B)^{\Delta\downarrow}, B \subseteq M \tag{6}$$

Such Δ-derivation operators allow defining Δ-concepts ordered within a lattice in the similar way to the classical formal concepts. Moreover, since any Δ-closed set of attributes is a closed set of attributes, than the set of Δ-concepts are subset of formal concepts and the order of Δ-concepts is a suborder of the corresponding formal lattice.

Finally, we should discuss Δ-implications since they provide a useful tool for finding associations between attribute sets.

Definition 4. *A rule $A \underset{\Delta}{\rightarrow} B$ is called Δ-implication if $B = A^{\Delta} \setminus A$, i.e., A and $A \cup B$ are from the same Δ-class of equivalence.*

Since Δ-closure can change the support of the attribute set a Δ-implication is not necessary an implication. However, the set of Δ-implications is subset of all association rules and thus is more easy to analyse. In particular, given an implication $A \rightarrow B$, the set B is the set of attributes that are associated with A in most samples from the underlying distribution.

In the next section we experimentally show how such lattices of Δ-concepts and the corresponding implications can be used for data analysis. Moreover,

since Δ-concepts can be found in polynomial time[3] [3], such analysis is suitable for processing really big data.

3 Evaluation

3.1 Dataset

In this paper we use Titanic dataset downloaded (train dataset) from Kaggle[4]. This is one of the most known datasets with quite easily interpretable patterns that do not require deep diving into the domain knowledge. The dataset describes 891 passengers of the last Titanic ship travel. Every passenger is described with name, age, sex, the number of parents and/or children and the number of spouse and/or siblings travelled together with the passenger. The ticket price and the ticket class is also known as well as the survival state of the passenger after the Titanic shipwreck.

All numerical data is divided into 5 percentiles and then *inter-ordinal scaling* is used on top of these percentiles. For example, for the quantity `Age` it is known from the data that $\frac{1}{5}$ of the passengers were below 19 years old, the next $\frac{1}{5}$ between 19 and 25, then between 25 and 32 and then between 32 and 41 and finally the last $\frac{1}{5}$ of the passengers were above 41 years old. Then new binary attributes are formed based on these limits (19, 25, 32,41), these eight attributes are "`Age` ≥ 19," "`Age` ≤ 19". "`Age` ≥ 25," "`Age` ≤ 25". "`Age` ≥ 32," "`Age` ≤ 32". "`Age` ≥ 41," "`Age` ≤ 41".

Additionally, from the "Name" field the social status is extracted, including "Mr", "Mrs", "Master", etc. It makes in total 49 attributes.

3.2 Concept lattice navigation

Even for such relatively simple data the total number of concepts is 9002. It is not hard to build such a lattice. However, analysis of the lattice is quite hard. It is hardly possible to draw the whole lattice and the only way is to navigate it from the top or from the bottom concepts. However, Δ-classes of equivalence give another means for such analysis and navigation.

Let us first increase the Δ threshold for the lattice. If $\Delta = 90$, then the lattice size is only 11 and it can be drawn. It is shown in Figure 1. For every concept the corresponding extent size and Δ-measure are shown. Every attribute is shown outside of the concept with an arrow attached to the concept of the first attribute entry. It can be seen that the lattice involves only 9 attributes out of 49. All other attributes are attached to the `BOTTOM` concept and are not shown. It

[3] Being more precise the enumeration procedure can be set in such a way, that it finishes in input-polynomial time. It is achieved by updating the threshold θ if the number of the already found patterns is too large. The result is the set of all patterns with $\Delta \geq \theta$. However, if θ is automatically set to be too high, the procedure still finishes in input-polynomial time but the result set is empty.

[4] https://www.kaggle.com/c/titanic

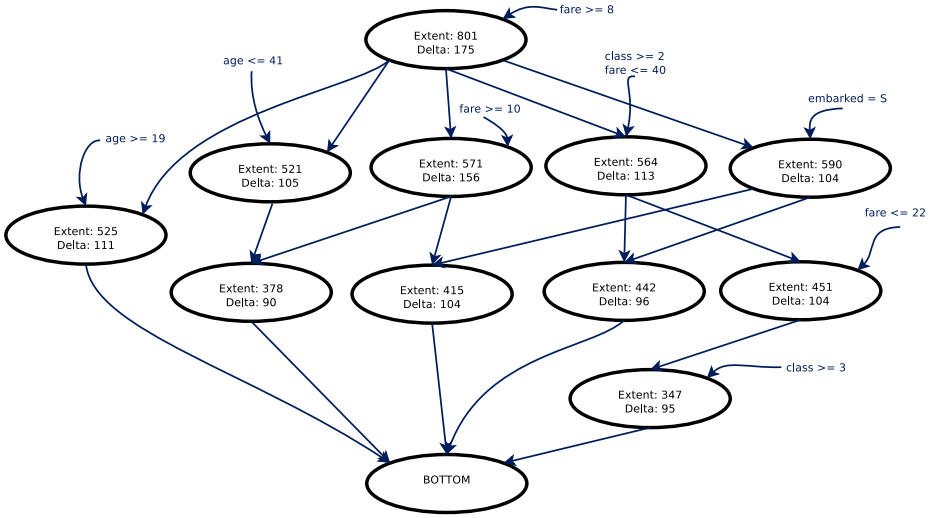


Fig. 1. The lattice of concepts with $\Delta \geq 90$ for Titanic dataset.

means that starting for any attribute a from this set, one has $\Delta(\{a\}^{\downarrow}) < 90$, i.e., smaller than the threshold. For example, the attribute $\texttt{fare} \geq 8$ is introduced to the lattice at the very top concept. It means that $|\emptyset^{\downarrow}| - |\{\texttt{fare} \geq 8\}^{\downarrow}| < 90$, i.e., for most of objects $\texttt{fare} \geq 8$. Moreover, since the top concept in this lattice has the maximal value of Δ, we can say that its extent is the most typical extent for all passengers. Since we can make it more precise without excluding less objects than 175.

Then we can see that there are 5 concepts below the top concept. They are the only concepts that are significantly different from their children. Accordingly, this concept lattice for $\Delta = 90$ shows the structure of different groups of passengers (formal concepts with Δ-closure for $\Delta = 90$. A concept is only shown if there is no more precise description that contains similar number of objects.

This setting allows finding and prioritizing the association rules related to a certain set of attributes.

3.3 Δ-Association rules

Let us study female subpopulation of the passengers. In particular, we can try to Δ-close the following description $\texttt{sex} = \texttt{female}$. What Δ should be used? One of the reasonable choice is the maximal Δ such that Δ-closure of $\texttt{sex} = \texttt{female}$ is different from M. For example in Figure 1 no concept with $\texttt{sex} = \texttt{female}$ is found. It means that for $\Delta \geq 90$, $\{\texttt{sex} = \texttt{female}\}^{\downarrow\uparrow_\Delta} = M$. If the whole lattice is available it corresponds to the concept with the maximal Δ-measure with the intent being a superset $\{\texttt{sex} = \texttt{female}\}$. In the form of Δ-implication it can be

written as[5]:

$$\{\texttt{sex} = \texttt{female}\} \underset{\Delta}{\rightarrow} \{\texttt{fare} \geq 10\}.$$

It suggests that women rarely buy cheapest tickets. Moreover, since we do not find $\texttt{class} \leq 2$ attribute, it means that they can afford the 3rd class, but nevertheless not the cheapest tickets.

What about men?

$$\{\texttt{sex} = \texttt{male}\} \underset{\Delta}{\rightarrow} \{\texttt{title} = \texttt{Mr}, \texttt{fare} \geq 8\}.$$

Now we see that the preference for more expensive tickets is missing. However, we see that most of the men are titled "Mr". It is not the classical closure assuming that all men are `Mr`. Indeed, the correspondence between the title and sex is shown in Table 1.

Table 1. Correspondence between passenger sex and title

	Miss	Mrs	Mr	Master	Other
female	182	128	0	0	4
male	0	0	525	40	12

Let us dive deeper into the title. What do we know about `Master`-title?

$$\{\texttt{title} = \texttt{Master}\} \underset{\Delta}{\rightarrow} \{\texttt{sex} = \texttt{male}, \texttt{age} \leq 19, \texttt{fare} \geq 10, \texttt{class} \geq 2\}.$$

We can see, that `Master` corresponds to young men from 2nd and 3rd class but not with the cheapest tickets. It is a quite strange combination and a deeper investigation is needed. However, it is not an artefact of the procedure. If we check the original dataset all findings are supported, i.e., they are mostly from the 2nd and 3rd class, but not so cheap. The proposed procedure was useful here only for highlighting such finding. In contrast, for `Miss`-title no new information is found.

Let us now formulate a question about women that had cheap tickets:

$$\{\texttt{sex} = \texttt{female}, \texttt{fare} \leq 10\} \underset{\Delta}{\rightarrow} \{\texttt{title} = \texttt{Miss}, \texttt{class} = 3, \texttt{fare} \leq 8\}.$$

In fact if decision is to travel cheap, then it is the cheapest option. Similar, answer will be given for men traveled cheap. However, the group of men that traveled cheep is about 7 times larger than the group of women. And thus, if we would have just requested "who traveled cheap", than the result would be the group of men.

Finally, let us ask how age affects the travel behavior.

$$\{\texttt{age} \leq 25\} \underset{\Delta}{\rightarrow} \{\texttt{class} \geq 2, 8 \leq \texttt{fare} \leq 40\}.$$

[5] For simplicity, the attribute sets are shown with reduction, i.e., if by knowing that $\texttt{fare} \geq 10$ we can conclude that $\texttt{fare} \geq 8$, the last attribute is not shown.

Thus, young people usually travel in the 2nd or 3d class. Similarly, for people aged more than 41 years.

$$\{\texttt{age} \geq 41\} \underset{\Delta}{\rightarrow} \{\texttt{class} \leq 2, \texttt{fare} \geq 22\},$$

i.e., such people usually prefer the 1st or 2nd class and the fare is more than 22.

Let us finally show that we can be interested also in combinations of attributes. For example, what about women of age more than 41 years?

$$\{\texttt{age} \geq 41, \texttt{sex} = \texttt{female}\} \underset{\Delta}{\rightarrow} \{\texttt{class} = 1, \texttt{fare} \geq 40, \texttt{title} = \texttt{Mrs}\}$$

So in contrast to generally aged people, women usually travel in the 1st class and they are titled `Mrs`.

4 Conclusion

Based on Titanic dataset Δ-classes of equivalence are shown to be useful for exploratory data analysis. In particular, it can be used to systematize the dataset and to prioritize association rules related to certain requests, e.g., every attribute can be associated with the most frequent attributes taken into account co-occurrences of the attributes. Since, Δ-classes of equivalence can be found in polynomial time [3], such approach is suitable for very large datasets. Moreover, such approach focuses on the distribution, where the dataset is taken from, rather than the dataset itself.

References

1. Aggarwal, C.C., Han, J.: Frequent pattern mining. Springer (2014)
2. Boley, M., Horváth, T., Wrobel, S.: Efficient discovery of interesting patterns based on strong closedness. Statistical Analysis and Data Mining: The ASA Data Science Journal **2**(5-6), 346–360 (2009)
3. Buzmakov, A., Kuznetsov, S.O., Napoli, A.: Fast generation of best interval patterns for nonmonotonic constraints. In: Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases. pp. 157–172. Springer (2015)
4. Dzyuba, V., van Leeuwen, M., De Raedt, L.: Flexible constrained sampling with guarantees for pattern mining. Data Mining and Knowledge Discovery **31**(5), 1266–1293 (2017)
5. Ganter, B., Wille, R.: Formal concept analysis–mathematical foundations (1999)
6. Hu, Q., Imielinski, T.: Alpine: Progressive itemset mining with definite guarantees. In: Proceedings of the SIAM International Conference on Data Mining. pp. 63–71. SIAM (2017)
7. Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Efficient mining of association rules using closed itemset lattices. Information systems **24**(1), 25–46 (1999)
8. Smets, K., Vreeken, J.: Slim: Directly mining descriptive patterns. In: Proceedings of the 12 SIAM International Conference on Data Mining, Anaheim, California. pp. 236–247 (2012)

FCA Went (Multi-)Relational, But Does It Make Any Difference?

Mickaël Wajnberg[1], Petko Valtchev[1], Mario Lezoche[2], Alexandre Blondin-Massé[1], and Hervé Panetto[2]

[1] Université du Québec À Montréal, Canada {name.firstname}@uqam.ca
[2] Université de Lorraine, France {firstname.name}@univ-lorraine.fr

Abstract. Relational Concept Analysis (RCA) was designed as an extension of Formal Concept Analysis (FCA) to multi-relational datasets, such as the ones drawn from Linked Open Data (LOD) by the type-wise grouping of the resource into data tables. RCA has been successfully applied to practical problems of AI such as knowledge elicitation, knowledge discovery from data and knowledge structuring. A crucial question, yet to be answered in a rigorous manner, is to what extent RCA is a true extension of FCA, i.e. reveals concepts that are beyond the reach of core FCA even using a suitable encoding of the original data. We show here that the extension is effective: RCA retrieves all concepts found by FCA as well as many further ones.

Keywords: Multi-relational Data · Formal Concept Analysis · RDF · Propositionalization.

1 Introduction

FCA provides the mathematical framework for several Knowledge Discovery in Databases (KDD) tasks whenever the data is purely, or at least predominantly, of categorical nature. Indeed, FCA-based association discovery and conceptual clustering have been applied to knowledge base structuring, ontology learning, anomaly detection, observation classification, etc. Most real datasets, though, stray from being purely categorical. FCA thus provides a set of scaling operators to deal with numerical and otherwise ordered scales. In AI, the majority of interesting data, such as those compatible with the LOD format, have relational structure. They can be represented either as graphs (for instance, named graphs in RDF) or as sets of relational tables. Approaches have been designed for the former, emphasizing the intra-data object links, e.g. *logical FCA* [7] and *pattern structures* [8], for graph datasets. For the latter, the focus is on inter-object links, e.g. in datasets structured as a unique RDF graph. In this second trend, more akin to *power-context familie*s [15] and *Graph-FCA* [6], we focus on the particular approach of relational concept analysis (RCA). It has already been successfully applied to a wide range of practical problems such as hydroecology [4], industrial decision making [12] or biology [1,13]. Rather than in a global graph, RCA shapes the data as a set of $\times$-tables, complying to the *Entity-Relationship* framework [2].

Part of the tables have the classical objects × properties format (entity types, FCA contexts) while the remainder represent objects × objects relations.

A natural question is whether RCA does extend the reach of FCA, knowing that for single datasets, whatever the level of complexity of the object descriptions (sequences, trees, graphs), the results of an FCA-based processing on those descriptions can be brought down to FCA on a context made of suitably-chosen derived **attributes**. The question is all the more important as prior studies seem to imply it does not [3] (though, for a reduced version of RCA). We make the case here for RCA as a true extension of FCA, in the sense that due to its multi-relational input and fixed point computation, it detects concepts that are out of reach for FCA while, in turn, retrieving all concepts that FCA is able to reveal. To that end, we chose some plausible re-encodings of a simple relational context family (RCF), the hypothesis being that with more complex datasets, the phenomenon only amplifies.

The remainder of the paper is as follows: Section 2 provides background on RCA while Section 3 presents our FCA-vs-RCA comparison. Next, Section 4 discusses the comparison outcome and Section 5 concludes.

2 Background

Formal concept analysis [14] is a mathematical method for eliciting the conceptual structure of "object × attribute" datasets. Data are gathered within a *(formal) context*, a triple $K = (O, A, I)$ where O is a set of objects, A is a set of attributes and $I \subseteq O \times A$ is the context incidence relation, where $(o, a) \in I$, also written oIa, means that the object o bears the attribute a. A context induces two derivation operators: one mapping objects to attributes, and the reciprocal. The object derivation $'$ maps a subset X of objects to the set of attributes shared by all members of X, $' : \wp(O) \longrightarrow \wp(A)$ with $' : X \mapsto \{a \in A \mid oIa \ \forall o \in X\}$. The dual attribute derivation, also denoted by $'$, works the other way around, $' : \wp(A) \longrightarrow \wp(O)$ with $' : Y \mapsto \{o \in O \mid oIa \ \forall a \in Y\}$. Inside a context K, a *(formal) concept* is a pair $(X, Y) \subseteq O \times A$ such that $X' = Y$ and $Y' = X$. The sets X and Y are called *extent* and *intent* of the concept (X, Y), respectively.

FCA extracts conceptual abstractions on objects by factoring out shared attributes. *Relational concept analysis* [10] extends it by factoring in relational information, as available in multi-relational datasets [5]. RCA admits multiple sorts of objects in its input format, each organized as a separate context, plus a set of binary relations between contexts. The input data structure, called *relational context family* (RCF), is thus a pair $(\mathbf{K}, \mathbf{R})$ where $\mathbf{K} = \{K_i\}_{i=1,\ldots,n}$ is a set of distinct contexts $K_i = (O_i, A_i, I_i)$ and $\mathbf{R} = \{r_k\}_{k=1,\ldots,m}$ a set of binary relations $r_k \subseteq O_i \times O_j$ where the contexts K_i and K_j are the domain and range contexts of r_k, respectively. Relational tables are also processed in their own way, as explained below. A cross in the table of relation r for $(domain\_object_i, range\_object_j)$, can be understood as the first order logic term $r(domain\_object_i, range\_object_j)$ being true.

RCA distills the shared relational information (i.e., inter-object links) using *propositionalization* [9]: It integrates new attributes into an extended version of the initial context, say $K_d = (O_d, A_d, I_d)$, to further refine the conceptual structure of the underlying object set. To increase shareability, rather than the individual objects from the target (range) context, say K_t, the new attributes refer to abstractions on them. In its most basic version, RCA exploits the natural conceptual structure provided by the concepts of each context. Indeed, two links of relation $r : d \longrightarrow t$ departing from o_1 and o_2 from O_d and referring to two distinct objects $\bar{o}_1$ and $\bar{o}_2$ from O_t, respectively, are distinct information. However, replacing $\bar{o}_1$ and $\bar{o}_2$ with a common abstraction, say $\{\bar{o}_1, \bar{o}_2\}'$, makes the new information shareable. Relational scaling follows a well-known schema from description logics: Given a relation r, for each concept c_t from the range context of r, it produces, for A_d, an attribute $q\,r : c_r$ where q is an operator chosen before-hand from a set Q. RCA admits, among others, standard description logics restrictions ($Q = \{\exists, \forall, \forall\exists, \ldots\}$), which behold their respective semantics (see [10] for details and example 1 for illustration).

Example 1. Assume a RCF made of contexts on *people* and *cars*, and an ownership relation, or *pos(sesses)*, which are given in Tables 1, 3 and 2, respectively. The cars lattice is shown in Figure 1. Now, an $\exists$-scaling of the relation *pos* using that lattice will add, for each car concept c, a new attribute $\exists pos : c$ to the person context, e.g. $\exists pos : cars\_4$ and $\exists pos : cars\_3$ that can be rewritten as $\exists pos : (cp)$ and $\exists pos : (el, pw)$, respectively, using intents as IDs.

K_P	Senior	Adult	Male	Female	I.T.	Sport
Fa	×		×			
La		×		×	×	
Sh		×		×		
Tr		×	×		×	×

Table 1: Person Context

pos	tw	$t3$	zo	$f5$
Fa	×			
La		×		×
Sh	×			×
Tr				

Table 2: Relation pos

K_C	el	pw	cp	ch
tw			×	×
$t3$	×	×		
zo	×		×	
$f5$		×	×	

Table 3: Car Context

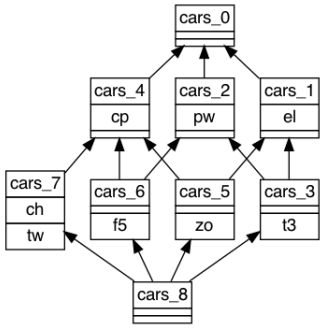


Fig. 1: Car Lattice

A scaling step results in the related contexts being extended, which, in turn, may lead to the emergence of new concepts. Thus, as the set of available abstractions increases, a scaling step with the differential set of concepts would produce further relational attributes and the whole process would go on cycling. The resulting iterative context refinement necessarily ends at a fixed point [10], i.e. a set of lattices whose concepts refer to each other via relational attributes.

3 What can RCA do for AI (that FCA can't) ?

Below, we examine two encoding strategies that bring a multi-relational dataset to a mono-relational one, i.e. aggregate several contexts into a single data table, so that they can be fed to classical FCA.

3.1 Encoding multiple contexts into a single one

Assume a simple RCF made of two contexts and a relation (see Figure 2). We use this simple case for our reasoning, knowing that in more complex cases, i.e. three or more contexts and several relations, it can be extended appropriately. Moreover, while there could be a wide range of concrete encoding disciplines [11], the principle behind them admits only two basic cases, i.e. entity-centric and relation-centric. In our FCA/RCA perspective this boils down to which sort of RCF element, i.e. context or relation, is put center-stage.

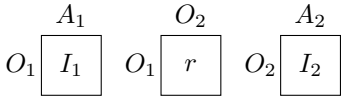


Fig. 2: Fictitious RCF

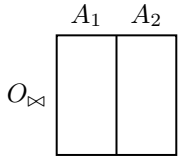


Fig. 3: Semi-join

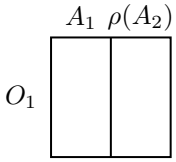


Fig. 4: Aggregation

The first encoding principle we examine below emphasizes the object-to-object relation as a primary construct and pivotal element of the encoding. Its member pairs become first-class objects which carry the attributes of both contexts incident to the relation. Technically speaking, the method is akin to the *(semi-)join* operation of relational algebra. The overall encoding schema is illustrated in Figure 3 whereas Section 3.2 proposes a formal definition thereof. It also provides a detailed comparison of the results from applying RCA to the RCF from Figure 2 with those of FCA on the semi-join of the two initial contexts.

A bit closer to the RCA propositionalization spirit, a second encoding principle emphasizes the context as a main construct and driver of the encoding: The domain context of the relation is extended with some additional attributes that translate the relation while following a technique akin to relational scaling. The main difference here is that the context is the one-shot context extension. The procedure whose details are discussed in Section 3.3, is schematically illustrated in Figure 4. Moreover, the asymmetric encoding of the relation and the one-shot extension amount to processing the range context as if it were aggregated into the domain one. Therefore, we termed the overall encoding principle the *aggregation* and the resulting context the aggregated one.

Finally, please notice that in the detailed investigation of each case (see below), our reasoning follows three steps: 1) We pick an arbitrary formal concept from the FCA output, 2) we show the RCA output comprises a concept with the same objects, and 3) we establish the link between the intents of both concepts.

3.2 Semi-join in single relation RCF

We consider here the concurrent case where the FCA is applied on a context encoding the semi-join of this RCF, as presented in Figure 3. This encoding consists in creating the objects of $O_{\bowtie}$ as the object pairs (o_1, o_2) where $o_1 \in O_1 \cup \{\bot\}$, $o_2 \in O_2 \cup \{\bot\}$, according to the RCF modeling of O_1, O_2 Figure 2. The $\bot$ object is a fictitious empty object with no attributes used to complete the semi-join. There are three cases to define the elements of $O_{\bowtie}$:

- If $o_1 \in O_1$, $o_2 \in O_2$, then $(o_1, o_2) \in O_{\bowtie}$ if and only if $(o_1, o_2) \in r$
- If $o_1 = \bot$, $o_2 \in O_2$, then $(o_1, o_2) \in O_{\bowtie}$ if and only if $r^{-1}(o_2) = \emptyset$, i.e. if there is no $x \in O_1$ such that $(x, o_2) \in r$
- If $o_1 \in O_1$, $o_2 = \bot$, then $(o_1, o_2) \in O_{\bowtie}$ if and only if $r(o_1) = \emptyset$, i.e. if there is no $x \in O_2$ such that $(o_1, x) \in r$

Example 2. As an illustration of the above modeling, assume an RCF made of contexts for people (Table 1) and for cars (Table 3) plus an ownership relation (*pos*session, Table 2). In the first context, *Farley, Lane, Shana,* and *Trudy* are described by being *senior* or *adult*, *male* or *female*, working in *IT*, and practicing a lot of *sports*. Cars –Twingo, Tesla 3, Zoe, and Fiat 500– can be electrical, powerful, compact or (not exclusive) cheap. The corresponding semi-join context is presented in Table 4.

$K_{\bowtie}$	Senior	Adult	Male	Female	I.T.	Sport	*el*	*pw*	*cp*	*ch*
(Fa,*tw*)	×		×						×	×
(La,*t*3)		×		×	×		×	×		
(La,*f*5)		×		×	×			×	×	
(Sh,*tw*)		×		×					×	×
(Sh,*f*5)		×		×				×	×	
(Tr,$\bot$)		×	×		×	×				
($\bot$,*zo*)							×		×	

Table 4: Semi-join context of Example 2 RCF

To avoid ambiguity, we consider the derivations in the K_1 and K_2 contexts always denoted x', while the derivation in the join context is denoted x^{∇} (and the double derivation $x^{\nabla\nabla}$).

We are first interested in describing a formal concept of the joined context. Let $X \subseteq O_{\bowtie}$ be a set of objects. So, for all $(o_1, o_2) \in X$ we have $o_1 \in O_1 \cup \{\bot\}$ and $o_2 \in O_2 \cup \{\bot\}$. Thus, by definition, $C = (X^{\nabla\nabla}, X^{\nabla})$ is a formal concept. Let us now take the projections on the first and second elements of the pairs of $X^{\nabla\nabla}$, i.e. $\pi_1 = \{o_1 \mid \exists o_2, (o_1, o_2) \in X^{\nabla\nabla}\}$ and $\pi_2 = \{o_2 \mid \exists o_1, (o_1, o_2) \in X^{\nabla\nabla}\}$. We start by defining $X^{\nabla\nabla}$ in terms of these projections.

Lemma 1 *We have* $X^{\nabla\nabla} = (\pi_1 \times \pi_2) \cap O_{\bowtie}$

Proof. Let $(u, v) \in X^{\nabla\nabla}$. By definition $X^{\nabla\nabla} \subseteq O_{\bowtie}$, so $(u, v) \in O_{\bowtie}$. Moreover, by construction $u \in \pi_1$ and $v \in \pi_2$ so $(u, v) \in \pi_1 \times \pi_2$. Thus, $X^{\nabla\nabla} \subseteq (\pi_1 \times \pi_2) \cap O_{\bowtie}$.

Let $(u, v) \in (\pi_1 \times \pi_2) \cap O_{\bowtie}$. Since $(u, v) \in (\pi_1 \times \pi_2)$, it exists $\tilde{u}$ and $\tilde{v}$ s.t. $(u, \tilde{u}) \in X^{\nabla\nabla}$ and $(\tilde{v}, v) \in X^{\nabla\nabla}$. But, by construction $\{(u, \tilde{u}), (\tilde{v}, v)\}^{\nabla} \subseteq u' \cup v'$. And, since $(u, v) \in O_{\bowtie}$ we can write $(u, v)^{\nabla} = u' \cup v'$. Thus, by derivation property we have $X^{\nabla\nabla\nabla} \subseteq \{(u, \tilde{u}), (\tilde{v}, v)\}^{\nabla}$, by transitivity $X^{\nabla\nabla\nabla} \subseteq (u, v)^{\nabla}$. Thus, by deriving this expression we obtain $(u, v)^{\nabla\nabla} \subseteq X^{\nabla\nabla\nabla\nabla}$. Finally, as $(u, v) \in (u, v)^{\nabla\nabla}$ and $X^{\nabla\nabla\nabla\nabla} = X^{\nabla\nabla}$ we have $(u, v) \in X^{\nabla\nabla}$. □

We first study the particular cases containing the object $\perp$ by starting with the case where this element appears in both projections.

Proposition 1 *If* $\perp \in \pi_1$ *and* $\perp \in \pi_2$ *then* $X^{\nabla} = \emptyset$ *and* $X^{\nabla\nabla} = O_{\bowtie}$

Proof. Suppose $\perp \in \pi_1$ and $\perp \in \pi_2$ then by definition of $\perp$ we have $X^{\nabla} \cap A_1 = \emptyset$ and $X^{\nabla} \cap A_2 = \emptyset$ and therefore $X^{\nabla} = \emptyset$. By definition of the derivation we have $\emptyset^{\nabla} = O_{\bowtie}$ therefore $X^{\nabla\nabla} = O_{\bowtie}$.The second assertion holds by symmetry. □

In the case described by the lemma 1, it is immediate to show that we can construct $(X^{\nabla\nabla}, X^{\nabla})$. We show that the same is true when only one of the components π_1 or π_2 contains $\perp$ by first describing X^{∇} then $X^{\nabla\nabla}$ in the lemmas 2 and 3.

Lemma 2 *If* $\perp \in \pi_1$ *and* $\perp \notin \pi_2$, $X^{\nabla} = \pi_2'$. *If* $\perp \in \pi_2$ *and* $\perp \notin \pi_1$, $X^{\nabla} = \pi_1'$.

Proof. Let us suppose $\perp \in \pi_1$ and $\perp \notin \pi_2$. We have $a \in X^{\nabla}$ iff $X^{\nabla\nabla} \subseteq a^{\nabla}$. Yet, since $\perp \in \pi_1$ we have construction $X^{\nabla} \cap A_1 = \emptyset$. thus, we have $a \in A_2$.therefore, we have $a \in X^{\nabla}$ iff for all $(o_1, o_2) \in X^{\nabla\nabla}$ o_2 carries the attribute a, i.e. $a \in \pi_2'$. Since we have $a \in X^{\nabla}$ iff $a \in \pi_2'$, we have $X^{\nabla} = \pi_2'$. We show the second assertion symmetrically. □

Lemma 3 *If* $\perp \in \pi_1$ *and* $\perp \notin \pi_2$, $X^{\nabla\nabla} = (O_1 \cup \{\perp\} \times \pi_2) \cap O_{\bowtie}$. *If* $\perp \in \pi_2$ *and* $\perp \notin \pi_1$, $X^{\nabla\nabla} = (\pi_1 \times O_2 \cup \{\perp\}) \cap O_{\bowtie}$.

Proof. Let us suppose $\perp \in \pi_1$ and $\perp \notin \pi_2$. Let $v \in \pi_2$. Any pair $(u, v) \in O_{\bowtie}$ verifies $\pi_2' \subseteq (u, v)^{\nabla}$. Since, by the lemma 2, we have $X^{\nabla} = \pi_2'$, we can write $X^{\nabla} \subseteq (u, v)^{\nabla}$ and thus, by derivation $(u, v)^{\nabla\nabla} \subseteq X^{\nabla\nabla}$. Finally, for any $u \in O_1 \cup \{\perp\}$, we have $(u, v) \in X^{\nabla\nabla}$ donc $X^{\nabla\nabla} = (O_1 \cup \{\perp\} \times \pi_2) \cap O_{\bowtie}$. We show the second assertion symmetrically. □

The lemmas 2 and 3 allow us to determine that in cases where only one of the projections contains $\perp$ we can write a formal concept of $K_{\bowtie}$ only with the other projection. Let us now study a formal concept based on this projection determined by RCA.

Lemma 4 *If $\perp \in \pi_1$ and $\perp \notin \pi_2$, there exists a concept $C_2 = (\pi_2, \pi_2')$ on K_2. If $\perp \in \pi_2$ and $\perp \notin \pi_1$, there exists a concept $C_1 = (\pi_1, \pi_1')$ on K_1.*

Proof. Let us suppose $\perp \in \pi_1$ and $\perp \notin \pi_2$. Since $\pi_2 \subseteq O_2$, (π_2'', π_2') is a concept son K_2. It is therefore sufficient to show that $\pi_2'' = \pi_2$, or more simply $\pi_2'' \subseteq \pi_2$. Let $o \in \pi_2''$. By construction at least one couple $(\bar{o}, o) \in O_{\bowtie}$ and $o' \subseteq (\bar{o}, o)^{\nabla}$. Now, we have $o \in \pi_2''$ so by derivation, $\pi_2' \subseteq o'$. Moreover, by the lemma 2, we have $X^{\nabla} = \pi_2'$. Thus, $X^{\nabla} \subseteq (\bar{o}, o)^{\nabla}$ so by derivation, $(\bar{o}, o) \in X^{\nabla\nabla}$. Finally, by definition of the projections $o \in \pi_2$. The second assertion holds by symmetry. □

The following proposition gathers the previous lemmas. It emphasizes that, in the case where only one of the two projections contains $\perp$, any concept of $K_{\bowtie}$ can be expressed with the other projection. Moreover, there exists a concept generated by RCA, of the same intent and whose extent corresponds to a projection of the extent of the concept generated by FCA.

Proposition 2 *Let $C = (X^{\nabla\nabla}, X^{\nabla})$. If $\perp \in \pi_1$ and $\perp \notin \pi_2$, $C = ((O_1 \cup \{\perp\} \times \pi_2) \cap O_{\bowtie}, \pi_2')$ and there exists a corresponding concept $C_2 = (\pi_2, \pi_2')$ on K_2. If $\perp \in \pi_2$ and $\perp \notin \pi_1$, $C = ((\pi_1 \times O_2 \cup \{\perp\}) \cap O_{\bowtie}, \pi_1')$ and there exists a corresponding concept $C_1 = (\pi_1, \pi_1')$ on K_1.*

Proof. Follows from the lemmas 2, 3 and 4. □

There remains a specific case, described by the lemma 5, to complete the exhaustive description of a formal concept on the join table.

Lemma 5 *For any $X \subseteq O_1 \times O_2$ we have $X^{\nabla} = \pi_1' \cup \pi_2'$ and $X^{\nabla\nabla} = \{(o_1, o_2) \mid \pi_1' \subseteq o_1' \wedge \pi_2' \subseteq o_2'\}$*

Proof. Let us show $X^{\nabla} = \pi_1' \cup \pi_2'$ by double inclusion.

(i) $X^{\nabla} \subseteq \pi_1' \cup \pi_2'$.

The RCF modeling assures us that $A_1 \cap A_2 = \emptyset$. Thus, an attribute $a \in X^{\nabla}$ is either in A_1 or in A_2. If $a \in X^{\nabla} \cap A_1$, it must be shared by all the elements of π_1; and so a is in π_1'. Similarly, if a is in $X^{\nabla} \cap A_2$, $a \in \pi_2'$. We deduce that $X^{\nabla} \subseteq \pi_1' \cup \pi_2'$.

(ii) $\pi_1' \cup \pi_2' \subseteq X^{\nabla}$.

On the other hand, if an attribute a is in π_1', then any pair of $X^{\nabla\nabla}$ has a first component that carries the attribute a. Since this property is true for any pair of $X^{\nabla\nabla}$ and $X \subseteq X^{\nabla\nabla}$, then any pair of X carries the attribute a. Therefore, we have $a \in X^{\nabla}$. In the same way, we show that if $a \in \pi_2'$, then $a \in X^{\nabla}$. thus, we have $\pi_1' \subseteq X^{\nabla}$ and $\pi_2' \subseteq X^{\nabla}$. We can therefore affirm that $\pi_1' \cup \pi_2' \subseteq X^{\nabla}$.

Finally, by (i) and (ii) we have $X^{\nabla} = \pi_1' \cup \pi_2'$. As $X^{\nabla\nabla}$ describes exactly the set of couples (o_1, o_2) having the attributes of $\pi_1' \cup \pi_2'$, by construction of the join table we have $\pi_1' \subseteq o_1'$ and $\pi_2' \subseteq o_2'$. □

The cases described by the lemmas 1 and 2 allow for the immediate selection of concepts from the RCA process corresponding in terms of extent to a concept in the join table. The proposition 3 relies on the lemma 5 to state the main result of this subsection, dealing with non-degenerate cases (without $\perp$ element).

Proposition 3 *Let $X \subseteq O_1 \times O_2$. There exists by RCA on K_1 a concept (X_1, Y_1) such that $X_1 = \pi_1$ and $\pi_1' \subseteq Y_1$ and there exists on K_2 a concept (X_2, Y_2) such that $X_2 = \pi_2$ and $\pi_2' \subseteq Y_2$.*

Proof. As $\pi_1' \subseteq A_1$ and $\pi_2' \subseteq A_2$, $C_1 = (\pi_1'', \pi_1')$ and $C_2 = (\pi_2'', \pi_2')$ are formal concepts on their respective contexts computed at step 0 of RCA.

Let us consider the contexts K_1 and K_2 after graduation by the operator $\exists$ on the relations r and r^{-1}. We then have the attributes $\exists r : C_2$ in K_1 and $\exists r^{-1} : C_1$ in K_2. We define the sets of attributes $Y_1 = \pi_1' \cup \{\exists r : C_2\}$ and $Y_2 = \pi_2' \cup \{\exists r^{-1} : C_1\}$ as well as the concepts $C_3 = (Y_1', Y_1'')$ and $C_4 = (Y_2', Y_2'')$ (it is possible that $C_1 = C_3$ or $C_2 = C_4$). We have $Y_1' = \pi_1'' \cap \{\exists r : C_2\}'$, let us show that $Y_1' = \pi_1$ by double inclusion.

Let $o \in \pi_1$, we have $o \in \pi_1''$. Moreover, by construction, any pair of $(o, \bar{o}) \in X^{\nabla\nabla}$ verifies $(o, \bar{o}) \in r$ with $\bar{o} \in \pi_2$ and, by hypothesis, $\bar{o} \neq \perp$. Thus, since $\pi_2 \subseteq \pi_2''$, o carries the attribute $\exists r : C_2$. Thus, we have $\pi_1 \subseteq Y_1'$.

Let $o \in Y_1'$. We have $\pi_1' \cup \{\exists r : C_2\} \subseteq o'$. Since $\{\exists r : C_2\} \subseteq o'$, there exists $\bar{o} \in \pi_2''$ such that $(o, \bar{o}) \in r$ and thus $(o, \bar{o}) \in O_{\bowtie}$. Moreover, since $\bar{o} \in \pi_2''$, we have $\pi_2' \subseteq \bar{o}'$. Since $\pi_2' \subseteq \bar{o}'$, $\pi_1' \subseteq o'$ and that by the lemma 5 we have $X^{\nabla} = \pi_1' \cup \pi_2'$, we can affirm $X^{\nabla} \subseteq (o, \bar{o})^{\nabla}$. Finally $(o, \bar{o})^{\nabla\nabla} \subseteq X^{\nabla\nabla}$ and by definition of π_1, on a $o \in \pi_1$ (In a completely analogous way, we show $\pi_2 = Y_2'$).

Finally, we have shown the existence of $C_3 = (Y_1', Y_1'')$ such that $Y_1' = \pi_1$ and $\pi_1 \subseteq Y_1''$ as well as of $C_4 = (Y_2', Y_2'')$ such that $Y_2' = \pi_2$ and $\pi_2 \subseteq Y_2''$. □

In conclusion, the propositions 1, 2 and 3 show that for any concept $C = (X^{\nabla\nabla}, X^{\nabla})$ we find on K_1 a concept (X_1, Y_1) such that $X_1 = \pi_1$ and $\pi_1' \subseteq Y_1$ and there exists on K_2 a concept (X_2, Y_2) such that $X_2 = \pi_2$ and $\pi_2' \subseteq Y_2$. It is to note that if $\perp \in \pi_1$ (respectively π_2) we have $\pi_1 = \{x \mid \exists y, (x, y) \in O_{\bowtie}\}$ (respectively $\pi_2 = \{x \mid \exists x, (x, y) \in O_{\bowtie}\}$). Example 3 illustrates these properties.

Example 3. Let us consider the relational family as well as the semi-join context defined in the Example 2.

On the joined context, we find the concept $C = (\{(La, t3), (La, f5)\}, \{\text{Adult}, \text{Female}, \text{IT}, pw\})$. Here, $\pi_1 = \{La\}$ and $\pi_2 = \{t3, f5\}$. We check that there exists on K_P a concept (π_1, π_1'), namely the concept $C_1 = (\{La\}, \{Adult, Female, IT\})$, and on K_C a concept (π_2, π_2'), the concept $C_2 = (\{t3, f5\}, \{pw\})$. After an iteration, RCA extends these concepts' intents to $\{Adult, Female, IT, \exists pos : C_2\}$ and $\{pw, \exists pos^{-1} : C_1\}$, respectively.

3.3 Aggregation operation in mono-relational case

Assume again the RCF in Figure 2 and let us consider FCA is applied on the context schematically visualized in Figure 4. Intuitively, this amounts to extending the domain context of the relation by appending some new attributes. These

are derived from the range context attributes by a technique akin to relational scaling, i.e. one basically simulating a one-shot RCA-like context refinement.

Formally speaking, we design the context $K_{\lessdot} = (O_{\lessdot}, A_{\lessdot}, I_{\lessdot})$ where $O_{\lessdot} = O_1$, $A_{\lessdot} = A_1 \cup \rho(A_2)$, and $\rho(A_2)$ are the attributes resulting from the application of the scaling operator ρ to the attribute concept of $a \in A_2$. We will denote such an attribute $\overline{\rho r : a}$ to avoid confusion with RCA's own relational attributes. Notice that $A_1 \cap \rho(A_2) = \emptyset$ holds. Next, we introduce $Constraint(\rho, r, o_p, (X, Y))$, a predicate verifying whether o_p and (X, Y), from the domain and the range of r, respectively, jointly comply to the semantic of ρ. Thus, $Constraint(\forall, r, o_p, (X, Y))$ is *true* iff $r(o_p) \subseteq X$. The predicate is a compact expression of the incidence $I_{\lessdot}$:

- if $a_p \in A_1$ then $(o_p, a_p) \in I_{\lessdot}$ iff $(o_p, a_p) \in I_1$,
- if $a_p \in \rho(A_2)$ then $(o_p, a_p) \in I_{\lessdot}$ iff $Constraint(\rho, r, o_p, (a'_p, a''_p))$ is true.

Example 4 illustrates the $\forall\exists$ case (reasoning with other operators is similar). For the O_2 perspective, it is enough to swap K_1 and K_2 and replace r by r^{-1}.

Example 4. Consider again the RCF in Example 2. We aggregate the family via $\forall\exists$: For an $o \in O_P$ and $a \in A_C$ s.t. $\overline{\forall\exists pos : a} \in \rho(A_C)$ it holds $(o, \overline{\forall\exists pos : a}) \in I$ iff 1) $\exists o_C \in O_C$ s.t. $(o, o_C) \in pos$ and 2) $\forall o_C \in O_C$, $(o, o_C) \in pos$ entails $o_v \in a'$ (there is at least one image of o by pos and all such images carry a. Table 5 depicts the resulting aggregated context $K_{\lessdot}$.

$K_{\lessdot}$	Senior	Adult	Male	Female	I.T.	Sport	$\overline{\forall\exists pos : el}$	$\overline{\forall\exists pos : pw}$	$\overline{\forall\exists pos : cp}$	$\overline{\forall\exists pos : ch}$
Fa	×		×						×	×
La		×		×	×			×		
Sh		×		×					×	
Tr		×	×		×	×				

Table 5: $K_{\lessdot}$ for Example 4

$K_{\lessdot}$	Senior	Adult	Male	Female	I.T.	Sport	$\overline{\exists pos : el}$	$\overline{\exists pos : pw}$	$\overline{\exists pos : cp}$	$\overline{\exists pos : ch}$
Fa	×		×						×	×
La		×		×	×		×	×	×	
Sh		×		×				×	×	×
Tr		×	×		×	×				

Table 6: $K_{\lessdot}$ for Example 6

To define a formal concept on the aggregated table, we first identify the component of the intent on the part $\rho(A_2)$. Again, we denote the derivations in K_1 and K_2 by $'$, and in the aggregated context by $^{\nabla}$.

Definition 1. *The* relational deviation *of $X \subseteq O_{\lessdot}$, denoted $\delta(X)$, is the set of its attributes from $\rho(A_2)$, i.e. $\delta(X) = X^{\nabla} \cap \rho(A_2)$.*

Proposition 4 *Given a $X \subseteq O_{\lessdot}$, $\delta(X) = \cap_{o \in X} \{\overline{\rho r : a} \mid Constraint(\rho, r, o, (a', a''))\}$.*

Proof. Let $o \in X$. By construction, for any $\overline{\rho r : a} \in \rho(A_2)$, holds $\overline{\rho r : a} \in o^{\nabla}$ iff $Constraint(\rho, r, o, (a', a''))$. Thus $o^{\nabla} \cap \rho(A_2) = \{\overline{\rho r : a} \mid Constraint(\rho, r, o, (a', a''))\}$. As $X' = \bigcap_{o \in X} o'$, we have $X' \cap \rho(A_2) = \bigcap_{o \in X} o^{\nabla} \cap \rho(A_2)$, hence $\delta(X) = \cap_{o \in X} \{\overline{\rho r : a} \mid Constraint(\rho, r, o, (a', a''))\}$. □

A formal concept on the aggregated context is then characterized by:

Proposition 5 *Let $X \subseteq O_{\lessdot}$, then the concept $C = (X^{\nabla\nabla}, X^{\nabla})$ of the aggregated context satisfies $X^{\nabla} = X' \cup \delta(X)$ and $X^{\nabla\nabla} = X'' \cap \{o \mid \delta(X) \subseteq o^{\nabla}\}$.*

Proof. By definition, we have $A_{\lessdot} = A_1 \cup \rho(A_2)$ and $A_1 \cap \rho(A_2) = \emptyset$. Thus we can write $X^{\nabla} = (X^{\nabla} \cap A_1) \cup (X^{\nabla} \cap \rho(A_2))$, that is $X^{\nabla} = X' \cup \delta(X)$.

By deriving X^{∇}, we determine that $X^{\nabla\nabla} = \{o \mid o \in O_1 \wedge X' \subseteq o^{\nabla} \wedge \delta(X) \subseteq o^{\nabla}\}$. Now, as $X' \subseteq A_1$, we have $X' \subseteq o^{\nabla}$ iff $X' \subseteq o^{\nabla} \cap A_1$, that is $X' \subseteq o'$. Finally, $X^{\nabla\nabla} = X'' \cap \{o \mid \delta(X) \subseteq o^{\nabla}\}$. □

Now, let us assume we have the result of RCA using the same relational scaling with ρ along r on the simple RCF in Figure 2. Let X be the extent of a concept from $K_{\lessdot}$. The set $\delta(X)$ is well-defined, hence we can denote its i-th member by $\overline{\rho r : a_{\delta,i}}$ (where $a_{\delta,i} \in A_2$). As every concept $C_{\delta(X),i} = (a'_{\delta,i}, a''_{\delta,i})$ is well defined on K_2, in RCA, K_1 will be refined with all the attributes $\rho r : C_{\delta(X),i}$ at the first relational scaling step. Let $Y_\delta = X' \cup_{i \in 1..|\delta(X)|} \rho r : C_{\delta(X),i}$, we claim that $C = (X^{\nabla\nabla}, X^{\nabla})$ and $C_\delta = (Y'_\delta, Y''_\delta)$ have the same extent:

Proposition 6 $Y'_\delta = X^{\nabla\nabla}$.

Proof. $Y'_\delta \subseteq X^{\nabla\nabla}$: Let $o \in Y'_\delta$. First, o carries all the attributes of X', thus $X' \subseteq o^{\nabla}$. Moreover, for each attribute $\overline{\rho r : a_{\delta,i}} \in \delta(X)$ a concept $C_i = (a'_{\delta,i}, a''_{\delta,i})$ exists such that o carries the attribute $\rho r : C_i$ (for which $Constraint(\rho, r, o, C_i)$ is true). Since $a_{\delta,i}$ is in the intent of C_i, we can verify that $\overline{\rho r : a_{\delta,i}} \in o^{\nabla}$. Since for all i, we have $\overline{\rho r : a_{\delta,i}} \in o^{\nabla}$, then we have $\delta(X) \subseteq o^{\nabla}$. Finally, since $\delta(X) \subseteq o^{\nabla}$ and $X' \subseteq o^{\nabla}$, we have $X^{\nabla} \subseteq o^{\nabla}$. By derivation, we have $o^{\nabla\nabla} \subseteq X^{\nabla\nabla}$. Finally, $Y'_\delta \subseteq X^{\nabla\nabla}$. □

$Y'_\delta \supseteq X^{\nabla\nabla}$: The 1st relational scaling step will necessarily produce $\rho r : (a'_{\delta,i}, a''_{\delta,i})$ for each $\overline{\rho r : a_{\delta,i}} \in \delta(X)$. Let $o \in X^{\nabla\nabla}$, then o carries all the attributes of X'. Moreover, after the scaling step, o gets incident to each attribute $\overline{\rho r : a_{\delta,i}} \in \delta(X)$ ($Constraint(\rho, r, o, (a'_{\delta,i}, a''_{\delta,i}))$ is necessarily satisfied). Thus, we have $Y_\delta \subseteq o'$ and therefore $o'' \subseteq Y'_\delta$. Finally, since $o \in o''$ we conclude that $X^{\nabla\nabla} \subseteq Y'_\delta$. □

Proposition 6 states that for any concept from the aggregated context, an RCA concept with the same extent exists. Definition 2 introduces the notion of relational weakening (illustrated by Example 5) to enable the mapping between both intents. The latter is given by proposition 7.

Definition 2. *Let a concept C be produced by RCA and let Y_r be the set of relational attributes of the intent of C. We call* relational weakening *of C, noted $\Omega(C)$, the set $\Omega(C) = \bigcup_{\rho r:(U,V) \in Y_r} \{\overline{\rho r : v} \mid v \in V\}$*

Example 5. Assume the contexts of Example 2: Context K_C gives rise to the concepts $C_1 = (\{t3\}, \{el, pw\})$, $C_2 = (\{f5\}, \{pw, cp\})$, $C_3 = (\{t3, zo\}, \{el\})$ and $C_4 = (\{zo, f5\}, \{cp\})$. After a scaling with $\exists$, K_P yields the concept $C = (\{La\}, \{Adult, Female, I.T., \exists pos : C_1, \exists pos : C_2, \exists pos : C_3, \exists pos : C_4, \exists pos : \top\})$. Then $\Omega(C) = \{\exists pos : el, \exists pos : pu, \exists pos : cp\}$.

Proposition 7 $\delta(X) \subseteq \Omega(C_\delta)$

Proof. Let's denote by Y_r the set of relational attributes in the intent of C_δ. Let $\overline{\rho r : a} \in \delta(X)$, then by scaling and construction of C_δ, it holds $\rho r : (a', a'') \in Y_r$ and as $a \in a''$, one concludes $\delta(X) \subseteq \Omega(C_\delta)$. □

While we've just shown that the extents of C and C_δ are equal, their intents might differ: As proposition 7 states, the intent of the aggregate table concept is a subset of the weakening of the RCA concept with the same extent (see Example 6 below). In this sense, we see the RCA concept as *more informative*.

Example 6. Assume the relational family defined by Example 2 with the $\exists$ operator. The aggregated context is presented in table 6. Now, after one iteration, RCA discovers the concept $(\{Fa\}, \{Senior, Male, \exists pos : (cp, ch)\})$ whereas FCA finds $(\{Fa\}, \{Senior, Male, \exists pos : (cp), \exists pos : (nd))\})$. While $\exists pos : (cp, ch)$ implies $\exists pos : (cp)$ and $\exists pos : (ch)$, the reverse does not hold.

4 Discussion

We've shown that for any FCA concept from an encoded context, RCA would reveal a counterpart concept, or a pair of such, conveying the same semantics (equal extent). Moreover, the syntactic expression of the RCA concept(s) is clearer than the FCA one, whatever the encoding. With semi-join, since separate RCA concepts map to the 1st and 2nd projections of a FCA concept, the clarity gain is immediate. Indeed, no confusion is ever possible as to which attribute of the semi-join intent is incident to which object. Moreover, redundancy in FCA concepts, e.g. shared 1st or 2nd projection, is avoided in RCA.

With aggregation, RCA trivially produces a concept of the same extent, yet it is more precise: The FCA counterpart is readily obtained by relational weakening. Here, higher-order encoding schemata are conceivable that mimic RCA iterations by nesting the scaling operators. Yet the maximal depth of these nestings in the resulting (pseudo-)relational attributes must be fixed beforehand. This is a serous limitation since we know of no simple way to determine the number of iterations required till the fixed point for a given RCA task, i.e. a RCF and a vector of scaling operators. This means that, at least in the realistic cases, RCA will be revealing concepts that FCA –over the aggregated context with all possible nestings of a depth up till the limit– will miss. A relevant question here is whether knowing the fixed point contexts in RCA, there is an equivalent aggregation context that comprises only nested operator attributes referring exclusively to attribute concepts. This would mean that a static encoding, i.e. without the need for explicitly composing RCA concepts popping at iterations 2+, exists. The cost of constructing such an encoding is, though, a separate concern.

5 Conclusion

We tackled here the question of whether RCA brings some effective scope extension to the realm of FCA, given that FCA is at its core. We've examined two complementary principles of encoding a relation into a single augmented context and compared FCA output on each of the contexts to the output of RCA on the original RCF. It was shown that in both cases, RCA is able to find counterpart concepts (same extent) to those found by FCA, while the RCA intents at its 1st iteration are at least as expressive as the FCA ones.

A more systematic study should allow us to demonstrate similar results in the more complex cases of multiple relations in the RCF as well as multiple relations between the same pair of contexts.

References

1. M. Alam et al. Lattice based data access: An approach for organizing and accessing linked open data in biology. In *DMoLD@ECML/PKDD*. Springer, 2013.
2. P. Chen. The Entity-Relationship Model - Toward a Unified View of Data. *ACM Transactions on Database Systems*, 1(1):9–36, 1976.
3. X. Dolques et al. RCA as a data transforming method: a comparison with propositionalisation. In *ICFCA*, pages 112–127. Springer, 2014.
4. X. Dolques et al. Performance-friendly rule extraction in large water data-sets with aoc posets and relational concept analysis. *Intl. J-l of General Syst.*, 2016.
5. S. Džeroski. Multi-relational data mining: an introduction. *ACM SIGKDD Explorations Newsletter*, 5(1):1–16, 2003.
6. S. Ferré and P. Cellier. Graph-fca: An extension of formal concept analysis to knowledge graphs. *Discrete Applied Mathematics*, 273:81–102, 2020.
7. S. Ferré and O. Ridoux. A logical generalization of formal concept analysis. In *ICCS*, pages 371–384. Springer, 2000.
8. B. Ganter and S. Kuznetsov. Pattern structures and their projections. In *ICCS*, pages 129–142, 2001.
9. S. Kramer et al. Propositionalization approaches to relational data mining. In *Relational Data Mining*, pages 262–291. Springer, 2001.
10. M. Rouane-Hacene et al. Relational concept analysis: mining concept lattices from multi-relational data. *AMAI*, 67(1):81–108, 2013.
11. E. Spyropoulou and T. De Bie. Interesting multi-relational patterns. In *ICDM*, pages 675–684. IEEE, 2011.
12. M. Wajnberg et al. Mining process factor causality links with multi-relational associations. In *K-Cap*, pages 263–266, 2019.
13. M. Wajnberg et al. Mining heterogeneous associations from pediatric cancer data by relational concept analysis. In *MSDM@ICDM*. IEEE, 2020.
14. R. Wille. *Restructuring Lattice Theory: An Approach Based on Hierarchies of Concepts*, pages 445–470. Springer Netherlands, Dordrecht, 1982.
15. R. Wille. Conceptual graphs and formal concept analysis. In *ICCS*, pages 290–303. Springer, 1997.

Likely-occurring itemsets for pattern mining

Tatiana Makhalova, Sergei O. Kuznetsov, and Amedeo Napoli

[1] Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France
tatiana.makhalova@inria.fr
[2] National Research University Higher School of Economics, Moscow, Russia
skuznetsov@hse.ru
[3] Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France
amedeo.napoli@loria.fr

Abstract. We consider the itemset mining problem in general settings, e.g., mining association rules and itemset selection. We introduce the notion of likely-occurring itemsets and propose a greedy approach to itemset search space discovery that allows for reducing the number of arbitrary or closed itemsets. This method provides itemsets that are useful for different objectives and can be used as an additional constraint to curb the itemset explosion. In experiments, we show that the method is useful both for compression-based itemset mining and for computing good-quality association rules.

1 Introduction

A generic objective of itemset mining is to discover a small set of non-redundant and interesting itemsets that describe together a large portion of data and that can be easily interpreted [1].

Itemset mining can be summarized into two steps: (i) discovering itemset search space and (ii) selecting interesting itemsets among the discovered ones.

This paper is devoted to the first step, i.e., the itemset search space discovery. Since the itemset search space contains exponentially many elements, it is important to discover as few useless itemsets as possible.

There are several approaches to discover the itemset search space: (i) an exhaustive enumeration of all itemsets followed by a selection of those satisfying imposed constraints [19], (ii) a gradual enumeration of some itemsets guided by an objective (or by constraints) [17], (iii) mining top-k itemsets w.r.t. constraints [15], (iv) sampling a subset of itemsets w.r.t. a probability distribution that conforms to an interestingness measure [6,7]. To reduce redundancy when enumerating itemsets, the search space can be shrunk to *closed itemsets*, i.e., the maximal itemsets among those that are associated with a given set of objects (support).

The exhaustive enumeration is the most universal way to discover itemset search space. There exists a lot of very efficient algorithms for its enumeration, e.g., CbO [12], In-Close [3], LCM [18], Alpine [11], and others [13].

Despite its wide usage and applicability for a large spectrum of interestingness measures, the exhaustive enumerators usually mine itemsets w.r.t. frequency, which results in the following issues: using too high frequency threshold results in a considerable amount of not interesting itemsets, while too low frequency threshold results in itemset explosion and intractability of itemset mining methods in practice.

However, considering the itemset mining problem in more general settings, e.g., mining association rules and implications, the exhaustive enumeration of frequent itemsets is usually the only (universal) remedy for the pattern explosion problem.

In this paper, we revisit the notion of *likely-occurring* itemsets introduced in [14] and propose a greedy approach to itemset search space discovery that allows for reducing the number of closed itemsets. This method provides itemsets that are useful for different objectives and can be used as an additional constraint to curb the itemset explosion. In experiments we show that the method is useful both for compression-based itemset mining and for computing good-quality association rules.

2 Preliminaries

We deal with binary datasets within the FCA framework [10].

A *formal context* is a triple $\mathbb{K} = (G, M, I)$, where G is a set of objects, M is a set of attributes and $I \subseteq G \times M$ is the incidence relation, i.e., $(g, m) \in I$ if object g has attribute m.

Two derivation operators $(\cdot)'$ are defined for $A \subseteq G$ and $B \subseteq M$ as follows:

$$A' = \{m \in M \mid \forall g \in A : gIm\}\,,\ B' = \{g \in G \mid \forall m \in B : gIm\}\,.$$

For $A \subseteq G$, $B \subseteq M$, a pair (A, B) such that $A' = B$ and $B' = A$, is called a *formal concept*, then A and B are closed sets and called *extent* and *intent* (or *closed itemsets*), respectively.

The (empirical or observed) probability of an itemset $X \subseteq M$ is given by $P(X) = fr(X) = |X'|/|G|$.

3 Likely-occurring itemsets

To reduce the itemset search space, we propose an additional constraint that consists in considering only the itemsets whose observed probability is greater than the estimated one. The estimated probability is computed under the independence model. We give the details on the chosen independence model below.

Definition 1. *A closed itemset $X \subseteq M$ is called* likely-occurring closed *(LOC) if there exists $m \in X$ and $Y \subseteq X \setminus \{m\}$, $(Y \cup \{m\})'' = X$ such that $P(X) > Q \cdot P(Y) \cdot P(\{m\})$, and $Q \geq 1$.*

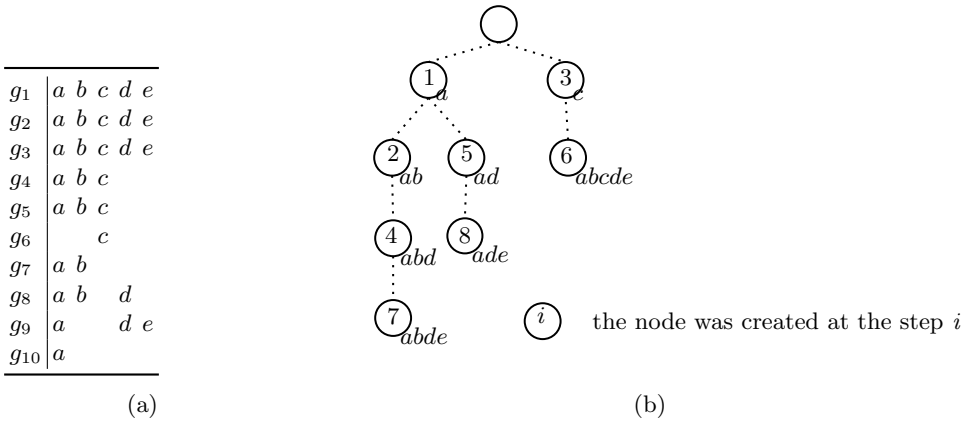


Fig. 1: A binary dataset and the execution tree of Algorithm 1 for this dataset

The empty itemset $\emptyset$ is considered to be likely-occurring by default. The parameter Q controls how large the difference between the observed probability $P(X)$ and the estimated probability $P(Y) \cdot P(\{m\})$, $Y \subseteq X \setminus \{m\}$ of the itemset X may be. The least restrictive constraint, i.e., $Q = 1$, requires the observed probability to be greater than the estimated one. The larger values of Q are more restrictive, i.e., they require the observed probability to be much larger than the estimated one.

According to the definition above, at most $|X|$ splittings should be enumerated to check whether an itemset X is LOC or not. To make it more tractable in practice, we propose a relaxation of the LOC itemset and a greedy approach for its computing, where one needs to check only one splitting per itemset. Let us proceed to this definition.

Definition 2. *Let $\{m_1, m_2, \cdots, m_k\}$ be a set of attributes arranged in order of decreasing frequency, i.e., $fr(m_i) \geq fr(m_j)$ for any $i \leq j$. A closed itemset X is* likely-occurring closed *(LOC) if there exists a LOC itemset $Y \subset X$ and $m \in X \setminus Y$ such that $fr(m) \geq \min_{m^* \in Y} fr(m^*)$, $X = (Y \cup \{m\})''$ and $P(X) > Q \cdot P(Y) \cdot P(\{m\})$.*

Example. Let us consider a running example from Fig. 1a, where the attributes are arranged by decreasing frequency. Itemset ab is an LOC itemset because a is an LOC itemset and $P(ab) > P(a) \cdot P(\{b\})$, the same for abd, namely, abd is an LOC itemset because ab is an LOC itemset and $P(abd) > P(ab) \cdot P(\{d\})$, etc.

We propose an algorithm to compute LOC itemsets using Definition 2, its pseudocode is given in Algorithm 1. This algorithm computes gradually LOC itemsets by considering one by one attributes of decreasing frequency. Apart from the threshold Q on the difference in probabilities, the algorithm also supports threshold F on frequency. By default, we use minimal restrictions, namely $Q = 1$ (we require the observed probability to be greater than the estimated one) and $F = 0$ (we do not impose any frequency constraints).

Algorithm 1 ComputeLOC

Procedure Merge (*node*,*candidate*)

Input: *node*, current node
candidate, candidate node

1: $I_n \leftarrow node.itemset$
2: $I_c \leftarrow candidate.itemset$
3: **if** $|I_n \setminus I_c| > 0$ **then**
4: $extent \leftarrow (I_c \cup I_n)'$ {computing shared objects}
5: **if** $extent = I_c'$ **then**
6: $I_n \leftarrow I_n \cup I_c$ {if I_n is included into all objects as I_n, just extend I_c}
7: **else if** $\left(\frac{|extent|}{|G|} > Q \cdot \frac{|I_c'|}{|G|}\frac{|I_n'|}{|G|}\right)$ **and** $|extent| \geq F$ **then**
8: **for all** $child \in node.children$ **do**
9: $merge(child, candidate)$
10: **end for**
11: **if** $candidate \notin \mathcal{T}$ **then**
12: $node.children.add(candidate)$
13: **end if**
14: **end if**
15: **end if**

Input: (G, M, I) formal context
F, frequency threshold
Q, threshold on LOC
Output: $\mathcal{T}$, a tree composed of LO/LOC-itemsets

1: $\mathcal{T} \leftarrow createEmptyTree()$
2: $root \leftarrow \mathcal{T}.getRoot()$
3: $M^* \leftarrow sortByDescendingFrequency(M)$
4: **for all** $m \in M^*$ **do**
5: $candidate \leftarrow m''$ {for LOC itemsets}
6: **for all** $child \in root.children$ **do**
7: $merge(child, candidate)$
8: **if** $candidate \notin \mathcal{T}$ **then**
9: $root.children.add(candidate)$
10: **end if**
11: **end for**
12: **end for**
13: **return** $\mathcal{T}$

Example. Let us consider the execution tree of the algorithm for a dataset from Fig. 1a. The algorithm starts constructing a tree adding the attributes of decreasing frequency. The order in which itemsets are enumerated is specified in the corresponding nodes.

4 Likely-occurring itemsets and related notions

Probability-based models are common in itemset and association rule mining. In this section we consider two widespread approaches to assess itemsets and

association rules, and discuss how they are related to likely-occurring closed itemsets.

Independence model and lift. The models based on the comparison of estimated and observed probabilities of itemsets are quite common in the scientific literature. The simplest model is the attribute independence model. Under this model, all items (attributes) are assumed to be independent. Attribute probability is approximated straightforwardly using the attribute frequency. Then, the probability of an itemset X is computed as follows:

$$P_{ind}(X) = \prod_{x \in X} P(x) = \prod_{x \in X} fr(x).$$

Despite its simplicity, this model is widely used in machine learning, e.g., Naïve Bayes classifiers are based on it. A natural extension of the attribute independence model is the partition independence model, where some partitions of X are assumed to be independent. Lift [8] is one of the most common measures to assess association rules under the partition independence model.

Definition 3. *Let $X \to Y$ be an association rule, then* lift *is given by*

$$lift(X \to Y) = \frac{P(XY)}{P(X)P(Y)} = \frac{fr(XY)}{fr(X)fr(Y)}.$$

Apart from lift, there is a lot of other measures (indices) based on the comparison of the antecedent and consequent supports, e.g., redundancy constraints [4,22], minimum improvement [5], etc. They are commonly used to select association rules.

The notion of lift can be also adapted in different ways for itemset assessment. For example, one may assess the probability of an itemset under the assumption that any partition of the itemset into two disjoint sets is independent. If the observed probability is greater than all the estimated probabilities obtained under this model, then the itemset is called *productive* [21].

The introduced above LOC itemsets, in a certain sense, represent a particular case of productive itemsets. Instead of considering all possible partitions of X into two sets of items, we consider only its proper subset Y and attribute $m \in X \setminus Y$. Reformulating the definition of LOC in terms of lift (for association rules), LOC itemset X is an itemset that consists of LOC itemset Y and attribute m such that $Y \cup \{m\}$ is the generator of X, and $lift(X \to m) > Q$, $Q \geq 1$. Since Y is also LOC, this reasoning can be done recursively.

If it is needed, one may reduce further the size of the discovered LOC itemsets by putting more tighter constraints, i.e., setting higher values for Q (in line 7 of the MERGE procedure given in Algorithm 1):

$$\frac{|(I_c \cup I_n)'|}{|G|} > Q \cdot \frac{|I_c'|}{|G|} \cdot \frac{|I_n'|}{|G|}.$$

The constraint above is equivalent to the constraint on lift of the association rule $I_n \to I_c$, i.e.,

$$lift(I_n \to I_c) = \frac{P(I_n \cup I_c)}{P(I_n) \cdot P(I_c)} > Q.$$

Moreover, because of the greedy strategy, the constraints hold recursively, i.e., there exist two disjoint subsets I_n^*, $I_c^* \subseteq I_n$ such that $lift(I_n^* \to I_c^*) > Q$. In experiments we consider how the proposed greedy strategy works for mining association rules on real-world datasets. Since the computing strategy is greedy, there are no guarantees that all LOC itemsets (see Definitions 1) will be enumerated.

Itemset mining through compression Likely-occurring itemsets may be also useful for selection of itemsets. We consider the relation between the itemsets selected by a compression-based itemset miner KRIMP [20] and LOC itemsets.

In KRIMP, and similar methods, the length of the code word corresponding to an itemset X is given by $length(X) = -\log P(X)$. Hence the compression is achieved by replacing several code words representing the itemsets B with a single code word, such that $length(B) < \sum_{X \in cover(B)} length(X)$. The latter is equivalent to $\log P(B) > log(\prod_{X \in cover(B)} P(X))$. Thus, we obtain the inequality $P(B) > \prod_{X \in cover(B)} P(X)$, which is very similar to one from the definition of the LOC itemsets.

Intuitively, in both cases, an itemset is considered optimal if its observed probability is greater than the estimated one. However, there are important differences between the models underlying the definition of "itemset optimality" (for the LOC itemsets and the model used in KRIMP):

1. the both methods use different probability estimates of itemsets, namely, $P(X) = fr(X)$ (for the LOC estimates) and $P(X) = \frac{usage(X)}{\sum_{Y \in \mathcal{P}} usage(Y)}$ (for the KRIMP-like models), where $usage(X)$ is frequency of X in the coverage, and $\mathcal{P}$ is the set of patterns;

2. the "optimality" of an itemset X in the compression-based model used in KRIMP is evaluated not only w.r.t. the dataset but also w.r.t. the other itemsets selected so far.

Thus, LOC itemsets may provide better results than the commonly used frequent closed itemsets, which are used by KRIMP. We compare different strategies for discovering itemset search space on real-world datasets in the next section.

5 Experiments

We use the discretized datasets from the LUCS-KDD repository [9] and study LOC itemsets[4] for two tasks, namely association rule and itemset mining.

Association rule mining. Frequent (closed) itemsets are commonly used to mine association rules. We study how useful LOC itemsets compared to frequent closed itemsets. In experiments we use 2 different sets of itemsets to compute rules: frequent closed (FR.CL.) and likely-occurring closed (LOC) itemsets. The itemsets are evaluated on 10 datasets, their parameters are given in Table 1. The number of objects and attributes is denoted by $|G|$ and $|M|$, respectively. The density of datasets (the ratio of 1's) is given in the column "density". The total number of closed itemsets is reported in the column "#CL". The total number of arbitrary itemset has not been computed.

Table 1: Parameters of datasets and the studied sets of itemsets

name	data description				closed itemsets			time (for itemsets)		#rules	
	$\|G\|$	$\|M\|$	density	#CL	#LOC	#FR.CL.	fr.thr.	LOC	FR	LOC	FR.CL
breast	699	14	0.64	360	74	74	0.33	0.01	0.12	4292	3980
ecoli	327	24	0.29	424	120	120	0.06	0.02	0.60	4768	2950
glass	214	40	0.22	3211	887	955	0.06	0.10	10.85	55262	18454
heart-dis.	303	45	0.29	25511	1928	1973	0.17	0.28	1.43	862252	22222
iris	150	16	0.25	106	45	47	0.05	0.00	0.03	274	320
led7	3200	14	0.50	1936	150	150	0.19	0.01	0.05	1484	1120
pima	768	36	0.22	1608	317	317	0.10	0.06	4.57	21294	7528
ticTacToe	958	27	0.33	42684	1880	1908	0.03	0.11	14.90	53816	13016
wine	178	65	0.20	13169	4914	5647	0.03	1.20	520.43	1771852	189378
zoo	101	35	0.46	4552	610	621	0.33	0.10	1.14	1609108	24736
average	**690**	**32**	**0.34**	**9356**	**1093**	**1181**	**0.14**	**0.19**	**55.41**	**438440**	**28370**

For each dataset we generate the whole set of LOC itemsets ($Q = 1$, $F = 0$), the sizes of these sets are indicated in the column "#LOC".

We chose the frequency threshold for closed itemsets in such a way that the number of closed itemsets is equal to the number of the LOC itemsets. The frequency threshold is indicated in the column "fr.thr." for closed itemsets. The frequency threshold varies a lot from dataset to dataset. For example, the smallest threshold is 0.06 for "ecoli" and "glass" datasets and the largest one is 0.33 for "breast" and "zoo" dataset. As we can see from the table, the sizes of "#LOC" and "#FR.CL." are quite close one to another.

To compute association rules we use a miner from MLxtent library implemented in Python[5]. The number of rules generated based on LOC and frequent closed (FR.CL.) is reported in the column "#rules".

[4] The source code for computing LOC itemsets is available at `https://github.com/makhalova/pattern_mining_tools/blob/master/modules/binary/likely_occurring_itemsets.py`

[5] `http://rasbt.github.io/mlxtend/`

The number of rules generated based on the LOC itemsets is higher than the number of rules generated based on frequent closed itemsets. For example, for the "ecoli" dataset, the number of rules computed on 120 LOC and 120 frequent closed itemsets is 4768 and 2950, respectively. It can be explained by the fact that the size of the LOC itemsets is usually larger than the size of frequent closed itemsets. Thus, a larger amount of rules can be built on LOC itemsets by splitting each itemset into an antecedent and consequent.

To evaluate their quality, we consider the most common quality measures for the association rules, namely *support*, *confidence*, *lift*, *leverage*, and *conviction*. We recall them below.

Let $X \to Y$ be an association rule with the antecedent X and the consequent Y, then the rule support is given by

$$support(X \to Y) = support(X \cup Y) = \frac{(X \cup Y)'}{|G|} \in [0, 1].$$

Confidence [2] of a rule $X \to Y$ is the conditional probability of $X \cup Y$ given X. It is defined as follows:

$$confidence(X \to Y) = \frac{support(X \to Y)}{support(X)} \in [0, 1].$$

The maximal value is achieved when Y always occurs with X.

Lift [8] was discussed in the previous section. We recall it below. For a rule $X \to Y$ lift is given by

$$lift(X \to Y) = \frac{support(X \to Y)}{support(X) \cdot support(Y)} \in [0, \infty).$$

Leverage [16], like lift, is based on the comparison of the observed probability (frequency) of the rule and the probability estimated under the assumption that the antecedent and consequent are independent. Leverage is given as follows:

$$leverage(X \to Y) = support(X \to Y) - support(X) \cdot support(Y) \in [-1, 1].$$

For independent X and Y leverage is equal to 0.

Let us proceed to the results of the experiments.

For the generated rules we consider mean values of the aforementioned quality measures as well as the 75th, 90th, and 95th percentiles. Considering the percentiles allows us to focus on the quality of the best itemsets, which are usually of interest to analysts. The averaged over 10 dataset values are reported in Fig. 2.

Since we do not set any frequency threshold for LOC, the support of LOC-based rules, as expected, is lower than the support of the rules based on frequent closed itemsets (FR.CL.). The top n% of LOC-based rules have higher values than the top n% of FR.CL.-based ones. For example, the top 10% values (the 90th percentile) of confidence are at least 0.935 for the LOC-based rules, and

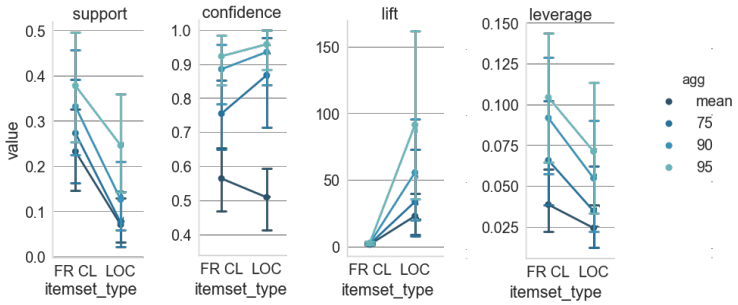


Fig. 2: The averaged quality for 2 types of rules: computed based on frequent closed (FR.CL.) and LOC itemsets. The quality is measured by support, confidence, lift, and leverage. For each type of rules and each quality measure, the average values of mean, the 75th, 90th, and 95th percentiles over 10 datasets from Table 1 are reported

only 0.885 for FR.CL.-based rules, respectively. Thus, considering the top rules, the LOC-based rules have higher confidence.

Regarding lift, LOC-based rules provide the best results. The difference in values is especially noticeable for the top 5% of rules (the 95th percentile). Top 5% LOC-rules have the highest values of lift, on average, 91.38. However, the lift values of the top 5% of rules vary a lot from dataset to dataset (the standard deviation is shown in plots by horizontal lines). Nevertheless, the quality, measured by lift, is consistently higher for LOC-based rules than for FR.CL.-based rules.

The leverage is higher for FR.Cl.-based rules. Despite the fact that lift and leverage differ only in the mathematical operations they use to compare the observed and estimated supports of rules and their parts, the analysis of rules based on these measures may lead to very different results. The high values of leverage (and low values of lift) for FR.CL.-based rules are caused by a different order of magnitude of the supports. Very low supports (that is the case of LOC-based rules) result in high values of lift and low values of leverage.

Thus, the analysis of the generated rules allows us to conclude that rules generated based on LOC itemsets have better quality than the rules generated using roughly the same amount of frequent arbitrary and closed itemsets, respectively.

Compression quality. In Section 4 we discussed the relation between LOC itemsets and the itemsets ensuring good compression in Krimp.

In this section we study the applicability of LOC itemsets for this task and compare them with closed itemsets (used in the original version of Krimp. We emphasize that, in the compared approaches, the itemset search space is discovered independently of the itemset mining process.

To evaluate the ability of the itemsets to compress data, we consider how many itemsets we need to obtain a certain compression ratio. Fig. 3 shows how the compression ratio changes w.r.t. the number of considered itemsets. The

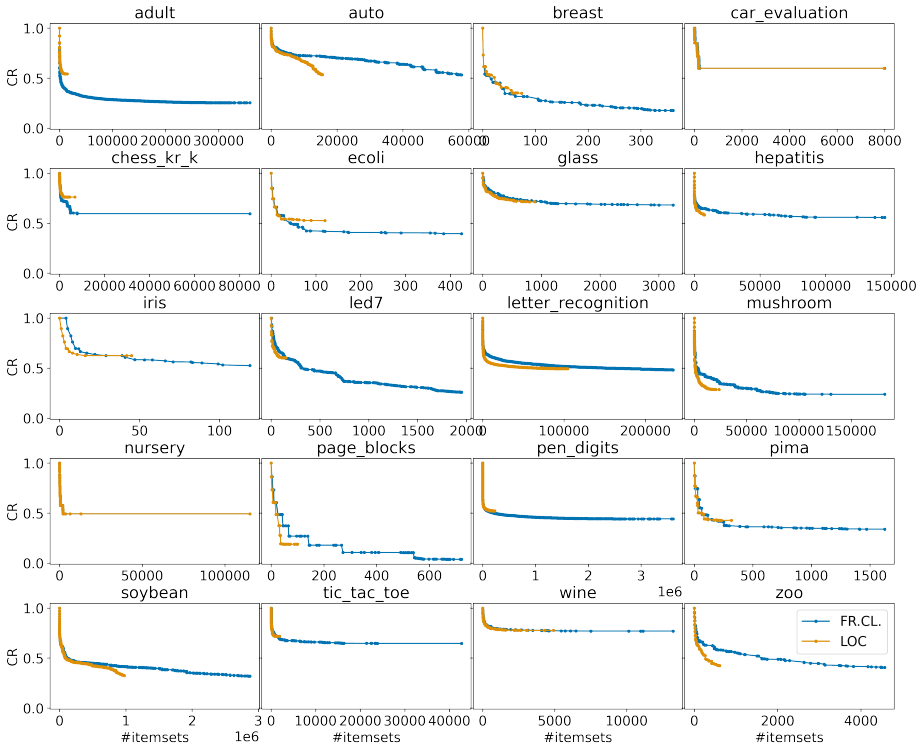


Fig. 3: Compression quality of closed (FR.CL.) and LOC itemsets. The lower values are better

initial state corresponds to the point (0,1), meaning that 0 itemsets have been used to compress data, and the compression ratio is maximal and equal to 1. The curves that are closer to the point (0,0) correspond to the best strategies of itemset search space discovery (i.e., the itemset set allows for compressing data better with a lower number of itemsets). The experiments show that for "car evaluation", "wine" and "nursery" datasets the LOC itemsets do not provide any benefits over the closed itemsets. For the majority of datasets, the number of LOC is too small to ensure as good compression as with the whole set of closed itemsets, e.g., "adult", "breast", "led7", and others. Among some of these datasets, we may still observe better behavior of LOC itemsets, e.g., for "hepatitis", "mushroom", "letter recognition", and "page blocks". There are also datasets where with the LOC itemsets we achieve as good compression as with the closed ones, but use a much lower number of itemsets, e.g., "auto", "hepatitis", "soybean", "zoo".

In general, LOC itemsets may be quite useful for itemset selection based on compression.

6 Conclusion

In this paper we studied likely-occurring closed itemsets in the context of association rule mining and itemset selection. In our experiments we show that the number of frequent enumerated LOC itemsets is much lower than the number of frequent closed itemsets. However, with LOC itemsets, we obtain association rules of better quality. The proposed approach may be useful for compression as well, however, it does not outperform the methods where itemsets are discovered towards the direction minimizing the total description length.

References

1. Aggarwal, C.C., Han, J. (eds.): Frequent Pattern Mining. Springer (2014)
2. Agrawal, R., Imieliński, T., Swami, A.: Mining association rules between sets of items in large databases. In: Proceedings of the International Conference on Management of Data. vol. 22, pp. 207–216. ACM SIGMOD (1993)
3. Andrews, S.: A partial-closure canonicity test to increase the efficiency of CbO-type algorithms. In: International Conference on Conceptual Structures. pp. 37–50. Springer (2014)
4. Bastide, Y., Taouil, R., Pasquier, N., Stumme, G., Lakhal, L.: Mining frequent patterns with counting inference. In: ACM SIGKDD Explorations Newsletter. vol. 2. ACM SIGKDD (2000)
5. Bayardo, R.J., Agrawal, R., Gunopulos, D.: Constraint-based rule mining in large, dense databases. Data Mining and Knowledge Discovery **4**(2-3), 217–240 (2000)
6. Boley, M., Lucchese, C., Paurat, D., Gärtner, T.: Direct local pattern sampling by efficient two-step random procedures. In: Proceedings of the 17th International Conference on Knowledge discovery and Data Mining. pp. 582–590. ACM SIGKDD (2011)
7. Boley, M., Moens, S., Gärtner, T.: Linear space direct pattern sampling using coupling from the past. In: Proceedings of the 18th International Conference on Knowledge Discovery and Data Mining. pp. 69–77. ACM (2012)
8. Brin, S., Motwani, R., Ullman, J.D., Tsur, S.: Dynamic itemset counting and implication rules for market basket data. In: Proceedings of the International Conference on Management of Data. pp. 255–264. ACM SIGMOD (1997)
9. Coenen, F.: The LUCS-KDD discretised/normalised ARM and CARM data library. `http://www.csc.liv.ac.uk/~frans/KDD/Software/LUCS_KDD_DN` (2003)
10. Ganter, B., Wille, R.: Formal Concept Analysis. Springer Berlin Heidelberg (1999). https://doi.org/10.1007/978-3-642-59830-2, `http://dx.doi.org/10.1007/978-3-642-59830-2`
11. Hu, Q., Imielinski, T.: Alpine: Progressive itemset mining with definite guarantees. In: Proceedings of the International Conference on Data Mining. pp. 63–71. SIAM (2017)
12. Kuznetsov, S.O.: A fast algorithm for computing all intersections of objects from an arbitrary semilattice. Nauchno-Tekhnicheskaya Informatsiya Seriya 2-Informatsionnye Protsessy i Sistemy (1), 17–20 (1993)
13. Kuznetsov, S.O., Obiedkov, S.A.: Comparing performance of algorithms for generating concept lattices. Journal of Experimental & Theoretical Artificial Intelligence **14**(2-3), 189–216 (2002)

14. Makhalova, T., Kuznetsov, S.O., Napoli, A.: On coupling FCA and MDL in pattern mining. In: Proceedings of the 15th International Conference on Formal Concept Analysis. pp. 332–340. Springer (2019)
15. Mampaey, M., Vreeken, J., Tatti, N.: Summarizing data succinctly with the most informative itemsets. ACM Transactions on Knowledge Discovery from Data **6**(4), 16 (2012)
16. Piatetsky-Shapiro, G.: Discovery, analysis, and presentation of strong rules. Knowledge Discovery in Databases pp. 229–238 (1991)
17. Smets, K., Vreeken, J.: Slim: Directly mining descriptive patterns. In: Proceedings of the International Conference on Data Mining. pp. 236–247. SIAM (2012)
18. Uno, T., Asai, T., Uchida, Y., Arimura, H.: An efficient algorithm for enumerating closed patterns in transaction databases. In: Proceedings of the 7th International Conference on Discovery Science. pp. 16–31. Springer (2004)
19. Vreeken, J., Tatti, N.: Interesting patterns. In: Aggarwal, C.C., Han, J. (eds.) Frequent Pattern Mining, pp. 105–134. Springer (2014)
20. Vreeken, J., Van Leeuwen, M., Siebes, A.: Krimp: mining itemsets that compress. Data Mining and Knowledge Discovery **23**(1), 169–214 (2011)
21. Webb, G.I.: Self-sufficient itemsets: An approach to screening potentially interesting associations between items. ACM Transactions on Knowledge Discovery from Data **4**(1), 1–20 (2010)
22. Zaki, M.J.: Generating non-redundant association rules. In: Proceedings of the 6th International Conference on Knowledge Discovery and Data Mining. pp. 34–43. ACM SIGKDD (2000)

Concept-based Chatbot for Interactive Query Refinement in Product Search

Elizaveta Goncharova[1][0000−0001−8358−9647], Dmitry Ilvovsky[1][0000−0002−5484−372X], and Boris Galitsky[2][0000−0003−0670−8520]

[1] National Research University Higher School of Economics, Moscow, Russia
{egoncharova,dilvovsky}@hse.ru
[2] Oracle Inc., Redwood Shores, CA, USA
boris.galitsky@oracle.com

Abstract. Nowadays, retrieval-based dialogue engines witness a significant interest in both industry and academic research. It is an important task to create an automated question-answering engine that may assist a user in the purchasing procedure. In this work, we describe a concept-based chatbot that utilizes a knowledge model to navigate a user to a subset of relevant objects. The knowledge model is built with pattern structures and organized to manipulate with both standard numerical and textual attributes describing the observed products. We provide an experimental evaluation of the introduced chatbot on the Flintkart e-commerce dataset and compare its performance with the faceted search approach.

Keywords: Query refinement · Information retrieval · FCA · Pattern structure.

1 Introduction

This paper describes ongoing research on creating an information retrieval (IR) chatbot that can assist customers in a search of suitable objects in the e-commerce database. IR chatbots have been an area of intense investigation [21, 24] in many spheres, from web search to electronic libraries. However, conventional search engines are still severely restricted. For example, when a user needs to refine his or her initial general request, a typical systems show either all attributes or the most frequently refined ones which results in the information overload problem and demands many choices to be made until the satisfactory result is found. Besides, such systems are not able to simultaneously operate with different types of data, such as numerical and textual ones.

We base our chatbot on utilizing the knowledge model constructed with the theory of Formal Concept Analysis (FCA) [22] and pattern structures (PS) [7] as its extension, therefore, further in the paper, we will refer to the chatbot as the concept-based one. The usage of the knowledge model allows a system to represent all the objects as hierarchically organized groups (concepts) and the description common for them. Thus, the chatbot can navigate a user through this model and following the specification or the generalization path, introduce only the relevant characteristics of a product.

This paper introduces the main aspects of the designed model, specifically, a method to combine the numerical and textual descriptions which are widely used to describe objects in the e-commerce datasets (Section 3). As for textual descriptions, we also

introduce a new way to define the generalization operation for two textual attributes that provide a better similarity assessment in comparison to the standard strict match of the keywords. We describe the bot architecture in Section 4. Finally, experimental evaluation of the introduced model on the e-commerce dataset is given in Section 5.

2 Related Work

The goal of IR is to find relevant objects that satisfy a user's request and, probably, refine it during a search procedure. Nowadays, the most popular IR systems are based on deep learning (DL) techniques [23, 9, 20]. These models encode the information describing an object with the neural network forming the object embeddings and, then, find the closest vectors to a query embedding. In most cases, this approach is applied to the text data. Lately, there have been some works proposing the neural networks for tabular data as well [6]. Despite the popularity of these methods, they still suffer from insufficient data representation, as we need to represent both a short query that reflects only partial information about the desired object and the whole objects described by many attributes within a vector of the same length.

FCA can serve as a natural mathematical tool for the knowledge-based semantic IR systems with the ability to effectively sustain data as a set of robust and hierarchically connected groups of objects and shared attributes. Traditionally, the studies that apply FCA to IR consider retrieving information from the large collections of unstructured documents, which is reasonable due to the analogy between the binary object-attribute and document-term tables [14]. While the standard IR FCA-based methods have been applied to the binary data which introduce some restrictions for real-life applications [15], PS allowed this type of models to be extended to more complex data such as numbers, graphs, or texts [13, 18, 5].

We claim that introducing a concept model to the e-commerce chatbot is beneficial due to its ability to represent both the numerical and textual attributes of objects. Moreover, the hierarchical organization of similar object groups can help the chatbot to effectively refine insufficient users' requests. However, first, we should define the rules for computing common descriptions for text attributes. Besides, due to the high computational requirements for building the concept lattice, we need to understand at which step of data exploration the concept model should be constructed.

3 Concept Model

A concept model is a principal part of the introduced IR chatbot as it is able to provide a convenient data representation that reflects the relations among the concepts, data, and entities. We build the model using PS, let us briefly recall its main definitions.

Formally, PS is defined as a triple $(G, (D, \sqcap), \delta)$, where G is a set of objects, $(D, \sqcap)$ is a complete meet-semilattice of descriptions and $\delta \rightarrow D$ is mapping of an object to its description. The Galois connection between the powerset of objects and the ordered set of descriptions is defined as follows $A^{\square} = \sqcap_{g \in A} \delta(g)$, $d^{\square} = \{g \in G | d \sqsubseteq \delta(g)\}$ for $A \subseteq G$, for $d \in D$. A pair (A, d) for which $A^{\square} = d$ and $d^{\square} = A$ is called a pattern concept.

A pattern concept is a pair, where the first element is a set of objects (called *pattern extent*) and the second element is the common description (called *pattern intent*) of the objects from the set. As in standard FCA, the sub-concept relation is given by the containment of extents. By defining the specific $\sqcap$ for different types of attributes, we can apply PS to processing complex heterogeneous data. For the introduced chatbot we have two main types of attributes: the numerical and textual ones.

Numerical Attributes We start building the knowledge model by distinguishing a subset of homogeneous objects $A_h \subseteq G$, i.e., the set of objects described by the high rate of similar attributes and construct PS. A semilattice operation $\sqcap$ is defined attribute-wise. The values of numerical attributes are given by intervals. Let $v_1, v_2 \subseteq R$ be the values of a numerical attribute for two different products. Then, $[v_1, v_1]$ and $[v_2, v_2]$ are their interval representations. The meet of these intervals is defined as $[v_1, v_1] \sqcap [v_2, v_2] = [min(v_1, v_2), max(v_1, v_2)]$. For nominal attributes, the operation $\sqcap$ is defined as $x \sqcap y = x$ if $x = y$ and $x \sqcap y = *$ otherwise, where $*$ means "any value". The navigation model is constructed in the form of *pattern concept lattice* that the chatbot can traverse during communication with a user. The example of this model for the numerical attributes can be found in github[3].

input : $\tau\{a_{1t_1}\}, \tau\{a_{2t_1}\}$ — textual descriptions (a set of key words) of t_1 attribute for two objects a_1 and a_2.
output: $\tau(\{a_{1t_1}\}, \{a_{2t_1}\})$ — a common description of t_1 attribute for two objects a_1 and a_2.

foreach *key term* kt_{a_1} *of the key terms set* $\tau(\{a_{1t_1}\})$ **do**
  foreach *key term* kt_{a_2} *of the key terms set* $\tau(\{a_{2t_1}\})$ **do**
    if $kt_{a_1} == kt_{a_2}$ **then**
      $\tau(\{a_{1t_1}\}, \{a_{2t_1}\}).add(kt_{a_1})$;
    else
      if `similarity(`$emb(kt_{a_1}), emb(kt_{a_2})$`)` $> threshold$ **then**
        $\tau(\{a_{1t_1}\}, \{a_{2t_1}\}).add(kt_{a_1})$;
      end
    end
  end
end
return $\tau(\{a_{1t_1}\}, \{a_{2t_1}\})$;

Algorithm 1: Computing the intersection of two textual attributes

Textual Attributes To process text data we, first, need to define its description, in our case we use simple keywords. Their usage is motivated by the specificity of the data we analyze. Textual attributes in e-commerce datasets are usually defined by a phrase or a sentence. Therefore, it is not necessary to utilize more complex descriptors,

[3] https://github.com/lizagonch/Chatbot.git

such as parse trees [19] or parse thickets [4] for such small texts. To calculate common description for two textual attributes (represented as the set of keywords), we define the $\sqcap$ operation as either a strict match between the keywords or by finding the synonyms in these keywords sets. The synonymity is assessed via pre-trained word embeddings (word2Vec [16] in our case). The algorithm 1 presents the procedure to calculate the intersection for two textual attribute descriptions.

4 Chatbot Overview

The idea of the introduced model is as follows, a user starts a search with an imprecise request, the search engine interactively refines a user's request by moving him or her down or up the lattice (concept model). The concept lattice is built using Close-By-One algorithm [1]. The overall system architecture is shown in Figure 1. At each turn of the dialogue, the user's utterance written in natural language goes through the NLP pipeline where sentence pre-processing is performed. This module is also responsible for identifying numerical attributes a user wants to refine and textual restrictions imposed on the product. Thus, at the output, we get the user's query description d_q. Let us consider each component in more detail.

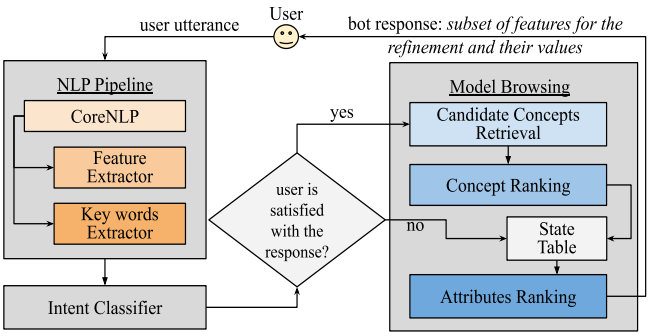


Fig. 1. System architecture

4.1 NLP Pipeline

We use Stanford CoreNLP toolkit [12] to perform sentence splitting, tokenization, and lemmatization. Then, the system reveals the restrictions on the numerical attributes. This part is performed with a predefined set of rules that determine attributes names and the values required by the user. Having retrieved these restrictions, we run the keyBERT model to return the set of keywords representing the user's request.

4.2 Intent Classifier

The proposed IR bot operates using the compiled hierarchical lattice-based knowledge model. It means that the system can navigate a user up (in case of generalization intent),

or down (refinement intent) the lattice. If a user wants to change the direction of a search, the bot should navigate him or her to another candidate concept at the same level of the concept lattice, where the level is the shortest way from the root concept to the current one. If a user clarifies some features proposed by the chatbot, then the chatbot moves in the right direction. Otherwise, asking to return on the previous step corresponds to generalization intent. We use manually constructed regular expressions to recognize the navigational intent.

4.3 Search Procedure

After identification of the intent, the system navigates a user through the knowledge model. Based on the user's intent, the bot provides various procedures to query processing. (*i*) If the user wants to *refine* his current position, the bot finds all most specific concepts that satisfy an input query description and updates the *State Table* with new candidate concepts. (*ii*) In the case of *update* intent the bot returns the user to the parent concept of the current (A_i, d_i) concept. (*iii*) *Change of interest* makes the bot return to the stack of promising concepts (*State Table*) and moves to the next candidate concept.

All candidate concepts in the *State Table* are ranked based on the stability measure [2, 8, 11], and each feature in the concept intent is assessed with respect to its diversity. The most stable concepts are introduced to the user first, whereas the most variable characteristics are proposed to the user for further refinement.

5 Experiments

We present the preliminary experiments to compare the concept-based chatbot performance with the faceted search[4] technique which is a standard approach for e-commerce IR. The experiments are performed on the publicly available Flipkart dataset[5] that contains information about more than 40000 objects from the e-commerce website. To compare the chatbot with the faceted search, which is not able to process textual data, we have retrieved 100 objects belonging to the category *'Computers – Laptops'* described only by the numerical attributes. To measure the effectiveness of the proposed $\sqcap$ operation for textual attributes we use the same concept-based bot, but define the generalization operation as the strict match of the keywords corresponding to the textual attributes. For this experiment, we have retrieved 300 objects belonging to the *'Clothing – Jeans'* category. The motivation for choosing these small datasets is that both the faceted search and the IR-chatbot should be launched after a user found the specific category of products that he or she is interested in that usually include up to 500 objects.

To assess the model performance we calculate the average number of iterations (**IN**) a user needs to achieve a satisfactory result. The lower this number is the faster a user finds relevant objects. Average Precision (**AP**) measures how many times during the dialogue a user had a *refinement* navigational intent. We also evaluate the number of cases where a user was not satisfied with the response of the bot and asked it to move in another direction (Unsatisfactory Rate (**UR**)).

[4] https://apex.oracle.com/en/

[5] https://data.world/promptcloud/product-details-on-flipkart-com

So far, we have not conducted the conversation of the bot with real customers, thus, in a preliminary experiment, we generated 60 random scenarios of possible users' requests. The scenario takes the form of a few relevant features and a range of their values that the user should sequentially introduce to the system. Table 1 presents the results of the experimental evaluation.

Table 1. Comparison of the knowledge-based search with the faceted search.

	Laptops (*only numerical attributes*)			Jeans (+ *textual attributes*)		
Model	**AP**	**UR**	**IN**	**AP**	**UR**	**IN**
Concept-based bot	0.7	0.24	5.6	0.72	0.14	6.6
Faceted search	—	0.31	6.2	—	—	—
Concept-based bot (*key words-strict match*)	—	—	—	0.68	0.25	8.6

AP and **UR** metrics indicate the ability of the bot to rank the promising pattern concepts and retrieve relevant features for refinement. The obtained results show that in nearly 70% of cases attributes introduced by the chatbot were assessed as significant. While, in almost 25% of cases in the "*Laptops*" category, a user had *change of direction intent*. The main metric evaluating the effectiveness of the search engine is the average number of dialogue turns (or clicks for the faceted search) that a user performed to find the set of items that are needed. In our experiments, this metric is in favor of the chatbot, 5.6 versus 6.2 respectively. For textual data processing, the approach that applies embeddings techniques to $\sqcap$ operation improves the performance of the model and makes a search procedure more precise (6.6 versus 8.6 dialogue turns).

6 Conclusion

In this paper, we have introduced an IR chatbot that utilizes the concept-based knowledge model to help users in finding a particular item in the e-commerce database. In comparison to a standard search engine, this system can operate with both structured data and textual descriptions that can also be obtained from any external resource.

We have compared the performance of the proposed model with that of the faceted search using the small product database retrieved from the online store. In this work in progress, we have not yet compared the performance of the model with the current state-of-the-art IR systems and other promising query enrichment FCA-based techniques [10, 17, 3], however, the obtained results based on the artificially generated scenarios of user-machine interaction has shown that the number of steps required by the proposed model is less than the one required by faceted browsing. In the future, we plan to add more functionality to the model, such as more advanced processing of textual information (e.g., a pre-defined set of syntactic patterns could be exchanged by the DL conversational engine) and more intelligent search of relevant attributes that should be introduced to the user. This can be achieved by encapsulating the history of users' purchases.

References

1. Andrews, S.: A partial-closure canonicity test to improve the performance of CbO-type algorithms. Lecture Notes in Computer Science **8577** (2014)
2. Babin, M.A., Kuznetsov, S.O.: Approximating concept stability. In: Proceedings of ICFCA 2012. pp. 7–15 (2012)
3. Bendella, M., Quafafou, M.: Patterns based query expansion for enhanced search on twitter data. In: CEUR Workshop Proceedings. vol. 2378 (2019)
4. Galitsky, B.A., Ilvovsky, D., Kuznetsov, S.O., Strok, F.: Finding maximal common sub-parse thickets for multi-sentence search. In: Graph Structures for Knowledge Representation and Reasoning. Lecture Notes in Computer Science. vol. 8323 (2014)
5. Hartmann, J., Stojanovic, N., Studer, R., Schmidt-Thieme, L.: Ontology-based query refinement for semantic portals. In: Lecture Notes in Computer Science. vol. 3379, pp. 41–50 (2005)
6. Katzir, L., Elidan, G., El-Yaniv, R.: Net-DNF: Effective deep modeling of tabular data. In: Proceeding of ICLR. vol. 125 (2021)
7. Kuznetsov, S.: Pattern structures for analyzing complex data. In: Rough Sets, Fuzzy Sets, Data Mining and Granular Computing, 12th International Conference, RSFDGrC 2009. Delhi, India
8. Kuznetsov, S.: Stability as an estimate of the degree of substantiation of hypotheses derived on the basis of operational similarity. Nauchno-Tekhnicheskaya Informatsiya, ser. 2 - Informatsionnye Protsessy i Sistemy pp. 21–29 (1990)
9. Li, J., Liu, C., Wang, J., Bing, L., Li, H., Liu, X., Zhao, D., Yan, R.: Cross-lingual low-resource set-to-description retrieval for global e-commerce. Proceedings of the AAAI Conference on Artificial Intelligence **34** (2020)
10. Lungley, D., Kruschwitz, U.: Automatically maintained domain knowledge: Initial findings. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). vol. 5478 (2009)
11. Makhalova, T., Ilvovsky, D., Galitsky, B.: Information retrieval chatbots based on conceptual models. In: Proceedings of International Conference on Conceptual Structures. pp. 230–238. Springer (2019)
12. Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S., McClosky, D.: The stanford CoreNLP natural language processing toolkit. In: Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations. Association for Computational Linguistics (2014)
13. Messai, N., Devignes, M., Napoli, A., Smaïl-Tabbone, M.: Many-valued concept lattices for conceptual clustering and information retrieval. In: Proceedings of ECAI (2008)
14. Messai, N., Devignes, M.D., Napoli, A., Smail, M.: Concept lattices and ontologies to query a directory of biological data sources (bioregistry). INFORSID 2005: Actes du XXIIIeme Congres Informatique des Organisations et Systemes d'Information et de Decision (2005)
15. Messai, N., Devignes, M.D., Napoli, A., Smail, M.: Using domain knowledge to guide lattice-based complex data exploration. In: Proceedings of the 2010 conference on ECAI 2010. pp. 847–852 (2010)
16. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. pp. 3111–3119 (2013)
17. Pattison, T.: Interactive query refinement using formal concept analysis. In: International Conference on Concept Lattices and Their Applications (2018)
18. Priss, U.: Lattice-based information retrieval. Knowledge Organization **27** (2000)

19. Punyakanok, V., Roth, D., Yih, W.T.: The necessity of syntactic parsing for semantic role labeling. In: Proceedings of IJCAI International Joint Conference on Artificial Intelligence (2005)
20. Song, S., Wang, C., Chen, H., Chen, H.: TCNN: Triple convolutional neural network models for retrieval-based question answering system in e-commerce. In: The Web Conference 2020 - Companion of the World Wide Web Conference, WWW 2020 (2020)
21. Tunkelang, D.: Dynamic category sets: An approach for faceted search. In: Proceedings of ACM SIGIR. vol. 6 (2006)
22. Wille, R.: Restructuring lattice theory: An approach based on hierarchies of concepts. Ordered Sets. NATO Advanced Study Institutes Series **83**, 445–470 (1982)
23. Yilmaz, Z., Wang, S., Yang, W., Zhang, H., Lin, J.: Applying BERT to document retrieval with Birch. In: Proceedings of IJCNLP 2019. pp. 19–24 (2019)
24. Zhang, L., Zhang, Y.: Interactive retrieval based on faceted feedback. In: Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval - SIGIR '10. ACM Press (2010)

Variability Extraction from Simulator I/O Data Schemata in Agriculture Decision-Support Software

Thomas Georges[1,2], Marianne Huchard[1], Mélanie König[2], Clémentine Nebut[1], and Chouki Tibermacine[1]

[1] LIRMM, Univ Montpellier, CNRS, Montpellier, France
{firstname.lastname}@lirmm.fr
[2] ITK, Montpellier, France
{firstname.lastname}@itk.fr

Abstract. The context of this work is the development of software systems that help in decision making in the agriculture domain at our industrial partner, ITK. These software systems include simulators which help farmers to understand and predict plants life cycle. Each plant and each kind of prediction has its own parameters. For example, yield prediction for wheat is very specific and different from vine disease prediction. There are however some common characteristics, like the fact that these simulators take as input weather data. The goal of the project on which we work is to build a software product-line in order to: i) enable an easy derivation of new products (by IT teams) with new simulators (built by agronomist teams), and ii) simplify the maintenance of the existing large code base of our industrial partner. The construction of this product-line passes through the extraction of variable and common characteristics of all existing products at ITK. The extraction process may be laborious and time consuming. We study in this work the automation of this process, by focusing on the schemata of data received as input and produced as output by simulators. We hypothesize that Formal Concept Analysis (FCA) is a useful tool for extracting software variability, i.e. highlight commonalities and specifics for assisting IT/agronomist teams in software construction. In this paper, we propose a process for variability extraction. This process is based on a set of pre-processing steps to prepare data for FCA tools. These tools build at the end of the process an AOC-Poset, i.e. a conceptual structure derived from the concept lattice in which we can identify common and variable characteristics. We implemented this process and experimented it on a set of six simulators. We obtained promising results towards the construction of the software product-line.

Keywords: Formal Concept Analysis · Software engineering · Software Product Line · Variability Extraction · Knowledge Extraction

1 Introduction

Many software companies face the problem of developing and maintaining a portfolio of products with some common purpose and context. Capitalizing knowledge acquired on the domain and on the previously developed software may be a help for developing new ones in the same business domain, and rationalizing the different activities around software. When software systems are sufficiently similar, migrating to the software product line paradigm may be appropriate for that capitalization. For example, our industrial partner ITK[3] provides a decision-support software systems platform for agriculture. It develops simulators for different purposes as yield expectation or disease prediction. The platform can be seen as a set of similar software systems, which allows a migration to a software product line. The expected benefit is to assist the agronomist team in the development of a new product, and speed up simulator integration by the IT team. This requires extracting and organizing knowledge from the code base and all existing documents. As a first step in that direction, this paper focuses on extracting knowledge from part of this description, which is composed of an input and an output data schema, with a tree structure embedding the data hierarchical organisation. We use Formal Concept Analysis to highlight input and output variability among the different simulators. We call variability the ability of a software to be configured, customized, extended, or changed for a specific context [3]. In our case, it concerns the variation in the data schemata of simulators. In this paper, we explain the process that leads us from raw data to a conceptual structure, here an AOC-Poset and how to exploit it. Concretely, the contribution of this paper is an application of Formal Concept Analysis to identify variability. This paper is organized as follows. Section 2 gives an overview of the approach. Section 3 addresses the dataset presentation. Section 4 explains how the preprocessing is performed on Data schemata. Section 5 presents the Formal Concept Analysis processing. Section 6 shows the results. Section 7 exposes the related work and Section 8 concludes the paper with a summary of the contribution and a few perspectives.

2 Overview of the approach

Research Question. The main question studied in this paper is: *How to extract software variability from simulator data schemata?* To answer this question, we need to use simulator data schemata to build a formal context usable with FCA, and more specifically AOC-Poset building algorithms. This is performed by a process, presented in the following subsection, which is able to exploit simulator data schemata to get as much knowledge as possible.

Process. Figure 1 illustrates the process to get variability from raw data schemata. First a pre-processing is performed, by cleaning, enriching and formatting data to obtain a formal context. Then from the formal context, an FCA-based structure,

[3] https://www.itk.fr/en/

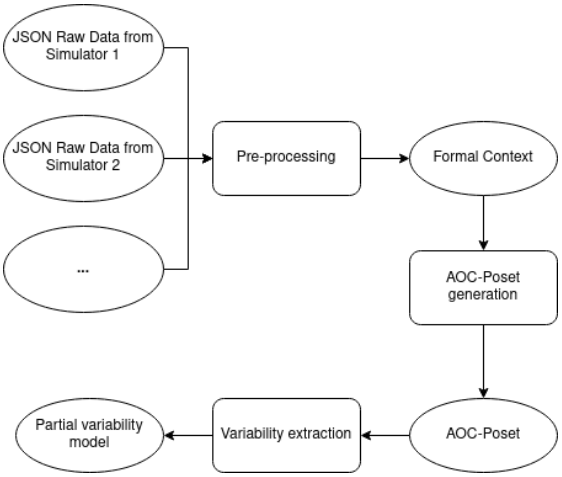


Fig. 1. Process from simulators to variability

i.e the AOC-Poset [10] is used to highlight commonalities and specifics. So far, only a partial variability model is created, with common terms and simulator-specific terms. We have not yet studied the identification of logical rules relating these terms, like implications, co-occurrences or mutual exclusions between two terms, but we are quite confident that this is possible by applying techniques proposed in previous work from our team [8].

An AOC-Poset has been chosen instead of a usual concept lattice because of the simplicity of this model which makes it easily readable and understandable in the context of variability analysis. The second theoretical reason is the complexity of its construction, which is polynomial in contrast to the construction of concept lattices that is exponential. The size of simulator data schemata is relatively small for the moment in our study, but in the future if the process is to be used with larger and numerous schemata, the construction of the variability model would be made more efficient.

3 Simulator description

Each simulator used in ITK products has its own purpose. It receives some data as input, like weather information or soil type. It produces some data as output, like predictions of yield or disease. All ITK products are defined as Web applications with simulators written mainly in Python. Input and output data are defined in *JSON* format[4] and have schemata defined in *JSON* too.

Output data schemata are in general less complex than input data ones and can be absent in some cases, if there is only a number or a string as the output from a simulator.

[4] https://www.json.org/json-en.html

Our study is based on six different simulators:

- *Cropwin simulator* to estimate yield from annual culture as wheat or corn.
- *Disease simulator* to predict plant's risk to contract a disease.
- *Grapes simulator* to estimate yield from grapes cultures.
- *nferti simulator* to predict plant's stress, as the lack of nitrogen.
- *Orchard simulator* to estimate yield from sustainable culture as apricots or walnuts.
- *Vine disease simulator* to predict vine's risk to contract a specific disease.

Simulator description Each simulator has a documentation which is provided by the agronomists who developed the simulator. This documentation includes a wiki with a description of the simulation model (the mathematical model), the API for using the simulator, its dependencies and its technical documentation, among other elements. The simulation model provides the schema of the data needed for running the simulator (input data) and also the schema of the data provided as an output. These schemata are defined using a JSON dialect. We collected the available schemata for the selected six simulators.

Input/output data schema. Each data schema is a tree of terms. From these six simulators, we have six input data schemata and four output ones. Input data schemata include from 31 to 127 different terms and output data schemata include 22 to 95 different terms (see Table 1 for details).

Simulator	nferti	Cropwin	Disease	Grapes	Orchard	Vine disease	**total**
Inputs terms	119	126	127	31	50	11	**464**
Outputs terms	95	32			53	22	**202**

Table 1. Simulator input and output description size

4 Pre-processing Data schemata

Raw data schemata cannot be exploited directly in our process. They need to be sanitized, formatted and prepared to be exploited by FCA. For this, we build a dictionary to exclude unwanted/technical terms, a dictionary to associate new terms to replace existing acronyms and abbreviations. In order to maximize variability extraction, we choose to exploit tuples of terms. The results have been formatted as a formal context.

The construction of our dictionaries by removing or associating more terms was an iterative manual work. An analysis was made after each iteration. Redundant and inappropriate terms were removed and new terms explaining existing abbreviations were added.

In Figure 2, we outline the complete process that goes from raw data to a formal context. First we transform raw data schemata in tree (*Treeify*) in order

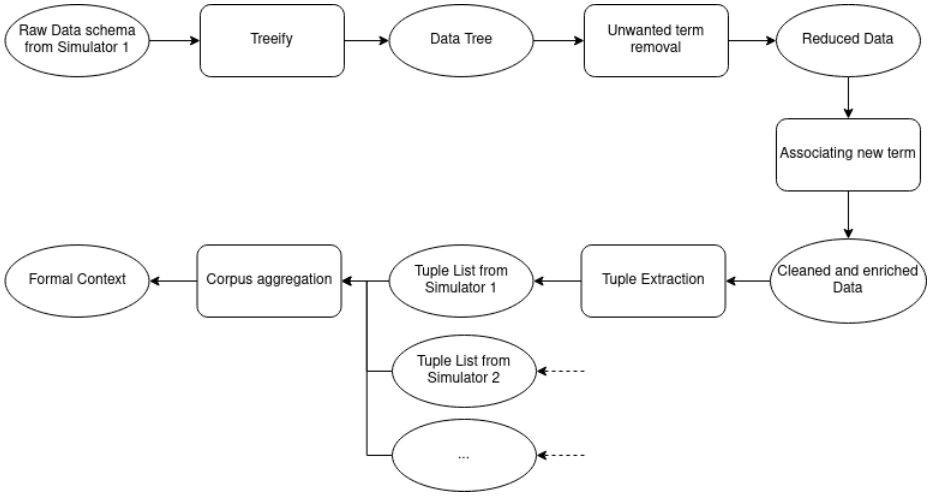


Fig. 2. Process to transform raw data into formal contexts.

to make the following processing steps on a tree and not on a text document. Then, we remove useless terms (*Unwanted term removal*). Next we replace abbreviations and acronyms (*Associating new term*) and we extract tuples from the tree (*Tuple Extraction*). At the end, for all simulators, data schemata of each kind (Input or Output) are merged and formatted as a formal context (*Corpus aggregation*).

4.1 Excluding unwanted terms

Stopwords are useless terms, which do not need to be kept in our variability extraction result. We choose to remove them to limit unnecessary too frequent terms. Without removing unwanted terms, variability extraction would be less relevant due to the introduced noise. To build the dictionary, we ranked all terms by their frequency, in order to select and remove all too frequent unwanted terms. We have used a well-known metric for that which is TF-IDF [16]. We built a dictionary with 30 different terms to be removed.

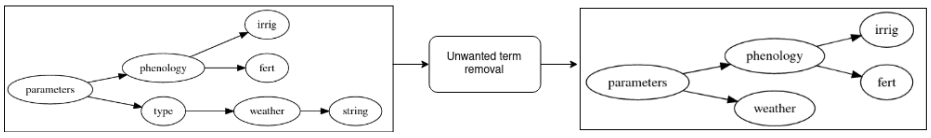


Fig. 3. Removing unwanted terms example

Figure 3 depicts an example of this processing. Each term in a data schema is searched in the unwanted terms dictionary and removed from the tree if present

(e.g. *type* and *string*). If the removed term is not a tree leaf, the subtree linked to this removed node is linked to its parent directly.

4.2 Associating new terms

The goal of building this second dictionary is to extend abbreviations and replace acronyms by the complete terms. The construction of this dictionary has been done manually. We checked each term to decide if it was an acronym or an abbreviation. If this was the case, we added it to the dictionary together with its complete name. For example, *irrig* has been added and associated to the term *irrigation*. The built dictionary includes around 30 different terms, 9/10 of them coming from the agronomic domain and 1/10 from IT domain.

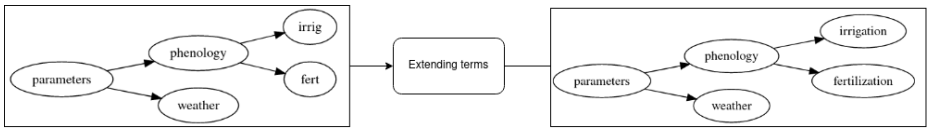


Fig. 4. Extending terms example

Figure 4 shows an example of this processing. Each term is compared with the associated term in the dictionary. If it matches, the current term is replaced by its complete version.

4.3 Tuple extraction

Extracting only single terms makes us loose the information about relationships provided by the tree structure. To keep information from data we need to refine the extraction. We extract each node alone, but also all father-son's pairs following a depth-first search method. The size of the extracted tuples is one or two terms, and this is enough for our process. After empirical observations, we indeed concluded that the use of tuples of size three or more terms does not help in identifying more commonalities in our data, because they are present in only one simulator. For example, the following tuple `(parameters,phenology,irrigation)` is specific to a single simulator, which is *nferti*.

Besides this, we did not base our process on raw text data, and choose to transform it into our own tree representation, in order to be independent from any kind of data structure format, such as *XML* or *JSON* in our case.

Figure 5 depicts an example of tuple extraction. In this example, we can observe the extraction of five tuples with a single term and four tuples with two terms, starting from a tree of five nodes and four father-son edges.

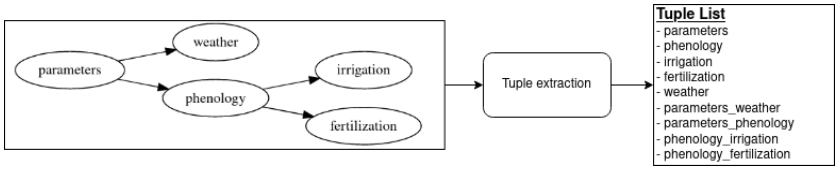


Fig. 5. From cleaned and enriched data to formal attributes

4.4 Data Formatting

To use Formal Concept Analysis, a last transformation is required. We use here FCA [12] as a knowledge engineering method, for its capacity to build *formal concepts* from a formal context (FC) $K = (G, M, I)$ that associates objects from a set G to attributes from a set M through relation $I \subseteq G \times M$. Object sets and attribute sets are associated thanks to two operators, both denoted by $'$. For $O \subseteq G$, the set of attributes shared by the objects of O is $O' = \{m | \forall g \in O, (g, m) \in I\}$. For $A \subseteq M$, the set of objects that share the attributes of A is $A' = \{g | \forall m \in A, (g, m) \in I\}$. A formal concept $\mathcal{C} = (Extent(\mathcal{C}), Intent(\mathcal{C}))$ is a maximal object group (extent) associated with their maximal shared attribute group (intent), i.e. $Extent(\mathcal{C}) = Intent(\mathcal{C})'$ (and equivalently $Extent(\mathcal{C})' = Intent(\mathcal{C})$). The concept order, denoted by $\preceq_{\mathcal{C}}$ is defined as follows: $\mathcal{C}_1 \preceq_{\mathcal{C}} \mathcal{C}_2$ if $Intent(\mathcal{C}_2) \subseteq Intent(\mathcal{C}_1)$ (and equivalently $Extent(\mathcal{C}_1) \subseteq Extent(\mathcal{C}_2)$). The concept lattice is the set of all concepts, provided with $\preceq_{\mathcal{C}}$. The lowest (w.r.t. $\preceq_{\mathcal{C}}$) concept owning one object is its introducer concept. The highest (w.r.t. $\preceq_{\mathcal{C}}$) concept owning one attribute is its introducer concept. The suborder of the concept lattice restricted to these introducer concepts is called the AOC-Poset (Attribute-Object Concept poset). In the following, we use the AOC-Posets, which are a scalable alternative to concept lattices, as the conceptual structures to highlight variability, as they contain all the information we need.

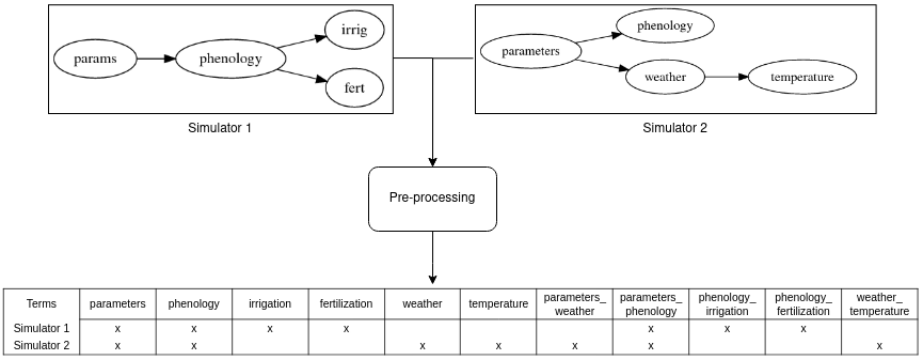
We need to generate two formal contexts from the extracted tuples. One for the input and the other for the output data schemata. In each formal context, G is the set of simulators, M is the set of tuples, i.e. 1-tuples for nodes or 2-tuples for edges. $(g, m) \in I$ if the tuple m exists in the data schema of the simulator g.

Figure 6 shows an example of the transformation. It starts from two simulators and generates a formal context. A cross means that the simulator has the tuple.

5 FCA processing

The cleaned data is now in the appropriate format to be processed by FCA tools. Extracting the variability using FCA is efficient and the obtained conceptual structures are a useful way to express this variability [6]. We used the RCA plugin of Cogui[5] to generate AOC-Posets [10]. For the input data schemata,

[5] http://www.lirmm.fr/cogui/



Terms	parameters	phenology	irrigation	fertilization	weather	temperature	parameters_ weather	parameters_ phenology	phenology_ irrigation	phenology_ fertilization	weather_ temperature
Simulator 1	x	x	x	x				x	x	x	
Simulator 2	x	x			x	x	x	x			x

Fig. 6. Example of raw data transformed into a formal context

we obtained the AOC-Poset depicted in Figure 7. This conceptual structure is discussed in the following section.

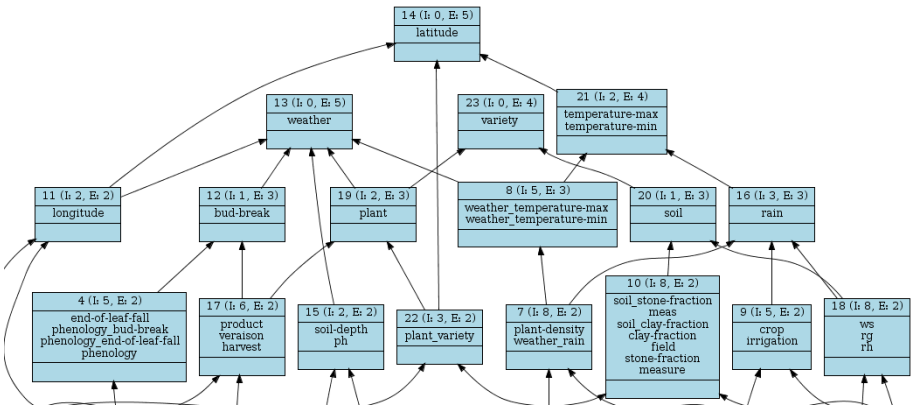


Fig. 7. Excerpt of the AOC-Poset from input data schemata

6 Evaluation

The variability extracted from AOC-Posets is used to understand how close the simulators are to each other.

6.1 Results

The excerpt of the AOC-Poset, presented in Figure 7, shows which simulators parts are specific and which are common. Precisely, beginning from the most common parts (from the top concepts of the AOC-Poset), we have:

– The concept 14 introducing attribute *latitude*, common to five simulators out of six.
– The concept 13 introducing attribute *weather*, common to five simulators out of six.
– The concept 23 introducing attribute *variety*, common to four simulators out of six.
– The concept 21 introducing attributes *temperature-max* and *temperature-min*, common to four simulators out of six.

Temperature and *weather* have a significant impact on cultures, each *variety* has its own specificity and *latitude* is useful for sunshine impact. Looking on agronomic domain, we can easily see why these are the most common terms.

We have 36 common terms to at least two simulators and 374 specific terms over 410 in total. This means that simulators have eight per cent of terms in common, all other concepts are only present once. These are specific to their own simulator. The bottom part of the AOC-Poset is not shown in Figure 7 because it contains useless information, which corresponds to six concepts that match with the six simulators and all their specific attributes (not common with other simulators). We can observe that the grouping of attributes in the AOC-Poset concepts is variable and ranges from one in the top concepts, discussed above, to seven in concept 10 of Figure 7. We can also observe that the latter concept corresponds to an abstraction of simulators that process soil information, *i.e.* there is a semantic cohesion between these seven attributes. Concept four (at the left of the figure) groups four attributes semantically related too, and which correspond to plant phenology.

For the outputs (the AOC-Poset is not shown in the paper for the sake of space), we found only three common terms:

– The concept introducing attribute *daily*, common to three simulators out of four.
– The concept introducing attribute *phenology*, common to three simulators out of four.
– The concept introducing attribute *yearly*, common to three simulators out of four.

We observed that there are only three common terms to at least two simulators and 195 terms specific over 198 in total. This means that simulators have one per cent of terms in common, all other concepts are only present once. This low degree of commonality was predictable since each simulator has its own purpose and returns only useful data targeting that purpose.

As a final step in this evaluation work, we plan in the near future to initiate a discussion with the agronomist teams in order to validate and potentially improve the extraction result (AOC-Posets).

6.2 Discussion

Our work has multiple purposes. The first goal is to assist agronomists in the design of new models and simulators, then support the simulator integration by the IT team and allow a better migration towards a software product-line.

The second purpose is to provide a common vocabulary. In addition, the association dictionary helps to understand which terms are used including acronyms, abbreviations or written in different ways and offer a new standardization. We are quite confident that this will help agronomists when a new model has to be built by providing a standardized vocabulary and naming conventions.

When the IT team has to develop a new application, its first task is the simulator integration. This is a fastidious and error prone task based on manually cloning an existing integration. Based on the generated variability, which will be linked to code artefacts, cloning can be automated and simplified.

The last envisaged use is the migration of ITK software products to a Software Product Line, considering the AOC-Poset as a step towards the production of a complete feature/variability model. This migration will not be based on the source code only but will be completed by ontologies [5]. The dictionaries and AOC-Posets are relevant artifacts to this aim.

7 Related work

FCA has already been used for variability extraction in the domain of software product lines, to synthesize a feature model by exploring the AOC-Poset, the AC-Poset (Attribute-Concept Poset) or implicative systems [15, 17, 1, 7]. In these works, the formal context associates software product configurations to features. The feature model is a kind of logical tree exposing mandatory and optional features, feature groups (Or, Xor), and feature refinement through tree edges [13]. Cross-tree constraints (such as binary implication or mutual exclusion) may accompany the description. We ground the variability extraction on the same principles, using FCA as the revealer of commonalities and specifics. Compared to these works, the difference is that we do not focus on feature variability, but on input/output data variability. This provides a complementary view on the future product line.

Dealing with tree description could have been dealt taking inspiration from genterms (labelled trees provided with a generalization relation) [9] or using the pattern structure paradigm [11, 14]. The pattern structure paradigm is a way we will explore. Nevertheless, it was initially not clear how to determine the similarity and subsumption operators for our labelled directed (rooted) trees. In this work, we preferred to conduct a first study using local information, based on common nodes and edges, that can be encoded with basic formal contexts. A drawback in our approach is that tree portions will have to be rebuilt in a post-processing operation if we want to have a global view, but this has the merit of providing a simple initial solution.

Finally, this work also shares similar objectives and techniques with database schema integration [4], ontology merging [18], and common model extraction [2],

including the need for linguistic analysis, and designing an integrated view, here on inputs or outputs of the simulators.

8 Conclusion

Linking software engineering and artificial intelligence with Formal Concept Analysis provides new tools and methods to improve current practices. We proposed to extract variability of simulators data schemata to improve future development of new simulators and to assist their integration in a new application.

In the short term, we have to integrate these in the software product line migration process, and to share this knowledge with the agronomist/IT team. All existing simulators do not have a data description schemata, especially outputs. This causes a lack of details about variability and reduces result impact. Including more data schemata coming from other simulators and asking the agronomist team to detail the outputs when they are absent will improve the quality of the results allowing us to give more assistance to ITK teams.

We plan in the future to work on variability extraction from source code of existing applications and linking this variability with what we extract from input/output data schemata. The ultimate goal is to provide a complete feature model which will enable agronomists and IT teams to easily configure new products by working together on common assets and using a unified vocabulary.

References

1. Al-Msie'deen, R., Huchard, M., Seriai, A., Urtado, C., Vauttier, S.: Reverse engineering feature models from software configurations using formal concept analysis. In: Proceedings of the Eleventh International Conference on Concept Lattices and Their Applications, Košice, Slovakia, October 7-10, 2014. pp. 95–106 (2014)
2. Amar, B., Guédi, A.O., Miralles, A., Huchard, M., Libourel, T., Nebut, C.: Using formal concept analysis to extract a greatest common model. In: Maciaszek, L.A., Cuzzocrea, A., Cordeiro, J. (eds.) ICEIS 2012 - Proceedings of the 14th International Conference on Enterprise Information Systems, Volume 1, Wroclaw, Poland, 28 June - 1 July, 2012. pp. 27–37. SciTePress (2012)
3. Bachmann, F., Clements, P.: Variability in software product lines. Tech. Rep. CMU/SEI-2005-TR-012, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA (2005), http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=7675
4. Batini, C., Lenzerini, M., Navathe, S.B.: A comparative analysis of methodologies for database schema integration. ACM Computer Survey **18**, 323–364 (1986)
5. Bécan, G., Acher, M., Baudry, B., Ben Nasr, S.: Breathing ontological knowledge into feature model synthesis: An empirical study. Empirical Software Engineering **21** (03 2015). https://doi.org/10.1007/s10664-014-9357-1
6. Carbonnel, J.: L'analyse formelle de concepts : un cadre structurel pour l'étude de la variabilité de familles de logiciels. PhD Thesis, Université de Montpellier (2018)
7. Carbonnel, J., Huchard, M., Nebut, C.: Modelling equivalence classes of feature models with concept lattices to assist their extraction from product descriptions. J. Syst. Softw. **152**, 1–23 (2019). https://doi.org/10.1016/j.jss.2019.02.027, https://doi.org/10.1016/j.jss.2019.02.027

8. Carbonnel, J., Huchard, M., Nebut, C.: Towards the Extraction of Variability Information to Assist Variability Modelling of Complex Product Lines. In: Proceedings of the 12th International Workshop on Variability Modelling of Software-Intensive Systems - VAMOS 2018. pp. 113–120. ACM Press, Madrid, Spain (2018). https://doi.org/10.1145/3168365.3168378, http://dl.acm.org/citation.cfm?doid=3168365.3168378
9. Daniel-Vatonne, M., Hemce, C.: On a tree-like representation for symbolic-numeric data and its use in galois lattice method. In: Proceedings of 18th International Conference of the Chilean Computer Science Society (SCCC '98), November 12-14, 1998, Antofagasta, Chile. pp. 48–57. IEEE Computer Society (1998). https://doi.org/10.1109/SCCC.1998.730782, https://doi.org/10.1109/SCCC.1998.730782
10. Dolques, X., Le Ber, F., Huchard, M.: AOC-posets: a scalable alternative to Concept Lattices for Relational Concept Analysis. In: CLA: Concept Lattices and their Applications. pp. 129–140. La Rochelle, France (Oct 2013), https://hal.archives-ouvertes.fr/hal-00916850
11. Ganter, B., Kuznetsov, S.O.: Pattern Structures and Their Projections. In: 9th Int. Conference ICCS'01, Stanford, CA, USA. pp. 129–142 (2001)
12. Ganter, B., Wille, R.: Formal Concept Analysis - Mathematical Foundations. Springer (1999)
13. Kang, K., Cohen, S., Hess, J., Novak, W., Peterson, A.: Feature-Oriented Domain Analysis (FODA) Feasibility Study. Tech. Rep. CMU/SEI-90-TR-021 (1990)
14. Leeuwenberg, A., Buzmakov, A., Toussaint, Y., Napoli, A.: Exploring pattern structures of syntactic trees for relation extraction. In: Baixeries, J., Sacarea, C., Ojeda-Aciego, M. (eds.) Formal Concept Analysis - 13th International Conference, ICFCA 2015, Nerja, Spain, June 23-26, 2015, Proceedings. Lecture Notes in Computer Science, vol. 9113, pp. 153–168. Springer (2015). https://doi.org/10.1007/978-3-319-19545-2_10, https://doi.org/10.1007/978-3-319-19545-2_10
15. Loesch, F., Ploedereder, E.: Restructuring variability in software product lines using concept analysis of product configurations. In: 11th European Conference on Software Maintenance and Reengineering, Software Evolution in Complex Software Intensive Systems, CSMR 2007, 21-23 March 2007, Amsterdam, The Netherlands. pp. 159–170 (2007). https://doi.org/10.1109/CSMR.2007.40
16. Ramos, J.: Using tf-idf to determine word relevance in document queries (01 2003)
17. Ryssel, U., Ploennigs, J., Kabitzsch, K.: Extraction of feature models from formal contexts. In: Software Product Lines - 15th International Conference, SPLC 2011, Munich, Germany, August 22-26, 2011. Workshop Proceedings (Volume 2). p. 4 (2011). https://doi.org/10.1145/2019136.2019141
18. Stumme, G., Maedche, A.: Ontology merging for federated ontologies on the semantic web. In: International Workshop for Foundations of Models for Information Integration (FMII-2001). pp. 413–418 (2001)

Multimodal Clustering with Evolutionary Algorithms

Mikhail Bogatyrev[1][0000-0001-8477-6006], Dmitry Orlov[1] and Tatyana Shestaka[1]

[1] Tula State University, 92 Lenin ave., Tula, Russia
okkambo@mail.ru

Abstract. The paper considers the application of evolutionary clustering algorithms on multidimensional formal contexts in order to solve fact extraction problem on database data. Formal contexts are built on the data that is the result of a database query. Clustering on such contexts allows one to find certain data combinations in clusters that can be interpreted as facts. Evolutionary clustering algorithm is chosen as an alternative to the FCA-based multimodal clustering algorithms. It is applied in solving the problem of phenotyping complications of myocardial infarction, formulated on the data of patient history, treatment methods and treatment results. The results of the work of evolutionary algorithms for their various parameters are presented.

Keywords: Evolutionary Computation, formal context, multimodal clustering fact extraction.

1 Introduction

A well-known problem in cluster analysis is the problem of interpreting results of clustering. Any clustering algorithm uses some proximity measure defined on the set of objects to be clustered. Therefore, the "meaning" of the resulting clusters is determined primarily by the used measure: the objects united into one cluster because we found them coinciding according to the chosen criterion. An attempt to interpret clusters from any other positions, for example, user-oriented, actually means switching to another proximity measure of the of objects, possibly more complex and not formalized.

Formal Concept Analysis (FCA) [3] offers a different approach to this problem. In FCA, clustering is used not on one, but on two, three and, in general, on an arbitrary number of sets, this is biclustering, triclustering and multimodal clustering [9]. Here, each cluster is a combination of data from clustered sets. The very fact of combining certain data in a cluster can be important from the user's point of view and carry new information. This interpretation of clusters is not directly related to the proximity measure of objects. The methods of multimodal clustering of FCA do not use a proximity measure in the traditional sense. Clustered objects are close if they are connected to each other by means of a relation that defines a formal context, and satisfy certain conditions of closure with respect to operators applied to elements of the data sets of formal context. This can be, for example, the Galois transformation which beget formal concepts or the prime and box operators which beget clusters [3, 13].

FCA-based methods of bi- and triclustering have been studied in sufficient detail [4, 7, 13]. There are also certain solutions for *n*-dimensional multimodal clustering [8, 9].

When applying FCA-based clustering methods to even not large data, a very large number of clusters is usually obtained, which makes it difficult to interpret them. In [18] and [19], a number of solutions to this problem have been proposed: algorithms for finding clusters of a given density and clusters with close values were developed, concept interestingness measures were introduced.

In this paper, it is proposed the solution of the problem of multimodal clustering of data of formal contexts created on the results of queries to databases. A query to an extensive database can return a large number of records with a large number of attributes. The representation of the query results in the form of a formal context with the subsequent finding clusters on it allows us to solve the problem of fact extraction from the database. Here, the facts are interpreted just as combinations of certain attributes in clusters. An evolutionary computation is used as a clustering method since it has a some advantages for applying in clustering, which are discussed below.

This work is a part of research aimed at modernizing the framework for evolutionary modelling [17] and applying it to new data structures.

The rest of the paper is organized as follows. We do not expound the known FCA statements, taking into account the topic of the workshop. In Section 2, there is brief description of evolutionary approach to multimodal clustering. In the Section 3, relations between evolutionary clustering and FCA are highlighted. The results of experimental study of proposed approach are presented in the Section 4. They are illustrated on the task of phenotyping of disease of myocardial infarction.

2 Evolutionary Approach to Multimodal Clustering

Evolutionary Approach to Multimodal Clustering is based on Evolutionary Computation [12]. Evolutionary computation is a term referring to several methods of global optimization, united by the fact that they all use the concept of the evolution of a set of solutions to an optimization problem, leading to solutions corresponding to the extreme value of some function that sets the optimization quality criterion. Evolutionary computation is effective when working with multimodal functions. If such a function has a global extremum, the evolutionary algorithm finds solutions corresponding to the range of values of the quality function that are sufficiently close to the that global extremum.

2.1 Principle of Evolutionary Computation

Let $\boldsymbol{X}$ is a set of solutions of a problem. Every solution $x \in \boldsymbol{X}$ can be characterized by a quality measure named as *fitness function* $f(x)$.

Let solutions of a problem depend on a set of parameters $\boldsymbol{P}$. Such a dependence may be very complex and not being expressed analytically. In this case, it is convenient to present the solution of the problem in the form of a black box. The black box inputs are the values of the parameters, and at the outputs we get the corresponding solutions, for which we calculate the values of the fitness function.

Most problems being solved by using Evolutionary Computation can be formulated as the following optimization problem: it is required to find optimal values of parameters p^*which deliver maximum value of fitness function, so the following is true:

$$p^* = \underset{p^* \in P}{argmax}\, f(x) \tag{1}$$

Parameter values indirectly determine the values of the fitness function calculated on the black box output, so in the expression (1) they were defined as arguments of fitness function.

Evolutionary approach to solving this problem consists in the following.

Building encoding scheme. *Encoding scheme* is the mapping φ: $P \rightarrow S$ where set S contains objects which *encode* parameters from P. Genetic algorithms, which are widely used in Evolutionary Computation often use binary encoding and every value of $p \in P$ is represented as binary string named as *chromosome*. Encoding scheme is not necessarily binary (as it is not binary in Nature): every string position contains a symbol (*gene*) from *encoding alphabet*, and there are variants of alphabets applied in encoding schemata [11]. However, necessarily there exists an inverse mapping φ^{-1}: $S \rightarrow P$, so for every $s \in S$ there exists $p \in P$.

Actually encoding is very important and represents the essence of evolutionary approach. There is *an atomic* principle of encoding which claims that encoding scheme has to be such that it generates minimal elements which influence on the values of elements of the set of solutions X. As in biology, heredity theory claims that gene (strictly gene combinations) is the minimal element which really determines individual characteristics, as here, in Evolutionary computation, atomic encoding principle plays the same role.

Evolutionary algorithm. For given encoding scheme, the following algorithm solves the problem (1).

A. Randomly generate an initial set (population) S_0 of objects from S.

B. Start *evolution* of the populations by applying a set of operators $\boldsymbol{A}$ to population $\boldsymbol{S}_0$ and further iteratively so that for every $\boldsymbol{S}_{k+1} = \boldsymbol{A}(\boldsymbol{S}_k)$ exists at least one

$$\boldsymbol{f}[\boldsymbol{\varphi}^{-1}(s_{k+1})] \geq \boldsymbol{f}[\boldsymbol{\varphi}^{-1}(s_k)], \tag{2}$$

where $s_k \in \boldsymbol{S}_k$ and $s_{k+1} \in \boldsymbol{S}_{k+1}$.

C. Finish the evolution of the population in accordance with the stopping criterion. Most often, the criterion for stopping is the immutability of the fitness function values over several steps of evolution.

If the set of operators $\boldsymbol{A}$ consists of genetic operators of *selection*, *mutation* and *recombination* (crossover) then evolutionary algorithm is named as *genetic algorithm* [11].

Selection works so that condition (2) is supported by the following "biological" principle: good parents produce good offspring (that is not true in Nature). Therefore, the higher fitness chromosomes have more opportunity to be selected than the lower ones and good solution is always alive in the next generation.

Crossover is the genetic operator that mixes two chromosomes together to form a new offspring. It does mixing by replacing fragments of chromosome's code divided in certain one or several randomly selected points.

Mutation involves modification of the gene values by randomly selecting new value from the alphabet at random point in the strings of genes.

Being realized, the algorithm (A. – C.) provides a fast and fairly accurate solution of the problem (1).

Fairly accurate means that evolutionary algorithm stops in a neighbourhood of global extreme of fitness function *f*. The size of a neighbourhood around extreme depends on the fitness function and parameters of genetic operators. When evolutionary algorithm works too fast it may stop at local extreme. This feature is traditionally considered as the lack of the algorithm but it may be useful for clustering since local extreme of quality measure may be semantically "better" than global extreme. In our experiments we have observed just that situation.

Operating speed of genetic algorithms could not be high because they have to manage not one but a whole set of possible solutions and evaluate fitness function N times on every step of evolution, where N is the size of population. Nevertheless, they are fast as compared to other algorithms for solving the problem (1) [12].

2.2 Evolutionary Multimodal Clustering

Evolutionary approach to clustering has quite a long history [10] and has contemporary applications [12]. Most applications belong to the field of gene expression analysis [2, 6, 12].

The gene expression data (GED) has been presented in two variants. These are either matrices containing expression values corresponding to various experiment conditions, or three-dimensional tensors, where a discrete time scale is used as the third dimension. Genetic algorithms with a full set of selection, mutation and crossover operators are used as evolutionary algorithms. The features of the genetic algorithms used here are determined by the choice of the chromosome encoding scheme, the fitness function and genetic operators.

In clustering, the encoding of chromosomes is the central problem on which the success of problem solution depends. Several encoding schemes have been proposed in this area [12]. Among them there are *binary encoding* and *integer encoding*. Some of them is shown on the Fig. 1 [10, 17]. In the binary encoding scheme for clustering, the length of chromosome may be very large if it corresponds to the number of clustering objects. Integer encoding is more compact but it is naturally redundant since different genotypes may correspond to the same clustering solution.

In [17] we have proposed the simple *chain-encoding scheme* also shown on the Fig. 1 which is not redundant and demonstrated its effectiveness in clustering when proximity measure is Euclidean. Another important advantage of the proposed encoding is that it provides quite fast work of the algorithm even on long chromosomes due to quasi parallelization of calculations: several genes in the chromosome may point to the same cluster simultaneously, so clusters are formed in a quasi parallel way. Below we also tested this encoding scheme in the current research for non-Euclidean proximity measure.

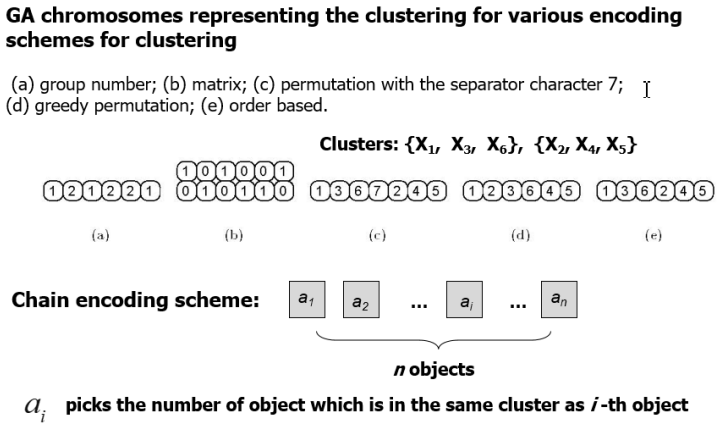


Fig. 1. Some encoding schemes for chromosomes in evolutionary algorithms for clustering.

As already mentioned, chromosomes in evolutionary clustering algorithm can have a significant length. The natural solution, which is known from practice [12, 15], is the use chromosomes with composite parts of which correspond to the data in the measurements. For GED, such chromosomes have two or three sections corresponding to genes and experiment conditions and third section corresponding to time stamps. Since number of genes in experiments may be over dozens of thousands the length of GED chromosomes may be giant. Nevertheless, the computational problem of processing very long chromosomes (usually binary) is solved now [15].

The application of genetic operators to such chromosomes has its own peculiarities. In the studying of gene expression, the genetic crossover and mutation operators are used in all sections of chromosomes. However, in other tasks, this is not justified, since the permutations of genes in certain places of the chromosomes contradict the meaning of the data and the atomic principle mentioned above.

The application crossover and mutation in each section of the chromosome entails large coverage of the search space and, possibly, fast convergence the algorithm to local extrema. However, there are other options for the implementation of operators, not necessarily covering all sections of the composite chromosomes. If, nevertheless, multisectional variants of genetic operators are used, parallelization of computations in accordance with the sectional arrangement of chromosomes is preferable.

In general, the most evolutionary clustering algorithms use fitness functions based on the distance between objects and either clusters' centroids or medoids [12].

If formal contexts are used as input data, then such proximity measures are not very effective, since instead of distances between objects and clusters (when centroids and medoids are used), the quality of clustering is characterized by such cluster' parameters as cluster density and volume of clusters.

All these and some other conditions were taken into account by when we modified our evolutionary modeling environment [17], which we used in this study.

3 Evolutionary Clustering and FCA

In the FCA, the application of Evolutionary Computation may be realized in two ways.

In the first way, Evolutionary Computation is used in FCA algorithms for constructing formal concepts, bi- and triclustering and for clustering of higher orders. Hybrid algorithms on the basis of existing FCA ones are created, and certain parameters of them are manipulated using Evolutionary Computation. The second way is creation of evolutionary algorithms for processing formal contexts as an alternative to existing FCA algorithms.

Our work relates mainly to the second way. However, we use OAC-triclustering [22] when processing formal contexts, so formally our approach partially belongs to the first way too.

A multidimensional, *n*-ary formal context is defined by a relation $R \subseteq D_1 \times D_2 \times ... \times D_n$ on data domains $D_1, D_2, ..., D_n$. The context is an $n+1$ set:

$$\mathbb{K} = <K_1, K_2, ..., K_n, R> \tag{3}$$

where $K_i \subseteq D_i$. The data from domains $D_1, D_2, ..., D_n$ is placed in a database and K_i may be treated as results of queries to database. Using queries, we can form formal contexts of certain dimension and content.

According to multimodal clustering, for any dimension of formal context, the purpose of its processing is to find *n* - sets which have the closure property [9]:

$$\forall u = (x_1, x_2, ..., x_n) \in X_1, X_2, ..., X_n, u \in R, \tag{4}$$

$\forall j = 1,2,...,n, \forall x_j \in D_j \setminus X_j < X_1, ..., X_j \cup \{x_j\}, ..., X_n >$ does not satisfy (4). The sets $H = < X_1, X_2, ..., X_n >$ constitute *multimodal clusters*. The multimodal *n*-adic concepts of an *n*-dimensional context (3) are exactly the maximal *n*-tuples with respect to component-wise set inclusion [8].

Two circumstances are important when applying multidimensional contexts to data analysis. The first is that not only formal concepts, but also multidimensional clusters are important for knowledge discovering from data. Multidimensional clusters are characterized by density, and formal concepts are absolutely dense clusters [7, 14].

The second feature is important for the interpretation of clusters. Each cluster is a combination of subsets of data from different domains. The very fact of combining certain data with each other can be of interest. It is this version of fact extraction that we use in this work. However, the higher the dimension of the cluster, the less certain is the information that is represented by the combination of data, and additional analysis of the clusters is required to refine it. Therefore, high-dimensional clusters are not built and mainly three-dimensional formal contexts are the subject of research here [14, 22, 24]. For simplicity, we also illustrate our approach with three-dimensional formal contexts.

The three-dimensional triadic context (tricontext) of the form $\mathbb{K} = (K_1, K_2, K_3, I)$, where $I \subseteq K_1 \times K_2 \times K_3$ is a ternary relation and in general $I \not\approx R$ after querying to

database. Traditionally, subsets K_1, K_2, K_3 are interpreted as *objects*, *attributes* and *conditions* (OAC), which have a specific meaning based on the content of the database.

The kind of triclusters as OAC-triclusters are studied in detail and demonstrated effectiveness in applications [22]. For triadic context $\mathbb{K} = (K_1, K_2, K_3, I)$ OAC-tricluster is defined in the form

$$\mathbb{C} = (X, Y, Z), X \subseteq K_1, Y \subseteq K_2, Z \subseteq K_3 \quad (5)$$

OAC-triclusters are characterized by cluster density [14], the presence of similar values in clusters [18], and interestingness measures [19].

We use cluster density, and volume of a cluster in our evolutionary clustering algorithm. The cluster density is defined as

$$d(\mathbb{C}) = \frac{|I \cap (X \times Y \times Z)|}{|X| \times |Y| \times |Z|}, \quad (6)$$

and volume of a cluster has the following form

$$\mathrm{v}(\mathbb{C}) = |X| \times |Y| \times |Z| \quad (7)$$

3.1 Genetic Clustering Algorithm

Algorithm 1 shown below is genetic algorithm, which realizes evolutionary algorithm A-C from the Section 2.1. Its chromosomes *chrom* may be encoded by two variants.

The first variant is classical one when processing GED. For three-dimensional formal context of GED, there are three sections in chromosomes, for example, “genes”, “conditions” and “time stamps”. Crossover and mutation operate in all three sections to maximize search space coverage [11]. However, among the new chromosomes generated in this way, there may be incorrect chromosomes, which do not correspond to the data in the original tensor. The *doMultipleCrossover* function accesses the original tensor in order to filter out the wrong chromosomes.

In the second variant of encoding, chromosomes have only one section containing positions of objects being clustering. Filtering out the wrong chromosomes is not needed but the algorithm may produce not proper solutions corresponded to local optima of fitness function. This variant of encoding is convenient for implementing FCA operators for clustering. Population of chromosomes represents variants of clustering and chromosomes contain only references to objects, so the corresponding them clusters can be constructed, for example, using OAC operators [22].

doSelection function realizes selection chromosomes according to selection method. There are *proportional*, *random universal*, *tournament* and *truncation* selection methods [11] realized in the algorithm.

The stopping criterion is that the fitness function of the population does not change with a certain accuracy over several steps of evolution. It may fail, and then the algorithm executes the specified maximum number of steps.

Algorithm 1 Genetic clustering algorithm

Input: *tensor* is multidimensional context as the set of *n* samples on the axes of measurements;
Parameters:
sizePop is the size of population of chromosomes;
numpoints is the number of points of crossover;
mutationRate is the probability of mutation;
coefDensity is the cluster density-scaling factor;
coefSize is the cluster volume-scaling factor;
limitPop is the maximal number of populations;
sel is the type of selection;
countPop is the number of steps of evolution;
popFitness is the value of the fitness function for the entire population.
Output: *clusters* is the set of clusters in the form of a set of subsets.

population ⟵ *createPopulation*[*tensor, sizePop*] creating a population of chromosomes *chrom*

```
1: while countPop ≤ limitPop do
2:     while stopping[population] is false do
3:         for all chrom do
4:             clusterDensity[chrom, tensor]
5:             clusterVolume[chrom, tensor]
6:             fitnessFunction[chrom, tensor, coefDensity, coefSize]
7:         end
8:         popFitness[population] calculating the value of the fitness function
                                  for the entire population.
9:         doMultipleCrossover[{chrom1, chrom2}, numpoints, tensor]
10:        doMutation[chrom, mutationRate, tensor]
11:        doSelection[chrom, popFitness, sel]
12:        for all chrom do
13:            {clusters} ⟵ getSubTensorChrom[chrom, tensor] forming
                            clusters from tensor
14:        end
15:    end
16: end
```

The purpose of other functions of the algorithm is clear from their names.

4 Experimental Study

Experimental study of the proposed approach was carried out in order to solve some tasks related to the problem of phenotyping diseases. Disease phenotyping refers to the determination of the form of the disease based on the clinical profile. A clinical profile

is a cluster that can include various data describing both the disease itself and the methods of its treatment, as well as the conditions of patients and sometimes the treatment results.

Our goal was also to study the efficiency and performance of the evolutionary clustering algorithm for its various parameters.

4.1 Data Sets

Usually the whole data about patients and disease is stored in clinical database and the data sets for the study can be obtained by performing queries to the database.
Depending on the database model and the DBMS platform, queries may be intellectual of some degree and DBMS generates corresponding complicated results in the output. However, relational databases and the SQL query language are still commonly used here. Standard results of the queries are tables, the fields of which contain data corresponding to the attributes of patients, diseases and treatment methods. Such tables constitute as binary as multi-valued formal contexts. Solving the clustering problem on such contexts, we get combinations of objects and attributes in clusters, which are further analyzed as sources of facts. A contexts of dimensions greater than two can be built on the results of queries, if we are interested in additional attributes retrieved from the database.

We use Myocardial Infarction Complications Data Set [16] for experiments. It contains information about 1700 patients having disease of myocardial infarction. All patients are anonymous and presented with identification numbers (ID). We use seven formal contexts acquired from the whole set which number of objects and attributes are shown in Table 1 where ECG is electrocardiogram. Among attributes, there are ones about patients (ID only), their anamnesis, their treatment methods, and complications after the treatment. An attribute may be binary or has a value as natural or real number.

Table 1. Number of objects and attributes of formal contexts

Context	Objects	Attributes
Anamnesis	1700	33
Therapy	1700	24
Analyzes	1700	19
Infarct	1700	6
ECG	1700	27
Therapy results	1700	14
Full data	1700	123

Some formal contexts such as Therapy, Analyzes and Therapy results have a third dimension in the form of days. The standard maximum treatment time for myocardial infarction is 21 days (in Russia), which defines the scale of the third dimension.

4.2 Evolutionary Clustering

Evolutionary clustering was performed using variants of genetic Algorithm 1 with two different encoding schemes and various types of crossover.

Chromosome encoding. After analyzing the existing variants of chromosome encoding [12], we settled on two of them. The first variant is our chained integer-encoding scheme [17] showed on Fig. 1.

The second encoding scheme is a binary scheme organized according to the principle of "one chromosome – one cluster". It has one, two or three sections in chromosomes according with the variant of encoding (see Section 3.1) and dimension of a context. Chromosomes for three-dimensional contexts have sections "patients", "attrbutes" and "days". In the sections, a number of gene is the number of patient, number of attribute from corresponding context from the Table 1 or number of a day according with objects order in the corresponding subsets in formal tricontext. Different chromosomes form different clusters. Because of the evolution of many such chromosomes, really k different chromosomes from n members of the population should remain. In this case, it turns out that some objects will be included in different clusters, i.e. there will be an intersection of clusters.

Fitness function. As in FCA, we control cluster density (6), its volume (7) and special kind of interestingness. There is the trade-off problem between the density and the volume of triclusters [7]. Depending on the data, density and volume may be contradictory characteristics of clusters. Myocardial infarction data are sparse, and if we collect enough units in a cluster, it will be simultaneously voluminous. Therefore, we do not use the volume of clusters in the fitness function, but only use their density. Nevertheless we calculate cluster volumes during evolution.

For the binary encoding scheme, fitness function has the form:

$$f(d) = \frac{1}{N}\sum_{i=1}^{N} \alpha_i d(C_i), \tag{8}$$

where α_i is user defined coefficient, which in general depends on cluster density, N is the number of chromosomes in population which is equal to the maximal number of clusters.

For the chained integer-encoding scheme fitness function is the following:

$$f(d) = \frac{1}{N}\sum_{j=1}^{N}\frac{1}{K_j}\sum_{i=1}^{K_j} \alpha_i d(C_i) \tag{9}$$

where K_j is the number of clusters in the j-th chromosome.

Interestingness of a cluster. It is known in clustering analysis that "*the criteria relate quite indirectly to the major goal of clustering which is improving of our understanding of the world*" [21]. According to the fitness of the chromosomes, the whole fitness of population is namely such criterion. It hides the features of individual chromosomes. But if selection leaves chromosomes with maximum fitness, then there is a chance that they will lead evolution to good solutions. Patient ID values found in clusters, other

attributes corresponding to them from the “treatment” and “treatment outcomes” domains are evaluated for the presence of information in them that can be treated as facts. The formal criteria for selecting such “interesting” clusters are:

- the presence of a single cluster at the end of evolution;
- the most dense clusters among the received;
- clusters of the maximum volume among received;
- clusters with given values of density and volume.

4.3 Fact Extraction with Clustering

The most serious complication of a myocardial infarction is a lethal outcome of the disease. In our data set, the lethal outcome is set by the attribute LET_IS, which has following 8 values: 0: unknown (alive), 1: cardiogenic shock, 2: pulmonary edema, 3: myocardial rupture, 4: progress of congestive heart failure, 5: thromboembolism, 6: asystole, 7: ventricular fibrillation. We selected attributes related to the treatment of patients to find out whether the treatment affects the lethal outcome. For this purpose, the formal context was constructed, containing 27 attributes and 110 objects as the numbers of patients who had a lethal outcome of any of the 7 variants. A similar formal context was also constructed for patients who did not have a lethal outcome. It contains 1590 objects. We have added a third dimension to these contexts, reflecting the use of drugs on certain days. For example, a point with coordinates (7, NA_R_1_n, 1) corresponds to a unit, which means that the patient number 7 got opioid drugs at the first day of hospitality.

To solve the task of the effect of patient treatment on the lethal outcome, triclustering was performed in both contexts using an evolutionary algorithm.

Facts Extracted. We were interested in special clusters. First of all, these are clusters with large groups of patients characterized by certain combinations of attributes from the domains "patient", "treatment", "treatment results". Several such groups were obtained.

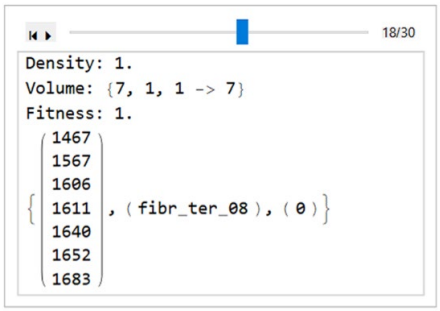
1. We have found that the lethal outcome of myocardial infarction is inherent in elderly patients over 60 years of age. This fact is consistent with the known data of cardiology.
2. In more detail, cases of heart attack in the anamnesis correlate with a fatal outcome, which also looks natural.

For both this groups of patients, we found absolutely dense clusters built on tensors with age and anamnesis attributes.

Unexpected result. We have found one unexpected result, which is as follows. On the data of myocardial infarction, there are stable (not changing according with different parameters of the genetic algorithm) and rather dense clusters in which a subgroup of patients with a lethal outcome have not got certain drugs. At the same time, patients with a non-lethal outcome had these drugs.

Comparison with Data-Peeler. We were also interested in absolutely dense clusters, the formal concepts. As expected, there were few such clusters, which follows

from the sparsity of the data. One of them is shown in Fig. 2. In it, we see that 7 patients had no fibrinolytic therapy by Streptodecase (attribute fibr_ter_08) what is confirmed by the query to the database.



Select 1 Results (MIC) - ISQL Session 1

Id	Age	Sex	Fibr Ter 08
1467	61	0	0
1567	67	0	0
1606	60	0	0
1611	55	1	0
1640	57	0	0
1652	68	1	0
1683	61	0	0

Fig. 2. The dense cluster and the query result.

To compare our results with well known another algorithm, we selected Data-Peeler [9] and modernized its code [23] by adding graphical user interface. Comparison of the results is shown in Table 2.

Table 2. Clustering results compared with Data-Peeler

Formal context	Number of clusters	Number of dense clusters	Number of Data-Peeler concepts
Anamnesis	30	14	449639
Therapy	30	19	28599
Analyzes	30	17	162
Infarct	30	20	65
ECG	30	10	689011
Therapy results	30	12	7798
Full Data	30	4	12564890

The results in the last row of Table 2 can be explained by the high sparsity of data in this formal context. Accordingly, the Data-Peeler algorithm has built a lot of small concepts.

4.4 Algorithm Performance.

The results of the algorithm performance study are as follows.

1. The algorithm processes very long three-section chromosomes of about 2000 genes fairly quickly. This allowed us to perform experiments in a wide range of changes in the parameters of the algorithm. Fig. 3 shows clustering execution time for each of

the seven contexts. On the Fig. 3-a it is shown for two-dimensional formal contexts and on the Fig. 3-b it is shown for three-dimensional formal contexts. At the same time, in some contexts, the third dimension was introduced artificially.

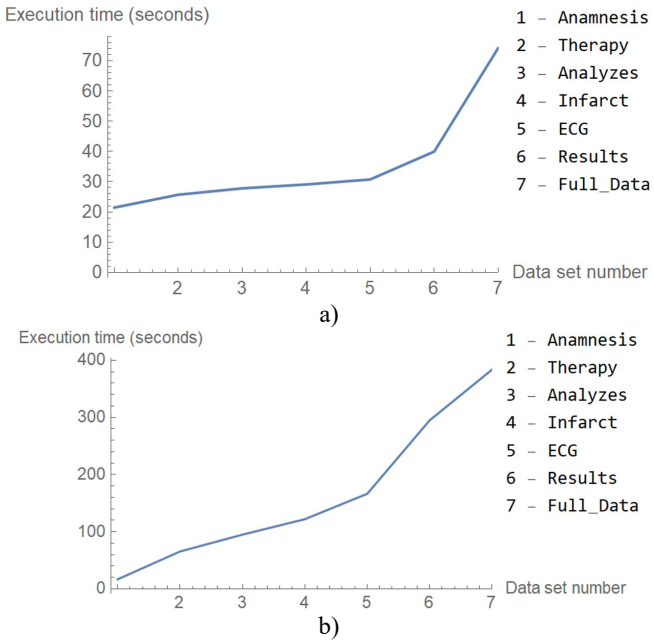


Fig. 3. Clustering execution time for several formal contexts.

The executions were performed on a standard PC 3.59 GHz with 4 Core-Processors and 8 GB RAM.

2. Encoding "one chromosome – one cluster" was more effective than chain encoding on a non-Euclidean fitness functions (6), (7) combining the density and volume of clusters. Since the chain encoding is more complex and multi-linked, the execution of crossover operators on chromosomes led to the "mixing"of genes, the appearance of many "incorrect" chromosomes, and as a result, a decrease in performance.

3. A multipoint crossover is more efficient than a single-point crossover. The use of multipoint crossover in all three sections of chromosomes accelerated the convergence of the algorithm and was effective namely on the encoding scheme "one chromosome – one cluster".

5 Conclusion and Future Work

This paper proposes an approach to multimodal clustering on multidimensional formal contexts using evolutionary computation. This approach is effective in experiments on clustering three-dimensional formal contexts based on data of patients with myocardial

infarction. The genetic algorithm builds dense clusters in any case, even for a local extrema of the fitness function.

The presented experimental results reflect the initial stage of research in this area. In the future, it is planned to do the following.

1. Evaluate the informativeness of the obtained clusters not manually, but using a user interface focused on doctors.

2. Experiments have confirmed that the criteria of cluster density and volume contradict each other. Therefore, it is necessary to apply multi-objective evolutionary clustering with appropriate algorithms.

3. Transition to the dimension of formal contexts greater than three. Separate groups of parameters can be represented as dimensions. Then their combinations obtained in clusters will reflect in more detail the relationships in heterogeneous data.

Acknowledgments. We thank anonymous reviewers for their remarkы and advices. The reported study was funded by Russian Foundation of Basic Research, the research project № 19-07-01178 and RFBR and Tula Region according to research project № 19-47-710007.

References

1. Hartigan J A. Direct clustering of a data matrix. Journal of the American statistical association, 67(337): 123—129 (1972)
2. Madeira SC, Oliveira AL. Biclustering algorithms for biological data analysis: a survey. IEEE/ACM Trans. Comput. Biol. Bioinform. Jan-Mar;1(1):24-45. (2004) DOI: 10.1109/TCBB.2004.2.
3. Ganter, Bernhard; Stumme, Gerd; Wille, Rudolf, eds., Formal Concept Analysis: Foundations and Applications, Lecture Notes in Artificial Intelligence, No. 3626, Springer-Verlag. Berlin (2005) DOI:10.1007/978-3-540-31881-1
4. Kaytoue, Mehdi, Kuznetsov, Sergei, Napoli, Amedeo. Biclustering Numerical Data in Formal Concept Analysis. 135-150. (2011). DOI: 10.1007/978-3-642-20514-9_12.
5. Ignatov D. I., Kuznetsov S. O., Zhukov L. E., Poelmans J., Can triconcepts become triclusters? // International Journal of General Systems, Vol. 42. No. 6 (2013)
6. Henriques R., Madeira S. Triclustering Algorithms for Three-Dimensional Data Analysis: A Comprehensive Survey. ACM Comput. Surv. V. 51. № 5. P. 1–43. (2019) DOI: 10.1145/3195833.
7. Dmitry V. Gnatyshak, Dmitry I. Ignatov, Sergei O. Kuznetsov, From Triadic FCA to Triclustering: Experimental Comparison of Some Triclustering Algorithms. In: Proceedings of the Tenth International Conference on Concept Lattices and Their Applications (CLA'2013), La Rochelle: Laboratory L3i, University of La Rochelle, pp. 249-260, (2013)
8. Voutsadakis, G. Polyadic concept analysis. – Order. Vol. 19 (3). Pp. 295–304 (2002)
9. Cerf, L., Besson, J., Robardet, C., Boulicaut, J.F.: Closed Patterns Meet N-ary Relations. In: ACM Trans. Knowl. Discov. Data. 3, 1, Article 3, 36 p. (2009)
10. R. M. Cole, Clustering with Genetic Algorithms, MSc Thesis, University of Western Australia, Australia (1998)
11. Goldberg D.E. Genetic Algorithms in Search Optimization and Machine Learning. Addison-Wesley, Reading, MA, USA (1989)
12. Hruschka E., Campello R., Freitas A., de Carballo A. A Survey of Evolutionary Algorithms for Clustering. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE

Transactions on Evolutionary Computation. V. 39. P. 133–155. (2009) DOI: 10.1109/TSMCC.2008.2007252.
13. S.O. Kuznetsov and S.A. Obiedkov, Comparing Performance of Algorithms for Generating Concept Lattices. Journal of Experimental and Theoretical Artificial Intelligence, Vol. 14, no. 2-3, pp. 189-216, 2002.
14. Ignatov D. I., Gnatyshak D. V., Sergei O. Kuznetsov, Boris G. Mirkin, Triadic Formal Concept Analysis and triclustering: searching for optimal patterns. In: Machine Learning, April, 2015, pp. 1-32.
15. Ma P., Chan K., Yao X., Chiu D. An evolutionary clustering algorithm for gene expression microarray data analysis. IEEE Transactions on Evolutionary Computation. V. 10. P. 296–314 (2006) doi: 10.1109/TEVC.2005.859371.
16. Myocardial infarction complications Data Set. http://archive.ics.uci.edu/ml/machine-learning-databases/00579/
17. M. Y. Bogatyrev, A. P. Terekhov. Framework for Evolutionary Modelling in Text Mining. - Proceedings of the SENSE'09 - Conceptual Structures for Extracting Natural Language Semantics. Workshop at 17th International Conference on Conceptual Structures (ICCS'09), p.p. 26-37 (2009)
18. Mehdi Kaytoue, Sergei O. Kuznetsov, Juraj Macko, Wagner Meira Jr., Amedeo Napoli, Mining Biclusters of Similar Values with Triadic Concept Analysis. In: Proc. 8th International Conference on Concept Lattices and Their Applications (CLA 2011), INRIA Nancy - Grand Est and LORIA, pp. 175 - 190, 2011.
19. Kuznetsov S., Makhalova T. Concept interestingness measures: a comparative study, in: Proceedings of the Twelfth International Conference on Concept Lattices and Their Applications Clermont-Ferrand, France, October 13-16, 2015 Vol. 1466. Clermont-Ferrand : CEUR Workshop Proceedings. P. 59-72 (2015)
20. Mirkin, B. G., & Kramarenko, A. V. Approximate bicluster and tricluster boxes in the analysis of binary data. In Rough sets, fuzzy sets, data mining and granular computing, LNCS, Vol. 6743, pp. 248–256. (2011)
21. Mirkin, Boris, Muchnik, Ilya. Combinatoral Optimization in Clustering. Handbook of Combinatorial Optimization. D.-Z. Du and P.M. Pardalos (Eds.) pp. 261-329 (2000)
22. Dmitry I. Ignatov, Alexander Semenov, Daria Komissarova, Dmitry V. Gnatyshak: Multimodal Clustering for Community Detection. Formal Concept Analysis of Social Networks 2017: 59-96
23. https://github.com/ibrahim85/d-peeler

On Suboptimality of GreConD for Boolean Matrix Factorisation of Contranominal Scales

Dmitry I. Ignatov[1,2] and Alexandra Yakovleva[1]

[1] National Research University Higher School of Economics, Moscow
[2] St. Petersburg Department of Steklov Mathematical Institute of Russian Academy of Sciences, Russia
dignatov@hse.ru,alexandrayakovlevaa@gmail.com

Abstract. In this paper we study certain properties of the GreConD algorithm for Boolean matrix factorisation, a popular technique in Data Mining with binary relational data. This greedy algorithm was inspired by the fact that the optimal number of factors for the Boolean matrix factorisation can be chosen among the formal concepts of the corresponding formal context. In particular, we consider one of the hardest cases (in terms of the numerous of possible factors), the so-called contranominal scales, and show that the output of GreConD is not optimal in this case. Moreover, we formally analyse its output by means of recurrences and generating functions and provide the reader with the closed form for the returned number of factors. An algorithm generating the optimal number of factors and the corresponding product matrices P and Q is also provided by us for the case of contranominal scales.

Keywords: Boolean Matrix Factorisation, Formal Concept Analysis, Schein rank, generating functions, greedy algorithms

1 Introduction

Boolean data analysis and Formal Concept Analysis are closely related [1]. For example, Boolean matrices describing binary relations can be considered as formal contexts and vice versa, and decomposition of Boolean matrices into the product of two Boolean matrices of possibly smaller sizes is one of such crossroads where two disciplines meet each other. Thus, it was shown that the optimal number of factors, that is the minimal size of common dimension of these two product matrices, can be found based on the family of corresponding formal concepts considered as factors for the original Boolean matrix [2]. Decomposition of object-attribute matrices into products of object-factor and factor-attribute matrices plays important role in Machine Learning and Data Mining [3]. One of the desired properties is the dimensional reduction that normally preserves with high accuracy similarly between objects or attributes in terms of dot product and makes it possible to recover the input matrix [4]. For example, in collaborative filtering domain Boolean Matrix Factorisation (BMF) was on par with the (truncated) Singular Value Decomposition approach in terms of obtained

quality metrics [5,6]. It speeds up the computation on the decomposed matrices and allows finding homogeneous taste communities as those latent factors.

Another fruitful property of Boolean matrices is their cheap bit representation and related bit operations. The only obstacle for Boolean Matrix Factorisation to be widely adopted technique so far is that of determination of the optimal number factors k for Boolean matrices or Schein rank is NP-hard problem [7,2]. So, every good approximate algorithm geared towards minimisation of the number of factors can be taken into account [8].

One of the earlier proposed algorithm for BMF is GreConD. It follows a greedy strategy adding attributes one-by-one with subsequent computation of their closures and is not optimal in general. In this paper we address one very important for practice case of the input for this algorithm, the contranominal scale of arbitrary size n, i.e. square Boolean matrix with all ones except the main diagonal [9]. It is well-known that the number of patterns (formal concepts) for this case is 2^n. It is easy to show experimentally that GreConD is not optimal for this particular case by comparing its output with the theoretically deduced values of Schein rank for contranominal scales. However, the output solution follows an interesting pattern deserving a special treatment in terms of recurrences and generating functions. It allows us to formally analyse the discrepancy between this suboptimal solution and theoretically optimal one. Moreover, to know the theoretically optimal solution as the number of factors does not mean to provide a concrete factorisation. To fill the gap, we sketch a correct algorithm to this end.

The paper is organised as follows. In Section 2, we recall the reader the basic definitions of FCA and BMF and describe GreConD algorithm. In Section 3, we shortly describe GreConD with its pseudocode. In Section 4, we provide the reader with our experimental and theoretical analyses of the algorithm's suboptimality. The penultimate section, Section 5, presents the optimal algorithm to find BMF for formal contexts of contranominal scales. Finally, Section 6 briefly discusses future prospects and concludes the paper.

2 Boolean Matrix Factorisation and GreConD

2.1 BMF based on Formal Concept Analysis

Basic FCA definitions. Formal Concept Analysis (FCA) is a branch of applied algebra and it studies (formal) concepts and their hierarchies [10]. The adjective "formal" indicates a strict mathematical definition of a pair of sets, called, the extent and intent. This formalisation is possible because of the use of the algebraic lattice theory.

Definition 1. Formal context $\mathbb{K}$ *is a triple* (G, M, I)*, where* G *is a set of* objects*,* M *is a set of* attributes*, and* $I \subseteq G \times M$ *is an incidence binary relation.*

The binary relation I is interpreted as follows: for $g \in G$, $m \in M$ we write gIm if the object g has the attribute m.

For a formal context $\mathbb{K} = (G, M, I)$ and any $A \subseteq G$ and $B \subseteq M$ a pair of mappings is defined:

$$A^{\uparrow} = \{m \in M \mid gIm \text{ for all } g \in A\}, \quad B^{\downarrow} = \{g \in G \mid gIm \text{ for all } m \in B\},$$

these mappings define Galois connection between partially ordered sets $(2^G, \subseteq)$ and $(2^M, \subseteq)$ on disjunctive union of G and M. The set A is called *closed set*, if $A^{\uparrow\downarrow} = A$ [11].

Definition 2. *A* formal concept *of the formal context $\mathbb{K} = (G, M, I)$ is a pair (A, B), where $A \subseteq G$, $B \subseteq M$, $A^{\uparrow} = B$ and $B^{\downarrow} = A$. The set A is called the* extent*, and B is the* intent *of the formal concept (A, B).*

It is evident that the extent and intent of any formal concept are closed sets.

The set of all formal concepts of a context $\mathbb{K}$ is denoted by $\mathfrak{B}(G, M, I)$.

The state-of-the-art surveys on advances in FCA theory and its applications can be found in [12,13].

Description of FCA-based BMF. Boolean Matrix Factorisation is a decomposition of the original matrix $I \in \{0,1\}^{n \times m}$, where $I_{ij} \in \{0,1\}$, into a Boolean matrix product $P \circ Q$ of binary matrices $P \in \{0,1\}^{n \times k}$ and $Q \in \{0,1\}^{k \times m}$ for the smallest possible number of k. We define Boolean matrix product as follows:

$$(P \circ Q)_{ij} = \bigvee_{l=1}^{k} P_{il} \cdot Q_{lj},$$

where $\bigvee$ denotes disjunction, and $\cdot$ conjunction.

For example, in collaborative filtering, matrix I can be considered as a matrix of binary relation between set X of objects (users), and a set Y of attributes (items that users have evaluated). In this case, we assume that xIy iff the user x evaluated object y. The triple (X, Y, I) naturally forms a formal context.

Consider a set $\mathcal{F} \subseteq \mathcal{B}(X, Y, I)$, a subset of all formal concepts of context (X, Y, I), and introduce matrices $P_{\mathcal{F}}$ and $Q_{\mathcal{F}}$:

$$(P_{\mathcal{F}})_{il} = \begin{cases} 1, i \in A_l, \\ 0, i \notin A_l, \end{cases} \quad (Q_{\mathcal{F}})_{lj} = \begin{cases} 1, j \in B_l, \\ 0, j \notin B_l. \end{cases},$$

where (A_l, B_l) is a formal concept from $\mathcal{F}$. We can consider decomposition of the matrix I into binary matrix product $P_{\mathcal{F}}$ and $Q_{\mathcal{F}}$ as described above. The following theorems are proved in [2]:

Theorem 1. *(Universality of formal concepts as factors). For every I there is $\mathcal{F} \subseteq \mathcal{B}(X, Y, I)$, such that $I = P_{\mathcal{F}} \circ Q_{\mathcal{F}}$.*

Theorem 2. *(Optimality of formal concepts as factors). Let $I = P \circ Q$ for $n \times k$ and $k \times m$ binary matrices P and Q. Then there exists a $\mathcal{F} \subseteq \mathcal{B}(X, Y, I)$ of formal concepts of I such that $|\mathcal{F}| \leq k$ and for the $n \times |\mathcal{F}|$ and $|\mathcal{F}| \times m$ binary matrices $P_{\mathcal{F}}$ and $Q_{\mathcal{F}}$ we have $I = P_{\mathcal{F}} \circ Q_{\mathcal{F}}$.*

There are several algorithms for finding $P_{\mathcal{F}}$ and $Q_{\mathcal{F}}$ by calculating formal concepts based on these theorems [2].

3 GreConD

There are several algorithms for finding $P_{\mathcal{F}}$ and $Q_{\mathcal{F}}$ by calculating formal concepts based on aforementioned theorems [2]. This paper studies the work of GreConD (Algoritm 2 from [2]), one of the existing algorithms for BMF. GreConD avoids computation of all possible formal concepts and therefore works much faster [2]. Time estimation of the calculations in the worst case yields $O(k|G||M|^3)$ [5], where k is the number of found factors (and can be omitted as a constant term), $|G|$ is the number of objects, $|M|$ is the number of attributes.

Define $\mathcal{U} = \{\langle i, j\rangle | I_{i,j} = 1\}$ for a Boolean matrix I. The main idea of the algorithm is to maximize the set

$$D \oplus y := ((D \cup \{y\})^{\downarrow} \times (D \cup \{y\})^{\downarrow\uparrow}) \cap \mathcal{U}$$

successively adding columns to intent D of formal concept (C, D).

Below we provide pseudocode for GreConD.

Algorithm 3.1 GreConD

```
 1: INPUT: I (Boolean matrix)
 2: OUTPUT: F (set of factor concepts)
 3:
 4: U ← {⟨i, j⟩ | I_{i,j} = 1}
 5: F ← ∅
 6: while U ≠ ∅ do
 7:     D ← ∅
 8:     V ← 0
 9:     while there is j ∉ D such that |D ⊕ j| > V do
10:         select j ∉ D that maximizes |D ⊕ j|
11:             D ← (D ∪ {j})^{↓↑}
12:             V ← |(D^↓ × D) ∩ U|
13:     end while
14:     C ← D^↓
15:     add (C, D) to F
16:     for each ⟨i, j⟩ ∈ C × D do
17:         remove ⟨i, j⟩ from U
18:     end for
19: end while
20: return F
```

The set $\mathcal{U}$ contains not yet covered object-attribute pairs by any of the previously found factors. When the newly found factor (C, D) is added to $\mathcal{F}$, all the pairs from $C \times D$ should be deleted from $\mathcal{U}$ (lines 15-18). When $\mathcal{U}$ is empty, the GreConD terminates (line 6, the main loop). The inner loop (lines 9–13) maximizes the cardinality $D \oplus j$ while it is still possible by examining attributes not in D.

4 GreConD on contranominal scale

In this section we show that GreConD is optimal for ordinal and nominal scales, but not optimal on contranominal scale. We also construct an optimal algorithm for contranominal scale.

4.1 Optimality on ordinal and nominal scales

In FCA, scales are used to represent the so-called multi-valued contexts (cf. relational tables in databases) as one-valued contexts; the latter we also consider here as Boolean matrices.

First, let us consider two elementary scales. The *nominal scale* is defined as a formal context $\mathbb{N}_n = (\{1,\dots,n\},\{1,\dots,n\},=)$ and is used to scale mutually exclusive attributes like traffic light signals (red, green, yellow). The *ordinal scale* is defined as $\mathbb{O}_n = (\{1,\dots,n\},\{1,\dots,n\},\leq)$ and is applied in cases where the values are ordered like university grades (poor, normal, good, excellent).

It follows from our experiment that the number of factors obtained by GreConD on ordinal and nominal scales are equal to the size of scales. We can prove that these numbers are optimal.

Proposition 1. *The number of factors n obtained by GreConD for a nominal scale $\mathbb{N}_n$ is optimal.*

Proof. Note that for a nominal scale of size n any concept with nonempty extent and intent has the form $(\{i\},\{i\})$ $(i \in \{1,\dots,n\})$. Furthermore, the number of formal concepts is equal to the number of factors by definition. □

Proposition 2. *The number of factors n obtained by GreConD for an ordinal $\mathbb{O}_n$ is optimal.*

Proof. Note that for the ordinal scale of size n and for any nonempty $A \subseteq \{max(A),\dots,n\}$ it holds that $A^{\uparrow} = \{1,\dots,n\}$. Besides, $\{max(A),\dots,n\}^{\downarrow} = \{1,\dots,max(A)\}$. Therefore, concepts for the ordinal scale are $(\{1,\dots,k\},\{k,\dots,n\})$ for $k \in \{1,\dots,n\}$. Since GreConD needs to cover every object-attribute pair, each pair $(\{i\},\{i\})$ for $i \in \{1,\dots,n\}$ should be covered as well, which requires exactly n concepts $(\{1,\dots,i\},\{i,\dots,n\})$. □

4.2 Suboptimality on contranominal scale

For every set S the *contranominal scale* is defined as $\mathbb{N}^c_S = (S,S,\neq)$. In what follows, we consider N^c_n with $S = \{1,\dots,n\}$ without loss of generality.

Factorizing contranominal scales of sizes from 1 to 128 by GreConD[3] we obtain a sequence of the number of factors a_n (n is the size of a scale):

$$a_1 = 0, \quad a_2 = 2, \quad a_3 = 3, \quad a_4 = 4,$$

[3] Our Python implementation of GreConD for these experiments: `https://bit.ly/GreConDsub`

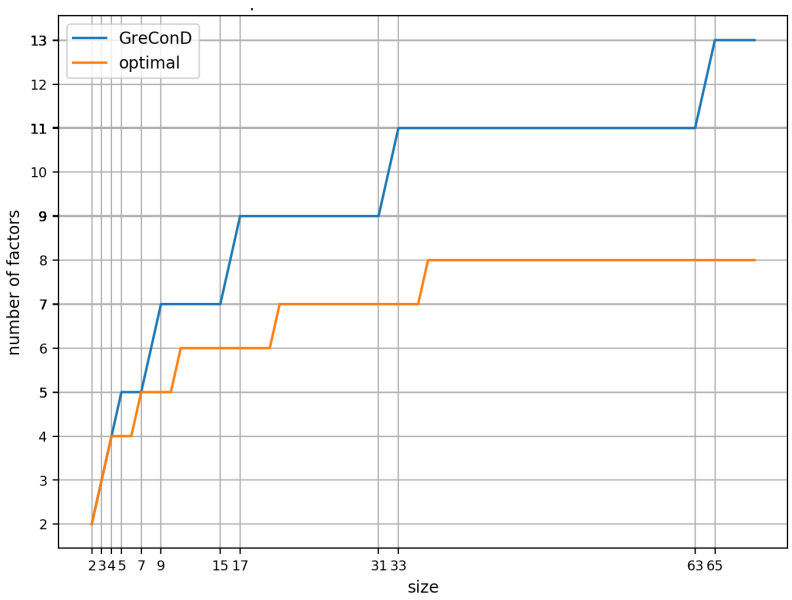


Fig. 1. The number of factors for GreConD and its theoretical optimum for contranominal scales of increasing size.

$$a_5 = a_6 = a_7 = 5, \quad a_8 = 6,$$
$$a_9 = \cdots = a_{15} = 7, \quad a_{16} = 8,$$
$$a_{17} = \cdots = a_{31} = 9, \quad a_{32} = 10,$$
$$a_{33} = \cdots = a_{63} = 11, \quad a_{64} = 12,$$
$$a_{65} = \cdots = a_{127} = 13, \quad a_{128} = 14\ldots$$

Note that the number of obtained factors increases by one when the size of a scale is a power of two or a power of two plus one.

The sequence can be defined as follows:

$$a_1 = 0, a_2 = 2,$$
$$a_n = 2\log_2 n \text{ if } \exists k : n = 2^k,$$
$$a_n = a_{2^{\lfloor \log_2 n \rfloor}} + 1 \text{ for other } n.$$

Thus analytic form for the sequence is the following:

Conjecture. The number of factors obtained by GreConD on contranominal scale is described by the sequence

$$a_n = 2 \cdot \lfloor \log_2 n \rfloor + 1 - [n = 2^{\lfloor \log_2 n \rfloor}],$$

n being the size of a scale.

One way to obtain a simpler closed form the considered sequence is to analyse its generating function [14].

Let $G(z) = \sum\limits_n a_n z^n$ is the associated generating function for the sequence a_n. The sequence a_n can be rewritten in the following way: $a_1 = 0$, $a_2 = 2$, while $a_n = a_{n-1} + \lfloor \log_2 n \rfloor - \lfloor \log_2(n-1) \rfloor + \lceil \log_2 n \rceil - \lceil \log_2(n-1) \rceil$. One can check that one of the respective differences of rounded logarithms takes on 1 when $n = 2^k$ or $n - 1 = 2^k$ for some $k > 0$.

Let us sum $a_n z^n$ as follows:

$$\sum_{n\geq 2} a_n z^n = \sum_{n\geq 2} a_{n-1} z^n + \sum_{n\geq 2} (\lfloor \log_2 n \rfloor - \lfloor \log_2(n-1) \rfloor + \lceil \log_2 n \rceil - \lceil \log_2(n-1) \rceil) z^n$$

Let $U_n = \lceil log_2 n \rceil$ and $L_n = \lfloor log_2 n \rfloor$, then

$$G(z) = zG(z) + \sum_{n\geq 2} L_n z^n - \sum_{n\geq 2} L_{n-1} z^n + \sum_{n\geq 2} U_n z^n - \sum_{n\geq 2} U_{n-1} z^n \ .$$

Now, let $L(z) = \sum\limits_{n\geq 2} L_n z^n$ and $U(z) \sum\limits_{n\geq 2} U_n z^n$, then

$$G(z) = zG(z) + L(z) - zL(z) + U(z) - zU(z) \text{ or}$$

$$G(z)(1 - z) = (L(z) + U(z))(1 - z) \ .$$

For $z \neq 1$ we have

$$a_n = [z^n]G(z) = \lfloor \log_2 n \rfloor + \lceil \log_2 n \rceil \ .$$

Next, we show that the number of factors obtained by GreConD on contranominal scale is not optimal.

First, we provide the definition of Schein rank.

Definition 3. *[15] For vectors v, w the matrix $v_i w_j$ is called cross-vector*[4]*.*

Definition 4. *[15] Schein rank of a Boolean matrix A is the least number of Boolean cross-vectors summing up to A.*

Theorem 3. *[16] Schein rank of contranominal scale of size n equals $N(n)$, where $l = N(k)$ ($k \in \mathbb{N}$) is defined as the least number, such that $k \leq \binom{l}{\lfloor l/2 \rfloor}$.*

We also provide several first values of $N(k)$[5]:
$N(1) = 1, N(2) = 2, N(3) = 3, N(4) = N(5) = N(6) = 4, N(7) = \cdots = N(10) =$

[4] Note that we deal with column vectors according to data analysis conventions; so, $v_i w_j$ is the outer product of v and w.

[5] See also OEIS sequence A305233: `https://oeis.org/A305233`

5,
$N(11) = \cdots = N(20) = 6, N(21) = \cdots = N(35) = 7, N(36) = \cdots = N(70) = 8,$
$N(71) = \cdots = N(126) = 9, N(127) = \cdots = N(252) = 10 \ldots$
Note that for contranominal scales of sizes $2, 3, 4, 7$ GreConD does find Schein rank, i.e. optimal number of factors. However, for the remaining sizes ($n > 1$) GreConD finds suboptimal number of factors.

5 Optimal algorithm for contranominal scale

Let us construct an algorithm that would factorize contranominal scale with optimal number of factors. We use Sperner's theorem.

Definition 5. *A family of incomparable (with respect to set inclusion) sets is called a Sperner family, or an antichain of sets.*

Theorem 4. *[17] (Sperner) For an n-element set the size of a largest antichain does not exceed $\binom{n}{\lfloor n/2 \rfloor}$.*
Equality holds iff an antichain consists of all subsets of size $\lceil n/2 \rceil$ or all subsets of size $\lfloor n/2 \rfloor$.

Theorem 3 [16] states that the optimal number of factors for contranominal scale of size n is equal to $N(n)$. Therefore, BMF with the optimal number of factors (we call it *optimal* BMF) has object-factor matrix of size $n \times N(n)$. From the proof [16] it follows that the minimal set of factors for contranominal scale is an antichain. Next, we show how to find an antichain of a given length n.

Let us find all combinations of elements from the set $\{1, \ldots, N(n)\}$ by $\lfloor N(n)/2 \rfloor$ elements (for example, by Algorithm T from [18][p. 359]). Note that by Sperner's theorem a set of those combinations is the largest antichain for the $N(n)$-element set of factors. Also, $\binom{N(n)}{\lfloor N(n)/2 \rfloor} \geq n$ by definition of $N(n)$. Next, for every combination we make a binary vector r of length $N(n)$ with $r_i = 1 \iff$ the corresponding combination contains the element i. Finally, we obtain object-factor matrix by choosing an n-element subset of binary vectors and placing it in object-factor matrix.

Based on the constructed object-factor matrix, we find the factor-attribute matrix. We apply the derivation operator $^{\downarrow}$ to every factor f in the object-factor matrix, then we apply the derivation operator operator $^{\uparrow}$ to the set of the obtained objects in object-attribute matrix. Finally, we make binary row for the obtained set of attributes and place it in f-row in factor-attribute matrix.

Thus, we get optimal BMF for contranominal scale.

Note that we can simplify the procedure of construction of the factor-attribute matrix using the following property.

Property 1. For contranominal scale of size n and any subsets A and B of sets of objects and attributes respectively it holds that

$$A^{\uparrow} = \{1, \ldots, n\} \backslash A; B^{\downarrow} = \{1, \ldots, n\} \backslash B.$$

Proof. Using the definition of contranominal scale and the derivation operator(s) we get:
$A^{\uparrow} = \cap_{a \in A}(\{1, \ldots, n\} \backslash a) = \{1, \ldots, n\} \backslash A$.
The proof for B is similar. □

Now we can remake the recovering of factor-attribute B matrix from object-factor matrix A. If A_k is the k-th column in the matrix A, then $\sim A_k$ (here $\sim$ is a logical negation) is a k-th row in the matrix B.

Example. Let us demonstrate our algorithm on contranominal scale of size 5.

$N(5) = 4$, hence BMF has 4 factors. Generate 5 different combinations from the set $\{1, 2, 3, 4\}$: $\{1, 2\}, \{1, 3\}, \{1, 4\}, \{2, 3\}, \{2, 4\}$. Therefore, we obtain the object-factor matrix A:
$A = \begin{pmatrix} 1\,1\,0\,0 \\ 1\,0\,1\,0 \\ 1\,0\,0\,1 \\ 0\,1\,1\,0 \\ 0\,1\,0\,1 \end{pmatrix}$. The first column of matrix A consists of vector $\big(1, 1, 1, 0, 0\big)^T$, so vector $\big(0, 0, 0, 1, 1\big)$ is the first row of factor-attribute matrix. Similarly, we fill the rest of the rows in matrix B and obtain the optimal BMF:

$$\begin{pmatrix} 0\,1\,1\,1\,1 \\ 1\,0\,1\,1\,1 \\ 1\,1\,0\,1\,1 \\ 1\,1\,1\,0\,1 \\ 1\,1\,1\,1\,0 \end{pmatrix} = \begin{pmatrix} 1\,1\,0\,0 \\ 1\,0\,1\,0 \\ 1\,0\,0\,1 \\ 0\,1\,1\,0 \\ 0\,1\,0\,1 \end{pmatrix} \circ \begin{pmatrix} 0\,0\,0\,1\,1 \\ 0\,1\,1\,0\,0 \\ 1\,0\,1\,0\,1 \\ 1\,1\,0\,1\,0 \end{pmatrix}$$

Proposition 3. *The number of optimal BMF of contranominal scale of size n is $n!\binom{q}{n}$, where $q = \binom{N(n)}{\lfloor N(n)/2 \rfloor}$.*

Proof. Recall that we choose an n-element set from all the combinations of numbers from the set $\{1, \ldots, N(n)\}$ by $\lfloor N(n)/2 \rfloor$ elements in order to get rows of an object-factor matrix. Further, there are $n!$ ways to arrange every obtained n-element set of combinations as rows of object-factor matrix.

We conclude the proof noting that there is a unique way to build factor-attribute matrix having the object-factor matrix. □

6 Conclusion

In the paper we considered important case for Boolean matrix factorisation based on our experimental and theoretical analyses of the behaviour of the GreConD algorithm. We hypothesise that the number of output factors for the contranomial scales in case of GreConD is $\lfloor \log_2 n \rfloor + \lceil \log_2 n \rceil$ based on the substantial observed fragment of its output for different values of the scale size n.

We have also proposed an optimal algorithm w.r.t. Schein rank to find one out of $n!\binom{q}{n}$ optimal Boolean matrix factorisation for this case, where $q = \binom{N(n)}{\lfloor N(n)/2 \rfloor}$.

As a future research direction we would like to continue our previous investigations of Boolean matrix factorisation for collaborative filtering problems [5,6] with an updated knowledge on suboptimality in case of contranominal scales presence as well to extend this approach to Boolean tensors.

Acknowledgments. The paper was prepared within the framework of the HSE University Basic Research Program and was also supported in part through computational resources of HPC facilities at HSE University. The first author was also supported by Russian Science Foundation under grant 17-11-01276 at St. Petersburg Department of Steklov Mathematical Institute of Russian Academy of Sciences, Russia and by RFBR (Russian Foundation for Basic Research) according to the research project No 19-29-01151. The foundations had no role in study design, data collection and analysis, writing the manuscript, and decision to publish.

We would like to thank Prof. Donald Knuth for personal written explanations of nontrivial pieces from Concrete Mathematics [14] regarding summation properties.

References

1. Janostik, R., Konecny, J., Krajca, P.: Interface between logical analysis of data and formal concept analysis. Eur. J. Oper. Res. **284**(2) (2020) 792–800
2. Belohlavek, R., Vychodil, V.: Discovery of optimal factors in binary data via a novel method of matrix decomposition. Journal of Computer and System Sciences **76**(1) (2010) 3 – 20 Special Issue on Intelligent Data Analysis.
3. Miettinen, P., Neumann, S.: Recent developments in boolean matrix factorization. In Bessiere, C., ed.: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020, ijcai.org (2020) 4922–4928
4. Belohlávek, R., Outrata, J., Trnecka, M.: Impact of boolean factorization as preprocessing methods for classification of boolean data. Ann. Math. Artif. Intell. **72**(1-2) (2014) 3–22
5. Ignatov, D.I., Nenova, E., Konstantinova, N., Konstantinov, A.V.: Boolean matrix factorisation for collaborative filtering: An fca-based approach. In Agre, G., Hitzler, P., Krisnadhi, A.A., Kuznetsov, S.O., eds.: Artificial Intelligence: Methodology, Systems, and Applications - 16th International Conference, AIMSA 2014, Varna, Bulgaria, September 11-13, 2014. Proceedings. Volume 8722 of Lecture Notes in Computer Science., Springer (2014) 47–58
6. Akhmatnurov, M., Ignatov, D.I.: Context-aware recommender system based on boolean matrix factorisation. In Yahia, S.B., Konecny, J., eds.: Proceedings of the Twelfth International Conference on Concept Lattices and Their Applications, Clermont-Ferrand, France, October 13-16, 2015. Volume 1466 of CEUR Workshop Proceedings., CEUR-WS.org (2015) 99–110
7. Miettinen, P., Mielikäinen, T., Gionis, A., Das, G., Mannila, H.: The discrete basis problem. IEEE Trans. Knowl. Data Eng. **20**(10) (2008) 1348–1362
8. Miettinen, P., Vreeken, J.: MDL4BMF: minimum description length for boolean matrix factorization. ACM Trans. Knowl. Discov. Data **8**(4) (2014) 18:1–18:31

9. Albano, A., Chornomaz, B.: Why concept lattices are large: extremal theory for generators, concepts, and vc-dimension. Int. J. Gen. Syst. **46**(5) (2017) 440–457
10. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer, Berlin/Heidelberg (1999)
11. Birkhoff, G.: Lattice Theory. eleventh printing edn. Harvard University, Cambridge, MA (2011)
12. Poelmans, J., Ignatov, D.I., Kuznetsov, S.O., Dedene, G.: Formal concept analysis in knowledge processing: A survey on applications. Expert Syst. Appl. **40**(16) (2013) 6538–6560
13. Poelmans, J., Kuznetsov, S.O., Ignatov, D.I., Dedene, G.: Formal concept analysis in knowledge processing: A survey on models and techniques. Expert Syst. Appl. **40**(16) (2013) 6601–6623
14. Graham, R.L., Knuth, D.E., Patashnik, O.: Concrete Mathematics: A Foundation for Computer Science, 2nd Ed. Addison-Wesley (1994)
15. Kim, K.H.: Boolean matrix theory and applications. Marcel Dekker, New York and Basel (1982)
16. Marenich, E.: Determining the schein rank of boolean matrices. Matrix Methods: Theory, Algorithms and Applications (2010) 85–103
17. Sperner, E.: Ein satz über untermengen einer endlichen menge. Math Z **27** (1928) 544–548
18. Knuth, D.E.: Combinatorial Algorithms. Volume 4A of The Art of Computer Programming. Addison-Wesley Professional (January 2011)

Summation of decision trees

Egor Dudyrev[1][0000−0002−2144−3308]
Sergei O. Kuznetsov[1][0000−0003−3284−9001]

National Research University Higher School of Economics, Moscow, Russia

Abstract. Ensembles of decision trees, like Random Forests are efficient machine learning models with state-of-the-art prediction quality. However, their predictions are much less transparent than those of a single decision tree. In this paper, we describe a prediction model based on a single decision tree in terms of Formal Concept Analysis. We define a differential way to describing a decision rule. We conclude by presenting an approach to summing an ensemble of decision trees into a single decision semilattice with the same predictions.

Keywords: Ensembles of Decision Trees · Formal Concept Analysis · Supervised Machine Learning.

1 Introduction

A decision tree [4] is a popular machine learning model. It can help face the challenge of interpretable machine learning. However, usually it is too simplistic to show good learning performance. Ensembles of decision trees show better learning quality. Some of them – such as random forest [3] and gradient boosting [7] – are considered state-of-the-art. However, ensembles miss the high interpretability of a single decision tree.

Formal Concept Analysis (FCA) [8] is a mathematically well-founded theory aimed at data analysis. In [1], [2], [9], [10], researchers show the connection between decision trees and FCA.

This paper continues our study on the connection between FCA and decision trees started in [6]. In that paper, we have presented the following pipeline. First, we convert a decision tree into a concept lattice. Second, we fuse an ensemble of concept lattices into a single concept lattice. Third, we convert a concept lattice into a decision (semi)lattice: a supervised machine learning model with prediction quality non-inferior to that of ensembles of decision trees.

In what follows, we present a method for constructing a decision semilattice that outputs *the same* predictions as an ensemble of decision trees. We propose a differential way for describing a decision rule and, consequently, a decision tree and a decision semilattice. We finish by summing the ensemble of decision trees into a single decision semilattice.

2 Basic definitions

For standard definitions of FCA and decision trees, we refer the reader to [8] and [4], respectively.

Here we use binary attributes to describe the algorithms. In the experimental section, we extend the algorithm to processing numerical data with interval pattern structures [11].

The standard FCA framework operates with a set M of binary attributes. In what follows we often replace a set of attributes M by a set $M^\star$ that consists both of attributes $m \in M$ and their complements $\overline{m}$ ("not m"):

$$M^\star = M \cup \{\overline{m} \mid \forall m \in M\} \tag{1}$$

3 The proposed approach

3.1 Decision tree and decision semilattice

Definition 1. *A* decision rule (p, t) *is a pair of a subset of attributes* $p \subseteq M^\star$ *called a* premise *and a real number* $t \in \mathbb{R}$ *called a* target. *The attributes in the premise* p *are non-complementary, i.e.* $\forall m \in M^\star$: *if* $m \in p$ *then* $\overline{m} \notin p$.

Given a description $x \subseteq M^\star$, a decision rule can be expressed as "if x contains p: $p \subseteq x$ then predict t".

We order decision rules $(p, t), (\tilde{p}, \tilde{t})$ by the reverse inclusion of their premises:

$$(p, t) < (\tilde{p}, \tilde{t}) \Leftrightarrow p \supset \tilde{p} \tag{2}$$

We cannot apply a single decision rule to any possible description $x \subseteq M^\star$. Therefore, we should use a set of decision rules. A popular means of structuring decision rules in a set is a decision tree DT.

Definition 2. *Decision tree* DT *is an ordered set of decision rules satisfying the following properties: (a) each premise in* DT *is unique, (b)* DT *contains a root decision rule with the empty premise, (c) each non-root decision rule in* DT *has exactly one direct bigger neighbour ("parent"), and one direct smaller neighbour of a parent ("sibling") which differ by one complementary attribute:*

$$a)\ \forall (p, t) \in DT \quad \nexists \tilde{t} \in \mathbb{R}, \tilde{t} \neq t : (p, \tilde{t}) \in DT \tag{3}$$

$$b)\ \exists t \in \mathbb{R} : (\emptyset, t) \in DT \tag{4}$$

$$c)\ \forall (p, t) \in DT, p \neq \emptyset, \quad \exists! (p_{par}, t_{par}), (p_{sib}, t_{sib}) \in DT, m \in p : \tag{5}$$
$$(p_{par}, t_{par}) \succ (p, t), \quad (p_{par}, t_{par}) \succ (p_{sib}, t_{sib}), p_{sib} \neq p$$
$$p_{par} = p \setminus \{m\}, \qquad p_{sib} = p \setminus \{m\} \cup \{\overline{m}\}$$

We propose a more general type of the ordered set of decision rules: a decision semilattice DSL. To define it, we relax the property "c" of a decision tree DT.

Definition 3. *Decision semilattice DSL is an ordered set of decision rules satisfying properties a-b (eq. 3-4) from Definition 2.*

A decision tree DT is a special case of a decision semilattice DSL. Thus, any operation defined for a decision semilattice can also be applied to a decision tree.

We define a "prediction" function $\phi(DSL, x)$ as a function outputting a single target prediction for a description $x \subseteq M^\star$ based on a decision semilattice DSL:

$$\phi(DSL, x) = \frac{1}{|DSL^x_{min}|} \sum_{(p,t) \in DSL^x_{min}} t \tag{6}$$

$$\text{where } DSL^x_{min} = \{(p,t) \in DSL^x \mid \nexists (\tilde{p}, \tilde{t}) \in DSL^x : (\tilde{p}, \tilde{t}) < (p,t)\} \tag{7}$$

$$DSL^x = \{(p,t) \in DSL \mid p \subseteq x\} \tag{8}$$

3.2 Differential decision tree

In this subsection we define a "differential" way for describing a decision rule: (given a prior prediction $\hat{y} \in \mathbb{R}$) "if x contains $p : p \subseteq x$ then add t to the prediction $\hat{y}$".

We define a function $\phi^\Delta(DSL, x)$ which outputs a single target prediction for a description $x \subseteq M^\star$ based on a decision semilattice DSL and differential approach:

$$\phi^\Delta(DSL, x) = \sum_{(p,t) \in DSL^x} t \tag{9}$$

It is unclear how to construct "differential" decision trees and semilattices. We suggest a solution to the former task. To construct a differential decision tree, one can construct a decision tree DT and then "differentiate" it with a function δ:

$$\delta(DT) = \{(p, t - \tilde{t}) \mid (p,t), (\tilde{p}, \tilde{t}) \in DT : (p,t) \prec (\tilde{p}, \tilde{t})\} \cup \{(\emptyset, t) \in DT\} \tag{10}$$

Proposition 1. *For a decision tree DT a prediction $\phi(DT, x)$ matches the prediction $\phi^\Delta(\delta(DT), x)$ for any x.*

Proof. The proof is derived from two facts: (i) a decision tree DT always uses only one decision rule to make a final prediction: $|DT^x_{min}| = 1, \forall x \subseteq M^\star$ (ii) each target of a decision rule in $\delta(DT)$ represents the difference between the target of the corresponding decision rule in DT and the target of its parent.

3.3 Summation of differential decision semilattices

We define an addition operation on decision semilattices in the following way:

$$\begin{aligned} DSL_1 + DSL_2 = &\{(p, t_1 + t_2) \mid \forall (p, t_1) \in DSL_1, t_2 \in \mathbb{R} : (p, t_2) \in DSL_2\} \\ &\cup \{(p, t_1) \in DSL_1 \mid \forall t_2 \in \mathbb{R} : (p, t_2) \notin DSL_2\} \\ &\cup \{(p, t_2) \in DSL_2 \mid \forall t_1 \in \mathbb{R} : (p, t_1) \notin DSL_1\} \end{aligned} \tag{11}$$

The addition operation leads to an important proposition:

Proposition 2. *Given a set of n decision semilattices* $\{DSL_i\}_{i=1}^n$*, the "differential" prediction of the sum of decision semilattices matches the sum of "differential" predictions of the summand decision semilattices :*

$$\phi^{\Delta}(\sum_{i=1}^{n} DSL_i, x) = \sum_{i=1}^{n} \phi^{\Delta}(DSL_i, x), \quad \forall x \subseteq M^{\star} \tag{12}$$

Proof. The proof follows from the definitions of the addition operation (eq. 11) and the function ϕ^{Δ} (eq. 9).

The summation of several identical decision semilattices can be represented as multiplication by a real number:

$$DSL * k = \sum_{i=1}^{k} DSL = \{(p, t * k) \mid (p, t) \in DSL\}, \quad \forall k \in \mathbb{R} \tag{13}$$

3.4 Ensembles of decision trees as decision semilattices

Random forest RF and gradient boosting GB are state-of-the-art ensembles of decision trees. They both operate with a set of decision trees $\{DT_i\}_{i=1}^n$ and, optionally, real-valued hyperparameters. Although the ensembles construct the set of decision trees differently, their prediction functions ϕ^{RF} and ϕ^{GB} are similar as they both sum the predictions of the underlying decision trees:

$$\phi^{RF}(\{DT\}_{i=1}^n, x) = \frac{1}{n} \sum_{i=1}^{n} \phi(DT_i, x) \tag{14}$$

$$\phi^{GB}\left((\{DT\}_{i=1}^n, \alpha, \lambda), x\right) = \alpha + \lambda \sum_{i=1}^{n} \phi(DT_i, x), \quad \alpha, \lambda \in \mathbb{R} \tag{15}$$

Proposition 3. *Given a set of n decision trees* $\{DT_i\}_{i=1}^n$ *and real numbers* $\alpha, \lambda \in \mathbb{R}$*, there is (i) a decision semilattice* DSL_{RF} *such that the prediction* $\phi^{\Delta}(DSL_{RF}, x)$ *matches the prediction* $\phi^{RF}(\{DT_i\}_{i=1}^n, x)$ *for any description* $x \subseteq M^{\star}$*; (ii) a decision semilattice* DSL_{GB} *such that the prediction* $\phi^{\Delta}(DSL_{GB}, x)$ *matches the prediction* $\phi^{GB}(\{DT_i\}_{i=1}^n, x)$ *for any description* $x \subseteq M^{\star}$*:*

$$1)\ \forall x \subseteq M^{\star} \quad \phi^{\Delta}(DSL_{RF}, x) = \phi^{RF}(\{DT_i\}_{i=1}^n, x) \tag{16}$$

$$DSL_{RF} = \frac{1}{n} \sum_{i=1}^{n} \delta(DT_i) \tag{17}$$

$$2)\ \forall x \subseteq M^{\star} \quad \phi^{\Delta}(DSL_{GB}, x) = \phi^{GB}\big((\{DT_i\}_{i=1}^n, \alpha, \lambda), x\big) \tag{18}$$

$$DSL_{GB} = \{(\emptyset, \alpha)\} + \lambda \sum_{i=1}^{n} \delta(DT_i) \tag{19}$$

Proof. (i) By proposition 1, for any decision tree DT_i, there is a differential decision tree $\delta(DT_i) : \phi(DT_i, x) = \phi^{\Delta}(\delta(DT_i), x)$, $\forall x \subseteq M^{\star}$, (ii) By proposition 2, one can sum a set of differential decision trees into a single differential decision semilattice keeping predictions unchanged.

4 Experiments

This section presents an empirical proof that a decision semilattice can produce the same predictions as ensembles of decision trees. The experiments are run via FCApy[1] python package.

The experimental setup is as follows. First, we construct the "base" models: a decision tree, a random forest, a gradient boosting from sci-kit learn package [12], and a gradient boosting from XGBoost package [5]. Then we convert each decision tree of these models into a unified decision tree format used in FCApy. Finally, we aggregate the unified decision trees of ensemble models into a decision semilattice as defined in equations 17, 19.

We use three real-world datasets for regression to compare the models. They are: Boston Housing Data[2]("Bost."), California Housing dataset[3]("Cal."), Diabetes Data[4]("Diab.").

To construct each decision semilattice in less than a minute (on average), we limit each ensemble model by only ten decision trees with a maximum depth of six. The sole decision tree models are limited by a maximal depth of ten.

Table 1 shows the weighted average percentage error (WAPE) of the decision semilattices copying the predictions of the base models on both train and test parts of a dataset. The error does not exceed 1.9%.

The slight difference in the errors comes from the real-valued nature of the datasets. The premises of decision trees built on such data are of the form either "is $m \leq \theta$" or "is $m > \theta$" where m is a real-valued attribute and $\theta \in \mathbb{R}$. These premises are sensitive to the precision of θ. They also use both closed and open intervals, while our FCA-based implementation operates only the former ones. We replace each premise of the form "is $m > \theta$" by the premise "is $m \geq \theta + 10^{-9}$".

Base model	DecisionTree			GradientBoosting			RandomForest			XGBoost		
Dataset	Bost.	Cal.	Diab.	Bost.	Cal.	Diab.	Bost.	Cal.	Diab.	Bost.	Cal.	Diab.
Train error	0.00	0.00	0.00	0.44	0.00	0.35	0.88	1.75	0.10	0.00	0.00	0.00
Test error	0.02	0.01	0.25	0.63	0.00	0.31	0.84	1.88	0.30	0.22	0.03	0.59

Table 1. WAPE (in %) of the decision semilattices copying the predictions of the base models

5 Conclusion

In this paper, we have introduced a method for summing an ensemble of decision trees into a single decision semilattice model with the same predictions. To do so,

[1] https://github.com/EgorDudyrev/FCApy
[2] https://archive.ics.uci.edu/ml/machine-learning-databases/housing
[3] https://scikit-learn.org/stable/datasets/real_world.html#california-housing-dataset
[4] https://www4.stat.ncsu.edu/~boos/var.select/diabetes.html

we have presented a "differential" way to describe decision rules and a function for differentiating a single decision tree.

In the future work, we plan to extend this approach to decision semilattices. We also plan to study the application of decision semilattice to improving interpretability of ensembles of decision trees.

Acknowledgments

The work of Sergei O. Kuznetsov on the paper was carried out at St. Petersburg Department of Steklov Mathematical Institute of Russian Academy of Science and supported by the Russian Science Foundation grant no. 17-11-01276

References

1. Assaghir, Z., Kaytoue, M., Jr., W.M., Villerd, J.: Extracting decision trees from interval pattern concept lattices. In: Napoli, A., Vychodil, V. (eds.) Proc. 8th Int. Conf. Concept Lattices and Their Applications, Nancy, France, October 17-20, 2011. CEUR Workshop Proc., vol. 959, pp. 319–332. CEUR-WS.org (2011)
2. Belohlávek, R., Baets, B.D., Outrata, J., Vychodil, V.: Inducing decision trees via concept lattices. Int. J. of General Systems **38**(4), 455–467 (2009)
3. Breiman, L.: Random forests. Machine Learning **45**(1), 5–32 (10 2001)
4. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. Wadsworth (1984)
5. Chen, T., Guestrin, C.: Xgboost. Proc. 22nd ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (08 2016)
6. Dudyrev, E., Kuznetsov, S.O.: Decision concept lattice vs. decision trees and random forests. In: Braud, A., Buzmakov, A., Hanika, T., Ber, F.L. (eds.) Formal Concept Analysis - 16th Int. Conf., ICFCA 2021, Strasbourg, France, June 29 - July 2, 2021, Proc. LNCS, vol. 12733, pp. 252–260. Springer (2021)
7. Friedman, J.: Greedy function approximation: A gradient boosting machine. Annals of Statistics **29**, 1189–1232 (10 2001)
8. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer Berlin Heidelberg (1999)
9. Krause, T., Lumpe, L., Schmidt, S.E.: A link between pattern structures and random forests. In: Valverde-Albacete, F.J., Trnecka, M. (eds.) Proc. 15th Int. Conf. Concept Lattices and Their Applications, Tallinn, Estonia, June 29-July 1, 2020. CEUR Workshop Proc., vol. 2668, pp. 131–143. CEUR-WS.org (2020)
10. Kuznetsov, S.O.: Machine learning and formal concept analysis. In: Eklund, P.W. (ed.) Concept Lattices, 2nd Int. Conf. on Formal Concept Analysis, ICFCA 2004, Sydney, Australia, February 23-26, 2004, Proc. LNCS, vol. 2961, pp. 287–312. Springer (2004)
11. Kuznetsov, S.O.: Pattern structures for analyzing complex data. In: Sakai, H., Chakraborty, M.K., Hassanien, A.E., Slezak, D., Zhu, W. (eds.) Rough Sets, Fuzzy Sets, Data Mining and Granular Computing, 12th Int. Conf., RSFDGrC 2009, Delhi, India, December 15-18, 2009. Proc. LNCS, vol. 5908, pp. 33–44. Springer (12 2009)
12. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al.: Scikit-learn: Machine learning in python. J. of machine learning research **12**(Oct), 2825–2830 (2011)

Ensemble Techniques for Lazy Classification Based on Pattern Structures

Ilya Semenkov[1] and Sergei O. Kuznetsov[1]

Higher School of Economics – National Research University, Moscow, Myasnitskaya street 20, 101000, Russia

Abstract. This paper presents different versions of classification ensemble methods based on pattern structures. Each of these methods is described and tested on multiple datasets (including datasets with exclusively numerical and exclusively nominal features). As a baseline model Random Forest generation is used. For some classification tasks the classification algorithms based on pattern structures showed better performance than Random Forest. The quality of the algorithms is noticeably dependent on ensemble aggregation function and on boosting weighting scheme.

Keywords: Formal Concept Analysis (FCA) · pattern structures · boosting algorithms · ensemble algorithms

1 Introduction

Pattern structures were introduced in [1] for the analysis of data with complex structure. In [6] [5] a model of lazy (query-based) classification using pattern structures was proposed. The model shows quite good performance, which, however, is lower than that of ensemble classifiers that employ boosting techniques. The main goal of this paper is to study various ensemble approaches based on pattern structures, boosting, and different aggregation functions. The model is known for good interpretability, so we would try to use ensemble techniques to improve its prediction quality.

2 Model description

2.1 Pattern structures

The main idea of the pattern structures is the use of intersection (similarity) operation, with the properties of a lower semilattice, defined on object descriptions to avoid binarization (discretization) prone to creating artifacts [1]. An operation of this kind allows one to define Galois connection and closure operator, which can be used to extend standard FCA-based tools of knowledge discovery to non-binary data without scaling (binarizing) them.

At the first step of the model we transform features in the following way: consider $x \in \mathbb{R}^n$ to be an observation, then we transform it using the following

transformation $T : (x_1, x_2, ..., x_n) \rightarrow ((x_1, x_1), (x_2, x_2), ..., (x_n, x_n))$. Basically, for each feature j, its value in the observation x gets transformed from a number x_i into a 2-dimensional vector (x_i, x_i) with the same value repeated twice. This is used later in the definition of similarity operation.

After the transformation each observation x has 2 values for each feature i, namely $x_{i,1}$ and $x_{i,2}$. In this paper, we define the similarity of two transformed observations x, y in the standard way of interval pattern structures [4] [5] [6]:

$$x \sqcap y = ((min(x_{1,1}, y_{1,1}), max(x_{1,2}, y_{1,2})), ..., (min(x_{n,1}, y_{n,1}), max(x_{n,2}, y_{n,2})))$$

where in $x_{i,j}$ and $y_{i,j}$ i indicates a feature and j indicates one of two values for a feature. In other words for each feature i there were $x_i = (x_{i,1}, x_{i,2}), y_i = (y_{i,1}, y_{i,2})$. After similarity operation $(x \sqcap y)_i = (min(x_{i,1}, y_{i,1}), max(x_{i,2}, y_{i,2}))$. Below, for simplicity this operation will be called intersection.

The subsumption relation is defined as the natural order of the semilattice: $x \sqsubset y \equiv x \sqcap y = x$.

A hypothesis for a description x is a description $x_h \in C_j : \nexists y \in C_i (i \neq j) : (x \sqcap x_h) \sqsubset y$, where C_i stands for a set of elements from class i. So, if a description x_h does not fit any observation from classes C_j $(j \neq i)$, but fits at least 1 observation of the class C_i, then it is considered to be a hypothesis for the class C_i.

The aggregation function is applied either to the whole set of all intersections between a test observation and every element of the training sample, or to the set of hypotheses (extracted from the training set).

Aggregation functions There are many reasonable aggregation functions. In our experiments we have used the following ones:

1. *avglengths*: $C = arg\min_{C_i} \frac{1}{\sum_{k \in A_{C_i}} w_k} \sum_{k \in A_{C_i}} dist(k)$, where $dist(k) = \sum_{j=1}^{n} (k_{j,2} - k_{j,1})$, w_k, the weight of k-th observation in a training set;
2. *k_per_class_closest_avg*: $C = arg\min_{C_i} \frac{1}{\sum_{k \in L_{m,C}} w_k} \sum_{k \in L_{m,C}} dist(k)$, where $dist(k) = \sum_{j=1}^{n} (k_{j,2} - k_{j,1})$, $L_{m,C}$ is the set of m elements from class C which are closest to the prediction observation, m is a hyperparameter;
3. *k_closest*: $C = arg\max_{C_i} \sum_{k \in L_m} w_k \cdot \mathbb{1}(k = C_i)$, where $dist(k) = \sum_{j=1}^{n} (k_{j,2} - k_{j,1})$, L_m, the set of m elements which are closest to the prediction observation regardless of their class, m is a hyperparameter;
4. *count_with_threshold_t*: $C = arg\max_{C_i} \sum_{k \in A_{C_i}} \mathbb{1}\left(\frac{dist(k)}{w_k} < t\right)$, where $dist(k) = \sum_{j=1}^{n} (k_{j,2} - k_{j,1})$, t, a threshold.

Note: in each case A_{C_i} is a set of intersections of the test observation with each observation in a class C_i, $A_{C_i} = \{x_{test} \sqcap x : (x \in X_{train}) \wedge (c(x) = C_i)\}$ if hypotheses are not used and $A_{C_i} = \{x_{test} \sqcap x_h : (x_h \in X_{train}) \wedge (c(x_h) = C_i)\}$ if hypotheses are used (specified in tables with results if they are used), $c(x)$ is a function which returns class of training object.

2.2 SAMME pattern structures

It is not possible to directly use the gradient boosting techniques as there is no loss function which gets directly optimized. Thus, an analog of AdaBoost is used as we know how to implement weighted datasets in pattern structures.

The general scheme called Stagewise Additive Modeling using a Multi-Class Exponential loss function or SAMME is presented in [3]. After adapting it to pattern structures it looks as follows:

Algorithm 1: SAMME for pattern structures analysis, training

Input: (X, y), training dataset, M is the number of models in ensemble ;

initialize $w_i = \frac{1}{n}$, n - number of objects in X;

for $m \in 1, ..., M$ **do**

initialize model $T^{(m)}$;

classify each observation in X using weighted dataset with a new model;

calculate error:

$$err^m = \frac{1}{\sum_{i=1}^{n} w_i} \sum_{i=1}^{n} w_i \mathbb{1}\left(y_i \neq T^{(m)}(x_i)\right)$$

calculate model weight:

$$\alpha^m = \log\left(\frac{1 - err^m}{err^m}\right) + \log(K - 1)$$

where K is the amount of classes;

recalculate object weights:

$$w_i \leftarrow w_i \cdot e^{\,a^m \cdot \mathbb{1}\,(y_i \neq T^{(m)}\,(x_i\,))} \quad , \; i \in \{\, 1, \ldots, n \,\}$$

end

Prediction of ensemble:

$$C(x) = arg\min_k \sum_{m=1}^{M} \alpha^m \cdot \mathbb{1}\left(T^{(m)}(x) = k\right)$$

Methods of weighting models Additionally to the original method of calculation of α, others were used such as:

1. Uniform: $\alpha^m = \frac{1}{M}$
2. Linear: $\alpha^m = \frac{1}{\sum_{i=1}^{n} w_i} \sum_{i=1}^{n} w_i \mathbb{1}\left(y_i = T^{(m)}(x_i)\right)$
3. Exponential: $\alpha^m = \frac{\sum_{i=1}^{n} e^{w_i \mathbb{1}\left(y_i = T^{(m)}(x_i)\right)}}{\sum_{i=1}^{n} e^{w_i}}$
4. Logarithmic: $\alpha^m = \log(1 + (1 - err^m)) = \log(2 - err^m)$
5. Iternum: $\alpha^m = \frac{1}{m}$

3 Datasets description

3.1 Cardiotocography Data Set

The dataset consists of preprocessed 2126 fetal cardiotocograms (CTGs). It contains 23 numerical attributes with no missing values. It could be used for 3 class prediction as well as a 10 class classification (classes are more specific). The dataset is available here [7]. The dataset is unbalanced. Before the classification, data was standartized.

3.2 Male Fertility dataset

The dataset [8] consists of 9 features about patient health, injuries, lifestyle and bad habits. All features are nominal. There are 100 observations in the dataset. The final goal is binary classification: normal semen sample or sample with deviations.

3.3 Divorces

The dataset [9] consists of 54 features which are the answers to questions about relationship experience. All features are nominal. There are 170 observations in the dataset. The final goal is binary classification: got divorced or not.

4 Random Forest

For each dataset Random Forest was chosen as a baseline, since this algorithm is an efficient ensemble of decision trees (which are also good in explanation) [2].

In this algorithm the ensemble of Decision trees is built, where each tree is tuned using random subsample from the training sample (Bagging) and random subsample of features.

Gridsearch with crossvalidation was used to tune it.

5 Results

Accuracy, precision, recall and F1-score were chosen as quality metrics. For datasets with more than 2 target classes, precision, recall and F1-score were calculated for each class separately and averaged afterwards.

Metrics are measured on a randomly chosen test set.

SAMME was also run with the aggregation function k_per_class_closest_avg, since it has shown good performance.

Due to space limitations, we present only results of the most interesting experiments, together with the baseline model Random Forest.

In the table pattern structures are referred to as FCA and SAMME pattern structures is referred to as SAMME FCA.

Table 1. Cardiotocography Data Set (3 classes)

Algorithm	Accuracy	Precision	Recall	F1-score	Ensemble size
SAMME FCA ('k_per_class_closest_avg' original method)	0.934	0.892	0.833	0.858	5
FCA ('k_per_class_closest_avg')	0.934	0.892	0.833	0.858	1
Random Forest	0.925	0.867	0.839	0.852	100
SAMME FCA ('k_per_class_closest_avg' uniform method)	0.925	0.877	0.829	0.848	5
SAMME FCA ('k_per_class_closest_avg' linear method)	0.906	0.838	0.804	0.819	5
SAMME FCA ('k_per_class_closest_avg' exponential method)	0.761	0.582	0.630	0.602	5
SAMME FCA ('k_per_class_closest_avg' logarithmic method)	0.864	0.760	0.735	0.747	5
SAMME FCA ('k_per_class_closest_avg' iternum method)	0.897	0.828	0.784	0.803	5

Table 2. Cardiotocography Data Set (10 classes)

Algorithm	Accuracy	Precision	Recall	F1-score	Ensemble size
SAMME FCA ('k_per_class_closest_avg' original method)	0.784	0.803	0.680	0.712	5
FCA ('k_per_class_closest_avg' original method)	0.784	0.803	0.680	0.712	1
Random Forest	0.793	0.720	0.723	0.704	100
SAMME FCA ('k_per_class_closest_avg' uniform method)	0.765	0.727	0.636	0.66	5
SAMME FCA ('k_per_class_closest_avg' linear method)	0.681	0.609	0.591	0.594	5
SAMME FCA ('k_per_class_closest_avg' exponential method)	0.465	0.423	0.412	0.409	5
SAMME FCA ('k_per_class_closest_avg' logarithmic method)	0.728	0.652	0.623	0.629	5
SAMME FCA ('k_per_class_closest_avg' iternum method)	0.643	0.602	0.575	0.582	5

As it can be seen in Table 1 and Table 2, with original weighting the metrics are identical to the simple FCA one-model. This happens because the first model has a significantly bigger α^1 and dominates others in ensemble. That is why other model weightings are tested.

However, in both cases best SAMME FCA and FCA models have higher average F1-score than the tuned baseline and for the first case they also win in terms of accuracy. Comparing SAMME FCA models it can be seen that the original weighting is better in terms of the presented metrics even though it effectively uses the first model only. In both SAMME and FCA the ensemble size is smaller than in the random forest.

On the divorces dataset (Table 3) due to its size and simplicity Random Forest manages to come out as an absolute winner having 100% in every score. A lot of SAMME FCA and FCA models show the same relatively high scores. SAMME again uses only the first model with original weights. However, a lot of other weighting techniques have similar metric values on this dataset. Because of that even though random forest uses 5 models, FCA can use less models.

On the fertility dataset (Table 4) the tuned Random Forest wins again. Especially the difference is significant in F1-score, precision and recall. The behaviour of the SAMME FCA metrics is similar to the previous dataset: different model weighting methods give similar results. In both SAMME and FCA the ensemble size is smaller than in random forest.

Table 3. Divorce Predictors Data Set

Algorithm	Accuracy	Precision	Recall	F1-score	Ensemble size
FCA "avglengths"	0.953	0.958	0.952	0.953	1
FCA "k_per_class_closest_avg"	0.953	0.958	0.952	0.953	1
FCA "k_closest"	0.953	0.958	0.952	0.953	1
FCA "count_with_treshold_t"	0.674	0.806	0.667	0.629	1
FCA "avglengths" (with hypotheses)	0.953	0.958	0.952	0.953	1
FCA "k_per_class_closest_avg" (with hypotheses)	0.953	0.958	0.952	0.953	1
FCA "k_closest" (with hypotheses)	0.953	0.958	0.952	0.953	1
FCA "count_with_treshold_t" (with hypotheses)	0.512	0.256	0.500	0.338	1
SAMME FCA "avglengths"	0.953	0.958	0.952	0.953	5
SAMME FCA original "k_per_class_closest_avg"	0.953	0.958	0.952	0.953	5
SAMME FCA "k_closest"	0.953	0.958	0.952	0.953	5
SAMME FCA "count_with_treshold_t"	0.674	0.806	0.667	0.629	5
SAMME FCA "avglengths" (with hypotheses)	0.953	0.958	0.952	0.953	5
SAMME FCA "k_per_class_closest_avg" (with hypotheses)	0.953	0.958	0.952	0.953	5
SAMME FCA "k_closest" (with hypotheses)	0.953	0.958	0.952	0.953	5
SAMME FCA "count_with_treshold_t" (with hypotheses)	0.512	0.256	0.500	0.338	5
Random Forest	1.0	1.0	1.0	1.0	5
SAMME FCA uniform "k_per_class_closest_avg"	0.953	0.958	0.952	0.953	5
SAMME FCA linear "k_per_class_closest_avg"	0.953	0.958	0.952	0.953	5
SAMME FCA exponential "k_per_class_closest_avg"	0.953	0.958	0.952	0.953	5
SAMME FCA logarithmic "k_per_class_closest_avg"	0.953	0.958	0.952	0.953	5
SAMME FCA iternum "k_per_class_closest_avg"	0.953	0.958	0.952	0.953	5

6 Conclusion

Even though the model itself seems promising, right now it has multiple issues. First, SAMME boosting technique does not give additional quality. The original SAMME method of calculating α effectively uses only the first classifier, while the other weighting methods do not give stronger metric values than the original method. The problem seems to be in the fact that classifiers with indices > 1 in ensemble are just not good enough. So, there is a reason while original methods stick to the first classifier instead of using all of them. While other weighting might use several parts of the ensemble, it does not improve the metrics, because the classifiers built after the first one have bad metrics most of the time. The potential room for improvement is to make consequent classifiers to produce a better quality results possibly by changing the way dataset gets weighted.

Secondly, we can see that this algorithm performed worse on several datasets. Even though it was better than Random Forest on the complex ones, there is still a room for improvement for simpler ones. This again can be done by improving the quality of the whole ensemble, which Random Forest seems to efficiently perform.

SAMME FCA works slower than Random Forest. However, SAMME FCA has much better explainability: it generates only 5 classifiers (in some cases

Table 4. Fertility Data Set

Algorithm	Accuracy	Precision	Recall	F1-score	Ensemble size
FCA "avglengths"	0.680	0.600	0.826	0.561	1
FCA "k_per_class_closest_avg"	0.920	0.460	0.500	0.479	1
FCA "k_closest"	0.920	0.460	0.500	0.479	1
FCA "count_with_treshold_t"	0.560	0.577	0.761	0.476	1
FCA "avglengths" (with hypotheses)	0.760	0.557	0.641	0.554	1
FCA "k_per_class_closest_avg" (with hypotheses)	0.080	0.272	0.272	0.080	1
FCA "k_closest" (with hypotheses)	0.920	0.460	0.500	0.479	1
FCA "count_with_treshold_t" (with hypotheses)	0.520	0.571	0.739	0.449	1
SAMME FCA "avglengths"	0.920	0.460	0.500	0.479	5
SAMME FCA "k_per_class_closest_avg"	0.920	0.460	0.500	0.479	5
SAMME FCA "k_closest"	0.920	0.460	0.500	0.479	5
SAMME FCA "count_with_treshold_t"	0.920	0.460	0.500	0.479	5
SAMME FCA "avglengths" (with hypotheses)	0.760	0.557	0.641	0.554	5
SAMME FCA "k_per_class_closest_avg" (with hypotheses)	0.800	0.455	0.435	0.444	5
SAMME FCA "k_closest" (with hypotheses)	0.920	0.460	0.500	0.479	5
SAMME FCA "count_with_treshold_t" (with hypotheses)	0.920	0.460	0.500	0.479	5
Random Forest	0.960	0.979	0.750	0.823	100
SAMME FCA uniform "k_per_class_closest_avg"	0.920	0.460	0.500	0.479	5
SAMME FCA linear "k_per_class_closest_avg"	0.920	0.460	0.500	0.479	5
SAMME FCA exponential "k_per_class_closest_avg"	0.920	0.460	0.500	0.479	5
SAMME FCA logarithmic "k_per_class_closest_avg"	0.920	0.460	0.500	0.479	5
SAMME FCA iternum "k_per_class_closest_avg"	0.920	0.460	0.500	0.479	5

effectively uses only one of them), while Random Forest consists of 5 trees only on Divorces dataset and consists of 100 trees in every other case.

References

1. Bernhard Ganter, Sergei O. Kuznetsov. Pattern Structures and Their Projections. In: Delugach H.S., Stumme G. (eds) Conceptual Structures: Broadening the Base. ICCS 2001. Lecture Notes in Computer Science, Volume 2120 (2001). Springer, Berlin, Heidelberg.
2. Leo Breiman. Random Forests. Machine Learning, Volume 45 (2001), 5–32.
3. Ji Zhu, Hui Zou, Saharon Rosset, Trevor Hastie. Multi-class AdaBoost. Statistics and Its Interface, Volume 2 (2009), 349–360.
4. Mehdi Kaytoue, Sergei O. Kuznetsov, Amedeo Napoli, Sébastien Duplessis. Mining gene expression data with pattern structures in formal concept analysis. Information Sciences, Volume 181 (2011), Issue 10, 1989-2001.
5. Sergei O. Kuznetsov. Fitting Pattern Structures to Knowledge Discovery in Big Data. In: Cellier P., Distel F., Ganter B. (eds) Formal Concept Analysis. ICFCA 2013. Lecture Notes in Computer Science, Volume 7880 (2013). Springer, Berlin, Heidelberg.
6. Sergei O. Kuznetsov. Scalable Knowledge Discovery in Complex Data with Pattern Structures. In: Maji P., Ghosh A., Murty M.N., Ghosh K., Pal S.K. (eds) Pattern Recognition and Machine Intelligence. PReMI 2013. Lecture Notes in Computer Science, Volume 8251 (2013). Springer, Berlin, Heidelberg.
7. UCI Machine Learning Repository. URL: https://archive.ics.uci.edu/ml/datasets/Cardiotocography [date of access: 27.08.2020].
8. UCI Machine Learning Repository. URL: https://archive.ics.uci.edu/ml/datasets/Divorce+Predictors+data+set [date of access: 21.03.2021].
9. UCI Machine Learning Repository. URL: https://archive.ics.uci.edu/ml/datasets/Fertility [date of access: 21.03.2021].

A concept of self-supervised logical rule inference in symbolic classifications

Xenia Naidenova[1][0000-0003-2377-7093] and Vladimir Parkhomenko[2][0000-0001-7757-377X]

[1] Military Medical Academy, Saint Petersburg, Russian Federation
E-mail: ksennaidd@gmail.com
[2] Peter the Great St. Petersburg Polytechnic University, Saint Petersburg, Russian Federation
E-mail: parhomenko.v@gmail.com

Abstract. An approach to modelling self-supervised learning for automated inferring good classification tests is proposed. The concepts of internal and external learning contexts are formulated. A model of intelligent agent, capable of improving own learning process of inferring good classification tests in the external context is advanced. Internal evaluation is used in an internal process of learning with the aim of tuning the external learning process. The same learning algorithm is used for supervised learning both in the external context and in the internal context. The structure of good test inferring is described and a procedure to recognize the end of inferring process is proposed.

Keywords: Self-supervised learning, Good classification tests, Internal context, External context, Intelligent agent, Deep learning.

1 Introduction

Self-learning embodies one of the essential properties of human intelligence related to an internal evaluation of the mental process quality. A deeper level of learning – self-learning – allows to manage the learning process in an external context in terms of its effectiveness through the internal evaluation and developing rules to select the best learning strategies and parameters without a teacher.

We shall understand self-learning as a process of improvement of an agent's (or system's) actions on the basis of self-evaluation of his (its) actions in a variable context. When the agent selects sub-contexts and some actions in learning process, he (it) uses some criteria. The self-learning is related to the ability to change these criteria, to form new criteria, which is essentially to improve the learning algorithms, making them more consistent with the external context and more effective.

The purpose of this paper is to model a self-learning process in the logical or symbolic supervised algorithms of machine learning. This mode of learning covers mining logical rules and dependencies from data: "if-then" rules, decision trees, functional, implicative and associative dependencies. We shall consider a special kind of symbolic machine learning, namely, inferring good tests from data [1] in multi-valued dynamic contexts (external contexts) for recognizing classes of objects represented by

their symbolic descriptions. The self-learning at the internal (deep) level implements the analysis and internal evaluation of classification rule inferring in the external context and allows one to reveal the relationships between the external contexts (subcontexts) and the parameters of learning. The implementation of self-learning in the internal context can be based on the same algorithm of symbolic machine learning that works in the external context.

The paper is organized as follows. The related works are discussed in Section 2. Sections 3 and 4 deal with defining a software agent capable of self-learning and the structure of the internal context. Sections 5, 6, and 7 cover the description of self-learning in inferring good maximally redundant classification tests from data. To complete the paper, we give a short conclusion.

2 Related works

The analysis of modern researches has been implemented in the following directions: modeling of self-learning (self-supervised learning), deep learning [2] and models of learning in robots and robotic systems.

In the first direction, it is particularly interesting [3] the principles and technologies of creating a robot that can move in the environment, manipulate objects and avoid obstacles. The robot is designed as an autonomous system. It requires from the robot a good spatial and semantic understanding of the environment. The self-learning robot should be aware of its own localization and realize an internal reflection of spatial situation taking into account different scenes (semantic understanding) in order to recognize new objects. It is declared by the author that the robot should be self-esteemed and self-managed on the basis of previous experience. It must constantly adapt its spatial and semantic models in order to improve the performance of its tasks. Some concepts and algorithms are proposed to evaluate the robot's own movement (Self-Supervised Visual Ego Motion Learning) [4]. Note that the concept of self-learning proposed in [3] coincides with the concept of self-learning offered by us.

In [5], the role of curiosity in self-learning is analyzed and the concepts of self-learning with the phenomenon of curiosity are developed.

It is an ordinary practice to associate self-learning with deep learning. Impressive successes in deep learning achieved in simulation games [6] and image analysis [7-16]. However, deep learning does not mean self-learning. Using neural networks for segmenting images traditionally requires a large quantity of training data marked manually. In [14] an algorithm is proposed on the basis of which 130000 images were generated with automatic marking for 39 objects. In [15], a robot’s internal evaluation of its future path cost is based on the probabilistic Bayesian method.

Neural networks recognize classes of objects and form a feature hierarchy of classes, but do not form their symbolic logical descriptions or rules to recognize them. There are a number of works in which attempts are made to find the interconnection between artificial neural networks and symbolic machine learning within the framework of the analysis of formal concepts (FCA) [17-20]. The main purpose of these works is to use the algorithms of constructing the concept lattice to configure the

artificial neural networks in order to make it interpretable in terms of concepts. However, an improvement of the artificial neural network learnability has not yet obtained.

In some works, the authors propose the use of robot's manipulation reflection in learning algorithms for improving and accelerating robot's training. For example, industrial Robot of Japanese Company Fanuc uses a method known as "training with reinforcement" to grab objects by a manipulator. In this process, the robot fixes its work on video and uses this video for correcting own activity. Domestic development of robots is also based on the use of artificial neural networks [21-24].

3 Software agent capable of self-learning

Intelligence acts always in a changing context. Several examples of changing contexts can be: the descriptions of patient's conditions supplemented by doctor's decisions and patient's responses, images of the Earth's surface, and student personal characteristics. The task of self-learnable individual or an automatic device in a such changing context is to support any purposeful action or function (search of food, search for exit from a labyrinth, etc.). Intelligence must have some abilities to act in the context by choosing sub-contexts and/or actions in them, as well as by assessing the extent to which its actions bring it closer to the goal. We shall refer to the context in which an intellectual being or device is acting as the external context.

The objects in the external context (training samples) are described in terms of their properties (features, attributes) and they are specified by splitting into classes. The task of learning is to find rules in a given space of object descriptions in order to repeat the classification of objects represented by splitting objects into disjoint classes. Good tests approximate the specified object classification in the best way and give the minimum sets of attributes (values) that carry out the greatest possible generalization within object classes and distinguish in pairs all objects from different classes [1]. As a task in the external context, we have chosen the task of constructing good maximally redundant classification (diagnostic) texts, because the algorithms developed for this task have a number of convenient properties for self-monitoring the process of inferring tests [25]:

- external context is partitioned into sub-contexts in which good tests are inferred independently;
- sub-contexts are chosen and formed by the logical rules based on analyzing sub-contexts' characteristics; the choice of sub-context determines the speed and efficiency of classification task.

The strategies for selecting sub-contexts of the external context and the algorithms to find good tests in them are easy to describe (to represent) with the use of special multi-valued attributes. In what follows, we shall call the intellectual being an agent, although it does not mean that we identify it with the agent in multiagent systems. Summing up the foregoing, we conclude that for self-learning the agent should have:

1. A display of the external context in terms of the internal context);
2. A set of rules (possible actions) for selecting context (sub-context);
3. A display of the desired target (state);

4. An operation (a function) for comparing the desired target with the achieved result.

During the training process, the agent must develop a sequence of actions that will lead to the goal. We shall consider the permanent external context and its changes only in connection with the activity of the agent, for example, a sub-context can be deleted when the agent has completely solved the problem for this sub-context. Decomposition of contexts into sub-contexts in the tasks of inferring good classification tests have been considered in [25-26].

When the agent selects sub-contexts and its (his) actions in learning process, it (he) uses some criteria. These criteria can be: the number of sub-contexts to be considered, the number of tests already extracted in sub-context, the number of objects and values of attributes in sub-context, the number of essential objects and values of attributes (attributes) in sub-context [1], temporal characteristics and some others. The agent needs to memorize the situations of learning and the activity associated with them.

Let us assume that the internal context necessarily contains:

1. Description of selected sub-context in terms of its properties;
2. Description of selected action and the rule for its selection;
3. Internal estimation of learning process with the use of some criteria of its efficiency.

4 The structure of the internal context and realizing self-learning

Let K be the descriptions of external sub-context via its properties, $A = \{A_1, A_2, \ldots. A_n\}$ be the descriptions of algorithms of good tests inferring via their properties in this sub-context, $R = \{R_1, R_2, \ldots,.R_m\}$ be the set of rules for selecting sub-contexts, and $V = \{V_1, V_2, \ldots, V_q\}$ be the set of rules for evaluating the process of good test inferring. Then the internal context is described by the direct product of sets K, A, R and its mapping on V: $K \times A \times R \rightarrow V$. There are more simple variants of the internal context: $K \times A \rightarrow V$ and $K \times R \rightarrow V$.

The same algorithm can be used in both the external and the internal context in order to infer the logical rules for distinguishing the variants of learning in the external context evaluated as good ones from the variants evaluated as not good ones. A few algorithms for good test inferring have been elaborated: ASTRA [27], DIAGARA, NIAGARA, and, INGOMAR [28].

We come to the realization of deep learning for the symbolic machine learning tasks. The internal context is a memory of the agent, the rules extracted from the internal context represent the agent's knowledge about the effectiveness of its actions in the external context. Actions in the internal and external contexts can be represented as actions of two agents functioning in parallel and exchange data (Fig. 1).

Agent A1 transmits the data (the descriptions of contexts, algorithms, rules for selecting sub-contexts) to Agent A2. Agent A2 acts in the internal context (obtained from agent A1) and passes to agent A1 the rules, which the latter applies to select the best variant of learning with each new external sub-context.

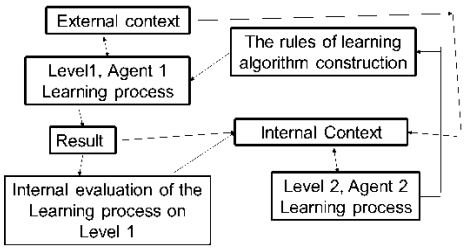


Figure 1. Scheme of self-learning with the interaction of two agents

For Agent A2, the internal context (memory) should not be empty, but this agent (as well as Agent A1) can use an incremental mode of learning [28]. A few incremental algorithms for good test inferring in symbolic contexts are described in [28].

5 The structure of good maximally redundant test inferring

Good test analysis (GTA) deals with the formation of best descriptions of a given object class (class of positive objects) against the objects do not belonging to this class (class of negative objects) on the basis of lattice theory. We assume that objects (or patterns) are described in terms of values of a given set U of attributes. The key notion of GTA is the notion of classification. To give a target classification of objects, we use an additional attribute $k \notin U$. This attribute partitions a given set of objects into disjoint classes the number of which is equal to the number of values of this attribute. We need in the following series of definitions.

Denote by M the set of attribute values such that $M = \cup a \in U\ rng(a)$, where rng(a) is the set of all values of a. Let $G = G+ \cup G-$ be the set of objects, where G+ and G− are the sets of positive and negative objects, respectively.

Let T be a table with many-valued data, where lines correspond to objects and columns correspond to attributes. For representing data, we do not use any scaling.

Denote a description of $g \in G$ by $\delta(g)$, and descriptions of positive and negative objects by $D+ = \{\delta(g) | g \in G+\}$ and $D- = \{\delta(g) | g \in G-\}$, respectively. The Galois connections [29] between the ordered sets $(2^G, \subseteq)$ and $(2^M, \subseteq)$, i.e. $2^G \rightarrow 2^M$ and $2^M \rightarrow 2^G$, are defined by the following mappings called derivation operators [30]:

for $A \subseteq G$ and $B \subseteq M$, $val(A) = \cap g \in A\ \delta(g)$ and

$obj(B) = \{g | B \subseteq \delta(g), g \in G\}$.

There are two closure operators [30, 31]: generalization_of(B) = val(obj(B)) and generalization_of(A) = obj(val(A)). A is closed if A = obj(val(A)) and B is closed if B = val(obj(B)). If (val(A) = B) & (obj(B) = A), then a pair (A,B) is called a formal concept [30, 32], subsets A and B of which are called concept extent and intent, respectively. A triplet (G,M,I), where I is a binary relation between G and M, is a formal context K. According to the values of a goal attribute, we get some possible

forms of the formal contexts: $K\varepsilon := (G\varepsilon, M, I\varepsilon)$ and $I\varepsilon := I \cap (G\varepsilon \times M)$, where $\varepsilon \in rng(k)$, $rng(k) = \{+,-\}$ (if necessary the value τ can be added to provide undefined objects) [32]. A classification context $K\pm$ is formed by the sub-position of contexts $K+$ and $K-$, and the apposition of the resulted context with $(G\pm, k, G\pm \times k)$, i.e. after adding the classification attribute k. Let us rewrite the definitions of tests by using notation of classification contexts and semi-concepts [33]: pairs like $(obj(B),B)$, $B \subseteq M$, the left side of which is called an extent, and pairs like $(A,val(A))$, $A \subseteq G$, the right side of which is called an intent. Here and later words "diagnostic test" (and GMRT) will be used for semi-concepts (or concepts), the right part of which is a test.

Definition 1. A diagnostic test (DT) for $K+$ is a pair (A,B) such that $B \subseteq M$, $A = obj(B) \neq \emptyset$, $A \subseteq G+$, and $obj(B) \cap G \neq \emptyset$.

Definition 2. A diagnostic test (A,B) for $K+$ is to be said maximally redundant if $obj(B \cup m) \subset A$ for all $m \in M \setminus B$.

Definition 3. A diagnostic test (A,B) for $K+$ is to be said good iff any extension $A1 = A \cup i$, $i \in G+ \setminus A$, implies that $(A1,val(A1))$ is not a DT for $K+$.

A maximally redundant test which is simultaneously good is called a good maximally redundant test (GMRT).

Definitions of tests (as well as other definitions), associated with $K+$, are applicable to $K-$.

If a good DT (A,B) for $K+$ is maximally redundant, then any extension $B1 = B \cup m$, $m \notin B$, $m \in M$ implies that $(obj(B1),B1)$ is not a good DT for $K+$.

In the general case a set B is not closed for DT (A,B), consequently, DT is not obligatorily a formal concept. A GMRT can be regarded as a special type of formal concept [1]. Note that the definition of GMRTs is equivalent to the definition of inclusion-minimal concept-based hypothesis in the FCA [30].

To transform inferring GMRTs into an incremental process, we introduce two kinds of subtasks for $K+$ ($K-$), called subtasks of the first and second kind, respectively [34]:

1. Given a positive object g, find all GMRTs $(obj(B),B)$ for $K+$ such that B is contained in $\delta(g)$. In the general case, instead of $\delta(g)$ we can consider any subset of values B1, such that $B1 \subseteq M$, $obj(B1) \neq \emptyset$, $B1 \not\subseteq \delta(g)$, $\forall g \in G-$.

2. Given a non-empty set of values $B \subseteq M$ such that $(obj(B),B)$ is not a DT for positive objects, find all GMRTs $(obj(B1),B1)$ such that $B \subset B1$.

Accordingly, we define two kinds of sub-contexts of a given classification context called object and attribute value projections, respectively. If (G,M,I) is a context and if $H \subseteq G$, and $N \subseteq M$, then $(H,N,I \cap H \times N)$ is called a sub-context of (G,M,I) [35].

Definition 4. The object projection $\psi(K+,g)$ returns sub-context $(N,\delta(g),J)$, where $N = \{n \in G+ \mid n \text{ satisfies } (\delta(n) \cap \delta(g) \text{ is a test for } K+)\}$, $J = I+ \cap (N \times \delta(g))$.

Definition 5. The attribute value projection $\psi(K+,B)$ returns sub-context (N,B,J), where $N = \{n \in G+ \mid n \text{ satisfies } (B \subseteq \delta(n))\}$, $J = I+ \cap (N \times B)$. In the case of negative objects, symbol + is replaced by symbol − and vice versa.

The decomposition of inferring GMRTs into the subtasks requires the following actions:

1. Select an object or value to form a subtask.
2. Form the subtask.

3. Reduce the subtask.

4. Delete the object or value when the subtask is over.

The following theorem gives the foundation for reducing sub-contexts formed by object and attribute value projections [27, 28].

Theorem 1. Let B ⊆ M, (obj(B),B) be a maximally redundant DT for positive objects and obj(m) ⊆ obj(B), m ∈ M. Then m cannot belong to any GMRT for positive objects different from (obj(B),B).

6 A procedure for mining the all GMRTs in the projections of both kinds

Let Sgood+ (Sgood−) be the partially ordered set of obj+(m), m ∈ M satisfying the condition that (obj+(m), val(obj+(m))) is a current good DT for K+ (K−). The basic recursive procedure (BRP) for K+ is defined in Fig. 2, where

• the first step of recursion is omitted for simplicity;

• the output Sgood+ is implicitly given via a globally defined set, which is modified during the procedure; algorithm formSgood is given in Fig. 3;

• variable ψtype has two possible values: object or attribute value projection;

• algorithm choiceOfprojection returns ψtype, and X, which can be either g or B w.r.t. value of ψtype;

• algorithm formSubcontext implements a definition of object or attribute value projection and returns new subcontext K∗; conditions for the end of recursion are described in steps 7, 25;

• after the end of the current recursion iteration the control goes to the previous recursion iteration from steps 13, 31;

• checking whether (obj+(m),val(obj+(m))) is a DT for K+ is performed as follows: val(obj+(m)) is a test for K+ iff obj(val(obj+(m))) = obj+ (m).

Procedure BRP

Input: K+,K−,Sgood+

Output: Sgood+

1. f := 0;
2. forall m ∈ M do
3. if val(obj+(m)) is a test for K+ then
4. formSgood(obj+(m),Sgood+);
5. M := M \m, f := 1;
6. end
7. if |M|≤ 1 then
8. return;
9. if f = 0 then
10. ψtype, X choiceOfprojection (K+,K−);
11. K∗ + formSubcontext(ψtype,X,K+);
12. BRP (K∗ +,K−,Sgood+);
13. if ψtype = object projection then

14. G+ := G+ \X; 15. else
16. M := M \X;
17. else
18. f := 0;
19. end
20. forall g ∈ G+ do
21. if val(g) is not a test for K+ then
22. G+ := G+ \g;
23. f := 1;
24. end
25. if |G+|≤ 1 then
26. return;
27. if f = 0 then
28. ψtype, X choiceOfprojection (K+,K−);
29. K∗ + formSubcontext(ψtype,X,K+);
30. BRP (K∗ +,K−,Sgood+);
31. if ψtype = object projection then
32. G+ := G+ \X;
33. else
34. M := M \X;
35. else
36. go to 1;
37. end

Figure 2. Pseudo code of basic recursive procedure

7 Forming SGOOD as the main problem of good test inferring

Essentially, the process of forming Sgood is an incremental procedure of finding all maximal elements of a partially ordered (by inclusion relation) set. It is based on topological sorting of partially ordered sets. Thus, when the algorithm is over, Sgood contains the extents of all the GMRTs for K+ (for K−) and only them. The operation of inserting an element A∗ into Sgood (in algorithm formSgood) under lexicographical ordering of these sets is reduced to lexicographically sorting a sequence of k-element collections of integers.

A sequence of n-collections whose components are represented by integers from 1 to |M|, is sorted in time of O(|M| + L), where L is the sum of lengths of all the collections of this sequence [36]. Consequently, if Lgood is the sum of lengths of all the collections A of Sgood, then the time complexity of inserting an element A∗ into Sgood is of order O(|M| + Lgood). The set Tgood of all the GMRTs is obtained as follows: Tgood = {t|t = (A,val(A)), A ∈ Sgood}.

Algorithm formSgood
Input: A∗ ⊆ G+,Sgood+
Output: Sgood+

1. forall A ∈ Sgood do
2. if A ⊂ A∗ then
3. Sgood+ := Sgood+ \A;
4. else
5. if A∗ ⊆ A then
6. return;
7. end
8. Sgood+ := Sgood+ ∪A∗;
9. return;

Figure 3. Pseudo code of algorithm formSgood

8 Some problem to be solved

In self-learning, it is very important determining the nearness of the current result to the goal of learning process. The goal in mining GMRTs is to find the all GMRTs for a given external context. Generally, a situation can be when there exist sub-contexts of the external context to be solved, but the saturation of S_{GOOD} is already achieved (i. e., all GMRTs are obtained). A procedure of determining the saturation of S_{GOOD} can be based on the properties of the set of all GMRTs of a formal context to be the Sperner System [37].

It is important to formulate some unsolved and nontrivial problems related to the decomposition considered in this paper. These problems are:

- How to recognize a situation that current formal context contains only the GMRTs already obtained?
- How to evaluate the number of recurrences necessary to resolve a subtask in inferring GMRTs? (in case we use a recursive algorithm like DIAGARA)?
- How to evaluate the perspective of a selected sub-context with respect to finding any new GMRT?

These problems are interconnected and the subject of our further research. The effectiveness of the decomposition depends on the properties of the initial classification context (initial data). Now we can propose some characteristics of data (contexts and sub-contexts) useful for choosing a projection:

- The number of objects;
- The number of attribute values;
- The number of the GMRTs already obtained and covered by this projection.

Some unsolved problems cited above are difficult for analytical solution. It is possible that realizing the proposed approach to self-improving learning algorithms permits one to investigate these problems and enables us to overcame the above difficulties.

One of the advantages of our approach is related to the possibilities to reduce the process of choosing sub-contexts and to obtain the best variant of learning to the plausible deductive reasoning, one of the models of which is described in [28]. Modeling of on-line human reasoning is a key problem in creating intelligent computer systems.

However, any attention is hardly paid to this topic in computer science. Knowledge engineering has arisen from a paradigm in which knowledge is considered as something to be separated from its bearer and to function autonomously with a problem-solving application. This paradigm ignores the very essential feature of intelligence, namely, its continuous cognitive activity. Knowledge is corrected constantly. This means that the mechanism of using knowledge cannot be separated from the mechanism of discovering knowledge. The future realization of our approach to self-improving good test inferring will support using logical rules extracted from the internal context for deductive process of choosing variants of learning.

9 Conclusions

The concept of self-learning in the processes of inferring good classification tests is proposed in the paper. The inferring of good classification tests is a task of symbolic machine learning, for which the questions of self-learning has been not considered earlier. The results of this article are the following:

A model of self-learning was proposed allowing to manage the process of inferring good tests in terms of its effectiveness through an internal evaluation of the learning process and the development of rules for choosing the best strategies, algorithms, and learning characteristics.

The concepts of internal and external learning contexts were formulated.

The structure of the internal context was proposed.

A model of intelligent agent, capable of improving own learning process of inferring good classification tests in the external context was advanced;

It was shown that the same learning algorithm can be used for supervised learning both in the external context and in the internal context. The proposed approach is a model of deep learning implemented by inferring logical rules from examples.

Acknowledgments. The research is partially supported by RFR grant № 18-07-00098A.

References

1. Naidenova, X.: Good diagnostic tests as formal concepts. In: F. Domenach, D.I. Ignatov, J. Poelmans (eds.) ICFCA-2012, LNCS vol. 7278, Springer, pp. 211-226 (2012).
2. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. The MIT Press (2016).
3. Pillai, S.: Towards richer and self-supervised perception in robots. PhD Thesis Proposal (2017) http://people.csail.mit.edu/spillai/research/ and http://people.csail.mit.edu/spillai/data/papers/2017-phdthesis-proposal-nocover.pdf
4. Pillai, S., Leonard, J.: Towards Visual Ego-motion Learning in Robots. Submitted to IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS) (2017). http://people.csail.mit.edu/spillai/learning-egomotion/learning-egomotion.pdf
5. Pathak, D., Agraval, P., Efros, A., Darrell, T.: Curiosity-driven exploration by self-supervised prediction. In: Proc. of the 34th Int. Conf. on Machine Learning, JMR: W&CP, pp. 12 (2017) https://pathak22.github.io/noreward-rl/

6. Silver, D., Huang, A., Maddison, Ch. J. et al.: Mastering the game of Go with deep neural networks and tree search. Nature **529**, 484-489 (2016). doi: 1038/nature 16961
7. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv: 1409.1556v6 [cs.CV] (2015).
8. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. Computer Vision and Pattern Recognition, arXiv:1512.03385v1[cs.CV] (2015).
9. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: F. Pereira, C.J.C. Burges et al. (eds.) Advances in neural information processing systems, pp. 1097-1105 (2012).
10. Potapov, A., Batishcheva, V., Pan Shu: Improving the quality of recognition in deep learning networks using the method of simulation of annealing. Scientific and Technical Bulletin of Information Technologies, Mechanics, and Optics, 17(4), (2017). (in Russian)
11. Hossain, D., Capi, G. Jinday, M.: Evolution of Deep Belief Neural Network Parameters for Robot Object Recognition and Grasping. Procedia Computer Science 105, 153-157 (2017).
12. Schmidt, T., Newcombe, R., Fox, D.: Self-supervised Visual Descriptor Learning for Dense Correspondence. IEEE Robotics and Automation Letters, 2(2), 420-427 (2016). doi: 10. 1109/LRA.2016.2634089
13. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. arXiv:1408.5093 [cs.CV] (2014).
14. Zeng, A., Yu, Kuan-Ting, Song, S., Suo, D., Walker, Ed., Rodrigues, A., Xiao, J.: Multi-View Self -Supervised Learning for 6D Pose Estimation in the Amazon Picking Challenge. In: Proceedings of IEEE International conference on Robotics and Automation (ICRA), pp. 1986-1993 (2017). doi: 10.1109/ICRA.2017.7989165
15. Sofman, B., Line, E. et al.: Improving Robot Navigation Through Self-Supervised Online Learning. Journal of Field Robotics, 23(11-12), 1059-1075 (2016). doi: 10.1002/rob.20169
16. Long, J., Shelhamer, E., Darrell, T.: Fully Convolutional Networks for Semantic Segmentation. arXiv:1605.06211v1 [cs.CV] (2016).
17. Endres, D., Foldiak, P.: Interpreting the neural code with formal concept analysis. In: Koller, D., Schuurmans, D., Bengio, Y., Bottou, L. (eds.) Advances in Neural Information Processing Systems, vol. 21, pp. 425–432. MIT Press, Cambridge (2008).
18. Kuznetsov, S.O., Makhazhanov, N., Ushakov, M.: On Neural Network Architecture Based on Concept Lattices, LNAI, vol. 10352, pp. 653–663 (2017). Available https//doi.org/10.1007/978-3-319-60438-1_64
19. Rudolph, S.: Using FCA for Encoding Closure Operators into Neural Networks. In: Priss, U., Polovina, S., Hill, R. (eds.) ICCS 2007, LNAI, vol. 4604, pp. 321–332. Springer, Berlin- Heidelberg (2007).
20. Tsopzé, N., Nguifo, E.M., Tindo, G.: CLANN: Concept lattice-based artificial neural network for supervised classification. In: Proceedings of the Fifth International Conference on Concept Lattices and Their Applications, vol. 331, pp. 153–164 (2007).
21. Pavlovsky, V. Savitsky, A: A Quadrocopter neural network control algorithm on typical trajectories. Nonlinear World, 13(6), 47-54 (2016). (in Russian)
22. Aliseychik, A., Orlov, I., Pavlovsky, V., Smolin, V., Podoprosvetov, A., Shishova, M.: Pneumatic manipulation with neural network control. LNCS, vol. 9719, pp. 292-301 (2016).
23. Savitsky, A. V., Pavlovsky, V. E.: Model of quadrotor and algorithm of vehicle control based on neural network. Keldysh Institute preprint 077 (2017). http://library.keldysh.ru/preprint.asp?id=2017-77 (in Russian)

24. Pavlovsky, V.E., Pavlovsky V.V.: Technologies SLAM for moving robots: State and Prospects. Mechatronics, Automation, Management, 17(6), 384-394 (2016). (in Russian)
25. Naidenova, X., Parkhomenko, V., Shvetsov, K.: Context-Dependent Incremental Learning Good Maximally Redundant Tests. SAI Intelligent Systems Conference 2015, pp. 1-6. London, UK: IEEE (2015). DOI:10.11.1109/IntelliSys.2015.7361258 https://www.researchgate.net/publication/292608731_Context-Dependent_Incremental_Learning_of_Good_Maximally_Redundant_Tests
26. Naidenova, X., Parkhomenko, V.: Context-Dependent Classification Reasoning Based on Good Diagnostic Tests. Proc. of FCA&A' 2015 (co-located with ICFCA'2015), J. Baixeries, Ch. Sacarea, and M. Ojeda-Aciego (eds), pp. 65-80. University de Malaga (2015). ISSN-84-606-7410-8. http://ceur-ws.org/Vol-1434/proceedings-fcaa.pdf
27. Naidenova, X., Plaksin, M. Shagalov, V.: Inductive inferring all good classification tests. Proceedings of International Conference "Knowledge-Dialog-Solution", vol. 1, pp.79-84. Jalta, Ukraine (1995).
28. Naidenova, X.: An incremental learning algorithm for inferring logical rules from examples in the framework of the common reasoning process. In: Triantaphyllou, E., & Felici, G. (eds.), Data mining and knowledge discovery approaches based on rule induction techniques, pp. 89–146. New York, NY: Springer. (2006).
29. Ore, O.: Galois connections. Trans. Amer. Math. Soc 55 (1944) 494–513.
30. Ganter, G., Kuznetsov, S. O.: Pattern Structures and Their Projections. In: H. S. Delugach, G. Stumme (eds.), Conceptual Structures: Broadening the Base, Proceedings of the 9th International Conference on Conceptual Structures, 129–142 (2001).
31. Naidenova, X.: The Data-Knowledge Transformation. In: V. Solovyev (ed.), Text Processing and Cognitive Technologies, vol. 3, Pushchino, 130–151 (1999).
32. Ganter, B., Kuznetsov, S. O.: Formalizing Hypotheses with Concepts. In: Conceptual Structures: Logical, Linguistic, and Computational Issues, Proceedings of the 8th International Conference on Conceptual Structures, 342–356 (2000).
33. Luksch, P., Wille, R.: A Mathematical Model for Conceptual Knowledge Systems. In: H.-H. Bock, P. Ihm (eds.), Proceedings of the 14th Annual Conference of the Gesellschaft fur Klassifikation (GfKl 1990), 156–162 (1991).
34. Naidenova, X., Parkhomenko, V.: Attributive and Object Sub-contexts in Inferring Good Maximally Redundant Tests. In: K. Bertet, S. Rudolph (eds.), Proceedings of the Eleventh International Conference on Concept Lattices and their Applications, Košice, Slovakia, October 7-10, 2014., vol. 1252 of CEUR Workshop Proceedings, 181–193 (2014).
35. Ganter, B., Wille, R.: Formal concept analysis: mathematical foundations. Springer, Berlin (1999).
36. Aho, A. V., Hopcroft, J.E., Ullman, J.D.: The design and analysis of computer algorithms. Addison-Wesley (1975).
37. Sperner, E.: Ein Satz über Untermenge einer Endlichen Menge. Math. Z. 27, 544-548 (1928).

Non-Redundant Link Keys in RDF Data: Preliminary Steps

Nacira Abbas[1], Alexandre Bazin[1], Jérôme David[2], and Amedeo Napoli[1]

[1] Université de Lorraine, CNRS, Inria, Loria, F-54000 Nancy, France
Nacira.Abbas@inria.fr, Alexandre.Bazin@loria.fr, Amedeo.Napoli@loria.fr
[2] Université Grenoble Alpes, Inria, CNRS, Grenoble INP, LIG, F-38000 Grenoble, France Jerome.David@inria.fr

Abstract. A link key between two RDF datasets D_1 and D_2 is a set of pairs of properties allowing to identify pairs of individuals, say x_1 in D_1 and x_2 in D_2, which can be materialized as a x_1 `owl:sameAs` x_2 identity link. There exist several ways to mine such link keys but no one takes into account the fact that `owl:sameAs` is an equivalence relation, which leads to the discovery of non-redundant link keys. Accordingly, in this paper, we present the link key discovery based on Pattern Structures (PS). PS output a pattern concept lattice where every concept has an extent representing a set of pairs of individuals and an intent representing the related link key candidate. Then, we discuss the equivalence relation induced by a link key and we introduce the notion of non-redundant link key candidate.

Keywords: Linked Data · RDF · Link Key · Formal Concept Analysis · Pattern Structures.

1 Introduction

In this paper, we are interested in data interlinking which goal is to discover identity links across two RDF datasets over the web of data [5,8]. The same real world entity can be represented in two RDF datasets by different subjects in RDF triples `(subject,property,value)` (instead of "object" usually used in RDF data we will use "value"). It is important to be able to detect such identities, for example using rules expressing sufficient conditions for two subjects to be identical. A link key takes the form of two sets of pairs of properties associated with a pair of classes. The pairs of properties express sufficient conditions for two subjects, from the associated pair of classes, to be the same. An example of a link key is $(\{(\texttt{designation}, \texttt{title})\}, \{(\texttt{designation}, \texttt{title}), (\texttt{creator}, \texttt{author})\}, (\texttt{Book}, \texttt{Novel}))$ which states that whenever an instance a of the class `Book` has the same (non empty) values for the property `designation` as an instance b of the class `Novel` for the property `title` (universal quantification), and that a and b share at least one value for the properties `creator` and `author` (existential quantification), then a and b denote the same entity, i.e., an `owl:sameAs` relation can be established between a and b.

A link key can be understood as a "closed set" in the sense that it is maximal w.r.t. the set of pairs of individuals to which it applies. This was firstly discussed in [2] and then extended in [3]. Hence the question of relying on Formal Concept Analysis (FCA [7]) to discover link keys is straightforward as FCA is based on a closure operator. Then, given two RDF datasets, FCA is applied in [3] to a binary table where rows correspond to pairs of individuals and columns to pairs of properties. The intent of a concept is a link key candidate which should be validated thanks to suitable quality measures. The extent of the concept is the set of identity links between individuals. Furthermore, a generalization of the former approach proposed in [1] is based on pattern structures [6] and takes into account different pairs of classes at the same time in the discovery of link keys.

Link key candidates over two RDF datasets have to generate different and maximal link sets. However it appears that two different link key candidates may generate the same link set. This means that there exists some redundancy between the two link key candidates, that they should be considered as equivalent and merged. This can be achieved by looking at `owl:sameAs` which is an equivalence relation stating that two individual should be identified. The `owl:sameAs` relation generates partitions among pairs of individuals that can be used to detect redundant link key candidates and thus reduce their number, i.e., two candidates relying on the same partition are declared as redundant and thus merged.

In this paper, we present the discovery of link key candidates within the framework of pattern structure. Then, we introduce the notion of non-redundant link key candidate based on the equivalence relation induced by a link key candidate. Finally, we discuss how these candidates can be merged to reduce the search space of link keys.

2 Basics and Notations

2.1 RDF data

In this work, we deal with RDF datasets which are defined as follows:

Definition 1 (RDF dataset).
Let U be a set of IRIs (Internationalized Resource Identifier), B a set of blank nodes and L a set of literals. An RDF dataset is a set of triples $(s, p, v) \in (U \cup B) \times U \times (U \cup B \cup L)$.

Given a dataset D, we denote by:

- $I(D) = \{s \mid \exists p, v \ (s, p, v) \in D\}$ the set of individual identifiers,
- $P(D) = \{p \mid \exists s, v \ (s, p, v) \in D\}$ the set of property identifiers,
- $C(D) = \{c \mid \exists s \ (s, \texttt{rdf:type}, c) \in D\}$ the set of class identifiers. A triple $(s, \texttt{rdf:type}, c)$ means that the subject s is an instance of the class c.
- $I(c) = \{s \mid (s, \texttt{rdf:type}, c) \in D\}$ the set of instances of $c \in C(D)$,
- $p(s) = \{v \mid (s, p, v) \in D\}$ is the set of values (or "RDF objects") related to s through p.

An identity link is an RDF triple $(a, \texttt{owl:sameAs}, b)$ stating that the IRIs a and b refer to the same real-world entity. Fig. 1 represents two RDF datasets D_1 and D_2, where $P(D_1) = \{p_1, p_2, p_3, p_4\}$ and $P(D_2) = \{q_1, q_2, q_3, q_4\}$. Then $C(D_1) = \{c_1\}$ and $C(D_2) = \{c_2\}$ with $I(c_1) = \{a_1, a_2, a_3, a_4, a_5\}$ and $I(c_2) = \{b_1, b_2, b_3, b_4, b_5\}$. For example, the set of values of b_3 for the property q_2 is $q_2(b_3) = \{v_8, v_9\}$.

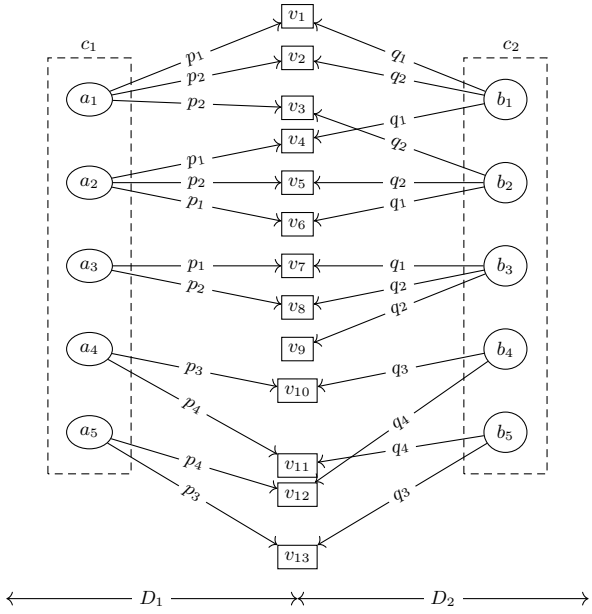


Fig. 1. Example of two RDF datasets. On the left-hand side, the dataset D_1 is populated with instances of the class c_1, and on the right-hand side the dataset D_2 is populated with instances of the class c_2.

2.2 Link Keys

Link keys are logical constructors allowing to deduce identity links (`owl:sameAs`). In this paper we will call *link key expression* the syntactic formulation of a link key, when we do not know whether the expression is a valid link key. For example, the link key expression $(\{\}, \{(\texttt{nbrOfPages}, \texttt{nbrOfBeds})\}, (\texttt{Book}, \texttt{Hospital}))$ will not satisfy the link key semantics since an instance of class `Book` and an instance of class `Hospital` cannot represent the same entity since `Book` and `Hospital` are disjoint classes.

Definition 2 (Link key expression, link key candidate). *Let D_1 and D_2 be two RDF datasets, $k = (Eq, In, (c_1, c_2))$ is a* link key expression *(over D_1 and D_2) iff $In \subseteq P(D_1) \times P(D_2)$, $Eq \subseteq In$, $c_1 \in C(D_1)$ and $c_2 \in C(D_2)$.*

The set of links $L(k)$ (directly) generated by k is the set of pairs of instances $(a, b) \in I(c_1) \times I(c_2)$ satisfying:

(i) for all $(p, q) \in Eq$, $p(a) = q(b)$ and $p(a) \neq \emptyset$,
(ii) for all $(p, q) \in In \setminus Eq$, $p(a) \cap q(b) \neq \emptyset$.

A link key expression $k_1 = (Eq_1, In_1, (c_1, c_2))$ is a link key candidate *if:*

(iii) $L(k_1) \neq \emptyset$,
(iv) k_1 is maximal i.e. there does not exist another link key expression $k_2 = (Eq_2, In_2, (c_1, c_2))$ such that $Eq_1 \subset Eq_2$, $In_1 \subset In_2$, and $L(k_1) = L(k_2)$.

The number of link key expressions may be exponential w.r.t. the number of properties. Then link key discovery algorithms only consider link key candidates which are link key expressions generating at least one link and that are maximal w.r.t. the set of links they generate.

3 Link Key Discovery

Here after we assume that all link key expressions are defined on the same pair of datasets D_1 and D_2 w.r.t. one pair of classes, yielding link key expressions of the form $k = (Eq, In, (c_1, c_2))$. In the following, we show how link keys may be discovered within the formalism of pattern structures (see details in [1]) and then we discuss the notion of non-redundant link keys.

Example 1. Let us consider the pattern structure $(G, (E, \sqcap), \delta)$ displayed in Table 1. Here we skip the details for building this table and the related PS lattice which can be found in [1].

The rows termed "PS objects" correspond to the set of objects G of the pattern structure and include pairs of related instances. The set of descriptions $(E, \sqcap)$ includes all possible pairs of properties preceded either by $\forall$ or $\exists$. The mapping δ relates a pair of instances $(a, b) \in I(c_1) \times I(c_2)$ to a description as follows: (i) $\delta(a, b)$ includes $\forall(p, q)$ whenever $p(a) = q(b)$ and $p(a) \neq \emptyset$, (ii) $\delta(a, b)$ includes $\exists(p, q)$ whenever $p(a) \cap q(b) \neq \emptyset$. Then the descriptions correspond to link key expressions (Eq, In) w.r.t. the pairs of classes (c_1, c_2). It should be noticed that it is possible to simultaneously work with several pairs of classes as explained in [1].

We have that $\delta(a_1, b_1) = \{\exists(p_1, q_1), \exists(p_2, q_2)\}$ because $p_1(a_1) \cap q_1(b_1) \neq \emptyset$ and $p_2(a_1) \cap q_2(b_1) \neq \emptyset$ while $\delta(a_2, b_1) = \{\exists(p_1, q_1)\}$ because $p_1(a_2) \cap q_1(b_1) \neq \emptyset$. Then $\delta(a_1, b_1) \sqcap \delta(a_2, b_1) = \{\exists(p_1, q_1)\}$ and thus $\delta(a_2, b_1) \sqsubseteq \delta(a_1, b_1)$. This can be read in the pattern concept lattice where the pattern concept pc_5 is subsumed by the pattern concept pc_4, i.e., the extent of pc_5 $\{(a_1, b_1), (a_2, b_2), (a_3, b_3)\}$ is included in the extent of pc_4 $\{(a_1, b_1), (a_2, b_1), (a_2, b_2), (a_3, b_3)\}$, while the intent $\{\exists(p_1, q_1)\}$ of pc_4 is included in the intent of pc_5, $\{\exists(p_1, q_1), \exists(p_2, q_2)\}$.

The set of all pattern concepts is organized within the pattern concept lattice LKPS-lattice displayed in Fig. 2. Moreover, all potential link key candidates are lying in the intents of the pattern concepts in the lattice. The corresponding set of link key candidates is denoted by LKC.

PS objects (g)	descriptions ($\delta(g)$)
(a_1, b_1)	$\{\exists(p_1, q_1), \exists(p_2, q_2)\}$
(a_1, b_2)	$\{\exists(p_2, q_2)\}$
(a_2, b_1)	$\{\exists(p_1, q_1)\}$
(a_2, b_2)	$\{\exists(p_1, q_1), \exists(p_2, q_2)\}$
(a_3, b_3)	$\{\forall(p_1, q_1), \exists(p_1, q_1), \exists(p_2, q_2)\}$
(a_4, b_4)	$\{\forall(p_3, q_3), \exists(p_3, q_3)\}$
(a_4, b_5)	$\{\forall(p_4, q_4), \exists(p_4, q_4)\}$
(a_5, b_4)	$\{\forall(p_4, q_4), \exists(p_4, q_4)\}$
(a_5, b_5)	$\{\forall(p_3, q_3), \exists(p_3, q_3)\}$

Table 1. The pattern structure related to link key discovery over c_1 and c_2 introduced in Fig. 1.

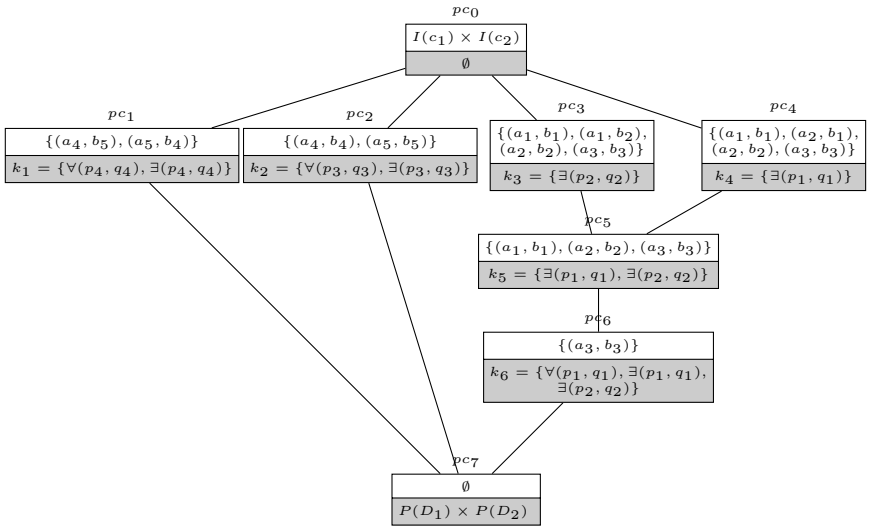


Fig. 2. The pattern concept intents in the pattern concept lattice LKPS-lattice include the complete set of link key candidates.

Let us consider the so-called LKPS-lattice and $pc = (L(k), k)$ a pattern concept, where the extent $L(k)$ corresponds to the set of links generated by k, and the intent k corresponds to a link key candidate. Let I denotes the set of instances $I = I(c_1) \cup I(c_2)$ and the binary relation $\simeq_k \subseteq I \times I$ such as $(a, b) \in L(k) \rightarrow a \simeq_k b$. The interpretation of $a \simeq_k b$ is: "k states that there exists a `owl:sameAs` relation between a and b". Actually $\simeq_k$ is an equivalence relation based on the fact that `owl:sameAs` itself is an equivalence relation. We say that k induces the equivalence relation $\simeq_k$ over I. Moreover $\simeq_k$ forms a partition over I where each element of this partition is an equivalence class. In fact the $\simeq_k$ equivalence relation will help us to build more concise set of link key candidates since it allows to identify *non-redundant link key candidates* termed nr-LKC. A link key candidate k_1 is a nr-LKC in LKC if there is no other candidate k_2 in LKC such that $\simeq_{k_1}$ and $\simeq_{k_2}$ form the same partition. Otherwise, k_1 is redundant.

In Fig. 2, it can be observed that $\simeq_{k_3}$ and $\simeq_{k_4}$ form the same partition, namely $\{(a_1, b_1, a_2, b_2), (a_3, b_3)\}$ (it should be noticed that singletons are omitted for the sake of readability). Then the link key candidates k_3 and k_4 are redundant. By contrast, k_1 is a nr-LKC because there is no other candidate k in LKC such that $\simeq_{k_1}$ and $\simeq_k$ form the same partition.

Let us briefly explain how $\simeq_{k_3}$ and $\simeq_{k_4}$ are inducing the same partition, namely $\{(a_1, b_1, a_2, b_2), (a_3, b_3)\}$. The extent of k_3 in LKPS-lattice is given by $\{(a_1, b_1), (a_1, b_2), (a_2, b_2), (a_3, b_3)\}$. By transitivity and symmetry of `owl:sameAs`, we have that (a_1, b_2) and (b_2, a_2) yields (a_1, a_2), then (a_2, a_1) and (a_1, b_1) yields (a_2, b_1), and finally (b_1, a_2) and (a_2, b_2) yields (b_1, b_2) and the complete graph between (a_1, a_2, b_1, b_2). The same thing applies when we consider k_4 instead of k_3. This intuitively shows how $\simeq_{k_3}$ and $\simeq_{k_4}$ are inducing the same partition.

One main straightforward application of identifying nr-LKC is the ability to reduce the search space of link keys since the set of nr-LKC is included in LKC. Indeed, this can be seen as a refinement where redundant link key candidates inducing the same partition are merged. For example, since $\simeq_{k_3}$ and $\simeq_{k_4}$ form the same partition, then, k_3 and k_4 can be merged into a nr-LKC $k_{34} = \{k_3, k_4\}$. Among the perspectives is to consolidate the theory and practice of link key discovery based on partition pattern structures initially introduced for mining functional dependencies in [4].

References

1. Abbas, N., David, J., Napoli, A.: Discovery of link keys in RDF data based on pattern structures: Preliminary steps. In: Proceedings of ICFCA. CEUR Workshop Proceedings, vol. 2668, pp. 235–246. CEUR-WS.org (2020)
2. Atencia, M., David, J., Euzenat, J.: Data interlinking through robust linkkey extraction. In: Proceedings of ECAI. pp. 15–20 (2014)
3. Atencia, M., David, J., Euzenat, J., Napoli, A., Vizzini, J.: Link key candidate extraction with relational concept analysis. Discrete applied mathematics **273**, 2–20 (2020)
4. Baixeries, J., Kaytoue, M., Napoli, A.: Characterizing functional dependencies in formal concept analysis with pattern structures. Annals of Mathematics and Artificial Intelligence **72**, 129–149 (2014)
5. Ferrara, A., Nikolov, A., Scharffe, F.: Data Linking for the Semantic Web. International Journal of Semantic Web and Information Systems **7**(3), 46–76 (2011)
6. Ganter, B., Kuznetsov, S.O.: Pattern Structures and Their Projections. In: Proceedings of the International Conference on Conceptual Structures (ICCS). pp. 129–142. LNCS 2120, Springer (2001)
7. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer (1999)
8. Nentwig, M., Hartung, M., Ngonga Ngomo, A.C., Rahm, E.: A survey of current link discovery frameworks. Semantic Web **8**(3), 419–436 (2017). https://doi.org/10.3233/SW-150210

Formal Concept Analysis for Semantic Compression of Knowledge Graph Versions

Damien Graux[1,2], Diego Collarana[3,4], Fabrizio Orlandi[2]

[1] Inria, Université Côte d'Azur, CNRS, I3S, France
[2] ADAPT SFI Centre, Trinity College Dublin, Ireland
[3] Fraunhofer IAIS and University of Bonn, Germany
[4] Universidad Privada Boliviana, Bolivia
damien.graux@inria.fr, orlandif@tcd.ie,
diego.collarana.vargas@iais.fraunhofer.de

Abstract. Recent years have witnessed the increase of openly available knowledge graphs online. These graphs are often structured according to the W3C semantic web standard RDF. With this availability of information comes the challenge of coping with dataset versions as information may change in time and therefore deprecates the former knowledge graph. Several solutions have been proposed to deal with data versioning, mainly based on computing data deltas and having an incremental approach to keep track of the version history. In this article, we describe a novel method that relies on aggregating graph versions to obtain one single complete graph. Our solution semantically compresses similar and common edges together to obtain a final graph smaller than the sum of the distinct versioned ones. Technically, our method takes advantage of FCA to match graph elements together. We also describe how this compressed graph can be queried without being unzipped, using standard methods.

1 Introduction

Knowledge Graphs (KG) are becoming the preferred data model for integrating heterogeneous data into actionable knowledge. General domain knowledge graphs such as Wikidata [21] and DBpedia [16] have been used as core knowledge sources to develop intelligent applications. Moreover, domain-specific knowledge graphs are being constructed in almost all domains. Knowledge graphs provide a flexible data model allowing the addition and deletion of facts in the graph. Therefore, KGs are dynamic and evolve over time: facts are either added or removed. Such a paradigm leads to the availability of several versions of the *same* KG *e.g.* each version may correspond to a specific release published by the data providers.

Practically, KGs are often modeled according to the RDF standard, proposed by the W3C. In a nutshell, the RDF[1] data model implements multi-relational directed labelled graphs using triples. Indeed, a triple $t = (subject, predicate, object)$ encodes a fact, *e.g.*, `ex:CR7 ex:born ex:Madeira` states that Cristiano Ronaldo was born in Madeira (Figure 1). To efficiently handle the continuously growing knowledge graphs, there is a need for efficient compression techniques to allow practitioners to share and

[1] https://www.w3.org/TR/rdf11-primer/

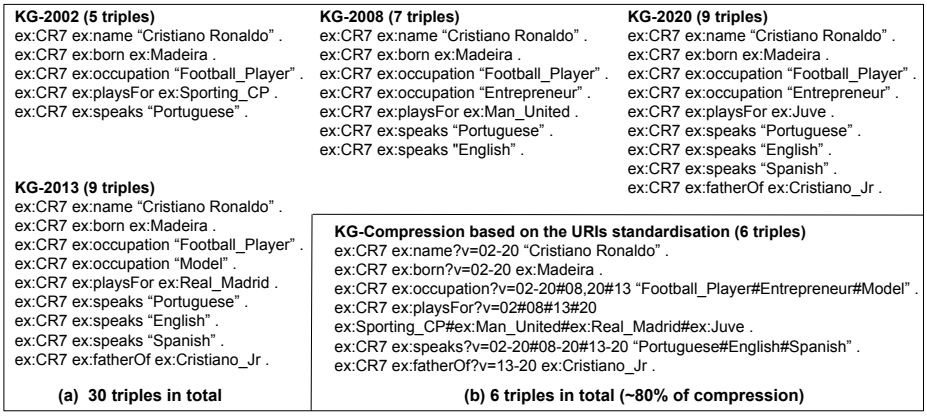


Fig. 1: **Motivating example:** (a) shows four different KG deltas of CR7 entity containing 30 triples in total. (b) depicts our vision of a semantically compressed KG with six triples gaining 80%. The most critical challenge is searching for implicational dependencies in the different deltas. Hence, FCA plays a crucial role in our approach.

store their graphs more easily. Ideally, knowledge graph compression algorithms should serialize RDF data compacting RDF representation in a manner that still allows for querying. By doing so, it should then be possible to query directly compressed graphs without having to “unzip” them prior; this strategy would thereby save memory.

To date, most RDF compression approaches focus on syntactic compression, with systems that modify the standard RDF data model. These systems require the implementation of complex encoders and decoders to deal with various KG versions, computing deltas of triples to capture changes spanning across several versions. For example, Álvarez-García *et al.* [2] implement algorithms directly in the storage solution. The authors apply compact tree structures to the well-known vertical-partitioning technique reaching a great compression degree. However, additional data structures are needed, including a mapping dictionary and adjacency matrices. In this work, we focus on a semantic compression of a set of RDF KGs, *i.e.*, reducing the number of triples by replacing or grouping repetitive parts observed in the various graphs of the set. Technically, our method takes advantage of *formal concept analysis* (FCA) to match graph elements together. Moreover, our method allows to aggregate together concepts considered as “similar” according to a chosen similarity metric.

2 CR7’s career as a motivating example

First, let us take as example four different versions of the same small knowledge graph (Figure 1) encoding facts about Cristiano Ronaldo (*URI = ex:CR7*). These versions provide facts about CR7 at various moments of his career: in 2002, 2008, 2013 & 2020.

We notice that there are redundant facts among the knowledge graph versions, *e.g.*, `ex:CR7 ex:name "Cristiano Ronaldo"` is present in all of them. Intuitively, a compressed graph of these four versions should carry only once this specific state-

ment, mentioning that it is present in each of the considered versions. Such mentions could be done by enriching the URIs of both predicates and objects, specifying e.g. the range of the version where the statement holds. Similarly, the `ex:playsFor` concept changes across versions as Cristiano played for a different team in each version. Our semantic compression, using the same kind of URI annotation, encompasses such changes to wrap all these statements within a single triple (cf. the 4th triple of Figure 1-b).

At the end of this process, the four versions of the CR7 graph are compressed into 6 triples. Moreover, the overall information contained in the obtained compressed KG is strictly the same as the sum of the information available in the various distinct versions. Therefore, for this basic example, our approach allows practitioners to carry one small KG of 6 triples instead of 4 distinct KGs gathering 30 triples.

3 Definitions

In this section, we introduce the main concepts used for our description of the approach and formally define them adapting existing definitions from the literature. First we define the concepts related to RDF Knowledge Graphs and then those related to FCA.

3.1 Knowledge Graph

Definition 1. ***Sets:*** *Let U and L be the mutually disjoint sets of URI references and literals, respectively. Let $P \subseteq U$ be the set of all properties.*

Definition 2. ***RDF triple:*** *An RDF triple $t = (s, p, o) \in U \times P \times (U \cup L)$ displays the statement that the subject s is related to the object o via the predicate p. In this work, we do not consider blank nodes as speficied in the W3C RDF standard.*

Definition 3. ***RDF Knowledge Graph:*** *An RDF Knowledge Graph G is a finite set of RDF triples, where $t = (s, p, o) \in G$. An RDF graph can also be viewed as a finite set of edges t, of the form $s \xrightarrow{p} o$, in a directed edge-labelled graph.*

3.2 Formal Concept Analysis

Formal concept analysis (FCA) is a methodology for extracting a concept hierarchy from sets of entities and their properties [22]. In other words, FCA is based on extracting *formal concepts* from *formal contexts*. Adapting the definitions in [8,14]:

Definition 4. ***Formal Context:*** *A formal context is a triple $X = (E, A, I)$, where E is a set of entities, A is a set of attributes, and $I \subseteq E \times A$ is the incidence: a set of pairs such that $(e, a) \in I$ if and only if the attribute a is defined for entity e.*

Definition 5. ***Formal Concept:*** *Let $X = (E, A, I)$ be a formal context; for a subset of entities $F \subset E$, let $H(F) := \{a \in A \mid \forall f \in F : (f, a) \in I\}$, conversely, for a subset of attributes $G \subset A$, let $K(G) := \{e \in E \mid \forall g \in G : (e, g) \in I\}$. A formal concept of the formal context X is an ordered pair (F, G) such that $H(F) = G$ and $K(G) = F$. If (F, G) and $(F1, G1)$ are formal concepts of X, then $(F, G) \leq (F1, G1)$ if $F \subset F1$ or, equivalently, if $G1 \subset G$.*

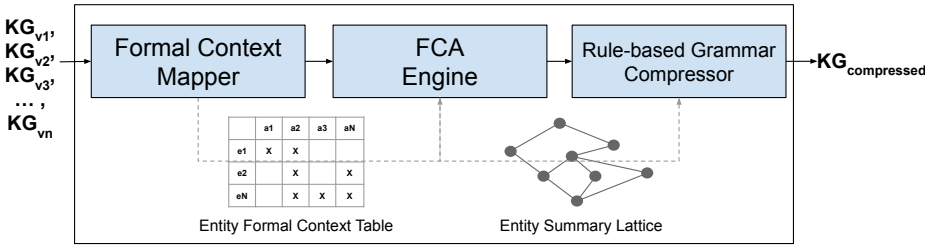


Fig. 2: **Zip function**: our compression approach creates a formal context from the same entities in different KG deltas. Then, we apply FCA to obtain an entity summary lattice. Finally, we execute a rule-based grammar to produce a compressed KG as output.

Definition 6. ***Concept Lattice:*** *Let $X = (E, A, I)$ be a formal context. The set of all formal concepts of X with the partial ordering defined in Definition 5 is called the concept lattice of X.*

4 An approach for zipping Knowledge Graph versions together

Grounded on FCA, we propose a zip function for compressing RDF knowledge graphs providing a solution to the problem of semantically compressing RDF graphs. Figure 2 depicts the main steps defined for our zip function. Our zip function follows a three-fold approach. We receive as input a set of KG versions *i.e.* several *complete* versions of the same Knowledge Graph (containing thereby redundant triples). First, we compute and prepare the formal context. Then, we run an FCA Engine to produce the formal concepts. Finally, we apply a set grammar rules to synthesize a (single) semantically compressed KG from the initial set of graphs.

Formal Context Mapper: First, from the set of KG deltas, we create a formal context, as defined in Definition 4. As E we consider objects of the same type in the KG deltas to be the entities. As A we consider all the properties in E to be the attributes, and the incidence I is given by the use of that property as a predicate on the given subject. Table 1 presents the entity formal context for the example introduced in Section 2, for instance, the RDF triple "`ex:CR7 ex:playsFor ex:Juve`" is only stated in 'KG-2020' as marked in the bottom-right corner cell.

FCA Engine: Second, we apply an FCA implementation to compute the formal concepts out of the formal context as defined in Definition 5. More visually, Figure 3 provides a concept lattice (Definition 6) corresponding to the Section 2 example. For instance at a glance one may see that "`name-CR`, `born-Madeira`, `occu-Player`, `speak-Por`" are statements made in every versions of the Knowledge Graph (practically it would be useful to store only once this information instead of four times).

Rule-based Grammar Compressor: Taking the formal concepts output by the FCA Engine, the idea to obtain a single compressed Knowledge Graph is to "*group*" the redundancies by subjects: meaning that for each distinct subject of the versions we establish the list of (predicate,object) available and then we tag the predicates and the object using the version name. In practice, we take advantage of the fact that the URIs

	name-CR	born-Madeira	occu-Player	plays-Sporting	speaks-Por	occu-Enter	plays-MU	speaks-En	occu-Model	plays-Madrid	speaks-Es	father-CRJ	plays-Juve
CR7-02	x	x	x	x	x								
CR7-08	x	x	x		x	x	x	x					
CR7-13	x	x	x		x			x	x	x	x	x	
CR7-20	x	x	x		x	x		x			x	x	x

Table 1: Example of the entity formal context that our approach creates for CR7 entity.

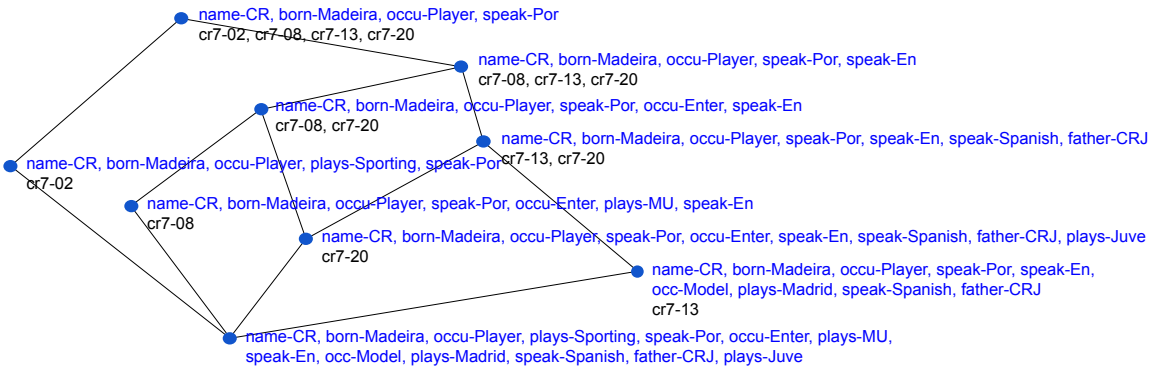


Fig. 3: Example of a resulting lattice after applying FCA to an entity formal context. We browse this lattice, employing rules to compress facts into a compressed KG.

used in RDF can be enriched, and we apply (on the predicates and the objects) the following rule-based grammar to compress the KGs semantically.

- **Hash** "`#`" is used for ordered separation of version numbers and their corresponding objects, both in new predicate IRIs and new concatenated object IRIs;
- **Hyphen** "`-`" indicates a continuous range of versions (*i.e.* from-to);
- **Comma** "`,`" indicates a discrete list of individual versions.

For example, in Figure 1, "`ex:CR7 ex:born?v=02-20 ex:Madeira`" means that "`ex:CR7 ex:born ex:Madeira`" was present in all the versions from '02' to '20'.

5 Querying the compressed Knowledge Graph

Fernández et al. [11] categorised all possible retrieval needs for versioned RDF archives. They identify six different types of retrieval needs, regarding the query type (materialisation or structured queries) and the target (version/delta) of the query. These types are listed below, including example queries based on our CR7 example (Figure 1).

1. **Single-version structured queries** are performed only on one specific version.
   `Q1-a`: In 'KG-2013' what team did CR7 play for?
   `Q1-b`: What "occupations" did CR7 have in 'KG-2013'?
2. **Cross-version structured queries** must be satisfied across different versions.
   `Q2-a`: In which KG version did CR7 play for ex:Juve ?
   `Q2-b`: In which KG versions did CR7 have the "occupation" of 'Entrepreneur'?
   `Q2-c`: Which predicates connect CR7 to ex:Madeira and in which versions?

3. **Single-delta structured queries** are the counterparts of the above version-focused queries, but must be satisfied on change instances instead.
   `Q3-a`: What triple with subject 'CR7' and predicate 'ex:speaks' was added between the two consecutive versions 'KG-2002' and 'KG-2008'?
4. **Cross-delta structured queries** are the counterparts of the above version-focused queries, but must be satisfied on change instances instead.
   `Q4-a`: What has changed between the non-consecutive versions 'KG-2002' and '2020'?
5. **Delta materialisation queries** retrieve the delta between two or more versions.
   `Q5-a`: What has changed between the consecutive versions 'KG-2013' and '2020'?
6. **Version materialisation queries** correspond to the retrieval of a full version.
   `Q6-a`: What are all the statements about CR7 valid in version 'KG-2008'?

Naively, these retrieval needs can be satisfied unzipping the KG to re-obtain the different versions. Nevertheless, one advantage of our approach lies in the expressive power of the *de facto* RDF query language: SPARQL[2]. Indeed, practitioners can express each of the aforementioned queries using one single SPARQL query involving filters based on `regular expressions` to grasp the relevant predicates. This therefore allows any standard-compliant triplestore to load and query the compressed KG. For instance, the aforementioned `Q6-a` could correspond to the following SPARQL query:[3]

```
SELECT DISTINCT ?s ?p ?o
WHERE{
  {
    ?s ?p ?o .
    FILTER regex(str(?p), "[?&]v=([^&]*)08.*$").
  }UNION{
    ?s ?p ?o .
    BIND (REPLACE(str(?p), "(..)*-.*", "$1") AS ?strFrom).
    BIND (REPLACE(str(?p), ".*-(..).*", "$1") AS ?strTo).
    FILTER (xsd:int(?strFrom) < 8).
    FILTER (xsd:int(?strTo) > 8).
  }
}
```

Different string functions, from the SPARQL 1.1 standard, are operated in order to check if the predicate `?p` was present in 'KG-2008'. Technically, this is done using regular expressions, for instance above, regex are used to extract the starting and ending versions in case the sought version is "*hidden*" within an interval using a hyphen. As a consequence, the compressed KG can be used to deal with version-related needs together with *conventional* querying while being kept "light" in terms of triples.

6 Related Work

We now provide an overview of the most pertinent related efforts in the areas of FCA for KGs, compression approaches for KGs, and KG versioning.

FCA on KGs: We see FCA supporting different tasks, including: Data Integration using ontologies [13], Entity Matching [20], Entity Temporal Evolution [19] and Modelling Dynamics [14], or Knowledge Exploration where FCA helps assess the completeness of Linked Datasets by mining definitions from RDF annotations [1]. FCA

[2] `https://www.w3.org/TR/sparql11-overview/`

[3] The presented query is simplified for space reasons; it might not generalize to all possible cases, but it could be adapted using additional `FILTER`s like the ones shown. See `https://github.com/badmotor/FCACompressRDF` for test data and all query examples.

is also used in specific cases such as in [4] to propose an alternative semantics for `owl:sameAs`, or to verbalize KG evolution [3]. Similarly, Aquin & Motta describe how to extract relevant questions to an RDF dataset [7]. Furthermore, Formica [12] and Rouane-Hacene et al. [17] extend FCA to respectively support formal ontology constructions in presence of uncertain data and to process multi-relational datasets.

KG Compression Techniques: In terms of knowledge graph compression techniques, several paradigms have been explored. In [9], the authors list the basic and naive techniques to compress an RDF graph. Later, solutions involving pre-processing and rewriting of the graph were proposed: for instance, the HDT representation of RDF triples was suggested [10]. Others describe methods to search the KGs for frequent patterns to factorise them [15]. Additionally, some solutions[4] summarise the KG hence compressing it, *e.g.* [23] compresses parts of a KG considering a set of user-selected queries.

KG Versioning: The literature has been focused on designing systems able to deal with many knowledge graph versions by computing deltas of triples *e.g.* OSTRICH [18]. Closer to our strategy, Cuevas & Hogan explored solutions for representing archives of versioned RDF data using the SPARQL standard [6].

7 Conclusion and Future Work

In this article, we described an architecture to enable the compression of a set of knowledge graph versions into a compressed one, while guaranteeing no loss of information. Furthermore, we presented how such a compressed knowledge graph can be queried directly without decompression, using any existing SPARQL-compliant endpoint.

To strengthen our solution, we will identify and evaluate more robust characters as delimiters for the version parameters and the concatenated object IRIs. This is because the "#" character could be present also in the original (non-concatenated) IRIs, and the "," and "−" characters could break our parser when parsing the version numbers embedded in the predicates. Potentially, the proposed approach generates a high number of unique predicates and unique objects. This could create some performance issues at query time if the data is loaded into a triplestore, as these engines are not usually optimised for such conditions (the RDF Singleton Property model also suffers from the same issue). In the near future, we could allow the configuration of the threshold for the similarity metric and thereby open the discussion towards uncertain data as the zipped KG could be carrying triples whose subjects were considered equal. Finally, an evaluation on real and large datasets will be conducted.

Acknowledgements: We acknowledge the support of the EU H2020 Projects Opertus Mundi (GA 870228), LAMBDA (GA 809965), the EDGE Marie Skłodowska-Curie grant (No. 713567) at the ADAPT SFI Research Centre at Trinity College Dublin (cofunded under the ERDF Grant #13/RC/2106_P2), and the Federal Ministry for Economic Affairs and Energy (BMWi) project SPEAKER (FKZ 01MK20011A).

[4] See [5] for a survey on summarizing semantic graphs.

References

1. Alam, M., Buzmakov, A., Codocedo, V., Napoli, A.: Mining definitions from RDF annotations using formal concept analysis. In: IJCAI, AAAI Press (2015) 823–829
2. Álvarez-García, S., Brisaboa, N.R., Fernández, J.D., Martínez-Prieto, M.A.: Compressed k^2-triples for full-in-memory RDF engines. In: AMCIS. (2011)
3. Arispe, M., Tasnim, M., Graux, D., Orlandi, F., Collarana, D.: Verbalizing the evolution of knowledge graphs with formal concept analysis. In: NLIWOD colocated with ISWC. (2020)
4. Beek, W., Schlobach, S., van Harmelen, F.: A contextualised semantics for `owl:sameAs`. In: ESWC. Volume 9678 of Lecture Notes in Computer Science., Springer (2016) 405–419
5. Čebirić, Š., Goasdoué, F., Kondylakis, H., Kotzinos, D., Manolescu, I., Troullinou, G., Zneika, M.: Summarizing semantic graphs: a survey. VLDB journal **28**(3) (2019) 295–327
6. Cuevas, I., Hogan, A.: Versioned queries over RDF archives: All you need is SPARQL? In: MEPDaW @ ISWC. (2020) 43–52
7. d'Aquin, M., Motta, E.: Extracting relevant questions to an RDF dataset using formal concept analysis. In: K-CAP, ACM (2011) 121–128
8. Denniston, J.T., Melton, A., Rodabaugh, S.E.: Formal Contexts, Formal Concept Analysis, and Galois Connections. Electronic Proceedings in Theoretical Computer Science **129** (2013) 105–120
9. Fernández, J.D., Gutierrez, C., Martínez-Prieto, M.A.: RDF compression: basic approaches. In: WWW. (2010) 1091–1092
10. Fernández, J.D., Martínez-Prieto, M.A., Gutierrez, C.: Compact representation of large RDF data sets for publishing and exchange. In: ISWC, Springer (2010) 193–208
11. Fernández, J.D., Umbrich, J., Polleres, A., Knuth, M.: Evaluating query and storage strategies for RDF archives. Semantic Web **10**(2) (2019) 247–291
12. Formica, A.: Semantic web search based on rough sets and fuzzy formal concept analysis. Knowl. Based Syst. **26** (2012) 40–47
13. Fu, G.: FCA based ontology development for data integration. Inf. Process. Manag. **52**(5) (2016) 765–782
14. González, L., Hogan, A.: Modelling dynamics in semantic web knowledge graphs with formal concept analysis. In: WWW, ACM (2018) 1175–1184
15. Karim, F., Vidal, M.E., Auer, S.: Compacting frequent star patterns in RDF graphs. Journal of Intelligent Information Systems **55**(3) (2020) 561–585
16. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P.N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., Bizer, C.: DBpedia - A large-scale, multilingual knowledge base extracted from wikipedia. Semantic Web **6**(2) (2015) 167–195
17. Rouane-Hacene, M., Huchard, M., Napoli, A., Valtchev, P.: Relational concept analysis: mining concept lattices from multi-relational data. Ann. Math. Artif. Intell. **67**(1) (2013)
18. Taelman, R., Vander Sande, M., Verborgh, R.: OSTRICH: versioned random-access triple store. In: Companion of WWW. (2018) 127–130
19. Tasnim, M., Collarana, D., Graux, D., Orlandi, F., Vidal, M.: Summarizing entity temporal evolution in knowledge graphs. In: Companion volume of WWW, ACM (2019) 961–965
20. Tasnim, M., Collarana, D., Graux, D., Vidal, M.: Context-based entity matching for big data. In: Knowledge Graphs and Big Data Processing. Springer (2020)
21. Vrandecic, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Commun. ACM **57**(10) (2014) 78–85
22. Wille, R.: Restructuring lattice theory: an approach based on hierarchies of concepts. In: International conference on formal concept analysis, Springer (2009) 314–339
23. Zhang, H., Duan, Y., Yuan, X., Zhang, Y.: ASSG: Adaptive structural summary for RDF graph data. In: ISWC (Posters & Demos). (2014) 233–236