# Why and How Knowledge Discovery Can Be Useful for Solving Problems with CBR

Amedeo Napoli

# Why and How Knowledge Discovery Can Be Useful for Solving Problems with CBR
## (Extended Abstract)

Amedeo Napoli

LORIA, B.P. 70239, Équipe Orpailleur, Bâtiment B, F-54506 Vandœuvre-lès-Nancy
Amedeo.napoli@loria.fr

**Abstract.** In this talk, we discuss and illustrate links existing between knowledge discovery in databases (KDD), knowledge representation and reasoning (KRR), and case-based reasoning (CBR). KDD techniques especially based on Formal Concept Analysis (FCA) are well formalized and allow the design of concept lattices from binary and complex data. These concept lattices provide a realistic basis for knowledge base organization and ontology engineering. More generally, they can be used for representing knowledge and reasoning in knowledge systems and CBR systems as well.

**Keywords:** knowledge discovery in databases, Formal Concept Analysis, knowledge representation and reasoning.

## 1 Introduction

In this talk, we will discuss and illustrate links existing between knowledge discovery in databases (KDD), case-based reasoning (CBR), and knowledge representation and reasoning (KRR). KDD techniques especially based on Formal Concept Analysis (FCA) are well formalized and allow the design of concept lattices (from binary and complex data). These concept lattices provide a realistic basis for knowledge base organization, ontology engineering, and hierarchical reasoning. They can be used as a backbone for an ontology by transforming formal concepts into concepts representing knowledge [5,4]. From the point of view of CBR, concept lattices may be used for a series of tasks such as:

- case retrieval: assessing similarity between cases with similarity paths, and case or information retrieval [19,20],
- case adaptation: traversing the lattice structure for building adaptation paths between cases [8,9],
- case learning: organizing and updating the case base and the underlying concept lattice.

Moreover, FCA is related to data mining techniques, such a as itemset search and association rule extraction. The use of such techniques can improve the scalability of FCA w.r.t. the volume of data to be analyzed [28,29].

I. Bichindaritz and S. Montani (Eds.): ICCBR 2010, LNAI 6176, pp. 12–19, 2010.

By contrast, we will also investigate some aspects of CBR that can be reused in KDD and ontology learning, such as searching and managing ontology design patterns.

## 2 Knowledge Discovery in Databases (KDD)

Knowledge discovery in databases (KDD) consists in processing a large volume of data in order to extract useful and reusable knowledge units from these data. An expert of the data domain, the analyst, is in charge of guiding the extraction process, on the base of his/her objectives and domain knowledge. The extraction process is based on data mining methods returning information units from the data. The analyst selects and interprets a subset of the units for building "models" that may be further interpreted as knowledge units with a certain plausibility. The KDD process is performed with a KDD system based on components such as domain ontologies, data mining modules (either symbolic or numerical), and interfaces for interactions with the system, e.g. editing and visualization.

The KDD process can be considered along three main steps: data preparation, data mining, and interpretation of the extracted units. At each step, domain knowledge, possibly represented within ontologies, can play a substantial role for improving the KDD process [18]. Moreover, data mining methods can be either numeric or symbolic. In this talk, we will mainly focus on the second type and especially itemset search, association rule extraction, and Formal Concept Analysis (and extensions) [21].

The search for frequent itemsets consists in extracting from binary tables itemsets occurring with a support that must be greater than a given threshold [22,3,29,31]. Given a set of objects and a set of properties, an item corresponds to an attribute or a property of an object, and an itemset (a pattern) to a set of items. The support of an itemset corresponds to the proportion of objects owning the itemset, with respect to the whole population of objects. An itemset is frequent if its support is greater than a given frequency threshold $\sigma_{\mathtt{S}}$: a proportion at least equal to $\sigma_{\mathtt{S}}$ of objects own all items included in the itemset. The search for frequent itemsets is based on monotony constraints (base of the Apriori algorithm [1]). The search of frequent itemsets begins with the search of frequent itemsets of minimal length (or length 1). Then, the frequent itemsets are recorded and combined together to form the candidate itemsets of greater length. The non-frequent itemsets are discarded and all their super-itemsets. The candidate itemsets are tested and the process continues in the same way, until no more candidates can be formed.

From frequent itemsets it is possible to generate association rules of the form $\mathtt{A} \longrightarrow \mathtt{B}$ relating an itemset $\mathtt{A}$ with an itemset $\mathtt{B}$, that can be interpreted as follows: the objects owning $\mathtt{A}$ also own $\mathtt{B}$ with a support and a confidence [1,23]. More precisely, an association rule $\mathtt{A} \longrightarrow \mathtt{B}$ has a support defined as the support of the itemset $\mathtt{A} \cup \mathtt{B}$ and a confidence defined as the quotient $\mathtt{support}(\mathtt{A} \cup \mathtt{B})/\mathtt{support}(\mathtt{A})$ (that can be interpreted as a conditional probability). Then, a rule is valid if its confidence is greater than a confidence threshold $\sigma_{\mathtt{C}}$, and its support is greater

than the frequency threshold for itemsets $\sigma_s$ (a valid rule can only be extracted from a frequent itemset).

The numbers of extracted itemsets and rules may be very large, and thus there is a need for pruning the sets of extracted itemsets and rules for ensuring a subsequent interpretation of the extracted units. This is especially true when the interpretation has to be done –and this is usually the case– by the analyst who is in charge of interpreting the results of the KDD process [6].

Actually, the search for itemsets and association rules are related to concept lattices: they correspond to a breadth-first search in the concept lattice associated with the formal context under study.

## 3 Formal Concept Analysis and Derived Formalisms

### 3.1 The Basic Framework of FCA

The framework of FCA is fully detailed in [12]. FCA starts with a formal context $(G, M, I)$ where $G$ denotes a set of objects, $M$ a set of attributes, or items, and $I \subseteq G \times M$ a binary relation between $G$ and $M$. The statement $(g, m) \in I$ is interpreted as "the object $g$ has attribute $m$". Two operators $(\cdot)'$ define a Galois connection between the powersets $(2^G, \subseteq)$ and $(2^M, \subseteq)$, with $A \subseteq G$ and $B \subseteq M$:
$A' = \{m \in M \mid \forall g \in A : gIm\}$ and $B' = \{g \in G \mid \forall m \in B : gIm\}$.

For $A \subseteq G$, $B \subseteq M$, a pair $(A, B)$, such that $A' = B$ and $B' = A$, is called a formal concept. In $(A, B)$, the set $A$ is called the extent and the set $B$ the intent of the concept $(A, B)$. Concepts are partially ordered by $(A_1, B_1) \leq (A_2, B_2) \Leftrightarrow A_1 \subseteq A_2$ ($\Leftrightarrow B_2 \subseteq B_1$). With respect to this partial order, the set of all formal concepts forms a complete lattice called the concept lattice of $(G, M, I)$. As already mentioned above, natural links exist between between concept lattices, itemsets, and association rules [3,31,28].

When one consider non binary contexts, e.g. numerical or interval data, conceptual scaling is often used for binarizing data and for obtaining a binary formal context [12]. Then, a numerical dataset is described by a many-valued context. $(G, M, W, I)$ is a many-valued context where $G$ is a set of objects, $M$ a set of numerical attributes, $W$ a set of values (e.g. numbers), and $I$ a ternary relation defined on the Cartesian product $G \times M \times W$. The fact $(g, m, w) \in I$ or simply $m(g) = w$ means that the object $g$ takes the value $w$ for the attribute $m$.

Then, classical algorithms can be applied for designing concept lattices from scaled contexts [16]. However, adapted algorithms for designing a concept lattice may be directly applied on more complex data such as numbers, intervals, or graphs [17,15,14,13].

### 3.2 Pattern Structures

Instead of applying discretization leading to space and time computational hardness, one may directly work on original data. A pattern structure is defined as a generalization of a formal context describing complex data [11,15].

In classical FCA, object descriptions are sets of attributes, which are partially ordered by set inclusion, w.r.t. set intersection: let $P, Q \subseteq M$ two attributes sets, then $P \subseteq Q \Leftrightarrow P \cap Q = P$, and $(M, \subseteq)$, also written $(M, \cap)$, is a partially ordered set of object descriptions. Set intersection $\cap$ behaves as a meet operator and is idempotent, commutative, and associative. A Galois connection can then be defined between the powerset of objects $(2^G, \subseteq)$ and a meet-semi-lattice of descriptions denoted by $(D, \sqcap)$ (standing for $(M, \cap)$). This idea is used to define pattern structures in the framework of FCA as follows.

Formally, let $G$ be a set of objects, let $(D, \sqcap)$ be a meet-semi-lattice of potential object descriptions and let $\delta : G \longrightarrow D$ be a mapping associating each object with its description. Then $(G, (D, \sqcap), \delta)$ is a pattern structure. Elements of $D$ are patterns and are ordered by a subsumption relation $\sqsubseteq$: $\forall c, d \in D,\ c \sqsubseteq d \Longleftrightarrow c \sqcap d = c$. A pattern structure $(G, (D, \sqcap), \delta)$ gives rise to two derivation operators $(\cdot)^{\square}$:

$$A^{\square} = \prod_{g \in A} \delta(g), \quad for\ A \subseteq G \quad and \quad d^{\square} = \{g \in G | d \sqsubseteq \delta(g)\}, \quad for\ d \in (D, \sqcap).$$

These operators form a Galois connection between $(2^G, \subseteq)$ and $(D, \sqcap)$. Pattern concepts of $(G, (D, \sqcap), \delta)$ are pairs of the form $(A, d)$, $A \subseteq G$, $d \in (D, \sqcap)$, such that $A^{\square} = d$ and $A = d^{\square}$. For a pattern concept $(A, d)$, $d$ is a pattern intent and is the common description of all objects in $A$, the pattern extent. When partially ordered by $(A_1, d_1) \leq (A_2, d_2) \Leftrightarrow A_1 \subseteq A_2 \ \ (\Leftrightarrow d_2 \sqsubseteq d_1)$, the set of all concepts forms a complete lattice called pattern concept lattice. More importantly, the operator $(.)^{\square\square}$ is a closure operator and pattern intents are closed patterns. Existing FCA algorithms (detailed in [16]) can be used with slight modifications to compute pattern structures, in order to extract and classify concepts. Details can be found in [11,14,15].

Below, we analyze object descriptions as interval in numerical data. Pattern structures allows to directly extract concepts from data whose object descriptions are partially ordered. Considering a numerical dataset with objects in $G$ and attributes in $M$, a meet operator $\sqcap$ on interval patterns can be defined as follows. Given two interval patterns $c = \langle [a_i, b_i] \rangle_{i \in \{1,...,|M|\}}$, and $d = \langle [e_i, f_i] \rangle_{i \in \{1,...,|M|\}}$, then: $c \sqcap d = \langle [minimum(a_i, e_i), maximum(b_i, f_i)] \rangle_{i \in \{1,...,|M|\}}$ meaning that a convexification of intervals on each vector dimension is operated. The meet operator induces the following subsumption relation $\sqsubseteq$ on interval patterns: $\langle [a_i, b_i] \rangle \sqsubseteq \langle [c_i, d_i] \rangle \Leftrightarrow [a_i, b_i] \supseteq [c_i, d_i],\ \forall i \in \{1, ..., |M|\}$ where larger intervals are subsumed by smaller intervals.

A numerical dataset with objects $G$ and attributes $M$ can be represented by an interval pattern structure. Let $G$ be a set of objects, $(D, \sqcap)$ a meet-semi-lattice of interval patterns ($|M|$-dimensional interval vectors), and $\delta$ a mapping associating to any object $g \in G$ an interval pattern $\delta(g) \in (D, \sqcap)$. The triple $(G, (D, \sqcap), \delta)$ is an interval pattern structure (see examples and details in [15,13]).

Pattern structures are very useful for building concept lattices where the extents of concepts are composed of "similar objects" with respect to a similarity measure associated to the subsumption relation $\sqsubseteq$ in $(D, \sqcap)$ [13].

### 3.3 Relational Concept Analysis

Relational datasets are composed of a binary tables ($\texttt{objects} \times \texttt{attributes}$) and inter-object relations ($\texttt{objects} \times \texttt{objects}$). Formally, these binary tables introduce a set of objects $\texttt{G}_\texttt{i}$ described by a set of attributes $\texttt{M}_\texttt{i}$, and, as well, a set of relations $\texttt{r}_\texttt{k} \subseteq \texttt{G}_\texttt{i} \times \texttt{G}_\texttt{j}$. Relational datasets arise in a wide range of situations, e.g. Semantic Web applications [26], relational learning and data mining [10], refactoring of UML class models and model-driven development [27].

Relational Concept Analysis (RCA) extends FCA to the processing of relational datasets in a way allowing inter-objects links to be materialized and incorporated into formal concept intents. Links are thus scaled to become relational attributes connecting first objects to concepts and then concepts to concepts as role restrictions do in Description Logics (DL) [2]. The new attributes are complex properties reflecting the relational aspects of a formal concept. They nevertheless abide to the same classical concept formation mechanisms from FCA which means that the relational concept intents can be produced by standard FCA methods. Due to the strong analogy between role restrictions and relational attributes in RCA, formal concepts can be readily translated into a DL-based formalism [24], e.g. for ontology engineering purposes as in [5,4,25].

RCA was introduced and detailed in [24]. The data structure is described by a relational context family, composed of a set of contexts $\{\mathcal{K}_\texttt{i}\}$ and a set of binary relations $\{\texttt{r}_\texttt{k}\}$. A relation $\texttt{r}_\texttt{k} \subseteq \texttt{G}_\texttt{j} \times \texttt{G}_\ell$ connects two object sets, a domain $\texttt{G}_\texttt{j}$ ($\texttt{dom}(\texttt{r}_\texttt{k}) = \texttt{G}_\texttt{j}$) and a range $\texttt{G}_\ell$ ($\texttt{ran}(\texttt{r}_\texttt{k}) = \texttt{G}_\ell$). RCA is based on a "relational scaling" mechanism that transforms a relation $\texttt{r}_\texttt{k}$ into a set of relational attributes that are added to the context describing the object set $\texttt{dom}(\texttt{r}_\texttt{k})$. To that end, relational scaling adapts the DL semantics of role restrictions.

For each relation $\texttt{r}_\texttt{k} \subseteq \texttt{O}_\texttt{j} \times \texttt{O}_\ell$, there is an initial lattice for each object set, i.e. $\mathcal{L}_\texttt{j}$ for $\texttt{O}_\texttt{j}$ and $\mathcal{L}_\ell$ for $\texttt{O}_\ell$. For a relation $\texttt{r}_\texttt{k} \subseteq \texttt{O}_\texttt{j} \times \texttt{O}_\ell$, a relational attribute, is associated to an object $\texttt{o} \in \texttt{O}_\texttt{j}$ whenever $\texttt{r}_\texttt{k}(\texttt{o})$ satisfies a given constraint, where $\texttt{r}_\texttt{k}(\texttt{o})$ denotes the set of objects in $\texttt{O}_\ell$ in relation with $\texttt{o}$ through $\texttt{r}_\texttt{k}$. The relational attribute is denoted by $\forall\texttt{r}_\texttt{k}.\texttt{C}$ (universal scaling) when $\texttt{r}_\texttt{k}(\texttt{o}) \subseteq \texttt{extent}(\texttt{C})$ with $\texttt{r}_\texttt{k}(\texttt{o})$ possibly empty. The relational attribute is denoted by $\exists\texttt{r}_\texttt{k}.\texttt{C}$ (existential scaling) when $\texttt{r}_\texttt{k}(\texttt{o}) \cap \texttt{extent}(\texttt{c}) \neq \emptyset$. Other relational scaling operators exist in RCA and follow the classical role restriction semantics in DL.

Actually, RCA is a powerful mechanism for managing relations in the framework of FCA. In CBR, it could be used for example for associating elements of problem statements with elements of problem solutions, an association that was not possible in [9].

## 4 Elements for Discussion

Usually, considering knowledge systems, and CBR systems as well, knowledge units may have two major different origins: explicit knowledge (and cases) can be given by domain experts and implicit knowledge can be extracted from databases of different kinds, e.g. domain data or textual documents. Moreover, a KDD system, as any other knowledge system, improves its performance when it is

guided by domain knowledge [18]. Hereafter, some requirements for KDD systems, adapted from [6,30], are listed for discussion:

- A KDD system is a knowledge system: it should present to the user the underlying domain in an appropriate fashion and rely on domain knowledge (e.g. an ontology).
- Extending the system knowledge: domain representation should be extensible by addition of new concepts or classes resulting from mining or querying processes. Concepts and their instances must be reusable in queries. The question of extracting cases from data, which have to be made precise, remains open [7].
- Alternative classification and mining tools: it should be possible to define alternative classifications of data, e.g. alternative concept lattices. A set of different classification and mining tools should be available, possibly combining numerical and symbolic methods.
- Support to analysts: analysts should be supported by adequate visualization tools and in the interpretation of extracted units as well, in particular by domain knowledge.
- Monitoring and documenting the system evolution: tools managing versions can be used for monitoring changes in classes or concepts over time. The system should document the different steps of the knowledge discovery process.
- KDD is a flexible process and its results should reflect the plural nature of knowledge, i.e. extracting procedural or declarative knowledge units, and, as well, meta-knowledge units.
- KDD provides knowledge units for extending ontologies, and, reciprocally, knowledge systems and CBR systems can be used to guide and improve KDD.

Finally, the relations between knowledge representation, reasoning, and knowledge discovery with FCA, are explained as follows in [30]. Formal concepts and concept lattices provide a mathematization of real-world concept hierarchies. This yields a mathematical support to human reasoning, especially using the graphical representation of concept lattices. Then, conceptual knowledge discovery, considered as pattern discovery plus knowledge creation, can be guided by the design of concept lattices and a subsequent representation of the formal concepts within a knowledge representation formalism such as description logics. The process can be repeated until a satisfactory knowledge base is obtained.

## References

1. Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A.I.: Fast Discovery of Association Rules. In: Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R. (eds.) Advances in Knowledge Discovery and Data Mining, pp. 307–328. AAAI Press/MIT Press (1996)
2. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (eds.): The Description Logic Handbook. Cambridge University Press, Cambridge (2003)

3. Bastide, Y., Taouil, R., Pasquier, N., Stumme, G., Lakhal, L.: Mining frequent patterns with counting inference. SIGKDD Exploration Newsletter 2(2), 66–75 (2000)
4. Bendaoud, R., Napoli, A., Toussaint, Y.: Formal concept analysis: A unified framework for building and refining ontologies. In: Gangemi, A., Euzenat, J. (eds.) EKAW 2008. LNCS (LNAI), vol. 5268, pp. 156–171. Springer, Heidelberg (2008)
5. Bendaoud, R., Toussaint, Y., Napoli, A.: Pactole: A methodology and a system for semi-automatically enriching an ontology from a collection of texts. In: Eklund, P., Haemmerlé, O. (eds.) ICCS 2008. LNCS (LNAI), vol. 5113, pp. 203–216. Springer, Heidelberg (2008)
6. Brachman, R., Anand, T.: The Process of Knowledge Discovery in Databases. In: Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R. (eds.) Advances in Knowledge Discovery and Data Mining, pp. 37–57. AAAI Press / MIT Press (1996)
7. d'Aquin, M., Badra, F., Lafrogne, S., Lieber, J., Napoli, A., Szathmary, L.: Case base mining for adaptation knowledge acquisition. In: Veloso, M. (ed.) IJCAI 2007, pp. 750–755. Morgan Kaufmann, San Francisco (2007)
8. Diaz-Agudo, B., Gonzales-Calero, P.: Classification based retrieval using formal concept analysis. In: Aha, D.W., Watson, I. (eds.) ICCBR 2001. LNCS (LNAI), vol. 2080, pp. 173–188. Springer, Heidelberg (2001)
9. Diaz-Agudo, B., Gonzales-Calero, P.: Formal Concept Analysis as a support technique for CBR. Knowledge-Based Systems 14, 163–171 (2001)
10. Dzeroski, S., Lavrac, N. (eds.): Relational Data Mining. Springer, Berlin (2001)
11. Ganter, B., Kuznetsov, S.: Pattern structures and their projections. In: Delugach, H.S., Stumme, G. (eds.) ICCS 2001. LNCS (LNAI), vol. 2120, pp. 129–142. Springer, Heidelberg (2001)
12. Ganter, B., Wille, R.: Formal Concept Analysis. Springer, Berlin (1999)
13. Kaytoue, M., Assaghir, Z., Messai, N., Napoli, A.: Two Complementary Classication Methods for Designing a Concept Lattice from Interval Data. In: Link, S., Prade, H. (eds.) FoIKS 2010. LNCS, vol. 5956, pp. 345–362. Springer, Heidelberg (2010)
14. Kaytoue-Uberall, M., Duplessis, S., Kuznetsov, S., Napoli, A.: Two FCA-Based Methods for Mining Gene Expression Data. In: Ferré, S., Rudolf, S. (eds.) ICFCA 2009. LNCS (LNAI), vol. 5548, pp. 251–266. Springer, Heidelberg (2009)
15. Kuznetsov, S.: Pattern structures for analyzing complex data. In: Sakai, H., Chakraborty, M.K., Hassanien, A.E., Ślęzak, D., Zhu, W. (eds.) RSFDGrC 2009. LNCS, vol. 5908, pp. 33–44. Springer, Heidelberg (2009)
16. Kuznetsov, S., Obiedkov, S.: Comparing performance of algorithms for generating concept lattices. Journal of Experimental & Theoretical Artificial Intelligence 14(2/3), 189–216 (2002)
17. Kuznetsov, S., Samokhin, M.: Learning closed sets of labeled graphs for chemical applications. In: Kramer, S., Pfahringer, B. (eds.) ILP 2005. LNCS (LNAI), vol. 3625, pp. 190–208. Springer, Heidelberg (2005)
18. Lieber, J., Napoli, A., Szathmary, L., Toussaint, Y.: First Elements on Knowledge Discovery guided by Domain Knowledge (KDDK). In: Yahia, S.B., Nguifo, E.M., Belohlavek, R. (eds.) CLA 2006. LNCS (LNAI), vol. 4923, pp. 22–41. Springer, Heidelberg (2008)
19. Messai, N., Devignes, M.D., Napoli, A., Smaïl-Tabbone, M.: Many-valued concept lattices for conceptual clustering and information retrieval. In: Ghallab, M., Spyropoulos, C., Fakotakis, N., Avouris, N. (eds.) 18th European Conference on Artificial Intelligence (ECAI 2008), Patras, Greece, pp. 127–131. IOS Press, Amsterdam (2008)

20. Messai, N., Devignes, M.D., Napoli, A., Smaïl-Tabbone, M.: Using domain knowledge to guide lattice-based complex data exploration. In: 19th European Conference on Artificial Intelligence (ECAI 2010), Lisbon, Portugal. IOS Press, Amsterdam (to appear, 2010)
21. Napoli, A.: A smooth introduction to symbolic methods for knowledge discovery. In: Cohen, H., Lefebvre, C. (eds.) Handbook of Categorization in Cognitive Science, pp. 913–933. Elsevier, Amsterdam (2005)
22. Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Discovering frequent closed itemsets for association rules. In: Beeri, C., Bruneman, P. (eds.) ICDT 1999. LNCS, vol. 1540, pp. 398–416. Springer, Heidelberg (1998)
23. Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Pruning closed itemset lattices for association rules. International Journal of Information Systems 24(1), 25–46 (1999)
24. Rouane-Hacene, M., Huchard, M., Napoli, A., Valtchev, P.: A proposal for combining formal concept analysis and description logics for mining relational data. In: Kuznetsov, S.O., Schmidt, S. (eds.) ICFCA 2007. LNCS (LNAI), vol. 4390, pp. 51–65. Springer, Heidelberg (2007)
25. Rouane-Hacene, M., Napoli, A., Valtchev, P., Toussaint, Y., Bendaoud, R.: Ontology Learning from Text using Relational Concept Analysis. In: Kropf, P., Benyoucef, M., Mili, H. (eds.) International Conference on eTechnologies (MCETECH 2008), pp. 154–163. IEEE Computer Society, Los Alamitos (2008)
26. Staab, S., Studer, R. (eds.): Handbook on Ontologies, 2nd edn. Springer, Berlin (2009)
27. Stahl, T., Voelter, M., Czarnecki, K.: Model-Driven Software Development: Technology, Engineering, Management. John Wiley & Sons, Chichester (2006)
28. Szathmary, L., Valtchev, P., Napoli, A., Godin, R.: Constructing iceberg lattices from frequent closures using generators. In: Boulicaut, J.-F., Berthold, M.R., Horváth, T. (eds.) DS 2008. LNCS (LNAI), vol. 5255, pp. 136–147. Springer, Heidelberg (2008)
29. Szathmary, L., Valtchev, P., Napoli, A., Godin, R.: Efficient Vertical Mining of Frequent Closures and Generators. In: Adams, N., Boulicaut, J.F., Robardet, C., Siebes, A. (eds.) IDA 2009. LNCS, vol. 5772, pp. 393–404. Springer, Heidelberg (2009)
30. Wille, R.: Why can concept lattices support knowledge discovery in databases? Journal of Experimental & Theoretical Artificial Intelligence 14(2/3), 81–92 (2002)
31. Zaki, M.: Efficient algorithms for mining closed itemsets and their lattice structure. IEEE Transactions on Knowledge and Data Engineering 17(4), 462–478 (2005)