Gradual Discovery with Closure Structure of a Concept Lattice

Tatiana Makhalova[1], Sergei O. Kuznetsov[2], and Amedeo Napoli[1]

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

Abstract. An approximate discovery of closed itemsets is usually based on either setting a frequency threshold or computing a sequence of projections. Both approaches, being incremental, do not provide any estimate of the size of the next output and do not ensure that "more interesting patterns" will be generated first. We propose to generate closed itemsets incrementally, w.r.t. the size of the smallest (cardinality-minimal or minimum) generators and show that this approach (i) exhibits anytime property, and (ii) first generates itemsets of better quality and then those of lower quality.

Keywords: Closed itemsets · Pattern Mining · Anytime algorithms.

1 Introduction

Closed itemsets (or intents of concepts in FCA) are widely used in Pattern Mining (PM) and Knowledge Discovery [19]. They are of particular importance because (i) a closed itemset provides a concise representation of implications, (ii) it is a maximal set that embodies all other itemsets with the same support [17].

Itemsets (or patterns) are mostly used to solve the set cover problem, which is formulated for PM as follows. Let $\mathcal{D}$ be a dataset over attributes M and $\mathcal{C}$ be the set of all closed itemsets of $\mathcal{D}$. Then the goal is to find the smallest set of itemsets $\mathcal{S} \subseteq \mathcal{C}$ such that $\mathcal{S}$ covers entirely $\mathcal{D}$[3]. This problem is known to be NP-complete [12]. One of the most common approaches in finding approximate solutions is based on the Minimum Description Length (MDL) principle [11]. An MDL-based approach to PM [18, 21] is realized in two steps: (i) computing a set of candidates $\mathcal{F} \subseteq \mathcal{C}$, (ii) selecting greedily those itemsets that minimize the total description length.

The problem of reducing $\mathcal{C}$ to $\mathcal{F}$ is one of the most important in PM [1]. Given a set of itemsets $\mathcal{S}$ (solution of the set cover problem), the goal is to compute such a candidate set $\mathcal{F} \subseteq \mathcal{C}$ that contains (i) (almost) all itemsets from $\mathcal{S}$, and

[3] The problem can be studied in more general settings, i.e., by considering arbitrary itemsets $X \subseteq M$.

(ii) a small number of non-optimal itemsets, i.e., $|\mathcal{F} \setminus \mathcal{S}|$ should be as small as possible.

There are two main approaches to compute $\mathcal{F}$. The first one is based on frequency [1], while the second one is based on projections [9, 5]. Both approaches are incremental, i.e., they use the preceding output to compute the next one. A frequency-based approach consists in an incremental computing of itemsets of decreasing frequency. The problem is that by decreasing the threshold one can get exponentially many newly generated patterns. Moreover, the largest threshold that ensures the presence of all itemsets from $\mathcal{S}$ in $\mathcal{F}$ is unknown in advance. The second approach, in order to find "good projections", requires embedding of background knowledge, which is not always available. Moreover, adding a new projection does not always guarantee that the size of the output will not be exponential [4].

An efficient approach to compute $\mathcal{F}$ should be not only incremental and background-knowledge-free but also should (i) have a polynomial delay [10] (meaning that the time between the output of one itemset and the next is bounded by a polynomial function of the input size), (ii) generate itemsets of decreasing interestingness. Here, by interesting itemsets we mean those that provide a (sub)optimal solution of the set cover problem. The latter raises the question, which strategy of closed itemset discovery would meet all the requirements listed above?

In this paper we propose a minimum closure structure, which is induced by generators of the smallest size. We show that it (i) allows for exploring closed itemsets incrementally, (ii) has a polynomial delay, (iii) computes itemsets that ensure a complete data coverage after at most two iterations. Itemsets providing a better solution of the set cover problem might be found in subsequent iterations. We also revise two algorithms, namely Titanic [20] and Close-By-One (CbO) [14], and show that they exhibit the anytime property. Both of them allow for generating the same subsets of closed itemsets iteratively, but CbO computes them in a more efficient way from a PM perspective.

The paper has the following structure. In Section 2 we recall the main definitions. In Section 3 we introduce the closure structure and minimum closure structure of closed itemsets and discuss anytime properties of Titanic and CbO w.r.t. the introduced structures. In Section 5 we discuss the empirical properties of closure structures. In Section 6 we conclude and give directions of future work.

2 Preliminaries

Concepts and the partial order between them. A *formal context* [8] is a triple (G, M, I), where G is called a set of objects, M is called a set of attributes and $I \subseteq G \times M$ is a relation called incidence relation, i.e., $(g, m) \in I$ if object g has attribute m. The derivation operators $(\cdot)'$ are defined for $A \subseteq G$ and $B \subseteq M$ as follows: $A' = \{m \in M \mid \forall g \in A : gIm\}$, $B' = \{g \in G \mid \forall m \in B : gIm\}$.

Sets $A \subseteq G$, $B \subseteq M$, such that $A = A''$ and $B = B''$, are said to be closed. For $A \subseteq G$, $B \subseteq M$, a pair (A, B) such that $A'' = B$ and $B'' = A$, is called a *formal concept*, A and B are called *extent* and *intent*, respectively. In Data Mining, an *intent* is also called a *closed itemset (or closed pattern)*.

A partial order $\leq$ is defined on the set of concepts as follows: $(A, B) \leq (C, D)$ iff $A \subseteq C\,(D \subseteq B)$, a pair (A, B) is a subconcept of (C, D), while (C, D) is a superconcept of (A, B). With respect to this partial order, the set of all formal concepts forms a complete lattice $\mathcal{L}$ called the *concept lattice* of the formal context (G, M, I).

Equivalence classes and key sets. Let B be a closed itemset. Then all subsets $D \subseteq B$, such that $D'' = B$ are called *generators* of B and the set of all generators is called the equivalence class of B, denoted by $Equiv(B) = \{D \mid D \subseteq B, D'' = B\}$. A subset $D \in Equiv(B)$ is a *key* [20] or *minimal generator* of B if for every $E \subset D$ one has $E'' \neq D'' = B''$, i.e., every proper subset of a key is a member of the equivalence class of a smaller closed set. We denote a set of keys (*key set*) of B by $K(B)$ [3]. The set of keys is an order ideal, i.e., any subset of a key is a key [20]. The *minimum key set* $K^{min}(B) \subseteq K(B)$ is a subset of the key set that contains the keys of the minimum size, i.e., $K^{min}(B) = \{D \mid D \in K(B), |D| = min_{E \in K(B)}|E|\}$. In an equivalence class there can be several keys, but only one closed itemset, which is maximal in this equivalence class. An equivalence class is called trivial if it consists only of a closed itemset.

The size of the equivalence class is the nominator of (extentional) stability [13, 15], which assesses the probability that the set of objects given by the closed itemset B remains closed when removing a subset of attributes from intent B, or formally, $Stab_{\text{ext}}(A, B) = |\{D \subseteq B \mid D' = A\}|/2^{|B|}$. Intentional stability is defined similarly, i.e., $Stab_{\text{int}}(A, B) = |\{C \subseteq A \mid C' = B\}|/2^{|A|}$.

For the sake of simplicity, we denote attribute sets by strings of characters, e.g., *abc* instead of $\{a, b, c\}$.

Example. Let us consider a formal context given in Table 1. Five concepts have nontrivial equivalence classes, namely $(\{g_1\}, acf)$, $(\{g_3\}, ade)$, $(\{g_5, g_6\}, bdf)$, $(\{g_5\}, bdef)$ and $(\emptyset, abcdef)$. Among them, only *bdf* and *abcdef* have the minimum key sets that differ from the key sets, i.e., $K^{min}(bdf) = \{b\}$, $K(bdf) = \{b, df\}$. $K^{min}(abcdef) = \{ab\}$, $K(abcdef) = \{ab, adf, aef, cef\}$.

$Stab(bdf) = |\{\{g_5, g_6\}, \{g_6\}\}|/2^2 = 1/2$ and $Stab(ac) = |\{\{g_1, g_2\}\}|/2^2 = 1/4$, thus, if the analyzed dataset contains noise, *bdf* is more likely to be closed in the original dataset (without noise).

Table 1: Formal context and nontrivial equivalence classes.

	a	b	c	d	e	f
g_1	×		×			×
g_2	×		×		×	
g_3	×			×	×	
g_4			×		×	
g_5		×		×	×	×
g_6		×		×		×

C	$K^{min}(C)$	$K(C)$	$Equiv(C)$
ade	*ad*	*ad*	*ad*, *ade*
acf	*af*, *cf*	*af*, *cf*	*af*, *cf*, *acf*
bdf	*b*	*b*, *df*	*b*, *df*, *bd*, *bf*, *bdf*
bdef	*be*, *ef*	*be*, *ef*	*be*, *ef*, *bde*, *bef*, *def*, *bdef*
abcdef	*ab*	*ab*, *adf*, *aef*, *cef*	*ab*, *adf*, *aef*, *cef*, ..., *abcdef**

* The equivalence class includes all itemsets that contain a key from $K(abcdef)$.

Lectic order and lexicographic trees. Let M be a set of linearly ordered attributes $m_1 < m_2 < \ldots < m_k$. Then, an itemset $B_1 \subseteq M$ is *lectically smaller* than the set $B_2 \subseteq M$, if the smallest differing element belongs to B_1. A *lexicographic tree* is a tree that has the following properties: (i) the root corresponds to the null itemset; (ii) a node in the tree corresponds to an itemset, (iii) the parent of a node $m_1 \ldots m_{i-1} m_i$ is a node $m_1 \ldots m_{i-1}$.

Example. The lexicographic tree in Fig. 1 is built on closed itemsets with the minimum keys of size 1 and 2, i.e., all closed itemsets except *ace* from Fig. 1. In the tree we distinguish nodes that correspond to closed itemsets (dark) and intermediate nodes (white), needed to respect property (iii) from the definition of the lexicographic tree.

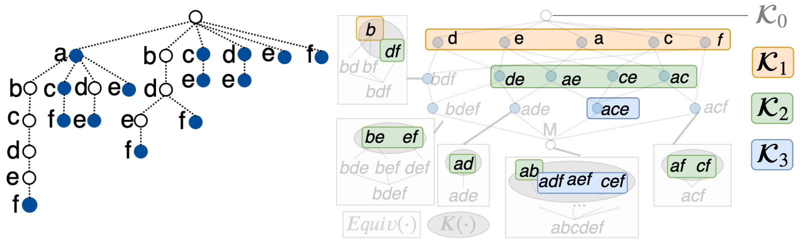

Fig. 1: Left: Lexicographic tree of a set of closed itemsets $\mathcal{C}_1^{min} \cup \mathcal{C}_2^{min} = \mathcal{C} \setminus \{ace\}$. A path from the root to a filled node corresponds to an itemset. Right: The closure structure is given by $\bigcup_{k=0,\ldots,3} \mathcal{K}_k$ (Section 3.1). Keys are given in bold.

3 Incremental Generation of Closed Itemsets

In this section we introduce the closure structure and discuss how Titanic and CbO can be used to discover this structure incrementally. Both algorithms exhibit the anytime property, meaning that they (i) compute the whole set of closed itemsets given enough time and space, (ii) can be interrupted at any time providing an estimated number of closed itemsets at the next (and following) levels. The algorithms differ in (i) key sets they use, (ii) the approaches to derivation of new keys based on previously computed ones, which results in different estimates of the output size for the next iteration and different computational complexity.

3.1 Level-wise Structure on Key Sets

Let $\mathcal{C}$ be the set of all closed itemsets and $K(B)$ be the key set of a closed itemset $B \in \mathcal{C}$. $Level : \mathcal{C} \to \{1, \ldots, |M|\}$ is a function that maps a closed itemset to the set of sizes of its keys, i.e., $Level(B) = \{|D| \mid D \in K(B)\}$. The structural level k is given by all the keys of size k, i.e., $\mathcal{K}_k = \{D \in K(B) \mid B \in \mathcal{C}, |D| = k\}$. We say that B belongs to the structural level k if there exists a key $D \in K(B)$ of size k, i.e., $B \in \mathcal{C}_k$ if $k \in Level(B)$. The set of the corresponding closed itemsets is given by $\mathcal{C}_k = \{B \mid B \in \mathcal{C}, k \in Level(B)\}$. The set of all structural levels will be called *closure structure*. We call *structural complexity* of $\mathcal{C}$ the maximal index of non-empty levels, $N_c = \max\{k \mid k = 1, \ldots, |M|, \mathcal{K}_k \neq \emptyset \}$.

Example. The levels of dataset from Fig. 1 are given in Fig. 1 (right). The structural complexity $N_c = 3$. Closed itemset *bdf* belongs to structural levels $\mathcal{C}_1$ and $\mathcal{C}_2$, because its equivalence class contains the keys from two structural levels, i.e., $b \in \mathcal{K}_1$ and $df \in \mathcal{K}_2$, while acf is included only in $\mathcal{C}_2$ since its keys af, cf are in $\mathcal{K}_2$.

Proposition 1 $\mathcal{I} = \bigcup_{k=1,\dots,|M|} \mathcal{K}_k$ *is an order ideal w.r.t.* $\subseteq$*.*

Proof. Consider an itemset $B \in \mathcal{I}$. Suppose the contrary, i.e., that an arbitrary subset $D \subset B$ of size $|B|-1$ is not in $\mathcal{I}$, thus, D is not a key. Then B can not be a key, since we can replace D with key $E \subset D$. Thus, our assumption is wrong and $\mathcal{I}$ is an order ideal.

Proposition 2 N_c *is the dimension of the maximal Boolean sublattice of the lattice of* $\mathcal{C}$*.*

It follows directly from the definition of N_c and Lemma 6 in [2].

Proposition 3 *Structural complexity* N_c *of* $\mathcal{C}$ *is the maximal* $n \in \{1, \dots, |M|\}$ *such that there exists a concept* (B', B) *with* $|B| = n$ *and* $Stab_{int}(B', B) = 1/2^n$*.*

Proof. By Proposition 1, there exists B, $|B| = N_c$ that corresponds to the intent of the bottom element of the Boolean sublattice. Since every subset $C \subset B'$ is an extent of a concept (C, C') such that $C' \neq B$, then $Stab_{\text{int}}(B', B) = 1/2^n$.

3.2 Generating closed itemsets using the closure structure.

Titanic algorithm [20] computes closed itemsets based on their keys. However, the authors do not consider this algorithm as incremental and anytime. Instead, in case of limited time or space, they propose its version for computing an iceberg lattice.

In this paper we state that, in case of limited resources, we can compute just several levels rather than considering only frequent itemsets. The pseudocode is given as Algorithm 1 (this version is a little bit different from the original one, since we put all the steps related to computing keys into one procedure).

To obtain the next set $\mathcal{K}_k$ it is sufficient to consider the union of two keys from $\mathcal{K}_{k-1}$ (line 1) and then check if each obtained key makes an order ideal with the previously computed keys (line 3-6, according to Proposition 1). However, itemsets from $\mathcal{S}$ may be just members of an equivalence class. To ensure that an itemset is a key, one needs to compare support $supp(D)$ of candidate $D \in \mathcal{S}$ with the minimal support $ind\_supp(D)$ of the keys of size $k-1$ contained in D (line 12). If these values are the same, then the support of D is inductively derived from the support of proper subsets, and D is not minimal (not a key).

Anytime property and the estimates of the size of the next output. Given the size of the current output $|\mathcal{K}_k|$ (the number of keys), the estimated size of the next output $|\mathcal{K}_{k+1}|$ is given by $|\mathcal{C}_{k+1}| \leq |\mathcal{K}_{k+1}| < |\mathcal{K}_k|(|\mathcal{K}_k| - 1)/2$. More generally, the number of keys at level $k+n$, $n \in \mathbb{N}$ is $O(|\mathcal{K}_k|^{2^n})$.

Titanic computes all keys of closed itemsets. A closed itemset can have keys of different size. The latter means that the same closed itemset will be computed at different levels (e.g., an itemset *bdf* from Fig. 1 has key $b \in \mathcal{K}_1$ and $df \in \mathcal{K}_2$). From a PM perspective, computing keys that generate the same closed itemsets is redundant.

Algorithm 1 Titanic-Gen

```
Require: $\mathcal{K}_{k-1}$, the key set of level $k-1$
Ensure: $\mathcal{K}_k$, the key set of level $k$
 1: $\mathcal{S} \leftarrow \{\{m_1 < \ldots < m_k\} \mid \{m_1, \ldots, m_{k-2}, m_{k-1}\}, \{m_1, \ldots, m_{k-2}, m_k\} \in \mathcal{K}_{k-1}\}$
 2: for all $D \in \mathcal{S}$ do
 3:     for all $(k-1)$-subsets $E \subseteq D$ do
 4:         if $E \notin \mathcal{K}_{k-1}$ then
 5:             $\mathcal{S} \leftarrow \mathcal{S} \setminus \{D\}$
 6:             exit forall
 7:         end if
 8:         $ind\_supp(D) \leftarrow \min(ind\_supp(D), supp(S))$
 9:     end for
10: end for
11: $ComputeSupport(\mathcal{S})$
12: $\mathcal{K}_k \leftarrow \{D \in \mathcal{S} \mid ind\_supp(D) \neq supp(D)\}$
13: return $\mathcal{K}_k$
```

Algorithm 2 CbO-Gen

```
Require: $\mathcal{K}^*_{k-1}$, a subset of the minimum key set $\mathcal{K}^{min}_{k-1}$ of level $k-1$,
    $\mathcal{T}_{k-1}$, the lexicographic tree containing all closed itemsets $\bigcup_{i=1,\ldots,k-1} \mathcal{C}^{min}_i$
Ensure: $\mathcal{K}^*_k$, a subset of the minimum key set of level $k$
 1: $\mathcal{K}^*_k \leftarrow \emptyset$
 2: $\mathcal{T}_k \leftarrow \mathcal{T}_{k-1}$
 3: for all $X \in \mathcal{K}^*_{k-1}$ do
 4:     $Y \leftarrow M \setminus X''$
 5:     for all $m \in Y$ do
 6:         $X^* = X \cup \{m\}$
 7:         $S \leftarrow (X^*)''$
 8:         if $S \notin \mathcal{T}_k$ then
 9:             $add(\mathcal{T}_k, S)$
10:             $\mathcal{K}^*_k \leftarrow \mathcal{K}^*_k \cup \{X^*\}$
11:         end if
12:     end for
13: end for
14: return $\mathcal{K}^*_k$
```

In the next section we propose an approach where only one minimum key for each closed itemset is considered.

3.3 Level-wise Structure on Minimum Key Sets

Similar to the closeness structure induced by key sets, we introduce the *minimum closeness structure* induced by minimum key sets.

Let $\mathcal{C}$ be a set of all closed itemsets and $K^{min}(B)$ be the minimum key set of a closed itemset $B \in \mathcal{C}$. We denote a function that maps a closed itemset to the size of its minimum key by *level*, i.e., $level : \mathcal{C} \to \{0, \ldots, |M|\}$, such that $level(B) = |D|$, where $D \in \mathcal{K}^{min}(B)$ is an arbitrary itemset chosen from $\mathcal{K}^{min}(B)$. The minimal structural level k is given by all minimum keys of size k, i.e., $\mathcal{K}^{min}_k = \bigcup_{\substack{B\in\mathcal{C},\\ level(B)=k}} K^{min}(B)$. We say that B belongs to the minimum structural level k if itemsets in $K^{min}(B)$ have size k. We denote the corresponding set of closed itemsets of level k by $\mathcal{C}^{min}_k$. More formally, $\mathcal{C}^{min}_k = \{B \mid B \in \mathcal{C}, level(B) = k\}$. We call *minimum structural complexity* of $\mathcal{C}$ the maximal number of not empty levels, $N^{min}_c = \max\{k \mid k = 1, \ldots, |M|, \mathcal{K}^{min}_k \neq \emptyset\}$.

Proposition 4 *$\mathcal{I}^{min} = \bigcup_{k=1,\ldots,|M|} \mathcal{K}_k^{min}$ is an order ideal.*

Proof. The proof is similar to the proof of Proposition 1.

Example. The structural levels $\mathcal{K}_k$ of the dataset from Fig. 1 are given in Fig. 1 (right). The minimum structural levels are the following: $\mathcal{K}_1^{min} = \mathcal{K}_1$, $\mathcal{K}_2^{min} = \mathcal{K} \setminus \{df\}$, and $\mathcal{K}_3^{min} = \{ace\}$. Set *df* is excluded because the size of *b* is smaller than *df*. The same for $\{adf, aef, cef\} \subseteq \mathcal{K}_3$. The minimum structural complexity is $N_c^{min} = 3$.

In contrast to $\mathcal{C}_k$ (induced by the structural level $\mathcal{K}_k$) $\mathcal{C}_k^{min}$ (induced by the minimum structural level $\mathcal{K}_k^{min}$) does not intersect with sets of concepts from other levels, e.g., *bdf* is in $\mathcal{C}_1 = \mathcal{C}_1^{min}$ and in $\mathcal{C}_2$, but not in $\mathcal{C}_2^{min}$.

3.4 Computing closed itemsets based on minimum closure structure.

A closed itemset may have more than one munimum key. In this section we propose to compute closed itemsets based on a subset $\mathcal{K}_k^*$ of $\mathcal{K}_k^{min}$.

The algorithm sequentially generates subsets $\mathcal{K}_k^* \subseteq \mathcal{K}_k^{min}$, $k = 1, \ldots, |M|$ such that for each $B \in \mathcal{C}_k^{min}$ there exists only one $D \in \mathcal{K}_k^*$. The pseudocode is given as Algorithm 2. By construction, $\mathcal{I}^* = \bigcup_{k=1,\ldots,|M|} \mathcal{K}_k^*$ is an order ideal.

An itemset from $\mathcal{K}_k^* \subseteq \mathcal{K}_k^{min}$ is computed as the closure of a union of an itemset $X \in \mathcal{K}_{k-1}^*$ and an attribute $m \in M \setminus X''$. The minimality of keys can be easily checked by using lexicographic tree $\mathcal{T}_{k-1}$ that contains the closed itemsets of the minimum keys computed earlier. The tree storage takes $O(2^{|}M|)$ memory. Checking the minimality takes $O(|M|)$ time.

The proposed algorithms is a version of Close-By-One (CbO) [14]. The call trees (records of an execution) of both algorithms are the same. It means that a closed itemset is computed based on the same minimum key (provided that the attributes in M are considered in the same order). They differ in the test for uniqueness of closed itemsets, namely the canonicity test (in CbO) and lexicographic tree (in the proposed version).

Example. Let us consider how Algorithm 2 works using the dataset from Fig. 1. The intermediate steps are given in Table 2. At input we get $\mathcal{K}_2^*$, a subset of minimum keys of size 2. For each closed itemset $\mathcal{C}_2^{min}$ it contains only one minimum key. The lexicographic tree for $\mathcal{C}_1^{min} \cup \mathcal{C}_2^{min}$ is given in Fig. 1.

Table 2: Intermediate steps of Algorithm 2 for the minimum structural level 2.

$X \in \mathcal{K}_2^*$	ab	ac	ad	ae	af	be	ce	de
X''	$abcdef$	ac	ade	ae	acf	$bdef$	ce	de
$Y = M \setminus X''$	$\emptyset$	$bdef$	bcf	$bcdf$	bde	ac	$abdf$	$abcf$
added to $\mathcal{K}_3^*$		ace						

Anytime property and the estimates of the size of the next output. Given the size $|\mathcal{K}_k^*|$ of the output at the current iteration, the number of keys $|\mathcal{K}_{k+1}^*|$ that will be generated at the next iteration is given by $\sum_{X \in \mathcal{K}_k^*} |M \setminus X''|$. More generally, the number of minimum keys at level $k+n$, $n \in \mathbb{N}$ is $O(|\mathcal{K}_k^*||M|^n)$.

4 Why Minimum Keys are Better?

In the previous section we introduced the structural levels $\mathcal{K}_k$ (used in Algorithm 1) and the miminum structural levels $\mathcal{K}_k^{min}$. The corresponding sets of closed itemsets are $\mathcal{C}_k$ and $\mathcal{C}_k^{min}$, respectively. We also consider subsets $\mathcal{K}_k^* \subseteq \mathcal{K}_k^{min}$ used in Algorithm 2.

At level k, both algorithms allow for discovering the same closed itemsets, i.e., $\bigcup_{i=1,\dots,k} \mathcal{C}_k^{min} = \bigcup_{i=1,\dots,k} \mathcal{C}_k$. However, $\mathcal{C}_k$ may intersect with another $\mathcal{C}_n$, while all $\mathcal{C}_k^{min}$ are disjoint. The latter means that in the second case closed itemsets will not be discovered twice. Redundant computations may take place not only w.r.t. different levels, but also within a (minimum) structural level, when a closed itemset has several keys of the same size. In Algorithm 2 this problem is solved by reducing $\mathcal{K}_k^{\min}$ to $\mathcal{K}_k^*$. Depending on data and the depth k of the closure structure, the performance of algorithms can differ drastically (see experiments in Section 5, Table 4). We illustrate this problem by means of a concrete example.

Example. Let us compare the structures that are used by the algorithms. The closure structure $\bigcup_{k=0,\dots,3} \mathcal{K}_k$ used by Titanic is given in Fig. 2. Four filled areas highlight the keys of four closed itemset. It means that Titanic discovers the same closed itemset several times (equal to the number of keys in the filled areas). The number of minimum keys (in bold) is much lower, while the number of minimum keys discovered by CbO (in dark bold) is even smaller. For instance, among the minimum keys $\mathcal{K}^{min}(bdef) = \{be, ef\}$ and $\mathcal{K}^{min}(acf) = \{af, cf\}$, CbO uses only be and af, respectively.

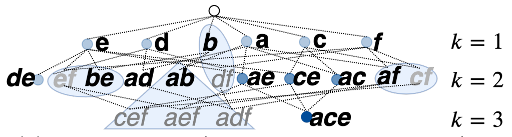

Fig. 2: Key sets $\bigcup_{k=1,\dots,|M|} \mathcal{K}_k$ (the closure structure), minimum key sets $\bigcup_{k=1,\dots,|M|} \mathcal{K}_k^{min}$ (the minimum closure structure, in bold), the subsets of minimum key sets used by Algorithm 2 (in dark bold). The keys corresponding to the same closed itemset are united by a filled area.

Computational redundancy can affect the estimates of the size of the next output. We will study how precise these estimates are in experiments (Section 5).

In addition to computational benefits, a minimum key is the best descriptor of an equivalence class according to the information theory [7]. Based on the reasoning from [16] it follows directly that the minimum keys provide the shortest description according to the Minimum Description Length principle [11].

Table 3: The characteristics of datasets and their closure structure. The cell color corresponds to the coverage ratio* of concepts in $\mathcal{C}_k^{min}$.

	$\|G\|$	$\|M\|$	$\|\mathcal{C}_k^{min}\|/\|\mathcal{C}\|$ (in percentage) k : **1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	N_c^{min}	$\|\mathcal{C}\|$
auto	205	129	0.2	4.7	21.7	**36.9**	26.4	8.7	1.3	0.1			8	57 788
breast	699	14	3.3	13.9	31.0	**34.6**	15.2	1.9					6	361
car ev.	1 728	21	0.3	2.3	10.6	27.5	**37.8**	21.6					6	7 999
ecoli	327	24	5.6	32.5	**43.5**	15.8	2.4	0.2					6	425
glass	214	40	1.2	13.9	**37.5**	32.1	11.9	3.0	0.4	0.0			8	3 245
heart	303	45	0.2	2.2	10.6	25.0	**31.2**	21.1	8.0	1.6	0.1		9	25 538
hepatitis	155	50	0.0	0.5	3.0	10.1	21.6	**29.0**	22.8	10.2	2.5	0.3	10	144 856
iris	150	32	0.7	5.9	17.7	28.5	**29.8**	15.2	2.2				7	4 481
led7	3 200	28	0.7	4.3	14.4	28.7	**32.3**	16.5	3.1				7	1 950
pima	768	36	2.2	17.5	**28.7**	27.8	16.7	5.9	1.2	0.1			8	1 625
soybean	683	99	0.0	0.1	1.4	6.9	18.0	**27.6**	25.6	14.4	5.0	1.0	10	2 870 926
tic tac toe	958	27	0.1	0.8	5.3	22.6	**39.8**	24.9	6.6				7	42 711
wine	178	65	0.5	9.7	36.1	**38.0**	13.5	2.0	0.1				7	13 228
zoo	101	35	0.8	8.3	26.1	**36.2**	23.2	5.2	0.2				7	4 569

*■ $CR = 1$, ▨ $0.8 \leq CR < 1.0$, ▢ $0.6 \leq CR < 0.8$, □ $CR < 0.6$.

5 Experiments

In this section we report the results of an experimental study of the minimum closure structure, i.e., closed itemsets within levels $\mathcal{C}_k^{min}$. We use freely available datasets from the LUCS/KDD data set repository [6] (see Table 3).

In our experiments we focus on the following questions: (i) what is the structure induced by the minimum keys, (ii) what concepts does level $\mathcal{C}_k^{min}$ ($k = 1, \ldots, |M|$) contain, (iii) what is the quality of concepts in $\mathcal{C}_k^{min}$?

5.1 Minimum Closure Structure

Considering Algorithms 1 and 2, the structural complexity (N_c and N_c^{min}) is equal to the number of iterations required to compute the whole lattice. Theoretically, for a dataset over M attributes, $N_c^{min} \leq N_c \leq |M|$, and the largest level $\frac{|M|}{2}$ has size $\binom{|M|}{\lfloor \frac{|M|}{2} \rfloor}$. In experiments we study how the theoretical bounds differ from the actual values. The results are reported in Table 3. The percentage of closed itemsets at level k w.r.t. the total number $|\mathcal{C}|$ is given in column "$|\mathcal{C}_k^{min}|/|\mathcal{C}|$", the largest level is highlighted in bold. Our experiments show that the theoretical bounds are overestimated 6 times (on average), i.e., $N_c^{min} \approx |M|/6$. The number of closed itemsets in the largest level on average is about 35% of the total number of closed itemsets. Thus, around 30% of closed itemsets can be computed before the largest level has been achieved. As expected, the largest levels are usually in the middle of the closure structure, i.e., $N_c^{min}/2$.

5.2 Characteristics of Structural Levels

In the previous section we study the overall closure structure. In this section we consider the composition of each level in more detail.

Coverage ratio. Referring to the set cover problem, we need to reduce the whole set of itemsets to a subset of candidates $\mathcal{F}$ (see Section 1). Algorithms 1 and 2 suggest to compute the candidate sets incrementally, which raises the question "when the computation can be stopped?" In our experiments we compute the level-wise coverage ratio of closed itemsets (formal concepts), i.e., the ratio of cells covered by closed itemsets $\mathcal{C}_k^{min}$ to all non-empty entries. The background colors of cells in Table 3 show the coverage ratio for each structural level. The coverage ratio monotonically decreases with levels. For example, for "iris" dataset, closed itemsets from $\mathcal{C}_1^{min}$ alone covers the whole dataset and $\mathcal{C}_2^{min}$ alone is able to cover the whole dataset. Each level from $\mathcal{C}_3^{min}$ to $\mathcal{C}_6^{min}$ covers more than 80% of dataset. The itemsets from the last level $\mathcal{C}_7^{min}$ cover more than 60%, but less than 80%. Thus, the coverage ratio does not increase with each consecutive level.

It follows directly from the definitions of the (minimum) structural levels that it is enough to compute at most 2 levels in order to compute the complete covering (if we use for covering itemsets of length at least 2).

Level composition. Along with the coverage ratio, we study the composition of the levels, i.e., the distribution of closed itemsets w.r.t. their frequency. The typical structures are given in Figure 3 (left and middle). For example, for "iris" dataset, frequent closed itemsets ($fr. \in (0.8, 1.0]$) make 20% of $\mathcal{C}_1^{min}$, 2.3% of $\mathcal{C}_2^{min}$, and only 0.1% of $\mathcal{C}_3^{min}$. The ratio of infrequent closed itemsets ($fr. \in (0., 0.2]$), on the contrary, increases, e.g., they make 20% of $\mathcal{C}_1^{min}$, 46% of $\mathcal{C}_3^{min}$, and 77% of the last level $\mathcal{C}_7^{min}$.

The described behavior is typical for all datasets. The only difference is the initial ratio of infrequent closed itemsets. For instance, $\mathcal{C}_1^{min}$ of "iris" and "heart-disease" contain 20% and 53% of infrequent closed itemsets, respectively. Only closed itemsets of "pima" and "ecoli" have level-wise frequency distribution that differs from the typical one. We show level-wise frequency of "pima" dataset in Figure 3, right. Further we will show that level-wise quality of "pima" and "ecoli" itemsets differs from others.

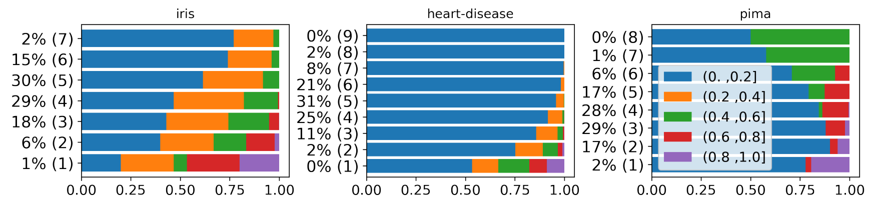

Fig. 3: Distribution of frequent concepts by levels. A level $\mathcal{C}_k^{min}$ is represented by a horizontal bar. The level number k is given in parentheses, the percentage of closed itemsets $|\mathcal{C}_k^{min}|/|\mathcal{C}_k|$ is on the left of the level number. The proportion of closed itemsets of frequency $(v_1, v_2]$ among $\mathcal{C}_k^{min}$ is highlighted in colors.

In those experiments we showed that the empirical structural complexity is much lower than its theoretical upper bound, i.e., $N_c^{min} \approx |M|/6 < |M|$, and closed itemsets of the first two levels contain a large number of frequent itemsets. This conclusion raises the following questions: (i) how useful are the closed itemsets of the first levels, (ii) do we need to compute the following levels to discover more interesting itemsets (or itemsets of better quality)?

Quality of itemsets. In our study we propose to assess itemset quality by average F_1 score by levels. All the studied datasets have class labels that are used only for evaluation. F_1 score is defined as follows: $F_1 = 2 \cdot prec \cdot recall/(prec + recall)$.

For a closed itemset B, objects from B' are considered as classified. The label of the majority of objects B' is taken as the label of B. The remaining objects $G \setminus B'$ are considered as unclassified by B. Then, the average values of F_1 score for concepts $\mathcal{C}_k^{min}$ are considered (see Fig. 4). In general, the quality of closed itemsets decreases with each subsequent level, except for "pima" and "ecoli" datasets. The reason of this unusual behavior needs a more careful study.

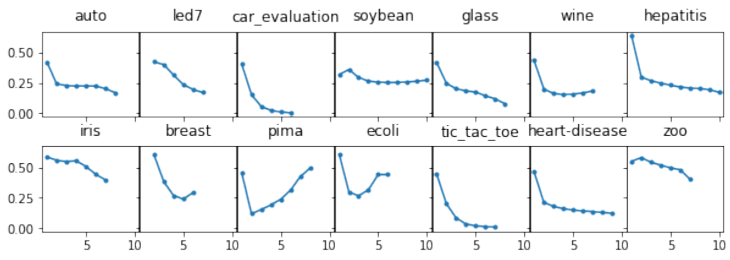

Fig. 4: Average value of F_1 by levels.

5.3 Anytime property. Precision of the boundaries.

Both algorithms provide boundaries on the number of keys that will be generated at the next iteration $k + 1$. For Algorithms 1 and 2 these values are $|\mathcal{K}_k|(|\mathcal{K}_k| - 1)/2$ and $\sum_{X \in \mathcal{K}_k^*} |M \setminus X''|$. We compare these values with the true number of newly generated closed itemsets $|\mathcal{C}_k^{min}| = |\mathcal{C}_k \setminus \cup_{i=1,\dots,k-1} \mathcal{C}_i|$. The ratio of the predicted size to the actual one is reported in Table 4. The results show that, except for levels 1 and 2, the estimates of Algorithm 2 provide more precise boundaries. Since the boundaries are closely related to the computational strategy, high values indicate the computational redundancy of Algorithm 1.

Table 4: The median values of the ratio of predicted size $|\mathcal{C}_k^{min}|$ to the actual one.

k	1	2	3	4	5	6	7
Titanic-Gen	1.00	1.61	58.81	347.82	1348.19	5907.63	12058.46
CbO-Gen	1.00	3.21	8.45	17.16	32.16	56.21	123.94

6 Conclusion

In this paper we have studied the minimum closure structure of closed itemsets. This structure is induced by the smallest (minimum) generators. We have proposed an alternative point of view on the Titanic and Close-By-One algorithms, namely we have shown that they exhibit the anytime property and allow for incremental generation of closed itemsets w.r.t. the (minimum) closure structure. In the experiments we have shown that the algorithms generate first itemsets of the highest quality and then itemsets of decreasing quality. One of the most interesting directions of future work is to study the connection between structural complexity and stability.

References

1. Aggarwal, C.C., Han, J.: Frequent pattern mining. Springer (2014)
2. Albano, A., Chornomaz, B.: Why concept lattices are large: extremal theory for generators, concepts, and VC-dimension. Int. Jour. of Gen. Syst. **46**(5) (2017)
3. Bastide, Y., Taouil, R., Pasquier, N., Stumme, G., Lakhal, L.: Mining frequent patterns with counting inference. ACM SIGKDD Explorations Newsletter **2**(2), 66–75 (2000)
4. Buzmakov, A., Kuznetsov, S.O., Napoli, A.: Fast generation of best interval patterns for nonmonotonic constraints. In: ECML PKDD. pp. 157–172 (2015)
5. Buzmakov, A., Kuznetsov, S.O., Napoli, A.: Revisiting pattern structure projections. In: ICFCA. pp. 200–215. Springer (2015)
6. Coenen, F.: The LUCS-KDD discretised/normalised ARM and CARM Data Library. http://www.csc.liv.ac.uk/ frans/KDD/Software/LUCS_KDD_DN,
7. Cover, T.M., Thomas, J.A.: Elements of inf. theory. John Wiley & Sons (2012)
8. Ganter, B., Wille, R.: Formal concept analysis–mathematical foundations (1999)
9. Ganter, B., Kuznetsov, S.O.: Pattern structures and their projections. In: ICCS. pp. 129–142. Springer (2001)
10. Goldberg, L.A.: Efficient algorithms for listing combinatorial structures, vol. 5. Cambridge University Press (2009)
11. Grünwald, P.: The minimum description length principle. MIT (2007)
12. Karp, R.M.: Reducibility among combinatorial problems. In: Complexity of computer computations, pp. 85–103. Springer (1972)
13. Kuznetsov, S.O.: Stability as an estimate of degree of substantiation of hypotheses derived on the basis of operational similarity. Nauchn. Tekh. Inf., Ser. 2 (12), 21–29 (1990)
14. 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)
15. Kuznetsov, S.O.: On stability of a formal concept. Annals of Mathematics and Artificial Intelligence **49**(1-4), 101–115 (2007)
16. Li, J., Li, H., Wong, L., Pei, J., Dong, G.: Minimum description length principle: Generators are preferable to closed patterns. In: AAAI. pp. 409–414 (2006)
17. Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Efficient mining of association rules using closed itemset lattices. Information systems **24**(1), 25–46 (1999)
18. Siebes, A., Kersten, R.: A structure function for transaction data. In: Proceedings of SIAM. pp. 558–569 (2011)
19. Singh, P.K., Kumar, C.A., Gani, A.: A comprehensive survey on formal concept analysis, its research trends and applications. International Journal of Applied Mathematics and Computer Science **26**(2), 495–516 (2016)
20. Stumme, G., Taouil, R., Bastide, Y., Pasquier, N., Lakhal, L.: Computing iceberg concept lattices with titanic. Data & knowledge engineering **42**(2), 189–222 (2002)

21. Vreeken, J., Van Leeuwen, M., Siebes, A.: Krimp: mining itemsets that compress. Data Mining and Knowledge Discovery **23**(1), 169–214 (2011)