Targeting Minimal Rare Itemsets from Transaction Databases

Amel Hidouri[1] , **Badran Raddaoui**[2,3] , **Said Jabbour**[1]
[1] CRIL & CNRS, Universite d'Artois, Lens, France
[2] SAMOVAR, Télécom SudParis, Institut Polytechnique de Paris, France
[3]Institute for Philosophy II, Ruhr University Bochum, Germany
{hidouri, jabbour}@cril.fr, badran.raddaoui@telecom-sudparis.eu

Abstract

The computation of minimal rare itemsets is a well-known task in data mining, with numerous applications, e.g., drugs effects analysis and network security, among others. This paper presents a novel approach to the computation of minimal rare itemsets. First, we introduce a generalization of the traditional minimal rare itemset model called *k-minimal rare itemset*. A k-minimal rare itemset is defined as an itemset that becomes frequent or rare based on the removal of at least k or at most $(k-1)$ items from it. We claim that our work is the first to propose this generalization in the field of data mining. We then present a SAT-based framework for efficiently discovering k-minimal rare itemsets from large transaction databases. Afterwards, by partitioning the k-minimal rare itemset mining problem into smaller sub-problems, we aim to make it more manageable and easier to solve. Finally, to evaluate the effectiveness and efficiency of our approach, we conduct extensive experimental analysis using various popular datasets. We compare our method with existing specialized algorithms and CP-based algorithms commonly used for this task.

1 Introduction

Pattern extraction is a core topic in data mining. It aims to automatically infer knowledge from data based on various kinds of interesting measures. This involves the identification of relevant patterns embedded in data. These patterns represent the properties of the data satisfying users' preferences. Frequent Itemset Mining (FIM, for short) is a typical task of data analysis area that aims to find objects that frequently occur together among data. The first original motivation for FIM was found in market basket analysis, which involves exploring transactions in retails to discover frequently co-occurring products.

Earlier work has often focused on mining frequent patterns (see [Fournier-Viger *et al.*, 2017] for a detailed survey). However, in some cases certain items have a much lower chance to frequently appear than others. Hence, patterns appearing rarely, i.e., itemsets having a low frequency in the database, are overlooked since they are considered uninteresting and eliminated using the frequency constraint. In contrast, it has been shown recently that in many real-life applications, targeting infrequent itemsets, called *rare itemsets*, may be more informative than discovering frequently occurring patterns [Darrab *et al.*, 2021]. For instance, in the medical field, infrequent patterns may be more appealing because their discovery would aid in the avoidance of negative consequences and the formulation of healthcare decisions. During the Covid-19 pandemic, one might be interested for example in analyzing clinical data to discover unusual combinations of disease symptoms and risk factors that have not previously been identified. These patterns could be used as surrogate indicators to identify patients with Covid-19 and predict their clinical outcomes, so that appropriate treatments can be provided. Another example of using rare itemsets is that in the banking field where finding irregular behaviors could be helpful for identifying potentially fraudulent transactions.

The problem of mining rare itemsets from transaction databases is quite challenging since they are practically much more numerous than frequent ones. To overcome this issue, a condensed representation of rare itemsets, coined *minimal* rare itemsets, has been considered. Minimal rare itemsets are of interest since they represent a *border* below which all subsets are frequent itemsets [Mannila and Toivonen, 1997]. Some algorithms have been proposed for computing minimal rare itemsets in transaction databases (e.g., [Szathmary *et al.*, 2007; Szathmary *et al.*, 2012; Belaid *et al.*, 2019]).

In this paper, we are primarily interested in generalizing the concept of minimal rare itemset, and then computing these motifs in large transaction databases. In fact, the frequency of some itemsets cannot exceed the minimum frequency threshold due to the presence of some items. As a result, these itemsets are regarded as rare. In this case, it may be more interesting to focus on finding this kind of itemsets while relaxing the fequency constraint over a fixed number of items. This representation will be referred to as *k-minimal rare itemsets*. In more detail, a k-minimal rare itemset X is an itemset that becomes frequent (resp. rare) when removing at least (resp. most) k (resp. $(k-1)$) items from it. Clearly, a 1-minimal rare itemset is a minimal rare itemset. Interestingly, such generalization could aid in the discovery of what may be relevant in data in real applications. In fact, k-minimal rare itemsets can be used to identify errors in certain processes. These item-

sets, for example, aid in the discovery of the causes of abnormal effects of drugs taken by patients in the medical field. Moreover, such patterns can assist a retailer in identifying irrelevant purchased products, allowing her/him to provide the appropriate product combinations. A retail store manager can use this knowledge to assess the risk of selling a product by removing it from the market.

Our Contributions. In this paper, we first present a generalization of the minimal rare itemset model, which we call k-minimal rare itemset. Note that our framework is general enough to encompass minimal rare itemsets as a specific case. Then, we provide a symbolic Artificial Intelligence (AI) approach to discovering k-minimal rare itemsets from transaction databases. In detail, we present a SAT-based framework for translating the task of finding k-minimal rare itemsets into a propositional satisfiability problem that can be solved using a SAT solver. To tackle the scalability issue, we provide afterwards a splitting-based approach enabling to partition the database into subsets with lower size that can be independently mined without jeopardizing completeness. Finally, we conduct extensive experiments on different popular real-world datasets to evaluate the efficiency of our approach w.r.t. the state-of-the-art methods.

2 General Setting

2.1 Propositional Satisfiability

Let $\mathcal{L}$ be a propositional language built up inductively from a countable set $\mathcal{P}$ of propositional variables, the Boolean constants $\top$ (*true* or 1) and $\bot$ (*false* or 0) and the classical logical connectives $\{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$ in the usual way. To range over the elements of $\mathcal{P}$, we use the letters x, y, z, and so on. A **literal** is a propositional variable (x) of $\mathcal{P}$ or its negation ($\neg x$). A **clause** is a (finite) disjunction of literals. Propositional formulas of $\mathcal{L}$ are denoted by Φ, Ψ, etc. For any formula Φ from $\mathcal{L}$, $\mathcal{P}(\Phi)$ denotes the symbols of $\mathcal{P}$ occurring in Φ. A formula in **conjunctive normal form** (or simply CNF) is a finite conjunction of clauses. A Boolean interpretation $\mathcal{I}$ of a CNF formula Φ is defined as a function from $\mathcal{P}(\Phi)$ to $\{0, 1\}$. A **model** of Φ is an interpretation $\mathcal{I}$ that satisfies Φ (denoted $\mathcal{I} \models \Phi$), that is, if there exists an interpretation $\mathcal{I} : \mathcal{P}(\Phi) \rightarrow \{0, 1\}$ that satisfies all clauses in Φ. The formula Φ is **satisfiable** if it has at least one model. We write $\mathcal{M}(\Phi)$ as shorthand for the set of models of a formula Φ.

The propositional satisfiability problem (in short, SAT) is the decision problem for propositional logic. Given a CNF formula Φ, SAT determines whether there exists a model of Φ. SAT solvers have been used in a variety of real-world scenarios, e.g., electronic design automation, software and hardware verification [Morgado and Marques-Silva, 2005], etc. Note that the aforementioned applications require the enumeration of all possible models of the corresponding CNF formula. This task, known as **model enumeration**, is of great importance. For instance, in diagnosis, the user is interested in finding all possible explanations rather than just whether one exists. Model enumeration problems find applications in a variety of tasks, including network verification [Zhang *et al.*, 2012], model checking [Hu and Martin, 2004] as well as several data mining tasks, e.g., [Dlala *et al.*, 2018; Jabbour *et al.*, 2018; Boudane *et al.*, 2016; Izza *et al.*, 2020; Hidouri *et al.*, 2021b; Hidouri *et al.*, 2021a; Hidouri *et al.*, 2022], and graph analysis [Jabbour *et al.*, 2016; Jabbour *et al.*, 2017; Jabbour *et al.*, 2022]. In the last decade, several SAT-based proposals for modeling and solving pattern mining problems have indeed studied. Because they rely on generic and declarative solvers, these approaches are suitable for modeling a variety of mining tasks.

2.2 Overview of the Rare Itemsets

Let Ω be a universe of items (or symbols) that may represent articles in a supermarket, web pages, or a collection of attributes or events. We denote the elements of Ω by the letters a, b, c, etc. An **itemset** (or simply a *motif*) is a subset of items in Ω, that is, $X \subseteq \Omega$. The set of all itemsets over Ω, denoted as 2^{Ω}, are represented by the capital letters X, Y, Z, etc. A **transaction database** is a finite non-empty set of transactions or records $\mathcal{D} = \{T_1, T_2, \ldots, T_m\}$. We denote by $\mathcal{D}_a$ the sub-base containing the set of transactions where a appears. Given a transaction database $\mathcal{D}$ and an itemset X, the **cover** of X in $\mathcal{D}$ is a mapping $2^X \rightarrow 2^{\mathcal{D}}$ which maps each itemset X to a set of transactions in $\mathcal{D}$ containing X. More formally, $\texttt{Cover}(X, \mathcal{D}) = \{i \in [1..m] \mid T_i \in \mathcal{D} \text{ and } X \subseteq T_i\}$. The cardinality of the cover of the itemset X represents its **support** (also called *frequency*). We write $\texttt{Supp}(X, \mathcal{D})$ for the support of X in the dataset $\mathcal{D}$, i.e., $\texttt{Supp}(X, \mathcal{D}) = |\texttt{Cover}(X, \mathcal{D})|$. When there is no confusion, the cover and support will be simply denoted as $\texttt{Cover}(X)$ and $\texttt{Supp}(X)$, respectively. We also write $X <_{\text{freq}} Y$ iff $\texttt{Supp}(X) \leq \texttt{Supp}(Y)$, for two itemsets X and Y. Next, we introduce some useful notions.

Definition 1 *Let $\mathcal{D}$ be a transaction database, X an itemset and $\lambda \geq 1$ a minimum support threshold. Then, X is a:*

- **frequent itemset** *in $\mathcal{D}$ iff* $\texttt{Supp}(X) \geq \lambda$.
- **rare itemset** *in $\mathcal{D}$ iff* $\texttt{Supp}(X) < \lambda$.
- **minimal rare itemset** *in $\mathcal{D}$ iff (i) X is rare, and (ii) $\forall\, Y \subset X$, Y is frequent.*

Let us write MRIs to be the set of minimal rare itemsets in $\mathcal{D}$. Obviously, every rare itemset is an infrequent itemset in $\mathcal{D}$, and any super-set of a MRI is also a rare itemset.

Definition 2 *Given a transaction database $\mathcal{D}$, an itemset X is called a* **minimal (key) generator** *in $\mathcal{D}$ iff $\forall\ Y \subset X, \texttt{Supp}(X) < \texttt{Supp}(Y)$.*

Given a database $\mathcal{D}$, it has been shown in [Szathmary *et al.*, 2012] that all minimal rare itemsets are generators in $\mathcal{D}$.

Example 1 *Let us consider the transaction database given in Table 1. Note that each transaction contains some products purchased in a grocery store. The minimum support threshold λ is set to 3.*

Figure 1 depicts the search space of (in)frequent itemsets (we refer to each product by a letter, e.g., a for apple, b for bread, etc.). The lattice is basically divided into two zones according to λ. The top zone represents frequent itemsets while the bottom one consists of rare itemsets. A line separates these two zones which are complementary subsets of the powerset $2^{|\Omega|}$. In this paper, we are interested in the second zone, i.e., the set of all rare itemsets in $\mathcal{D}$.

TID	Items				
1	*apple*	*bread*	*cheese*	*diapers*	
2	*apple*	*bread*	*cheese*		
3	*apple*	*bread*	*cheese*		*eggs*
4	*apple*	*bread*		*diapers*	*eggs*
5	*apple*	*bread*		*diapers*	
6					*eggs*

Table 1: A sample transaction database $\mathcal{D}$

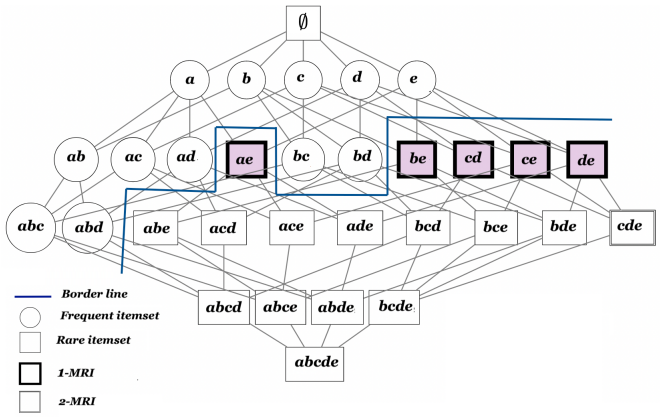

Figure 1: The powerset lattice of items $\{a, b, c, d, e\}$ in $\mathcal{D}$.

As we have seen, a minimal rare itemset is a rare itemset that by removing one item from it becomes frequent. Next, we generalize such condition to consider the deletion of one or more items. Following that, we present our formalization of this generalization as follows.

Definition 3 *Let $\mathcal{D}$ be a transaction database, λ a minimum support threshold, and k a positive integer. Then, an itemset X is called a k-***minimal rare itemset** *(k-MRI, in short) if (i) $\forall\, Y \subset X$ s.t. $|Y| \leq k-1$, $X \setminus Y$ is rare, and (ii) $\forall Y \subset X$ s.t. $|Y| \geq k$, $X \setminus Y$ is frequent.*

Intuitively, Definition 3 extends the minimal rare itemset model by requiring that the removal of $(k-1)$ items from the itemset does not make it frequent, but the deletion of at least k items does. It is also worthy to say that a k-minimal rare itemset is of size at least k. To our knowledge, this is the first generalization of minimal rare itemsets in data mining. Interestingly, our model encompasses the minimal rare itemsets as a specific case by setting k to 1.

Example 2 *Consider the transactions in Table 1 and $\lambda = 3$. Then, $X = \{c, d, e\}$ is a 2-*MRI*. In fact, $\{c\}, \{d\}, \{e\}$ are frequent while $\{c, d, e\}$, $\{c, d\}$, $\{c, e\}$, $\{d, e\}$ are not.*

Proposition 1 *Let $\mathcal{D}$ be a transaction database and $k > 1$. If X is a k-*MRI *in $\mathcal{D}$, then for all $x \in X$, $X \setminus \{x\}$ is a $(k-1)$-*MRI *in $\mathcal{D}$.*

Proposition 2 *Given a transaction database $\mathcal{D}$ and a minimum support threshold λ, if X is a 1-*MRI *in $\mathcal{D}$ with $X = Y \uplus Z$, then Z is a 1-*MRI *in $\{T_i \in \mathcal{D} \mid Y \subseteq T_i\}$.*

3 SAT-based Encoding for k-MRIs Mining

This section presents our SAT-based encoding for the k-minimal rare itemset enumeration problem. Given a transaction database $\mathcal{D}$, our key idea is to encode the task of k-MRIs computation into a propositional formula whose models correspond exactly to the set of k-minimal rare itemsets of $\mathcal{D}$. In this view, the problem of mining such motifs is translated to the one of computing all models of a propositional formula.

We start by providing a SAT encoding for the enumeration of 1-MRIs before proposing a generalization for the k-MRIs mining problem. For that purpose, to establish a one-to-one mapping between the models of our SAT-based encoding and the set of 1-MRIs in the database $\mathcal{D}$, each item $a \in \Omega$ is associated with a propositional variable x_a where x_a is *true* iff a belongs to the candidate rare itemset, while each transaction T_i in $\mathcal{D}$ is associated with two propositional variables p_i and q_i where p_i (resp. q_i) is *true* iff the transaction T_i contains the candidate rare itemset (resp. the candidate rare itemset except one item).

Now, in order to establish a mapping between the set of 1-MRIs in the database $\mathcal{D}$ and the models of the corresponding propositional formula, we introduce the following logical constraints.

Cover constraint. The first constraint allows to capture the cover of the itemset and it can be expressed as follows:

$$\bigwedge_{T_i \in \mathcal{D}} (p_i \leftrightarrow (\bigwedge_{a \in \Omega \setminus T_i} \neg x_a)) \tag{1}$$

Constraint (1) means that the itemset appears in the transaction T_i if all items that do not occur in that itemset are set to *false*.

Infrequency constraint. The second constraint ensures that the candidate itemset is not frequent in $\mathcal{D}$.

$$\sum_{T_i \in \mathcal{D}} p_i < \lambda \tag{2}$$

Given a minimum support threshold λ, Constraints (1) and (2) allow to find the set of all rare itemsets in $\mathcal{D}$.

Now, in order to find a 1-MRI, one needs to identify the set of transactions matching X in $\mathcal{D}$ as well as those that match $X \setminus \{a\}$ for all $a \in X$. This can be expressed by the following constraint.

Generator constraint. This constraint allows to catch the set of transactions in the database $\mathcal{D}$ involving the whole itemset except one item.

$$\bigwedge_{T_i \in \mathcal{D}} (q_i \leftrightarrow (\sum_{a \in \Omega \setminus T_i} x_a = 1)) \tag{3}$$

In other words, enforcing a rare itemset X to be minimal is equivalent to enforcing the itemset $X \setminus \{a\}$ ($\forall a \in X$) to be frequent. This requirement can be modeled with the next constraint.

Frequency constraint. This constraint requires that each itemset $X \setminus \{a\}$ ($\forall a \in X$) must be frequent in $\mathcal{D}$.

$$\bigwedge_{a \in \Omega} (x_a \rightarrow (\sum_{\substack{T_i \in \mathcal{D} \\ a \in T_i}} p_i + \sum_{\substack{T_i \in \mathcal{D} \\ a \notin T_i}} q_i \geq \lambda)) \tag{4}$$

Example 3 *Consider again Table 1 and let $\lambda = 3$. Then, the cover, infrequency, generator and frequency constraints can be written as given in Figure 2.*

Property 1 *Each model of the formula $\Phi_{\mathcal{D}}^{\lambda} = (1) \wedge (2) \wedge (3) \wedge (4)$ corresponds to a* 1-MRI *in the database $\mathcal{D}$.*

The next result shows that the formula $\Phi_{\mathcal{D}}^{\lambda}$ entails the following set of clauses.

Proposition 3 *For a given transaction database $\mathcal{D}$, we have $\Phi_{\mathcal{D}}^{\lambda} \models \Theta$:*

$$\Theta = \bigwedge_{a \in \Omega} (\neg x_a \vee \bigvee_{\substack{T_i \in \mathcal{D} \\ a \notin T_i}} q_i) \tag{5}$$

It is worth noting that Constraint (5) can be useful for unit propagation. In fact, this constraint enforces the rare itemset to be a generator. Indeed, it expresses that there exists a transaction in which $X \setminus \{a\}$ appears for each $a \in X$. As this property holds for every $a \in X$, then X is a generator.

Interestingly, the previous encoding can be generalized to deal with the problem of mining the set of all k-MRIs from a given transaction database. To do so, let us present the following three logical constraints.

$$\bigwedge_{j=0}^{k} \bigwedge_{T_i \in \mathcal{D}} (p_{i,j} \leftrightarrow (\sum_{a \in \Omega \setminus T_i} x_a = j)) \tag{6}$$

$$\bigwedge_{\substack{X \subseteq \Omega \\ |X|=k-1}} (\bigwedge_{a \in X} x_a \rightarrow (\sum_{j=0}^{k-1} (\sum_{\substack{T_i \in \mathcal{D} \\ |(\Omega \setminus T_i) \cap X|=j}} p_{i,j} < \lambda))) \tag{7}$$

$$\bigwedge_{\substack{X \subseteq \Omega \\ |X|=k}} (\bigwedge_{a \in X} x_a \rightarrow (\sum_{j=0}^{k} (\sum_{\substack{T_i \in \mathcal{D} \\ |(\Omega \setminus T_i) \cap X|=j}} p_{i,j} \geq \lambda))) \tag{8}$$

Let us now go over these constraints in greater depth. To compute the set of all k-MRIs in $\mathcal{D}$, we need first to identify the transactions involving the whole itemset (i.e., $p_{i,0}$), those containing the itemset except one item (i.e., $p_{i,1}$), until those with k missing items (i.e., $p_{i,k}$). Constraint (6) enables the identification of such a set of transactions. Next, Constraint (7) requires the deletion of $(k-1)$ items from the candidate itemset in order for it to be rare. This requires an additional constraint for each subset of items from Ω of size $(k-1)$. Note that in this case the number of constraints can be exponential w.r.t. k. Lastly, to allow the removal of k items, we need to identify the transactions containing partially the itemset (i.e., the candidate itemset without k items). This condition can be modelled with Constraint (8). Notice that for $k = 1$, the SAT encoding of 1-MRIs corresponds to $\Phi_{\mathcal{D}}^{\lambda}$ by substituting $p_{i,0}$ (resp. $p_{i,1}$) by p_i (resp. q_i). Also, it is worthy to note that Constraint (6) is a cardinality constraint of the form $\sum_{i=1}^{n} x_i = k$. Such constraint can be rewritten as follows:

$$\bigwedge_{j=0}^{k} \bigwedge_{T_i \in \mathcal{D}} (p_{i,j} \leftrightarrow (\bigwedge_{0 \leq l < j} \neg p_{i,l} \wedge \sum_{a \in \Omega \setminus T_i} x_a \leq j)) \tag{9}$$

To close this section, we note that the three constraints (6), (7) and (8) clearly induce a large number of propositional variables that grow rapidly with the values of k.

Proposition 4 *Given a transaction database $\mathcal{D}$, a minimum support threshold λ, and a positive integer $k > 0$, then the propositional formula $\Phi_{\mathcal{D}}^{k,\lambda}$ = (6) $\wedge$ (7) $\wedge$ (8) encodes the problem of mining k-MRIs in $\mathcal{D}$.*

4 Towards a More Efficient Computation of Rare Itemsets

The practical SAT-based encoding of k-MRIs mining problem, defined in Section 3, is polynomial in k, $|\Omega|$, and $|\mathcal{D}|$. In particular, the number of propositional variables and clauses is bounded by $O(|\Omega| \times |\mathcal{D}|^2 + |\Omega|^2 \times |\mathcal{D}|)$ for 1-MRIs. Unfortunately, even if such complexity is polynomially bounded, it becomes challenging and time-consuming for large datasets. This is due especially to the cardinality constraint (4) (for 1-MRIs) that involves all the transactions of $\mathcal{D}$. Hence, the size of the whole encoding may be huge. To obviate this scalability issue, we utilize a decomposition scheme for better efficiency by avoiding the generation of large CNF formulas. We provide next an illustration through the 1-MRIs case, which can be generalized to k-MRIs ($k > 1$).

In more details, the aim is to partition the initial mining problem into less complex and more manageable set of subproblems. The resolution task is then expected to be easier than that for the original problem. In other words, the search space is divided into disjoint parts by the use of the Shannon's principle [Zaki, 1999] (also referred to as *guiding path* [Zhang *et al.*, 1996]). In fact, the guiding path is a set of formulas added to the original formula in order to split the search space. More formally, let Φ be a propositional formula and $\{\Gamma_1, \ldots, \Gamma_n\}$ be a set of sub-formulas over $\mathcal{P}(\Phi)$ s.t. $\Gamma_1 \vee \ldots \vee \Gamma_n \equiv \top$ and $\Gamma_i \wedge \Gamma_j \models \bot, \forall\ i \neq j$. Then, the satisfiability of the formula Φ is related to the satisfiability of at least $\Phi \wedge \Gamma_{1 \leq i \leq n}$. Then, we have that $\mathcal{M}(\Phi) = \biguplus_{i=1}^{n} \mathcal{M}(\Phi \wedge \Gamma_i)$.

In [Jabbour *et al.*, 2020], the authors defined Γ_i as $\Gamma_i = \Phi \wedge x_{a_i} \wedge \bigwedge_{1 \leq j < i} \neg x_{a_j}$, allowing to enforce the enumeration of only frequent itemsets containing the item a_i. This is meant to avoid the encoding of the whole database by considering only the database $\mathcal{D}_{a_i}$ composed of transactions of $\mathcal{D}$ involving a_i. Note that all items $a_{j<i}$ are simply removed from $\mathcal{D}_{a_i}$. Nevertheless, for the k-MRIs computation problem, in addition to the base $\mathcal{D}_{a_i}$, the remaining transactions in $\mathcal{D}$ should also be considered. This will naturally increase our previous SAT-encoding size. To overcome this issue, we propose to apply the decomposition principle twice. The main idea is to recursively perform the decomposition over the items of D_{a_i} by considering the following set of formulas of the form: $\Gamma_{i,j} = x_{a_i} \wedge x_{a_j} \wedge \bigwedge_{l<j,\ l \neq i} \neg x_{a_l}$ if $i < j$. Intuitively, $\Gamma_{1 \leq i \leq n}$ is extended to $\Gamma_{i,j}$ by performing over Γ_j at each branch. More precisely, once an item a_i is chosen, we iterate through the set of transactions involving a_i (i.e., $\mathcal{D}_{a_i}$). The transaction database $\mathcal{D}$ can then be partitioned into two subbases $\mathcal{D}_{a_i} \cup \mathcal{D}_{a_j}$ and $\mathcal{D} \setminus (\mathcal{D}_{a_i} \cup \mathcal{D}_{a_j})$. The sub-table

Cover constraint
$p_1 \leftrightarrow (\neg x_e)$
$p_2 \leftrightarrow (\neg x_d \wedge \neg x_e)$
$p_3 \leftrightarrow (\neg x_b)$
$p_4 \leftrightarrow (\neg x_c)$
$p_5 \leftrightarrow (\neg x_c \wedge \neg x_e)$
$p_6 \leftrightarrow (\neg x_a \wedge \neg x_b \wedge \neg x_c \wedge \neg x_d)$
Infrequency constraint
$p_1 + p_2 + p_3 + p_4 + p_5 + p_6 < 3$

Generator constraint
$q_1 \leftrightarrow x_e = 1$
$q_2 \leftrightarrow x_d + x_d = 1$
$q_3 \leftrightarrow x_d = 1$
$q_4 \leftrightarrow x_c = 1$
$q_5 \leftrightarrow x_c + x_e = 1$
$q_6 \leftrightarrow x_a + x_b + x_c + x_d = 1$

Frequency constraint
$x_a \rightarrow (p_1 + p_2 + p_3 + p_4 + p_5 + q_6 \geq 3)$
$x_b \rightarrow (p_1 + p_2 + p_3 + p_4 + p_5 + q_6 \geq 3)$
$x_c \rightarrow (p_1 + p_2 + p_3 + q_4 + q_5 + q_6 \geq 3)$
$x_d \rightarrow (p_1 + p_4 + p_5 + q_2 + q_3 + q_6 \geq 3)$
$x_e \rightarrow (p_3 + p_4 + p_6 + q_1 + q_2 + q_5 \geq 3)$

Figure 2: SAT-based encoding for the transaction database shown in Table 1.

$\mathcal{D}_{a_i} \cup \mathcal{D}_{a_j}$ contains either the item a_i or a_j. Then, for each transaction $T_l \in \mathcal{D} \setminus (\mathcal{D}_{a_i} \cup \mathcal{D}_{a_j})$, both variables p_l and q_l are set to *false* which means that such transactions are useless under $\Gamma_{i,j}$. Thus, the encoding can be performed by considering only transactions of $\mathcal{D}_{a_i} \cup \mathcal{D}_{a_j}$. Interestingly enough, this allows us to avoid modeling the entire database, having a large number of clauses, and dealing with the associated computational issues.

Algorithm 1 summarizes our SAT-based approach for mining the set of 1-MRIs from transaction databases. It relies basically on a set of simplifications aiming to reduce the size of the generated sub-formulas. Following the decomposition principle discussed above, the algorithm starts by looping over the set of items of $\mathcal{D}$. More precisely, an item a is picked up (line 2). Then, if its support is less than the fixed minimum support λ, then $\{a\}$ is clearly a 1-MRI (line 4). Otherwise, the sub-database D_a is then considered (the set of items of D_a are named Ω_a). Obviously, each frequent item b not belonging to Ω_a forms with a a 1-MRI (line 9). A second loop is performed over the items of Ω_a. Then, we can distinguish the following two cases:

1. $\texttt{Supp}(\{b\}, \mathcal{D}_a) = |\mathcal{D}_a \cap \mathcal{D}_b| < \lambda$: in this case $\{a, b\}$ is clearly the only 1-MRI and it is added to S (line 16).
2. $|\mathcal{D}_a \cap \mathcal{D}_b| \geq \lambda$: in such a case, some rational simplifications are performed to reduce the size of the encoding. In fact, for each $c \in \Omega_a \cup \Omega_b$ no rare itemset of the form $\{a, b, c, \ldots\}$ can exist in the three following cases: (i) $\texttt{Supp}(\{c\}, \mathcal{D}_a) < \lambda$, (ii) $\texttt{Supp}(\{c\}, \mathcal{D}_b) < \lambda$, and (iii) $\texttt{Supp}(\{c\}, \mathcal{D}_a \cap \mathcal{D}_b) = |\mathcal{D}_a \cap \mathcal{D}_b|$ (lines 20-24). In such cases, the item c can be removed from $\mathcal{D}_a \cup \mathcal{D}_b$ (function `Simplify`, line 25). After this simplification step, the remaining database is finally encoded (function `Encode_in_CNF`, line 26), and a model enumeration procedure over the resulting formula (function `Enumerate_Models`, line 28) is performed to find the set of all models that correspond to the set of 1-MRIs.

Algorithm 1 SAT-based 1-**MRI C**omputation (**SAMRIC**)

Input: A transaction database $\mathcal{D}$, a support threshold λ
Output: The set of 1-MRIs S

1: $S = \emptyset, \Gamma = \emptyset, \Gamma' = \emptyset, \Delta = \emptyset$
2: **for** $a \in \Omega$ **do**
3: **if** $\texttt{Supp}(\{a\}, \mathcal{D}) < \lambda$ **then**
4: $S \leftarrow S \cup \{a\}$
5: **else**
6: $\Gamma \leftarrow \Gamma \cup \{a\}$
7: **for** $b \notin \Omega_a$ **do**
8: **if** $\{a\} <_{\text{freq}} \{b\}$ && $\texttt{Supp}(\{b\}) \geq \lambda$ **then**
9: $S \leftarrow S \cup \{a, b\}$
10: **end if**
11: **end for**
12: $\Gamma' \leftarrow \Gamma$
13: **for** $b \in \Omega_a$ **do**
14: **if** $\texttt{Supp}(\{b\}, \mathcal{D}_b) \geq \lambda$ **then**
15: **if** $\texttt{Supp}(\{b\}, \mathcal{D}_a) < \lambda$ **then**
16: $S \leftarrow S \cup \{a, b\}$
17: $\Gamma' \leftarrow \Gamma' \cup \{b\}$
18: **else**
19: $\Delta \leftarrow \emptyset$
20: **for** $c \in \Omega_a \cup \Omega_b$ **do**
21: **if** $\texttt{Supp}(\{c\}, \mathcal{D}_a) < \lambda \;||\; \texttt{Supp}(\{c\}, \mathcal{D}_b) < \lambda \;||\; \texttt{Supp}(\{c\}, \mathcal{D}_a \cap \mathcal{D}_b) = |\mathcal{D}_a \cap \mathcal{D}_b|$ **then**
22: $\Delta \leftarrow \Delta \cup \{c\}$
23: **end if**
24: **end for**
25: $\mathcal{D}_s \leftarrow \texttt{Simplify}(\mathcal{D}_a \cup \mathcal{D}_b, \Gamma', \Delta)$
26: $\Psi = \texttt{Encode\_in\_CNF}(\mathcal{D}_s)$
27: $\Phi \leftarrow \Psi \wedge x_a \wedge x_b \wedge \bigwedge_{c \in \Gamma'} \neg x_c \wedge \bigwedge_{c \in \Delta} \neg x_c$
28: $S = S \cup \texttt{Get\_Patterns}(\texttt{Enumerate\_models}(\Phi))$
29: $\Gamma' \leftarrow \Gamma' \cup \{b\}$
30: **end if**
31: **end if**
32: **end for**
33: $\Gamma \leftarrow \Gamma \cup \{a\}$
34: **end if**
35: **end for**
36: **return** S

5 Empirical Studies

We now present the experiments carried out to assess the performance of our symbolic AI approach to mining k-MRIs from transaction databases.

Algorithm 1 uses the MiniSAT solver, which is a popular SAT solver written in C++. The solver has been modified to disable restarts and perform a simple backtrack each time a model is found. This modification allows the solver to enumerate all models instead of stopping at the first satisfying assignment [Morgado and Marques-Silva, 2005]. Our source code and datasets are available at https://github.com/amel-hidouri/SAMRIC.git. It should be noted here that for the decomposition, items are sorted in increasing order of frequency. This strategy has been shown to generate small-scale

Instance	$\|D\|, \|\Omega\|, d(\%)$	1-MRI							2-MRI	
		λ	Walky-G	CP4MII	SAMRIC			#1-MRIs	SAMRIC	
			Time	Time	Time	#Clauses	#Conf		Time	#2-MRIs
Australian-credit	653, 125, 41	150	7.24	7.13	12.53	3728726	267836	447339	76.32	11232
		120	30.67	16.1	28.13	5439614	689799	1069099	176.96	10549
		100	94.72	33.23	51.3	6955828	1392377	2026756	327.69	7452
		80	331.13	81.85	99.25	8884832	3043183	4126577	713.01	9927
		50	OM	340.73	311.94	12968960	12579300	14764676	3455.5	15988
Chess	3196, 75, 49.33	2500	0.61	0.09	0.16	56181	389	511	0.14	1498
		1000	15.74	8.85	8.01	2160106	133582	152316	6.93	1231
		500	OOM	113.59	54.54	5205068	1250215	1353344	37.14	997
		160	OOM	1187.26	270.14	10338201	9470426	9364262	191.51	1344
Kr-vs-kp	3196, 73, 49	1000	17.32	8.71	7.61	2010900	129748	147402	6.32	1028
		500	494.37	104	47.78	4464959	1127491	1216675	28.86	933
		250	OOM	453.38	138.91	7095525	4252364	4358189	74.88	684
		100	OOM	1134.01	256.98	10217782	10447743	9766140	175.3	1417
		50	OOM	1111.14	261.7	12014148	11951221	9966663	266.38	793
Hypothyroid	3247, 88, 49	1000	228.25	51.84	15.01	1942537	659795	678192	8.35	1266
		750	OOM	114.17	26.77	2836790	1324214	1351554	25.72	1771
		500	OOM	254.66	48.82	4249107	2757496	2781524	26.94	851
		100	OOM	499.21	83.73	9700357	6357909	6036383	80.73	1322
		50	OOM	330.1	65.3	11573554	4677783	4151903	114.66	1696
Liquor	9284, 2626, 0.10	5000	0.78	OM	0.51	$\star$	0	4051	1.08	8066207
		1000	0.83	OM	5.04	20053	9	8482	97.78	7861133
		500	2,52	OM	13.26	147137	57	16961	272.4	8164759
		250	3,01	OM	39.95	1321041	284	38397	657.23	10128473
Retail	16470, 1030, 0.06	2000	0.41	720.88	0.21	51770	9	16561	0.34	135359185
		1000	0.57	723.63	0.41	84644	18	17912	1.47	134727253
		500	0.64	726.83	1.26	188118	71	33193	9.4	133613822
		250	1.39	867.8	4.07	700815	239	187553	64.16	164337941
Pumsb	49046, 2113, 3.5	48000	0.28	9.599	0.21	13	1	2115	0.17	2218675
		32000	15.87	63.01	44.19	7862046	54106	57425	50.5	2154049
		24000	OOM	1831.4	767.18	27515963	1425873	1257200	TO	–
		17000	OOM	TO	TO	–	–	–	TO	–
BMS(1)	59602, 497, 0.51	1000	0.11	2.9	0.07	$\star$	0	901	0.52	112843
		500	0.14	6.21	0.17	202	0	3724	3.62	175229
		100	0.47	71.65	1.95	494339	305	41949	45.01	4070866
		50	1.55	149.99	9.08	4924821	5551	77993	121.81	7195403
Connect	67557, 129, 33.33	7000	108.13	152.97	323.36	88980659	193771	165198	451.61	5916
		2000	OM	1049.27	1259.38	225726872	1673400	1180278	2899.82	13900
		1000	OM	TO	2357.76	307244491	4147061	2787960	TO	–
		676	TO	TO	3041.34	351500671	6406821	6407533	TO	–
T10I4D100K	100000, 870, 1.16	500	0.97	151.85	0.55	104730	84	161617	44.09	30645550
		400	1.22	202.26	1.18	429303	279	197796	53.16	42326895
		300	1.43	267.06	2.64	1349725	836	238334	66.92	57409762
		200	1.77	319.79	5.93	3541664	2098	273436	94.58	71668538
		100	2.41	390.26	12.05	6145539	4288	344651	1102.51	90024783
Fruithut	181970, 1265, 0.28	1000	0.26	13.83	0.41	2144	1	10576	8.37	1053040
		500	0.36	20.3	1.31	44707	39	21412	48.79	1829817
		250	0.91	32.63	4.88	1979320	317	41231	570.45	3430925
		150	1.57	54.81	12.53	8979961	1140	75567	1967.6	7068064
		100	2.49	97.30	24.46	23212662	2662	129198	TO	–
		50	3.25	234.13	70.06	80901751	10791	294690	TO	–
Kosarak	990002, 41270, 0.02	100000	14.79	OOM	0.94	3	1	41268	0.84	851420745
		80000	13.51	OOM	1.14	6	2	41275	1.47	851338220
		60000	12.87	OOM	1.41	22	3	41297	2.05	851173233
		50000	14.1	OOM	1.93	73402	8	41293	2.35	851173218
PowerC	1040000, 140, 5	10000	0.63	OOM	31.44	26627273	332	566	70.16	6041
		5000	0.68	OOM	45.28	44666695	562	829	99.53	7007
		2500	0.79	OOM	58.37	61139858	945	1447	144.21	11932
		1000	1.2	OOM	82.19	91129054	1727	2342	213.52	18969
		500	1.3	OOM	104.51	114377882	2801	4302	328.27	52308
Susy	5000000, 190, 10	500000	7.850	OOM	398.59	4290	237	1346	594.18	22698
		250000	26.65	OOM	1384.9	57655607	838	3204	1722.32	65811
		100000	84.72	OOM	1974.07	361944809	1582	5145	TO	–
		50000	250.73	OOM	TO	–	–	27187	TO	–

Table 2: Experimental results for mining k-minimal rare itemsets using real and synthetic datasets.

sub-problems [Boudane *et al.*, 2018]. Our experiments were performed on a Linux machine 32GB of RAM running at 2.66 GHz. We test our approach for $k = \{1, 2\}$ while varying the minimum support threshold (λ). The reported runtime is in seconds and the timeout is set to one hour for each test. Let us mention that for our algorithm, the computation time includes both the time needed for generating the CNF formulas and that for computing all models (i.e. the k-MRIs) of such formulas. For our empirical evaluation, experiments were carried out on different commonly used benchmark datasets taken from the well-known repositories FIMI[1], CP4IM[2] and SPMF[3]. In more details, we use small (i.e., *Australian-credit*, *Chess*, *Kr-vs-kp*, *Hypothyroid*, *Liquor*), medium (i.e., *Retail*, *Pumsb*, *BMS(1)*, *Connect*) as well as large (i.e., *T10I4D100K*, *Fruithut*, *Kosarak*, *PowerC*, *Susy*) scaled datasets with hundreds, thousands and even millions of transactions, respectively. Characteristics of all these datasets are given in Table 2: the number of transactions ($|\mathcal{D}|$), the number of items ($|\Omega|$), and the density (d).

For $k = 1$, we compared our algorithm, coined `SAMRIC`, to two baselines: the constraint programming based algorithm `CP4MII`[4] proposed recently in [Belaid *et al.*, 2019], and the specialized method `Walky-G`[5] introduced in [Szathmary *et al.*, 2012], according to the running time for different minimum support threshold values. For the computation of 2-MRIs, since we are the first to define such kind of itemsets, there are no state-of-the-art so far, we only discuss the results of our algorithm. Table 2 summarizes the empirical results of the tested algorithms for $k = 1$ and $k = 2$ on each dataset and the considered λ threshold values. Here, `TO` (resp. `OOM`) reported when an algorithm is not able to complete the mining process under the fixed time (resp. memory limitations).

Results for 1-MRIs. According to Table 2, the specialized approach `Walky-G` and `SAMRIC` have good performance in terms of computing time. They perform well on several datasets e.g., *T10I4D100K*, *Fruithut* and *Susy*. Moreover, experimental results show that `SAMRIC` surpasses the baseline `CP4MII` across almost of datasets. Interestingly enough, `CP4MII` reaches timeout/memory-out 20 times, with 15 times for `Walky-G` and 2 times for `SAMRIC`. Table 2 shows also that `Walky-G` becomes less efficient on dense datasets, e.g., *Chess* and *Hypothyroid*. This can be explained by the fact that this algorithm stores a large number of frequent generators. Interestingly, `SAMRIC` runs faster than `Walky-G`, which fails to discover the 1-MRIs for the dataset *Kr-vs-kp* with $\lambda \leq 250$, and for *Hypothyroid* with $\lambda \leq 750$. For *Retail* and *T10I4D100K* datasets, `SAMRIC` was respectively up to 700 and 66 times faster than `CP4MII`. It can also be observed that on *Liquor*, *Kosarak* and *PowerC* datasets, `CP4MII` was unable to mine the target itemsets under the timeout for all the fixed threshold values. However, for the same datasets our `SAMRIC` algorithm was able to scale for all minimum support threshold values with a maximum running time of 104 seconds. As one can see from Table 2, for the *Connect* dataset and $\lambda = 7000$, `Walky-G` is the best. However, if $\lambda \leq 2000$, the performance of `CP4MII` and `SAMRIC` become better than `Walky-G` that fails to discover 1-MRIs. Finally, it is worth noticing that our algorithm is able to scale for large datasets, e.g., *Susy* with 5 million of transactions, while `CP4MII` fails for all the considered minimum support threshold values. In our experiments, we also include the number of 1-MRIs in Table 2. Generally speaking, the minimum support threshold λ has a strong impact on the performance of the mining process. Thus, the number of patterns gets larger when the value of λ decreases and it can exceed 14 million (e.g., *Kr-vs-kp* dataset). Clearly, the performance of the three algorithms depends on the overall dataset characteristics. Interestingly, our `SAMRIC` algorithm scales well for large datasets and lower values of λ.

Table 2 reports in addition the size of our SAT encoding according to the number of clauses (#Clauses)[6] and conflicts (#Conf). The results show that the number of conflicts varies depending on the tested dataset and the λ values. Clearly enough, the encoding size is proportional to the size of the dataset, e.g., more than 361 million of clauses for *Susy*. Notably, despite this barrier, our algorithm remains efficient and able to solve almost datasets under the threshold values.

Results for 2-MRIs. Table 2 contains also the run-time and the number of found 2-MRIs for all datasets. For the decomposition, a variant of Algorithm 1 is performed. Let us first note that the number of 2-MRIs can be very large compared to the number of 1-MRIs, especially for lower λ values, e.g., more than 850 million on *Kosarak* dataset for $\lambda = 100000$. According to our results, as expected, the computation of 2-MRIs is time consuming than the one of 1-MRIs. For instance, on *Australian-credit* dataset, the time needed for discovering the 2-MRIs is about 9 times higher than for 1-MRIs. Surprisingly, this running time is similar to the one of 1-MRIs on some datasets (e.g., *Chess* and *Kosarak*). More interestingly, on 68 tests, `SAMRIC` is able to identify the complete set of 2-MRIs, even for lower threshold values.

6 Conclusions and Future Work

In this paper, we have presented a SAT-based formalization of the problem of mining k-MRIs, a generalization of minimal rare itemsets. Our approach is based on a set of logical constraints in connection with a decomposition scheme that enables high scalability without forgoing completeness and efficiency. An extensive set of experiments conducted over different datasets showed that our approach scales well and outperforms the CP-based approach `CP4MII` while competed with the specialized algorithm `Walky-G`.

In the future, we would like to improve our SAT encoding for k-MRIs mining, especially for $k > 2$. Furthermore, we plan to investigate more advanced and powerful partitioning techniques to improve the performance of our approach.

[1] http://fimi.ua.ac.be/data/

[2] http://dtai.cs.kuleuven.be/CP4IM/datasets/

[3] https://www.philippe-fournier-viger.com/spmf/index.php?link=datasets.php

[4] https://gite.lirmm.fr/belaid/cp4borders

[5] http://coron.loria.fr/

[6] Note that we didn't count clauses of formulas solved by unit propagation. We refer to that by $\star$ in Table 2.

Acknowledgments

This research has received support from the European Union's Horizon research and innovation programme under the MSCA-SE (Marie Skłodowska-Curie Actions Staff Exchange) grant agreement 101086252; Call: HORIZON-MSCA-2021-SE-01; Project title: STARWARS (STormwAteR and WastewAteR networkS heterogeneous data AI-driven management).

References

[Belaid *et al.*, 2019] Mohamed-Bachir Belaid, Christian Bessiere, and Nadjib Lazaar. Constraint programming for mining borders of frequent itemsets. In *IJCAI*, pages 1064–1070, 2019.

[Boudane *et al.*, 2016] Abdelhamid Boudane, Saïd Jabbour, Lakhdar Sais, and Yakoub Salhi. A SAT-based approach for mining association rules. In *IJCAI*, pages 2472–2478, 2016.

[Boudane *et al.*, 2018] Abdelhamid Boudane, Saïd Jabbour, Lakhdar Sais, and Yakoub Salhi. SAT-based data mining. *Int. J. Artif. Intell. Tools*, 27(1):1–24, 2018.

[Darrab *et al.*, 2021] Sadeq Darrab, David Broneske, and Gunter Saake. Modern applications and challenges for rare itemset mining. *Int J Mach Learn Comput*, 11(3):208–218, 2021.

[Dlala *et al.*, 2018] Imen Ouled Dlala, Saïd Jabbour, Badran Raddaoui, and Lakhdar Sais. A parallel SAT-based framework for closed frequent itemsets mining. In *CP*, pages 570–587, 2018.

[Fournier-Viger *et al.*, 2017] Philippe Fournier-Viger, Jerry Chun-Wei Lin, Bay Vo, Tin Truong Chi, Ji Zhang, and Hoai Bac Le. A survey of itemset mining. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 7(4):e1207, 2017.

[Hidouri *et al.*, 2021a] Amel Hidouri, Saïd Jabbour, Imen Ouled Dlala, and Badran Raddaoui. On minimal and maximal high utility itemsets mining using propositional satisfiability. In *IEEE International Conference on Big Data (Big Data)*, pages 622–628, 2021.

[Hidouri *et al.*, 2021b] Amel Hidouri, Saïd Jabbour, Badran Raddaoui, and Boutheina Ben Yaghlane. Mining closed high utility itemsets based on propositional satisfiability. *Data Knowl. Eng.*, 136:101927, 2021.

[Hidouri *et al.*, 2022] Amel Hidouri, Saïd Jabbour, and Badran Raddaoui. On the enumeration of frequent high utility itemsets: A symbolic AI approach. In *CP*, pages 27:1–27:17, 2022.

[Hu and Martin, 2004] Alan J Hu and Andrew K Martin. *Formal Methods in Computer-Aided Design: 5th International Conference, FMCAD 2004, Austin, Texas, USA, November 15-17, 2004, Proceedings*. Springer Science & Business Media, 2004.

[Izza *et al.*, 2020] Yacine Izza, Said Jabbour, Badran Raddaoui, and Abdelahmid Boudane. On the enumeration of association rules: A decomposition-based approach. In *IJCAI*, pages 1265–1271, 2020.

[Jabbour *et al.*, 2016] Saïd Jabbour, Nizar Mhadhbi, Abdesattar Mhadhbi, Badran Raddaoui, and Lakhdar Sais. Summarizing big graphs by means of pseudo-boolean constraints. In *IEEE International Conference on Big Data*, pages 889–894, 2016.

[Jabbour *et al.*, 2017] Saïd Jabbour, Nizar Mhadhbi, Badran Raddaoui, and Lakhdar Sais. A SAT-based framework for overlapping community detection in networks. In *PAKDD*, pages 786–798, 2017.

[Jabbour *et al.*, 2018] Saïd Jabbour, Fatima Ezzahra Mana, Imen Ouled Dlala, Badran Raddaoui, and Lakhdar Sais. On maximal frequent itemsets mining with constraints. In *CP*, pages 554–569, 2018.

[Jabbour *et al.*, 2020] Saïd Jabbour, Nizar Mhadhbi, Badran Raddaoui, and Lakhdar Sais. SAT-based models for overlapping community detection in networks. *Computing*, 102(5):1275–1299, 2020.

[Jabbour *et al.*, 2022] Saïd Jabbour, Nizar Mhadhbi, Badran Raddaoui, and Lakhdar Sais. A declarative framework for maximal k-plex enumeration problems. In *AAMAS*, pages 660–668, 2022.

[Mannila and Toivonen, 1997] Heikki Mannila and Hannu Toivonen. Levelwise search and borders of theories in knowledge discovery. *Data mining and knowledge discovery*, 1(3):241–258, 1997.

[Morgado and Marques-Silva, 2005] António Morgado and João Marques-Silva. Algorithms for propositional model enumeration and counting. Technical report, Instituto de Engenharia de Sistemas e Computadores, Investigação e Desenvolvimento em Lisboa, 2005.

[Szathmary *et al.*, 2007] Laszlo Szathmary, Amedeo Napoli, and Petko Valtchev. Towards rare itemset mining. In *ICTAI*, pages 305–312, 2007.

[Szathmary *et al.*, 2012] Laszlo Szathmary, Petko Valtchev, Amedeo Napoli, and Robert Godin. Efficient vertical mining of minimal rare itemsets. In *CLA*, 2012.

[Zaki, 1999] Mohammed J Zaki. Parallel and distributed association mining: A survey. *IEEE concurrency*, pages 14–25, 1999.

[Zhang *et al.*, 1996] Hantao Zhang, Maria Paola Bonacina, and Jieh Hsiang. Psato: a distributed propositional prover and its application to quasigroup problems. *Journal of Symbolic Computation*, pages 543–560, 1996.

[Zhang *et al.*, 2012] Shuyuan Zhang, Sharad Malik, and Rick McGeer. Verification of computer switching networks: An overview. In *International Symposium on Automated Technology for Verification and Analysis*, pages 1–16, 2012.