Rearrangement of fuzzy formal contexts for reducing cost of algorithms

1st International Joint Conference on Conceptual Knowledge Structures

Domingo López-Rodríguez, and Manuel Ojeda-Hernández

Departamento de Matemática Aplicada

UNIVERSIDAD DE MÁLAGA

CONCEPTS'24

Introduction and Motivation

- FCA allows us (and potential end-users) to represent and manage (semantically and syntactically) **all the available knowledge** in a data table.

  (Un)fortunately, real-world data tables are huge ⇒ lot of knowledge to exploit.

- The fundamental knowledge structures within a formal context are the concept lattice and the basis of valid attribute implications.
- The classical paradigm of absolute object-attribute relationships has been enriched by the incorporation of fuzzy set theory (see Burusco-Juandeaburre and Fuentes-González [1994] and Bělohlávek [2002], among others).
- Complexity of formal contexts: number of concepts exhibiting exponential growth relative to the context size.

  Even more computationally expensive in the fuzzy setting.

- As a consequence, we need to study and develop strategies to make faster algorithms. Or maybe… to **make algorithms faster**?

A brief reminder of L-fuzzy formal concept analysis

- Let us consider a residuated lattice $\mathbb{L} = (L, \wedge, \vee, \otimes, \rightarrow, 0, 1)$.
- A **formal context** is a tuple (G, M, I) where G and M are non-empty sets of objects and attributes, respectively, and $I \in L^{G \times M}$ is a fuzzy relation, commonly called the incidence relation.
- This means that each attribute m is present at each object g to degree $I(g, m) \in L$. $I(g, m) = 0$ and $I(g, m) = 1$ have the classical meanings.
- Let $A \in L^G$ and $B \in L^M$,

  $$A^{\uparrow}(m) = \bigwedge_{g \in G} (A(g) \rightarrow I(g, m)) \qquad \text{and} \qquad B^{\downarrow}(g) = \bigwedge_{m \in M} (B(m) \rightarrow I(g, m)).$$

  The pair of mappings $(^{\uparrow}, ^{\downarrow})$ forms a fuzzy Galois connection (Yao and Lu [2009]).
- A **formal concept** is a pair (A, B) where $A \in L^G$ and $B \in L^M$ and $A^{\uparrow} = B$ and $B^{\downarrow} = A$.

Attribute ordering

- Concept lattice construction algorithms (Bělohávek et al. [2010], Krajča et al. [2010], Andrews [2018]) inherently depend on a pre-defined attribute order to guide the exploration.

- The order in which attributes are analysed may affect algorithms' performance.

- *This idea is not new*: the utilisation of doubly-lexically ordered tables has been investigated in the context of maintaining standardisation and enabling recursive partitioning (Pattison and Nataraja [2023]).

- But **this idea has not been empirically explored**.

Rationale

- Different strategies for different algorithms?
- NextClosure (Ganter [2010]) employs the lexicographic order to prune the computation of intents (avoiding duplicates). In layman terms, the *first* attribute to be explored corresponds to the last column in the context table.
- The CbO family of algorithms (Kuznetsov [1993]) operates primarily on the context's extents. Pruning is performed also by the computation of intents.
- Intuitive idea: if the first attributes explored by the algorithm are able to produce larger intents, then a greater pruning will be produced in the first steps of the algorithm, reducing its computational time.
- But... **how do we decide/determine which attributes are going to produce a larger intent**?

Proposed rearrangements

- We expect that columns containing higher truth values will generate larger closed sets if explored jointly.

1. **Lower Cone Priority**. For each attribute m, the size of the lower cone of each $I(g, m) \in L$ is determined – let us call it $\mathrm{lcs}(g, m)$. Averaging over all $m \in M$, a preference score is obtained.

   What does this preference score tell us when comparing two attributes?

   The Lower Cone ordering prioritises attributes based on the average position of their values within the lattice L. A higher average lower cone cardinality for a column indicates, on average, that its truth values occupy higher positions within the lattice structure compared to another column with a lower average.

2. **Non-Zero Element Count**. This approach prioritises sparsity. The less sparse columns (the ones with more non-zero entries) are given a higher preference score.

   The nnz (number of non-zero elements) ordering offers a simplified alternative to the Lower Cone approach, particularly in crisp contexts (where values are binary). Here, the reordering computation prioritises columns with fewer non-zero elements. While these two orderings often coincide in crisp contexts, they can diverge in the case of L-fuzzy contexts.

- The attribute ordering is then established such that attributes with the highest preference scores (type 1 or 2) are processed first by the algorithm.

Experiments and results

- Recall that the NextClosure algorithm stands alone as the sole algorithm with extension to the L-fuzzy setting (Bělohlávek [2002]).
- We can explore what happens in the crisp case, analyzing if *nnz* can help reduce the computational cost of algorithms.

  We'll be back on fuzzy NextClosure later.

- Metrics used:
  - *Number of Attribute Intents Computed*: This metric quantifies the number of computations of the form $X^{\uparrow}(m)$, where $X \subseteq G$ and $m \in M$. These computations represent one of the most time-consuming operations within the core functionalities of the algorithms.
  - *Algorithm Execution Time*: This metric directly measures the time required for the algorithm to execute.

Table 1: Results in the binary setting. Time is measured in seconds.

Original ordering

				FastCbO		InClose5		NextClosure	
$\lvert G\rvert$	$\lvert M\rvert$	δ	$\lvert\mathbb{B}(\mathbb{K})\rvert$	Intents	t	Intents	t	Intents	t
125	25	0.10	200.7	19366.7	0.0007	4985	0.0003	5912	0.0123
		0.25	1585	122041.7	0.0049	51208.7	0.0022	35461	0.0244
	50	0.10	604	184816.7	0.0039	37594	0.0011	35102	0.0281
		0.25	9899	2310250	0.0598	744429.3	0.0197	440208	0.1580
	75	0.10	1135.3	698650	0.0120	114623	0.0027	100650	0.0573
		0.25	26855	11914900	0.2607	3317822.7	0.0715	1769732	0.6522

nnz ordering

				FastCbO		InClose5		NextClosure	
$\lvert G\rvert$	$\lvert M\rvert$	δ	$\lvert\mathbb{B}(\mathbb{K})\rvert$	Intents	t	Intents	t	Intents	t
125	25	0.10	200.7	21241.7	0.0007	4619.7	0.0003	4948	0.0145
		0.25	1585	132533.3	0.0051	48963	0.0023	31529	0.0235
	50	0.10	604	201266.7	0.0042	32806.7	0.0011	28839	0.0245
		0.25	9899	2550183.3	0.0637	683976.7	0.0208	347074	0.1287
	75	0.10	1135.3	753425	0.0135	97556	0.0028	76753	0.0460
		0.25	26855	13266100	0.2792	3001031.3	0.0723	1405992	0.5278

Highlights:

- On NextClosure, the nnz rearrangement is able to reduce the number of computations **and** the execution time.
- On InClose5, the number of computations with *nnz* is reduced **but** the execution time remains the same.
- On FastCbO, the execution time is not affected, but the number of computations is increased!!

- Back to the primary objective: to analyse the influence of the two ordering approaches, namely Lower Cone and nnz, on the efficiency of NextClosure in the context of enumerating formal concepts.
- Same metrics as above.
- We incorporate the *lower cone* strategy.

Table 2: Results in the fuzzy setting for NextClosure. Time is measured in seconds.

					Original		Lower Cone		nnz	
$\lvert G\rvert$	$\lvert M\rvert$	L	δ	$\lvert\mathbb{B}(\mathbb{K})\rvert$	Intents	t	Intents	t	Intents	t
125	25	L_2	0.10	452	34340	0.0194	28870.7	0.0183	28684	0.0184
			0.25	5928.7	331506.7	0.0806	283726.7	0.0716	279589.3	0.0734
		L_4	0.10	773.7	173152	0.0341	142184	0.0302	140128	0.0323
			0.25	15234	2500338	0.3264	2115076	0.2852	2023846	0.2811
		L_8	0.10	1049.3	766460	0.0666	622630	0.0563	593683.3	0.0543
			0.25	25782.3	14170673.3	1.0362	11834833.3	0.8700	11043656.7	0.8108
	50	L_2	0.10	1512.3	223874.7	0.0642	173634.7	0.0532	168849.3	0.0519
			0.25	40886	4404921.3	0.9925	3622664	0.8110	3555716	0.7872
		L_4	0.10	2584.7	1160386	0.1824	902794	0.1490	859176	0.1373
			0.25	101480.3	33736054	4.7487	25262250	3.5343	24844340	3.4286
		L_8	0.10	3821	5543853.3	0.4761	4233523.3	0.3622	3971513.3	0.3327
			0.25	174914.7	187260976.7	15.3390	147066386.7	11.9332	140793376.7	11.1887
	75	L_2	0.10	3123.7	693174.7	0.1973	519308	0.1519	505765.3	0.1458
			0.25	117370	20047082.7	4.9866	15224972	3.7696	14757693.3	3.5845
		L_4	0.10	5544.3	3540956	0.6140	2662388	0.4540	2578906	0.4394
			0.25	317589	153110538	24.3419	116877862	18.3480	112582956	17.8424
		L_8	0.10	7877.7	17343346.7	1.7396	12234906.7	1.2016	11507746.7	1.1215
			0.25	561812	877110786.7	81.9409	669309533.3	61.3007	632467553.3	57.4833

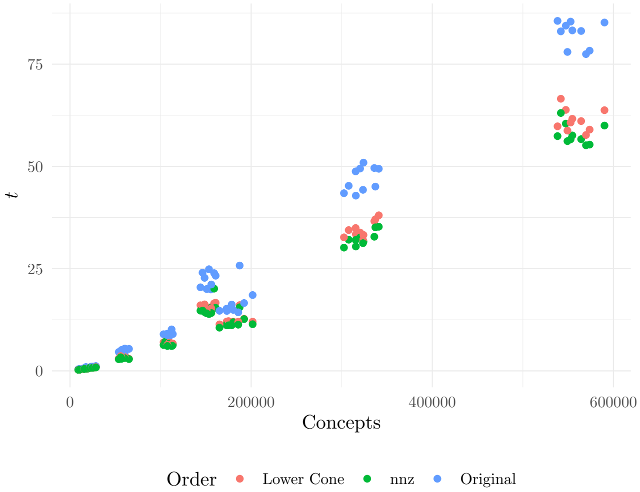

Figure 1: Execution time as a function of the number of computed concepts.

Conclusions

- We have started the exploration of formal context rearrangement in order to reduce the computational cost of constructing the concept lattice.
- Two strategies for the fuzzy setting have been defined.
- In the simplest case, for crisp contexts, the computations made by NextClosure and InClose are reduced wrt a random initial arrangement of the context columns.
- FastCbO does not follow this pattern, what reinforces the idea that different algorithms need different rearrangement strategies.
- In the fuzzy setting, both the *lower cone* and the *nnz* approaches seem to reduce the computational cost of NextClosure.

Note that the rearrangement of a formal context by means of any of these strategies is direct and requires much fewer calculations than the computation of the concept lattice. Thus, the implementation of these strategies as a ***pre-processing*** *step may be of great practical help.*

Future works

- Extension of FastCbO and InClose to fuzzy setting: analysis of the effect of rearranging a fuzzy formal context using the previous strategies. Will they follow the same pattern as in the crisp case?
- If so, are there other possible strategies for the rearrangement of a L-fuzzy formal context that could ensure a reduction of the computational effort of these algorithms?
- Explore the double-lexicographic ordering of formal contexts and exploit the obtained inherent structure to **make algorithms faster**.
- Explore the cost/benefit of computing (absolute and relatively) necessary attributes.
- Investigate how these strategies adapt to different data distributions. *Is there a "best" rearrangement for a given data distribution?*
- Extend this analysis to real-world problems (WIP).

References I

Simon Andrews. A new method for inheriting canonicity test failures in close-by-one type algorithms. In Dmitry I. Ignatov and Lhouari Nourine, editors, *Proceedings of the Fourteenth International Conference on Concept Lattices and Their Applications, CLA 2018, Olomouc, Czech Republic, June 12-14, 2018*, volume 2123 of *CEUR Workshop Proceedings*, pages 255–266. CEUR-WS.org, 2018. URL `https://ceur-ws.org/Vol-2123/paper21.pdf`.

Radim Běloháv́ek, Bernard De Baets, Jan Outrata, and Vilem Vychodil. Computing the lattice of all fixpoints of a fuzzy closure operator. *IEEE Transactions on Fuzzy Systems*, 18(3):546–557, 2010. ISSN 10636706. doi: 10.1109/TFUZZ.2010.2041006.

Radim Bělohlávek. *Fuzzy Relational Systems*. Springer, 2002.

Radim Bělohlávek. Algorithms for fuzzy concept lattices. In *Int. Conf. on Recent Advances in Soft Computing*, pages 200–205, 2002.

References II

Ana Burusco-Juandeaburre and Ramón Fuentes-González. The study of the L-fuzzy concept lattice. *Mathware and Soft Computing*, 1(3):209–218, 1994.

Bernhard Ganter. Two basic algorithms in concept analysis. In *Formal Concept Analysis: 8th International Conference, ICFCA 2010, Agadir, Morocco, March 15-18, 2010. Proceedings 8*, pages 312–340. Springer, 2010. doi: 10.1007/978-3-642-11928-6_22.

Petr Krajča, Jan Outrata, and Vilém Vychodil. Advances in algorithms based on CbO. In *CLA*, volume 672, pages 325–337. Citeseer, 2010.

Sergei O Kuznetsov. 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.

Tim Pattison and Aryan Nataraja. Doubly-lexical order supports standardisation and recursive partitioning of formal context. In *International Conference on Formal Concept Analysis*, pages 17–32. Springer, 2023.

W. Yao and L. X. Lu. Fuzzy Galois connections on fuzzy posets. *Mathematical Logic Quarterly*, 55:105–112, 2009.

Rearrangement of fuzzy formal contexts for reducing cost of algorithms

1st International Joint Conference on Conceptual Knowledge Structures

Domingo López-Rodríguez, and Manuel Ojeda-Hernández

Departamento de Matemática Aplicada

UNIVERSIDAD DE MÁLAGA

CONCEPTS '24