

Extracting Frequent Gradual Patterns Based on SAT

Jerry Lonlac¹^a, Imen Ouled Dlala²^b, Said Jabbour³^c, Engelbert Mephu Nguifo⁴^d,
Badran Raddaoui⁵^e and Lakhdar Sais³^f

¹Centre de Recherche, IMT Lille Douai, Université Lille, Lens, France

²Pôle Universitaire Léonard de Vinci, Research Center, Paris La Défense, France

³CRIL, University of Artois & CNRS, Lens, France

⁴Univ. Clermont Auvergne, CNRS, LIMOS, F-63000 Clermont-Ferrand, France

⁵SAMOVAR, Télécom SudParis, Institut Polytechnique de Paris, France

Keywords: Data Mining, Gradual Patterns, Propositional Satisfiability.


Abstract: This paper proposes a constraint-based modeling approach for mining frequent gradual patterns from numerical data. Our declarative approach provides a principle way to take advantage of recent advancements in satisfiability testing and several features of modern SAT solvers to enumerating gradual patterns. Interestingly, our approach can easily be extended with extra requirements, such as temporal constraints used to extract more specific patterns in a broad range of gradual patterns mining applications. An empirical evaluation on two real-world datasets shows the efficiency of our approach.


1 INTRODUCTION


Nowadays, numerical data, also coined quantitative data, are abundant due to the proliferation of measuring and data collection devices and sensors. These data can be obtained from a range of practical areas, including environment & ecology, humanities & social science, trade & finance, and life sciences & astronomy. Typically, machine learning, statistical, and logical data analysis techniques are employed for analyzing these large amount of numerical data. Nevertheless, only a small number of pattern mining techniques, which typically studied categorical data, are designed to handle numerical data. Among these approaches, we can cite itemset/association rule mining (Ramakrishnan and Rakesh, 1996; Aumann and Lindell, 1999; Salleb-Aouissi et al., 2007), interval patterns using formal concept analysis (Kaytoute et al., 2011), rank-correlated sets of numerical attributes mining (Calders et al., 2006), and gradual pattern mining (Di-Jorio et al., 2009). In particular, gradual itemsets mining aims at discovering frequent co-variations between attributes, such as “*the*


higher the age, the higher the salary, and the lower the free time”. Gradual patterns are very expressive as they represent the variability of numerical values, often sought in numerous application domains, including for instance biology where most advances are done by analyzing genome data, medicine where researchers in psychology focus on correlations between memory and feeling points from the “Diagnostic manual of mental disorders”, and also in financial markets for discovering covariation between various economic and financial indicators. More generally, discovering gradual patterns is relevant in all numerical data where one needs to recover the relationship between attributes in terms of variability (e.g. (Ngo et al., 2018; Aryadinata et al., 2014; Fan and Xiao, 2017)).


Numerous proposals have been introduced to tackle the problem of gradual itemset mining or its variants, e.g., (Hüllermeier, 2002; Berzal et al., 2007; Masegaglia et al., 2008; Di-Jorio et al., 2008; Di-Jorio et al., 2009; Do et al., 2010; Laurent et al., 2010; Oudni et al., 2013; Négrevergne et al., 2014; Do et al., 2015; Lonlac et al., 2018). For most of these methods, an extracted gradual pattern is provided with a unique support or sub-sequence of transactions, called *extension*, supporting it. Such information when provided, allows us to explain to the user why such itemset is gradual. Recently, in (Ngo et al., 2018), the authors addressed the problem of mining spatial gradual pat-


^a <https://orcid.org/0000-0003-3278-9969>

^b <https://orcid.org/0000-0001-6928-4599>

^c <https://orcid.org/0000-0002-8389-8332>

^d <https://orcid.org/0000-0001-9119-678X>

^e <https://orcid.org/0000-0003-4712-0811>

^f <https://orcid.org/0000-0003-2879-8627>

terns with an application to the measurement of potentially avoidable hospitalizations. In this context, they have shown that the analysis of the different sequences of objects associated to the gradual patterns could reveal new relevant knowledge to the expert. Let us mention that the sequence of objects verifying a gradual pattern is not unique, to the best of our knowledge, all the mentioned approaches do not provide all the possible extensions of the gradual itemset. Providing an exhaustive set of extensions associated to a given gradual itemset could be beneficial and would improve the robustness and the precision of the information characterizing the pattern. Moreover, in (Di-Jorio et al., 2009), the authors presented a first effective algorithm for the discovery of gradual itemsets and gradual rules that can handle databases with hundreds of attributes (contrary to the algorithm introduced in (Berzal et al., 2007) which is limited to only six attributes).

One of the primary issues with mining gradual pattern approaches is the exponential combination space to be explored, coupled with the challenge of the huge number of patterns, which can also be of exponential size. This combinatorial explosion is tackled in (Laurent et al., 2009; Ayouni et al., 2010; Do et al., 2015). In fact, in (Laurent et al., 2009), the authors proposed a method for extracting gradual patterns from large datasets which takes advantage of a binary representation of lattice structure. In addition, other work (e.g., (Ayouni et al., 2010; Do et al., 2015)) retrieved only closed frequent gradual patterns in order to reduce the total number of extracted patterns. In this paper we propose a new approach for extracting all frequent closed gradual patterns in a numerical dataset. Our approach differs from all the previous specialized approaches. It follows the SAT-based framework proposed in (Jabbour et al., 2013) for mining frequent closed itemsets. This new framework offers a declarative and flexible representation model. In fact, specialized approaches often require fresh implementations to accommodate novel constraints, whereas such constraints can be easily integrated within a Boolean satisfiability framework. This enables data mining problems to take advantage of multiple generic and efficient SAT solving techniques. We propose in this paper to heavily exploit the declarative language (SAT) and these associated efficient and generic solving techniques. First, for that purpose, we present a new SAT-based model to find frequent gradual patterns that includes different types of constraints. Then, we provide a boolean formulation of the closeness constraints in order to search for frequent closed gradual patterns. Different experiments carried out over real datasets are pre-

sented to show the feasibility of the new SAT-based approach. The paper is organized as follows: in Section 2.1, we present the problem of mining gradual itemsets from numerical databases and some efficient algorithms that have been proposed in the literature. We also recall the problem of Boolean satisfiability. Section 4 describes the SAT-based enumeration procedure to deal with the problem of enumerating the set of all models of a CNF formula. In Section 3, we present our SAT encoding of frequent gradual itemset mining problem and show through an example how it can be applied to find frequent gradual itemsets from a numerical dataset. Finally, section 5 presents detailed experiments carried out over real datasets, showing the applicability and the interest of our approach.

2 PRELIMINARIES

In this section, we formally describe the problem of mining frequent gradual itemsets (or patterns) in numerical datasets as well as the propositional satisfiability problem.

2.1 Gradual Itemsets Mining Problem

The problem of mining gradual patterns consists in retrieving attribute co-variations in numerical dataset of the form "The more/less X, \dots , the more/less Y ". We assume herein that we are given a dataset Δ containing a set of objects \mathcal{T} defining a relation on an attribute set I with numerical values, i.e., for $t \in \mathcal{T}$ and $i \in I$, $i \downarrow t$ denotes the value of the attribute i over object t . Table 1 gives an example of a numerical dataset built over the set of attributes $I = \{age, salary, cars\}$.

Table 1: An example of a numerical dataset Δ .

tid	age	salary	cars
t_1	22	1200	2
t_2	28	1850	3
t_3	24	1200	4
t_4	35	2200	4
t_5	38	2000	1
t_6	44	3400	1
t_7	52	3400	3
t_8	41	5000	2

Each attribute will hereafter be considered twice: once to indicate its increase (\leq), and another to indicate its decrease (\geq). This leads to new kinds of items, called *gradual items*.

Definition 1. Let Δ be a dataset defined on a numerical attribute set I . A gradual item is defined under the

form i^o , where i is an attribute of I and $o \in \{\geq, \leq\}$ represents an ascending or descending order or variation of the attribute values of i .

If we consider the numerical dataset of Table 1, age^{\geq} (respectively age^{\leq}) is a gradual item expressing that the values of the attribute age are increasing (respectively decreasing). Now, a *gradual itemset* (or simply *gradual pattern*) is a non-empty set of gradual items. Also, such an itemset is called a k -gradual itemset if it contains exactly k gradual items. For example, $g_1 = \{age^{\geq}, salary^{\geq}\}$ is a 2-gradual itemset, meaning that "the higher the age, the higher is the salary". A gradual itemset sets a variation order on several attributes simultaneously. The length of a gradual itemset is equal to the number of gradual items that it contains.

The *support* (also called *frequency*) of a gradual itemset amounts to the extent to which a gradual pattern is present in a numerical database. Several support definitions have been proposed in the literature (Hüllermeier, 2002; Calders et al., 2006; Berzal et al., 2007; Laurent et al., 2009; Di-Jorio et al., 2009; Kendall and Smith, 1939), showing that gradual itemsets can follow different semantics. In (Hüllermeier, 2002) the computation of the support of gradual itemset is based on linear regression. In (Calders et al., 2006; Berzal et al., 2007; Kendall and Smith, 1939), the authors considered the proportion of couples of tuples that verifies the constraints expressed by all the gradual items of the itemset, while in (Di-Jorio et al., 2009), the support is defined as the size of the longest sequence of tuples supporting the gradual itemset. In this paper, we adopt this last definition of support for its relevance and generality. To introduce formally this variant of support, let us first introduce the following additional definitions:

Definition 2. Let $g = (i_1^{o_1}, \dots, i_k^{o_k})$ be a gradual itemset and $s = \langle t_1 \rightarrow t_2 \rightarrow \dots \rightarrow t_n \rangle$ a sequence of tuples. Then, s is an extension of g if $\forall 1 \leq p \leq k$ and $\forall 1 \leq j < n$, we have:

$$(i_p \downarrow t_j) o_p (i_p \downarrow t_{j+1}) \quad (1)$$

It is important to note that there might be several extensions or sequences of tuples validating g .

Definition 3. Let g be a gradual itemset in a numerical database Δ . We define $Cover(g, \Delta)$ as the set of the longest extensions of g in Δ w.r.t. set inclusion.

Example 1. Let us consider the database Δ depicted in Table 1 and the gradual itemset $g_1 = \{age^{\geq}, salary^{\geq}\}$. $Cover(g_1, \Delta) = \{\langle t_1 \rightarrow t_3 \rightarrow t_2 \rightarrow t_4 \rightarrow t_6 \rightarrow t_7 \rangle, \langle t_1 \rightarrow t_3 \rightarrow t_2 \rightarrow t_5 \rightarrow t_6 \rightarrow t_7 \rangle, \langle t_1 \rightarrow t_3 \rightarrow t_2 \rightarrow t_4 \rightarrow t_8 \rangle, \langle t_1 \rightarrow t_3 \rightarrow t_2 \rightarrow t_5 \rightarrow t_8 \rangle\}$. From the same example, the cover of the gradual item

$salary^{\geq}$ (resp. $cars^{\geq}$) is $\{\langle t_1 \rightarrow t_3 \rightarrow t_2 \rightarrow t_4 \rightarrow t_5 \rightarrow t_8 \rightarrow t_6 \rightarrow t_7 \rangle\}$ (resp. $\{\langle t_5 \rightarrow t_6 \rightarrow t_8 \rightarrow t_1 \rightarrow t_2 \rightarrow t_7 \rightarrow t_3 \rightarrow t_4 \rangle\}$).

Now, we are ready to give the definition of support.

Definition 4. Let Δ be a numerical database and g be a gradual itemset of Δ . Then,

$$Supp(g, \Delta) = \frac{\max\{|s|, s \in Cover(g, \Delta)\}}{|\Delta|}.$$

Example 2. Referring again to the database Δ of Table 1 and the gradual itemset g_1 , we have $Supp(g_1, \Delta) = \frac{6}{8}$. So, six among the eight input tuples can be ordered according to g_1 . Note that the support of a gradual item is equal to 100% as it is always possible to order all of the tuples according to the values of a single attribute.

A gradual itemset is said to be *frequent* if its support is greater than or equal to a user-defined support threshold.

Definition 5. Let Δ be a numerical database and λ a minimum support threshold. The problem of mining gradual itemsets is to find the set of all frequent gradual itemsets of Δ w.r.t. λ , i.e., finding the set $\{g \mid Supp(g, \Delta) \geq \lambda\}$.

Definition 6. Let $g = (i_1^{o_1}, \dots, i_k^{o_k})$ be a gradual itemset, and \bar{f} be a function such that $\bar{f}(\geq) = \leq$ and $\bar{f}(\leq) = \geq$. Then $\bar{f}(g) = (i_1^{\bar{f}(o_1)}, \dots, i_k^{\bar{f}(o_k)})$ is the complementary (symmetric) gradual itemset of g .

Interestingly, any gradual itemset admits a complementary gradual one where the items are the same but the variations are all reversed. For instance, the complementary gradual itemset of g_1 is $(age^{\leq}, salary^{\leq})$.

Proposition 1 ((Di-Jorio et al., 2009)). Let g be a gradual itemset of a numerical database Δ . We have $Supp(g, \Delta) = Supp(\bar{f}(g), \Delta)$.

Proposition 1 avoids unnecessary computation, as generating only a half of the set of the gradual itemsets is sufficient to automatically deduce the complementary ones. As far as we know, there is no existing algorithm to mining both the gradual itemsets and all their corresponding extensions. For each gradual itemset, all the state-of-the-art algorithms looks for an extension with the maximum size while there are application domains where the extensions of the gradual itemsets bring new knowledge to the user.

2.2 Boolean Satisfiability Problem

This section introduces the Boolean satisfiability problem, or simply SAT. It corresponds to the prob-

lem of deciding if a formula of propositional classical logic is consistent or not. It is one of the most studied NP-complete decision problem. In this work, we consider the associated problem of boolean model enumeration.

We consider the conjunctive normal form (CNF, for short) representation for the propositional formulas. A CNF formula \mathcal{F} is a conjunction of clauses, where a *clause* is a disjunction of literals. A *literal* is a positive (l) or negated ($\neg l$) propositional variable. The two literals (l) and ($\neg l$) are called complementary. We note by \bar{l} the complementary literal of l . For a set of literals L , \bar{L} is defined as $\{\bar{l} \mid l \in L\}$. Let us recall that any propositional formula can be translated to CNF using linear Tseitin's encoding (Tseitin, 1968). The set of variables occurring in \mathcal{F} is noted $Var(\mathcal{F})$.

An *interpretation* ρ of a boolean formula \mathcal{F} is a function which associates a value $\rho(l) \in \{0, 1\}$ (0 correspond to *false* and 1 to *true*) to the variables $x \in Var(\mathcal{F})$. A model of a formula is an *interpretation* ρ that satisfies the formula. SAT problem consists in deciding if a given formula admits a model or not.

3 SAT-BASED ENCODING FOR DISCOVERING FREQUENT GRADUAL PATTERNS

In this section, we show how the problem of mining all the frequent gradual itemset in a numerical dataset w.r.t. a minimum support threshold $minSupp$ can be encoded as a propositional formula. In order to formally describe our encoding, we consider a numerical dataset $\Delta = \mathcal{T} \times \mathcal{A}$ where $\mathcal{A} = \{a_1, \dots, a_n\}$ is a set of attributes, $\mathcal{T} = \{t_1, \dots, t_m\}$ a set of transactions, and k the minimum support threshold. To model the frequent gradual itemset mining task into SAT, we associate with each gradual item a^{\geq} (resp. a^{\leq}) a Boolean variable $x_{a^{\geq}}$ (resp. $x_{a^{\leq}}$), meaning that the gradual item is included in the gradual itemsets or not. Similarly, with each transaction t_i , we associate a set of Boolean variables t_{i1}, \dots, t_{ik} where t_{ij} means that the transaction t_i is set on the j -th position. For constraints modeling, our objective is to range a set of k transactions from m that highlights a gradual itemset. Otherwise, we have k positions that have to be assigned with transactions. Constraint (2) allows to not consider gradual itemset involving both a^{\geq} and a^{\leq} of each attribute a .

$$\bigwedge_{a \in \{a_1, \dots, a_n\}} (\neg x_{a^{\geq}} \vee \neg x_{a^{\leq}}) \quad (2)$$

This first constraint solves the problem encoun-

tered with the specialized algorithm of frequent gradual itemsets mining GLCM (Do et al., 2015) which often returns the gradual itemsets containing both the gradual items and their corresponding complementary gradual items.

The second constraint (3) allows us to indicate that a position $j \in \{1, \dots, k\}$ must be associated with one transaction.

$$\bigwedge_{1 \leq j \leq k} \left(\sum_{i=1}^n t_{ij} = 1 \right) \quad (3)$$

The Constraint (4) is introduced to not allow a transaction to be placed in more than one position among $\{1, \dots, k\}$.

$$\bigwedge_{1 \leq i \leq n} \left(\sum_{j=1}^k t_{ij} \leq 1 \right) \quad (4)$$

Note that the two constraints (3) and (4) encodes the well known pigeon-hole problem.

Constraint (5) aims to express, given a gradual item a^* , the set of transactions that can be set in position $j+1$ if transaction t_i is putted in position j .

$$\bigwedge_{a^* \in \mathcal{A}^*} \bigwedge_{1 \leq i \leq n} \bigwedge_{1 \leq j \leq k} (x_{a^*} \wedge t_{ij} \rightarrow \bigvee_{t_k(a) \diamond t_i(a)} t_{k(j+1)}) \quad (5)$$

Note that such constraint can be expressed differently by considering only those that are not allowed as stated in Constraint (6).

In contrast to (5), constraint (6) allows to add only ternary clauses. However, their number is higher than those of (5).

$$\bigwedge_{a^* \in \mathcal{A}^*} \bigwedge_{1 \leq i \leq n} \bigwedge_{1 \leq j \leq k} (x_{a^*} \wedge t_{ij} \rightarrow \bigwedge_{t_k(a) \bar{\diamond} t_i(a)} \neg t_{k(j+1)}) \quad (6)$$

Example 3. Let us consider the transaction database of Table 1. Assume that $k = 5$. If the gradual itemset contains the gradual item car^{\geq} , and if the support is as t_7 is set on position 1, then the corresponding constraints is as follows:

$$x_{car^{\geq}} \wedge t_{71} \rightarrow t_{22} \vee t_{32} \vee t_{42}$$

Proposition 2. There is a one-to-one mapping between the gradual itemsets and the models of the formula $\Phi_{k,n}^D = (2) \wedge (3) \wedge (4) \wedge (5) \wedge (6)$.

Proposition 2 links the gradual itemsets to the models of our encoding.

Finally, in order to eliminate symmetrical gradual itemsets, we add the following constraint:

$$\bigwedge_{a_i \in a_1 \dots a_n} (\neg x_{a_i^{\geq}} \vee \bigvee_{1 \leq j < i} \neg x_{a_i^{\leq}}) \quad (7)$$

In fact, the permutation $\sigma = (a_1^{\geq}, a_1^{\leq}) \dots (a_n^{\geq}, a_n^{\leq})$ is a symmetry of the proposed encoding. Consequently, one can break such symmetry by adding the Symmetry Breaking Predicates as defined in (Crawford et al., 1996). More precisely, in (Crawford et al., 1996), for a symmetry $\sigma = (x_1, y_1) \dots, (x_n, y_n)$ the author show that to break this symmetry one can add the following constraint

$$\bigwedge_{i=1}^n \bigwedge_{j=1}^{i-1} (x_j = y_j) \rightarrow (x_i \leq y_i)$$

Combining this constraint with the one of (2) leads to the simplified Constraint (7).

Note that the constraint (7) allows to avoid computing all gradual patterns and their corresponding symmetric gradual pattern. However, this constraint will add a certain number of variables and clauses to the final boolean formula. We propose another direction to take into account this symmetrical without add the constraint (7) but by adding two blocking clauses in the NCF formula each time a model is found. One clause to avoid finding the same model and another to avoid finding a model corresponding to the symmetric pattern.

Note that $(\sum_{i=1}^n y_{ij} = 1)$ (respectively $(\sum_{j=1}^k y_{ij} \leq 1)$) represent linear equality (respectively inequality) commonly called exact-One (respectively atMostOne Constraint). Such constraint can be encoding in respectively $O(n)$ (respectively $O(k)$) clauses using $O(n)$ (respectively $O(k)$) additional variables as indicated in constraint (8) (Warners, 1998; Silva and Lynce, 2007). A possible encoding of $\sum_{i=1}^n x_i = 1$ is as follows using auxiliary variables $\{p_1, \dots, p_{n-1}\}$.

$$\left(\bigvee_{1 \leq i \leq n} x_i \right) \wedge (\neg x_1 \vee p_1) \wedge (\neg x_n \vee \neg p_{n-1}) \wedge \bigwedge_{1 < i < n} (\neg x_i \vee p_i) \wedge (\neg p_{i-1} \vee p_i) \wedge (\neg x_i \vee \neg p_{i-1}) \quad (8)$$

From complexity point of view, let us note that our encoding introduces $O(k \times n \times m)$ clauses. In fact, Constraint 2 is on $O(n)$. Constraint 3 and 4 leads to $O(n \times m)$. For Constraint 5 requires $O(k \times n \times m)$. Finally, Constraint 4 requires $O(n)$. So to summarize, the encoding is in $O(n + k \times m + k \times n \times m) = O(k \times n \times m)$. For the introduced variables. Let us mention that this number is in $O(k \times m)$. In fact, in addition to $x_{a \geq}$, $x_{a \leq}$ and t_{ij} , new variables must be added to encoded cardinality constraints of Constraints 3 and 4. This number remains bounded by $O(k \times m)$.

As mentioned encoding gradual itemsets mining into propositional satisfiability allows to have a more flexible approach where new constraints can be added to mine particular patterns. Typically, in many application fields, interesting gradual patterns can be

distinguished from irrelevant ones by specifying semantic constraints on the gradual pattern itself. For example, the authors of (Lonlac et al., 2017) designed an algorithm to mine temporal gradual patterns which are gradual patterns whose the longest sequence of transactions respect the temporal order. These kinds of gradual patterns are particularly interesting in the paleoecological domain where the experts search from their paleoecological numerical data the patterns which capture the simultaneously frequent co-evolutions between attributes. As the transactions are encoded in our CNF formula as Boolean variables, the temporal constraint can be captured by selecting in the temporal order the propositional variables t_{ij} representing the transaction identifiers of the numerical dataset.

4 SAT-BASED ENUMERATION PROCEDURE

In this section, we describe the SAT-Based enumeration procedure to deal with the problem of enumerating all models of the CNF formula $\Phi_{k,n}^D$. SAT is a decision problem. When the answer is positive, the current SAT solvers provide a model satisfying the formula. In the sequel, we briefly describe the basic components of modern SAT solvers, also called CDCL SAT solvers (Moskewicz et al., 2001; En and Sörensson, 2003) designed to enumerate all the models of a given CNF formula. To be exhaustive, these solvers incorporate unit propagation (enhanced by efficient and lazy data structures), variable activity-based heuristic, literal polarity phase, clause learning, restarts and a learnt clauses database reduction policy.

Algorithm 1 depicts the general scheme of CDCL SAT solver extended for model enumeration. A SAT solver is a tree-based backtrack search procedure; at each node of the search tree, the assigned literals (decision literal and the propagated ones) are labeled with the same decision level starting from 1 and increased at each decision (or branching).

Typically, this solver performs a tree-based backtrack search procedure. Each branch of the binary search tree can be seen as a sequence of decision and unit propagated literals. At each node, a decision variable is chosen (ligne 23), and assigned to the *true* or *false* polarity (*selectPhase(l)* - ligne 25). Then unit propagation is performed in line 6. All these literals (decision and propagated ones) assigned at a given node are labelled with the same level dl . If all literals are assigned without contadiction, then ρ is a model of \mathcal{F} and the formula is answered to be satisfiable (line 16). As our boolean formula represents

Algorithm 1: CDCL Based Enumeration solver.

```

Input: a CNF formula  $\Phi$ 
Output: All models of  $\Phi$ 
1  $\rho = \emptyset$ ; /* interpretation */
2  $\delta = \emptyset$ ; /* learnt clauses database */
3  $dl = 0$ ; /* decision level */
4 while (true) do
5   Prop;
6    $\gamma = \text{unitPropagation}(\Phi, I)$ ;
7   if  $\gamma \neq \text{null}$  then
8      $\beta = \text{conflictAnalysis}(\Phi, I, \gamma)$ ;
9      $btl = \text{computeBackjumpLevel}(\beta, I)$ ;
10    if  $btl == 0$  then
11      return UNSAT;
12       $\delta = \delta \cup \{\beta\}$ ;
13      if restart() then
14         $btl = 0$ ;
15        backjump( $btl$ );
16         $dl = btl$ ;
17    else
18      if  $\rho \models \Phi$  then
19        extractPatternFromModel( $\rho$ );
20        addBlockedClause( $\rho$ );
21        backjumpUntil(0);
22        goto Prop;
23      if (timeToReduce()) then
24        reduceDB( $\delta$ );
25         $l = \text{selectDecisionVariable}(\Phi)$ ;
26         $dl = dl + 1$ ;
27         $\rho = \rho \cup \{\text{selectPhase}(l)\}$ ;

```

the encoding of the closed frequent gradual itemset mining problem, each time a model is found, an gradual itemset is extracted from ρ (line 17). For model enumeration, the search continue by adding a blocked clause to avoid enumerating again the same models (line 18). Search restart at level 0, to search for the next models (lines 19-20). The other case, is reached when unit propagation (lines 8-14) leads to a conflict (γ is the conflict clause), a new asserting clause β is derived by conflict analysis (line 8), mostly following the First-UIP scheme ('Unique Implication Point' (Zhang et al., 2001)) A backtrack level (btl) is derived from the asserting clause (line 9). If btl is null, then the formula is answered unsatisfiable (line 10), otherwise β is added to the learnt clauses database (line 11) and the algorithm backjump to the level btl (line 13). Regularly, the CDCL solver performs restarts, by backtracking to level 0 (line 12) using one of the various restart strategies ((Huang,)). Such restarts define the frequency used by the solver to restart the search. Finally, another component concern the learnt clauses management policy. To maintain a learnt clauses database of reasonable size, a reduction is performed

(line 22) using one the various strategies proposed in the literature (Audemard and Simon, 2009; Eén and Sörensson, 2003; Lonlac and Mephu Nguifo, 2017; Jabbour et al., 2014).

5 EXPERIMENTS

In this section, we carried out an experimental evaluation of the performance of our proposed approach. we ran experiments both on the real-world paleoecological datasets and on the synthetic datasets. The paleoecological dataset are constituted of a set of numerical attributes whose the values correspond to the quantity of each paleoecological indicator contained in a sediment record taken, by coring operations, in a lake ecosystem. It contains 111 objects corresponding to different dates identified on the considered Lacustrine recording, and 117 attributes corresponding to paleoecological indicators. All the experiments were done on Intel Xeon quad-core machines with 32GB of RAM running at 2.66 Ghz.

To solve the obtained formulas, we use the solver *MiniSAT2.2* (Eén and Sörensson, 2003) adapted for model enumeration since as proposed enumerating all the models that satisfy the CNF formula which encodes the frequent gradual pattern mining problem is equivalent to enumerating all frequent gradual patterns.

The main procedure of our approach is given in algorithm 2. This procedure compute and return all frequent gradual patterns with respect to the minimum support threshold $minSupp$. The procedure *findAllModel* corresponds to the algorithm 1 modified by adding to the CNF formula two blocking clauses instead of one blocking clause at each time that a model is found during the resolution process. One blocking clause to avoid finding the same model and another one to avoid finding a model corresponding to the symmetric pattern of the extracted gradual pattern.

Algorithm 2: SAT Based Gradual Patterns Enumeration.

```

Input: a numerical database  $DS$ , a
          minimum support  $minSupp$ 
Output: Set of all frequent gradual patterns
1  $\mathcal{F} \leftarrow \text{SATEncoding}(DS, minSupp)$ ;
2 findAllModel( $\mathcal{F}$ );

```

Table 2 presents results obtained on the paleoecological dataset. It yields the size of the CNF formula (number of variables and clauses) encoding the gradual patterns mining problem with respect to a minimum support. In this table, we mention the formula encoding the whole problem in terms of number

of variables ($\#vars$) and clauses ($\#clauses$) with respect to a minimum support threshold ($\#minSupp$). The last column gives in seconds the cpu time need for encoding. The first observation that we can draw from Table 2 is that our SAT-based approach generates huge CNF formulas in short time. For instance, for a minimum support equal to 50%, our SAT encoding generates in 2.25 seconds, a CNF formula with 18383 variables and 1438734 clauses. It is also worth mentioning that the number of variables of the CNF formula increases when the minimum support increases (see Table 2) and the number of clauses strongly increases.

Table 2: CNF encoding characteristics by varying the minimum support threshold.

#minSupp	#vars	#clauses	#encodingTime (in seconds)
5%	2 115	133 521	0.22
10%	3 775	266 706	0.43
20%	7 427	559 713	0.86
30%	11 079	852 720	1.31
40%	14 731	1 145 727	1.74
50%	18 383	1 438 734	2.25
60%	22 035	1 731 741	2.69
70%	25 687	2 024 748	3.12
80%	29 339	2 317 755	3.54
90%	32 991	2 610 762	4.03

Table 3 compares (run-times, in seconds) on a synthetic dataset of 10 items and 100 transactions our proposed SAT-based approach, which we coined SAT4GIM to GRITE solver (Di-Jorio et al., 2009) and , the efficient specialized algorithm for extracting frequent gradual itemsets from numerical databases when varying the minimum support threshold. We generate the synthetic dataset using an adapted version of IBM Synthetic Data Generation Code for Associations and Sequential Patterns¹. We also compare our SAT-based approach to the one proposed in (Hidouri et al., 2021) called SATGIM. The results from Table 3 show that, for the small minimum support thresholds, our SAT-based approach is faster than the efficient specialized algorithm for extracting frequent gradual itemsets from numerical databases GRITE (Di-Jorio et al., 2009) and SATGIM. On the other hand, our proposal takes longer than other approaches to enumerate the complete set of gradual patterns when the minimum support threshold is high. It is worth mentioning that SAT4GIM makes it possible to know for each frequent gradual pattern, the position of each transaction belonging to its extension. That is not the case for GRITE and SATGIM.

¹www.almaden.ibm.com/software/projects/hdb/resources.shtml

Table 3: SAT4GIM vs (GRITE, SATGIM) for various minimum support values.

#minSupp	GRITE	SATGIM	SAT4GIM	#Gradual
0.02	3.82	1.51	0.61	59 001
0.03	3.71	5	0.91	38 923
0.04	3.51	7.4	1.40	14 507
0.05	3.45	8.32	1.90	5 741
0.1	3.29	7.38	4.71	411
0.15	3.09	6.98	7.2	201
0.2	2.62	6.70	10.35	75
0.3	2.59	2.04	16.37	33
0.4	2.58	0.80	22.04	27
0.5	2.50	0.17	28.19	21

6 CONCLUSION

In this paper, we proposed SAT encoding to address the problem of mining frequent gradual patterns. This declarative approach offers an additional possibility to benefit from the recent progress in satisfiability testing and to enumerate each gradual pattern with the sequence of objects supporting it. We also performed experiments with real-world and synthetic datasets to show the efficiency of our proposal w.r.t. state-of-the-art algorithms for mining gradual itemsets. Future directions can be pursued to address various challenges. First, we intend to develop a SAT-based encoding to enumerate maximal frequent gradual patterns which remains an open challenging and impactful problem in gradual pattern mining. We also plan to perform more experiments on large datasets.

REFERENCES

- Aryadinata, Y. S., Lin, Y., Barcellos, C., Laurent, A., and Libourel, T. (2014). Mining epidemiological dengue fever data from brazil: A gradual pattern based geographical information system. In *IMPU*, pages 414–423.
- Audemard, G. and Simon, L. (2009). Predicting learnt clauses quality in modern sat solvers. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence, IJCAI'09*, pages 399–404.
- Aumann, Y. and Lindell, Y. (1999). A statistical theory for quantitative association rules. In *SIGKDD*, pages 261–270.
- Ayouni, S., Laurent, A., Yahia, S. B., and Poncelet, P. (2010). Mining closed gradual patterns. In *Artificial Intelligence and Soft Computing, 10th International Conference, ICAISC 2010, Zakopane, Poland, June 13-17, 2010, Part I*, pages 267–274.
- Berzal, F., Cubero, J. C., Sánchez, D., Miranda, M. A. V., and Serrano, J. (2007). An alternative approach to discover gradual dependencies. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 15(5):559–570.

- Calders, T., Goethals, B., and Jaroszewicz, S. (2006). Mining rank-correlated sets of numerical attributes. In *KDD*, pages 96–105.
- Crawford, J., Ginsberg, M. L., Luck, E., and Roy, A. (1996). Symmetry-breaking predicates for search problems. In *Principles of Knowledge Representation and Reasoning (KR'96)*, pages 148–159.
- Di-Jorio, L., Laurent, A., and Teisseire, M. (2008). Fast extraction of gradual association rules: a heuristic based method. In *CSTST 2008: Proceedings of the 5th International Conference on Soft Computing as Transdisciplinary Science and Technology, Cergy-Pontoise, France, October 28-31, 2008*, pages 205–210.
- Di-Jorio, L., Laurent, A., and Teisseire, M. (2009). Mining frequent gradual itemsets from large databases. In *Advances in Intelligent Data Analysis VIII, 8th International Symposium on Intelligent Data Analysis, IDA 2009, Lyon, France, August 31 - September 2, 2009. Proceedings*, pages 297–308.
- Do, T. D. T., Laurent, A., and Termier, A. (2010). PGLCM: efficient parallel mining of closed frequent gradual itemsets. In *ICDM*, pages 138–147.
- Do, T. D. T., Termier, A., Laurent, A., Négrevergne, B., Tehrani, B. O., and Amer-Yahia, S. (2015). PGLCM: efficient parallel mining of closed frequent gradual itemsets. *Knowl. Inf. Syst.*, 43(3):497–527.
- Eén, N. and Sörensson, N. (2003). An extensible sat-solver. pages 502–518.
- En, N. and Sörensson, N. (2003). An extensible SAT-solver. pages 502–518.
- Fan, C. and Xiao, F. (2017). Mining gradual patterns in big building operational data for building energy efficiency enhancement. *Energy Procedia*, 143:119 – 124. Leveraging Energy Technologies and Policy Options for Low Carbon Cities.
- Hidouri, A., Jabbour, S., Raddaoui, B., and Yaghlane, B. B. (2021). Mining closed high utility itemsets based on propositional satisfiability. *Data Knowl. Eng.*, 136:101927.
- Huang, J. The effect of restarts on the efficiency of clause learning. pages 2318–2323.
- Hüllermeier, E. (2002). Association rules for expressing gradual dependencies. In *Principles of Data Mining and Knowledge Discovery, 6th European Conference, PKDD 2002, Helsinki, Finland, August 19-23, 2002, Proceedings*, pages 200–211.
- Jabbour, S., Lonlac, J., Sais, L., and Salhi, Y. (2014). Revisiting the learned clauses database reduction strategies. *CoRR*, abs/1402.1956.
- Jabbour, S., Sais, L., and Salhi, Y. (2013). The top-k frequent closed itemset mining using top-k SAT problem. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27*, pages 403–418.
- Kaytoue, M., Kuznetsov, S. O., and Napoli, A. (2011). Revisiting numerical pattern mining with formal concept analysis. In *IJCAI*, pages 1342–1347.
- Kendall, M. and Smith, B. (1939). The problem of m rankings. In *The annals of mathematical statistics - Volume 10*, pages 275–287.
- Laurent, A., Lesot, M., and Rifqi, M. (2009). GRAANK: exploiting rank correlations for extracting gradual itemsets. In *Flexible Query Answering Systems, 8th International Conference, FQAS 2009, Roskilde, Denmark, October 26-28, 2009. Proceedings*, pages 382–393.
- Laurent, A., Négrevergne, B., Sicard, N., and Termier, A. (2010). Pgp-mc: Towards a multicore parallel approach for mining gradual patterns. In *DASFAA, Part I*, pages 78–84.
- Lonlac, J. and Mephu Nguifo, E. (2017). Towards learned clauses database reduction strategies based on dominance relationship. *CoRR*, abs/1705.10898.
- Lonlac, J., Miras, Y., Beauger, A., Mazenod, V., Peiry, J.-L., and Mephu, E. (2018). An approach for extracting frequent (closed) gradual patterns under temporal constraint. In *FUZZ-IEEE*, pages 878–885.
- Lonlac, J., Miras, Y., Beauger, A., Pailloux, M., Peiry, J.-L., and Nguifo, E. M. (2017). Une approche d'extraction de motifs graduels (fermés) fréquents sous contrainte de la temporalité. *Revue des Nouvelles Technologies de l'Information, Extraction et Gestion des Connaissances*, RNTI-E-33:213–224.
- Masseglia, F., Laurent, A., and Teisseire, M. (2008). Gradual trends in fuzzy sequential patterns. In *In IPMU*, pages 456–463.
- Moskewicz, M. W., Madigan, C. F., Zhao, Y., Zhang, L., and Malik, S. (2001). Chaff: Engineering an efficient SAT solver. In *Proceedings of the 38th Design Automation Conference (DAC'01)*, pages 530–535.
- Négrevergne, B., Termier, A., Rousset, M., and Méhaut, J. (2014). Para miner: a generic pattern mining algorithm for multi-core architectures. *DMKD*, 28(3):593–633.
- Ngo, T., Georgescu, V., Laurent, A., Libourel, T., and Mercier, G. (2018). Mining spatial gradual patterns: Application to measurement of potentially avoidable hospitalizations. In *SOFSEM*, pages 596–608.
- Oudni, A., Lesot, M., and Rifqi, M. (2013). Processing contradiction in gradual itemset extraction. In *FUZZ-IEEE*, pages 1–8.
- Ramakrishnan, S. and Rakesh, A. (1996). Mining quantitative association rules in large relational tables. *SIGMOD Rec.*, 25(2):1–12.
- Salleb-Aouissi, A., Vrain, C., and Nortet, C. (2007). Quantminer: A genetic algorithm for mining quantitative association rules. In *IJCAI*, pages 1035–1040.
- Silva, J. P. M. and Lynce, I. (2007). Towards robust cnf encodings of cardinality constraints. In *CP*, pages 483–497.
- Tseitin, G. (1968). On the complexity of derivations in the propositional calculus. In Slesenko, H., editor, *Structures in Constructives Mathematics and Mathematical Logic, Part II*, pages 115–125.
- Warners, J. P. (1998). A linear-time transformation of linear inequalities into conjunctive normal form. *Information Processing Letters*, 68(2):63 – 69.
- Zhang, L., Madigan, C. F., Moskewicz, M. W., and Malik, S. (2001). Efficient conflict driven learning in Boolean satisfiability solver. In *IEEE/ACM CAD'2001*, pages 279–285.