Expert Systems with Applications 42 (2015) 2582–2592

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Association rule mining with mostly associated sequential patterns

Ömer M. Soysal [*]

Highway Safety Research Group, Louisiana State University, 3535 Nicholson Ext., Baton Rouge, LA, USA

ARTICLE INFO

Article history:
Available online 10 November 2014

Keywords:
Association rule mining
Interesting rules
Pattern recognition
Big data
Knowledge discovery
Data mining

ABSTRACT

In this paper, we address the problem of mining structured data to find potentially *useful* patterns by association rule mining. Different than the traditional *find-all-then-prune* approach, a heuristic method is proposed to extract mostly associated patterns (MASPs). This approach utilizes a maximally-association constraint to generate patterns without searching the entire lattice of item combinations. This approach does not require a pruning process. The proposed approach requires less computational resources in terms of time and memory requirements while generating a long sequence of patterns that have the highest co-occurrence. Furthermore, *k*-item patterns can be obtained thanks to the sub-lattice property of the MASPs. In addition, the algorithm produces a tree of the detected patterns; this tree can assist decision makers for visual analysis of data. The outcome of the algorithm implemented is illustrated using traffic accident data. The proposed approach has a potential to be utilized in big data analytics.

1. Introduction

In the last century, data-driven decision making is becoming more challenging due to production and processing of extremely huge amount of data from a variety of sensors. The decision makers are often required to understand relations within the multi-dimensional space before taking an action, making a law, producing a product, setting up regulations, etc. In this paper, we attempt to reveal *useful relations* by means of extracting mostly associated patterns from a structured data.

Many approaches have been proposed to discover *useful* information from structured data. Among these approaches, association rule mining (ARM) plays an important role. The ARM algorithms aim to discover hidden rules among enormous pattern combinations based on their individual and conditional frequencies. The traditional ARM algorithms first generate all of the possible patterns from the data while pruning out non-frequent ones and then produce rules from these frequent patterns. Once the rules are generated, some interesting measures (IMs) are applied to obtain interesting rules that can be used in decision making. A general process flow of an ARM framework is shown in Fig. 1. A brief description of the modules in this general framework is as follows: (a) the Preprocess module is used to localize data by filtering, to summarize data by sampling, or to transform data to speed up rule detection, (b) the C-Generator finds candidate patterns, (c) a pruning is applied before rule generation, (d) the R-Generator is used to generate *k*-items rules, (e) interesting rules are obtained by the R-Filter. The constraints define the rule search strategy. The ARM algorithms differ mainly from each other based on utilization of these constraints. Among the many, both thresholds, the minimum support and minimum confidence, would be considered as default constraints.

As we summarized in the section 'Related Works' of this paper, interesting rules are extracted through an exhaustive search if no constraint other than the default ones is used on patterns. In this paper, we propose an approach that imposes an interestingness constraint on patterns to detect the highest co-occurring ones without searching all possible pattern combinations (entire lattice) and filtering them out later. This approach offers an advantage of consuming significantly less computational resources for finding long rule sequences. During the search process, a most associated sequential pattern (MASP) tree is formed. After generating the MASP tree, the rules are generated in significantly less computation time. Besides obtaining MASPs, a traditional rule mining can be conducted within a relatively small data set of each MASP; the outcome of both MASPs' rules and traditionally obtained rules can be combined to find interesting rules as explained in the method section of this paper; this combination is named as 'MASP+'. Readers should refer to Lemma 4. Furthermore, the MASP tree has a sub-lattice rule generation property that reveals *k*-items rules from MASPs as stated in Theorem 1.

In general, real data to be mined has 'attribute = value' imbalances; that is, some distinct values of an attribute are

[*] Tel.: +1 225 578 6297; fax: +1 225 578 0240.
E-mail address: omsoysal@lsu.edu

http://dx.doi.org/10.1016/j.eswa.2014.10.049
0957-4174/© 2014 Elsevier Ltd. All rights reserved.

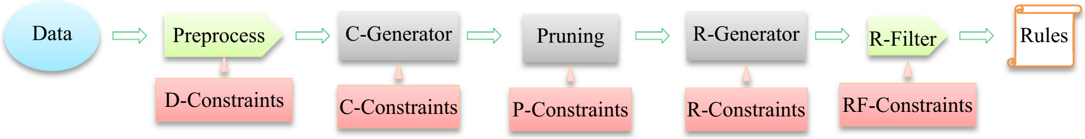

Fig. 1. The general process flow of an ARM framework for detection of interesting rules.

over-represented than other values of the same attribute. As an example, events of property-damage-only cases are extensively more than the injury-only cases, and injury-only cases are relatively higher than fatal-only cases in traffic accident data. When applied to such a data, a traditional ARM algorithm will favor the over-representing frequent items; consequently, these over-representing items will show up in most of the rules. The proposed approach can find these most *favorable* rules without spanning the entire search lattice. In addition, the proposed approach is capable of discovering long patterns while utilizing less resources compared to the exhaustive search approaches that consume a significant amount of resources.

The rest of the paper is as follows: The literature review about searching lattice and mining interesting rules are presented in the section *Related Works*. The proposed approach is introduced in the *Method* section. Data, the experiments conducted, and their results are summarized in the *Experiments and Results*. The paper is finalized with conclusion and future works.

2. Related Works

In general, the problem of mining association rules is solved in two steps (Das, Ng, & Woon, 2001): (1) first, all frequent itemsets are found, (2) then, association rules are generated from the frequent itemsets. Once the rules are obtained, the rules are ranked by their interestingness measure. In this section, we provide a brief review of approaches that aim to search combinatorial pattern space for finding frequent itemsets and to filter the interesting rules among the rule set.

Association rule mining algorithms can be classified based on the search strategy used to find frequent itemsets and on the scope of the search. The scope can be the entire lattice or a sub-lattice determined by constraints. The search strategies show varieties based on how to traverse data space; some algorithms find frequent itemsets directly from the transactional data while others form an intermediate data structure. In the former group, the Apriori algorithm (Agrawal & Srikant, 1994) utilizes the bread-first approach, and the Eclat algorithm (Klbsgen, 1996) uses the depth-first approach. In the latter group, the FP-growth algorithm (Han, Pei, Yin, & MAO, 2004) transforms transactional data into the form of a tree; the A-Close proposed in Pasquier, Bastide, and Taouil (1998) finds frequent closed itemsets from which all frequent itemsets are derived or rules are directly generated from the closed set; the MAFIA proposed in Burdick et al. (2001) obtains the maximal itemsets before rule generation.

The constraint-based algorithms perform a filtering operation on the data itself, on the patterns while being generated, or on the patterns after being generated (Kotsiantis & Kanellopoulos, 2006). A constraint can belong to a data set, to a measure (such as a statistic) for discovering patterns, or to the type of patterns to be discovered (Wojciechowski & Zakrzewicz, 2002); note that temporal and spatial constraints would be considered under the 'type of patterns'. Among the constraint-based mining, the RARM (Das et al., 2001) finds frequent 2-itemsets and utilizes an Apriori-based strategy to find frequent k-itemsets where $k \geqslant 3$. In Das et al. (2001), a schema constraint, which defines the structure of the patterns, and the opportunistic confidence constraint, which aims to discriminate significant and redundant rules, are introduced. The category-based (or concept hierarchy-based) approaches, e.g. in Do et al. (2003) at each pass, check whether the transaction has items belonging to the "categories" (or concepts) specified by the user.

Multiple-minimum supports proposed in Wojciechowski and Zakrzewicz (2002) discover sequential patterns by means of a tree structure. An improved version of the predictive (n,p) approach proposed in Denwattana and Getta (2001) is introduced in Hong, Horng, Wu, and Wang (2009), where the frequent itemsets are discovered through promising and non-promising candidate itemsets using two threshold parameters of minimum itemsets' length and minimum frequency. A review of association rule mining algorithms from the subgroup discovery perspective is provided in Herrera, Carmona, González, and Jesus (2011).

Among the most recent research on finding frequent patterns, as an emerging topic, mining top-k frequent patterns that does not require to set a minimum support value is studied by Pyun and Yun (2014), Deng (2014). The closed itemsets can be extracted from these top-k frequent patterns. The former researchers developed a new algorithm based on the FP-growth structure and the later proposed a new data structure named Node-list. Tseng (2013) addresses the problem of mining large databases. The author proposed a hierarchical partitioning approach on both the database and solution space. In discovery of patterns from large database, Király, Laiho, Abonyi, and Gyenesei (2014) reduced two well-known problems of frequent closed itemset mining and biclustering into a single problem for binary data. In Chen, Lan, Hong, and Lin (2013), propositional logic is utilized to find coherent rules that take into consideration of negations; this approach addresses to find an appropriate minimum support as well. Jin, Wang, Huang, and Hu (2014) employed causality between antecedent and consequent to discover interesting rules; they used causality as an objective measure. The frequent itemsets and useful rules are explored by similarity instead of attribute–value equivalence in Rodríguez-González, Martínez-Trinidad, and Carrasco-Ochoa (2013). They adapted the algorithm proposed in Agrawal and Srikant (1994) to generate interesting rules. In Vo, Coenen, and Le (2013), significance of items are considered while finding frequent itemsets and interesting patterns. They proposed the WIT-tree (Weighted Itemset-Tidset tree) as a data structure to mine high utility itemsets.

2.1. Rule interestingness

In ARM, the second main step after discovering frequent patterns is to generate the rules. As in the most cases, the ARM- based information discovery suffers from producing many trivial or uninteresting patterns when all possible rules are produced first and then redundant ones are eliminated (Ashrafi, Taniar, & Smith, 2004, 2005; Omiecinski, 2003). Sahar (2010) classifies IMs in three main categories as objective, subjective, and semantics-based measures. Many criteria have been proposed for elimination of redundant rules (or for revealing interesting ones) (Heravi & Zaïane, 2010; Sahar, 2010). Discovery of non-redundant rules based on logical inference is proposed in Lo, Khoo, and Wong (2009); the authors utilized the discovery over a compressed set of non-redundant rules. In Cheng, Ke, and Ng (2008), δ-Tolerance Association Rules (δ-TARs) is introduced as a non-redundant representation of association rules. The tree produced by this algorithm enables efficient generation of δ-TARs and provides means of querying the association rules. The authors in Xu, Li, and Shaw (2011) attacked the non-redundant association rule elimination through concise representation of frequent items based on closed itemsets. They defined Reliable Basis, which is proved to be a lossless representation of ARs based on their redundant rule definition, and proposed a Certainty Factor to measure the strength of association rules discovered.

Among the methods for selecting appropriate interestingness measures, we noted the rankings with the 'property matching' and with the 'matching by expert scoring with a measure ranking' (Tan et al., 2002), and multi-criteria decision approach towards measure selection (Guillet & Hamilton, 2007). The intuition behind the property-matching approach depends on the fact that no measure is consistently better than others in all application domains. This is because different measures have different intrinsic properties, some of which may be desirable for certain applications but not for others. Thus, in order to find the right measure, we need to match the desired properties of an application against the properties of the existing measures. In 'matching by expert scoring with the measure ranking', since it is practically not possible for an expert to rank all the tables manually, a smaller set of contingency tables are given to the experts for ranking and they use this information to determine the most appropriate measure. In the multi-criteria decision approach, each measure is analyzed with respect to the properties and then evaluated. Klemettinen et al. have considered the problem of mining interesting rules from the large set of association rules generated (Klemettinen, Heikki, Ronkainen, & Toivonen, 1994). Their idea is to classify the attributes of the original data to an inheritance hierarchy and to utilize templates defined in terms of that hierarchy. These templates are later used to prune the rules effectively according to the user's intuitions.

In Marukatat (2006), Marukatat proposed a structured-based rule selection framework, which classifies, selects, and filters the rules based on the rule structures. This framework consists of semantic rule classification and permutation analysis. The semantic rule classification involves classification of the rules as candidate, strongly abundant, or weakly abundant. The permutation analysis filters the equivalent but less significant ones; then the rules that cover the other rules are selected and the ones being covered are discarded. Chen and Tsai constructed a relationship graph according to the patterns identified from the transactional database and the priorities of condition attributes from the customer database (Chen & Tsai, 2004). An efficient graph-based algorithm based on the relationship graph is presented to discover interesting association rules. In Webb and Yu (2004), interesting rules are discovered by clustering frequent patterns; the interesting rules are the ones having dissimilar items. Mining association rules at different levels of taxonomy is considered in Han and Fu (1999).

3. Method

In association rule mining, data to be mined can be stored in the form of transactions. A transaction t is composed of some items in the form of (attribute = value) pairs as in t = {Age = Young, intoxicated = Yes, Day = Friday}. The goal of the ARM is to discover non-trivial patterns hidden in the transactional data set. An association rule $A \rightarrow C$ has two sets of items (*itemsets*), namely the antecedent A and the consequent C. An itemset of $A \cup C$ is considered to be a rule if its frequency (Support(*itemset*) = P(*itemset*)) satisfies a *minimum support* threshold and the conditional probability $\mathrm{P}(C|A)$ satisfies a *minimum confidence* threshold. The support can be considered as a global measure of being interesting, and the confidence is used as a localization measure. These two constraints are applied in C-Constraints of the MASP framework. These two measures may not be enough to reveal interesting patterns; as a general practice, the measure 'Lift' is used to find interesting patterns. The Lift measures how much C is dependent on A. It can be interpreted as the degree of lifting confidence in association between A and C from global existence (Support(C)) to a local association (Confidence($A \rightarrow C$)). The Lift($A \rightarrow C$) is given by

$$\mathrm{Lift}(A \rightarrow C) = \frac{\mathrm{Confidence}(A \rightarrow C)}{Support(C)} = \frac{\mathrm{P}(A, C)}{\mathrm{P}(A)\mathrm{P}(C)}$$

In the traditional ARM framework that utilizes the default constraints only, the combinatorial search space is first explored for finding all frequent items followed by a rule generation process, and then an interesting measure is applied. In contrast to generate-all-and-test (or find-all-then-prune), in the framework proposed in this paper, the rules are generated by utilizing an interestingness (or *preference*) constraint together as defined in Definition 3 below. The items that form the rules are ordered in sequence; each item in the rule sequence has a level value (the order index) of ordinals 1st, 2nd, etc. When a new item is added to the rule sequence at a level, a sub-set of data is retrieved to find the next item of the rule sequence; this sub-set data is obtained by predicates composed from the items in the sequence. This process iteratively continues until certain criterion is achieved. At the final stage, the traditional ARM is applied to the sub-set of the whole data. The proposed framework is depicted in Fig. 2. The C-Generator module applies both default constraints of minimum support and confidence. The R-Generator updates the MASP after applying the interestingness criteria. The R-Filter decides whether to use the MASP for further mining by a T-ARM algorithm. This process is iteratively applied to sub-set data of each MASP. Once the MASP tree is obtained, a T-ARM algorithm is utilized to find the rules in a subset of data extracted by the MASPs of the tree to generate MASP+ rules.

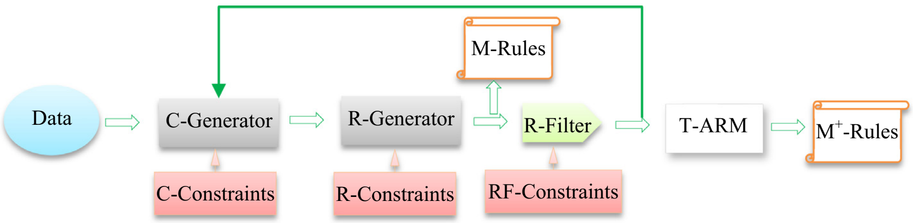

Fig. 2. The proposed MASP framework.

Detection of rules is performed in five main steps. First, the original data table is pre-processed to quantize each attribute's values into bins; following the quantization, the bin values are encoded for speeding up mining operations. After this pre-processing step, a MASP tree is constructed as described in the next section. Third, the data blocks are retrieved from the original data table for each path, which is from the root of the tree to terminal nodes. Then, some extra rules, 't-rules', are generated from these data blocks by means of a T-ARM based algorithm. The final step produces the rule sets 'MASP+' by combining the MASP and the t-rules. As an implementation note, we do not save data for each block; rather we keep record of its query.

3.1. The most associated sequential pattern tree

In this section, we explain how to obtain a MASP tree. First, we will introduce some definitions used to explain our approach.

Definition 1. Let's define some basic notations used throughout this paper. Given a set $S = \{A_1, A_2, \ldots, A_n\} = \{A_i\}$ with some elements A_i, where $i = 1, 2, \ldots, n$, $|S|$ denotes the cardinality of the set S; if the set S is a data table, its cardinality is given by the number of its records. The operator "$\subset$" denotes a subset relation as in "$X \subset Y$", which reads 'X is a subset of Y'. The operator "$\backslash$" denotes set minus as in "$X \backslash Y$", which reads 'Y is excluded from X'. The function $\text{Row}(P_Q)$ returns some records for the predicate P_Q applied to data source of a query Q. In this paper, we use record, transaction, or row, and attribute or column interchangeably. The function $\text{Parent}(M)$ returns the parent of a sequence (or set) M.

Definition 2. A *data table* D is a set of all records that are composed of attribute–value *cells* $C_{ri} \triangleq (A_i = V_{ij})_r$, where a row index $r = 1, 2, \ldots, |D|$ and attribute index $i = 1, 2, \ldots, |A|$. $\text{Atr}(D) = \{A_i\}$ is a function which returns the *attribute set* of the data table D. Each attribute A_i has a *value set* $V_i = \{V_{ij}\}$, where value index $j = 1, 2, \ldots, |V_i|$. Each unique attribute–value pair is called an *item* $I \triangleq (A_i = V_{ij})$; sometimes we use $I \triangleq V_{ij}$ for clarity. Notice that each cell is an instance of an item. As an example, let A_1 = "*Weather*" and $V_1 = \{V_{ij}\}$ = {Sunny, Rain, Fog}; then, an item $I = (A_1 = V_{11})$ or simply $I = V_{11}$ will correspond to "*Weather* = *Sunny*".

Definition 3. We define *MASP* as a sequence $M = (I_1, I_2, \ldots, I_{k-1}, I_k, \ldots, I_K)$ whose elements, called *items*, satisfy two default constraints: (1) A child item I_k at the level $k = 1, 2, \ldots, K$ must be a frequent item that satisfies the minimum support threshold τ_S and (2) this child has the highest association strength that satisfies the minimum confidence threshold τ_C given its parent $(I_1, I_2, \ldots, I_{k-1})$, among the other items existing in the same subset data; that is, (1) $P(I_1, I_2, \ldots, I_k) \geqslant \tau_S$ and (2) $P(I_k|I_1, I_2, \ldots, I_{k-1}) \geqslant \tau_C$ and $P(I_k|I_1, I_2, \ldots, I_{k-1})$ must be the maximum.

Definition 4. We define a *block* $B(M) \subseteq D$ of a MASP $M = (I_1, I_2, \ldots, I_K)$ as a set of transactions which is obtained by the *query* $Q = \{\text{SELECT } A_Q; \text{FROM } D; \text{WHERE } P_Q\}$ of M. Each item in a MASP is the most frequent item *Imax* at the level k given its parent MASP $(I_1, I_2, \ldots, I_{k-1})$. Notice that $P(\mathit{Imax};\ \text{parent MASP}) = \mathit{maxC}/|D| \geqslant \tau_S$, where *maxC* denotes the number of records where *Imax* appears in the block of the parent MASP.

A MASP query is composed of three parts: (1) Select clause having a set of the attributes $A_Q = \text{Atr}(D) \backslash \text{Atr}(P_Q)$, (2) data source D, and (3) the predicate P_Q which is composed with conjunctions of the items in the MASP. As an example, let $M = (I_1 = (A_1 = V_{11}), I_2 = (A_2 = V_{23}), I_3 = (A_4 = V_{42}))$ (or simply, $M = (V_{11}, V_{23}, V_{42})$) be a MASP given the set of attributes $\text{Atr}(D) = A_1, A_2, \ldots, A_6$; then we would get $Q = \{\text{SELECT } A_Q = \{A_3, A_5, A_6\}; \text{FROM } D, \text{WHERE } P_Q = (A_1 = V_{11} \text{ and } A_2 = V_{23} \text{ and } A_4 = V_{42})\}$ and $B(M)$ would be a set of all the rows $R = \text{Row}(P_Q)$ where each row is composed of the cells C_{ri}, where $i = 3, 5, 6$ and $r = 1, 2, \ldots, |R|$.

Definition 5. A *Parent block* of a MASP $M = (I_1, I_2, \ldots, I_K)$ is the block belonging to the parent sequence $\text{Parent}(M) = (I_1, I_2, \ldots, I_{K-1})$.

Definition 6. A *counter-block* $CB(M)$ of a MASP M is the set of all records $CB(M) = \text{Parent}(B(M)) \backslash \text{Row}(P_Q)$. The counter-block is composed of the records from the parent block excluding all the rows included in the block $B(M)$. The *CB* has all columns of its parent block while the block excludes the attribute of *Imax* found in its parent; note that $A_Q = \text{Atr(Parent block)}$ for a CB. As an example, given $M = (I_1 = (A_1 = V_{11}), I_2 = (A_2 = V_{23}))$ and the $\mathit{Imax} = I_3 = (A_4 = V_{42})$ at the level 3, the counter-block at this level will be from the rows retrieved by the predicate "$(A_1 = V_{11})$ *and* $(A_2 = V_{23})$ *and* $(A_4 \neq V_{42})$". Fig. 3 illustrates parent block, block, and counter-block relation.

Definition 7. A *most associated sequential pattern tree* 'MTree' is a directed graph whose nodes hold an item and edges connect a parent node to its most associated child. Lemma 1 is direct implication of this definition.

Lemma 1. *Each sub-path that originated from the root node of a MASP tree's branch is also a most associating sequence.*

Proof. By definition, the most associated item (MAI) is added to the sequence at each iteration. Let the MAIs be found until the iteration K as $M_K = (I_1, I_2, \ldots, I_K)$, where I_1 resides at the root node and the branch will be formed from I_1 until I_K. After removing I_K, rest of the sequence $M_{K-1} = (I_1, I_2, \ldots, I_{K-1})$ is still composition of MAIs as long as the order is not changed. Formally, $M_K \rightarrow M_{K-1}$, where "$\rightarrow$" means logical implication. Deductively, $M_{K-1} \rightarrow M_{K-2}, \ldots, M_2 \rightarrow M_1$. Therefore any sub-path of $M_2, M_3, \ldots, M_{K-1}$ obtained from the path of M_K is a MASP. □

Lemma 2. *In a MASP tree, the path that does not include any negations is an element within the entire itemset-lattice.*

Proof. By definition, the entire itemset-lattice L is a set of all k-itemsets without negations where $k = 1, 2, \ldots, |A|$ and $|A|$ is the number of attributes as described in Definition 2. Let $M_K = (I_1, I_2,$

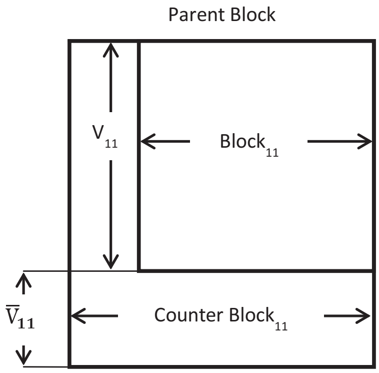

Fig. 3. Block and counter-block of a MASP M, where $M = (A_1 = V_{11})$ and $\bar{V}_{11} \equiv (A_1 \neq V_{11})$, *Imax* of the parent block is $A_1 = V_{11}$.

…, I_K) be the path of the items without negation, where $K \leqslant |A|$; then this K-itemset $M_K \in L$. □

Definition 8. A *terminal branch* is the path from the root to a terminal node of the MTree.

Now, we will explain building a MASP tree with a simple example before introducing the algorithm. Let *CanB* denotes the set of candidate blocks from which MASPs will be detected. Initially, *MTree* has a root node and $CanB = \{D\}$, where D is the original data table as defined in Definition 2. The algorithm first finds the most frequent item, say $Imax = (A_{11} = V_{11})$, in $CanB(1) = D$ satisfying $P(V_{11}) \geqslant \tau_S$ and $P(V_{11}|D) \geqslant \tau_C$; accordingly, the *MTree* will be updated by adding two children $\{V_{11}, \bar{V}_{11}\}$ to the *Root*. Then, the block B_{11} and the counter-block CB_{11} of *Imax* are added to the set *CanB* resulting in $CanB = \{B_{11}, CB_{11}\}$; notice that {D} is removed from *CanB*. Similarly, assume that $P(V_{41}, V_{11}) \geqslant \tau_S, P(V_{41}|V_{11}) = P(V_{41}|B_{41}) \geqslant \tau_C$, and $Imax = V_{41}$ has the highest number of occurrences among all other items existing in $CanB(1) = B_{11}$; consequently, the *MTree* will have two new children to the node "V_{11}". Then, the block $B_{41}$and the counter-block CB_{41} of *Imax* will be added to *CanB*; the updated set will be $CanB = \{CB_{11}, B_{41}, CB_{41}\}$. Formation of the *MTree* continues until no block is left in *CanB*. A sample *MTree* is illustrated in Fig. 4. As shown in this figure, currently eight MASPs ending at the terminal nodes are detected; they are

$$\begin{aligned}MTree = \{&(V_{11}, V_{41}, V_{31}, V_{21}), (V_{11}, V_{41}, V_{31}, \bar{V}_{21}), (V_{11}, V_{41}, \bar{V}_{31}),\\ &(V_{11}, \bar{V}_{41}), (\bar{V}_{11}, V_{13}, V_{21}), (\bar{V}_{11}, V_{13}, \bar{V}_{21}),\\ &(\bar{V}_{11}, \bar{V}_{13}, V_{14}), (\bar{V}_{11}, \bar{V}_{13}, \bar{V}_{14})\}.\end{aligned}$$

As Lemma 1 postulates, as an example, (V_{11}, V_{41}, V_{31}) and (V_{11}, V_{41}) are also a MAPS at the level 3 and 2, respectively. As Lemma 2 postulates the branch $(V_{11}, V_{41}, V_{31}, V_{21})$ is a combination within the entire lattice. Notice that the sub-sets of this branch are also covered by the entire lattice as pointed in Theorem 1 presented in the section 'Generating Rules' below. This branch is enclosed with an ellipse-like shape in Fig. 4. The items in *CanB* are obtained through a breadth-first traversal of MTree. As an example, the blocks and counter-blocks will be processed to generate the MTree shown in Fig. 4 in the order of

$$\left(D, B_{11}, CB_{11}, B_{41}, CB_{41}, B_{13}, CB_{13}, B_{31}, CB_{31}, B^{(1)}_{21}, CB^{(1)}_{21}, B_{14}, CB_{14}, B^{(2)}_{21}, CB^{(2)}_{21}\right)$$

In the following section, we introduced the algorithm to create a MASP tree and explain in some details. The 't-mined', which is used below, stands for 'traditionally mined'; that is, an ARM algorithm with find-all-then-prune is used.

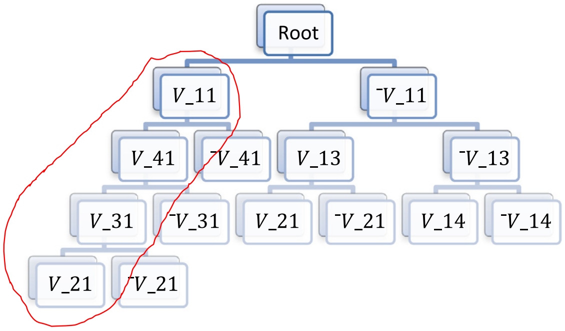

Fig. 4. An example of MASP tree. The left child of a node with ¯V as in ¯V_11 means negation of V_{11}.

Algorithm creating A MASP tree

```
Inputs: Data table $D$ to be mined and the thresholds minimum
   support $\tau_S$ and confidence $\tau_C$.
Output: MASP Tree $MT$.
Steps:
(A) Initialize:
  1. $CanB = \{D\}$;      //set of all candidate block queries to be
     processed
  2. $MTree = (Root)$;      //MASP Tree
  3. $Cmin = \tau_S|D|$;     //minimum number of records to be
                           considered as a frequent item
(B) Repeat until $CanB = \{\}$:
  1. $Can = CanB(1)$;
  2. Remove $CanB(1)$ from $CanB$;
  3. Find the most frequent item $Imax$, if any, having the
     number of occurrences $maxC$; that is, $P(Imax$;
     parent MASP) = $maxC/|D| \geqslant \tau_S$. Finding the $Imax$:
    (a) Obtain $C_i = \{Count_1, Count_2, \ldots\} \in C = \{C_1, C_2, \ldots, C_{|A|}\}$,
        $i = 1, 2, \ldots, |A|$, where $A$ denotes the attribute set of $Can$
        and $C_i$ is the set of distinct value counts for the
        attribute $A_i$;
    (b) Obtain the set $Cmax = \{Cmax_1, Cmax_2, \ldots\} = \{\max\{C_i\} \geqslant$
        $Cmin\}$ from $C$;
    (c) If there is a frequent item ($Cmax \neq \{\}$), then Compute
        $maxC = \max\{Cmax\}$ and obtain the most associated
        item $Imax = \mathrm{argmax}(\max\{Cmax\})$;
4. If there is a frequent item $Imax$
  4.1 If the association confidence strength $P(Imax$|parent
     MASP) = $maxC/|Can| \geqslant \tau_C$ is satisfied
    4.1.1. Add $Imax$ to $MTree$;
    4.1.2. Form the block $Bmax$ and the counter-block $CrBmax$
           of the item $Imax$;
    4.1.3. Add $Bmax$ to $CanB$;
    4.1.4. If $|CrBmax|/|D| \geqslant \tau_S$;
      4.1.4.1. If $|CrBmax|/|Can| \geqslant \tau_C$, then negation $\overline{Imax}$ of
               $Imax$ to $MTree$ and add $CrBmax$ to $CanB$; as an
               example, given $Imax = (A_1 = V_{11})$, then
               $\overline{Imax} = (A_1 \neq V_{11})$
      4.1.4.2. Else indicate that the block of the MASP ending
               at $\overline{Imax}$ can be t-mined Notice that the block for
               the negation part is obtained differently; it
               includes the attribute of negation.
    4.1.5. Else ($|CrBmax|$ does not satisfy the thresholds $\tau_S$)
           indicate that the block of the MASP ending at $\overline{Imax}$
           will not be t-mined
  4.2 Else indicate that the block of the MASP can be t-mined
      E.g., let A = {A1, A2, …, A5}, the MASP = (V11, V21),
      Imax = V31. The block of $A_Q$ = (A3, A4, A5) and
      $A_Q$ = (A1 = V11 And A2 = V21)) can be t-mined.
5. Else if the $Can$ is a block, then indicate that the block of the
   MASP can be t-mined
6. Else (the candidate block is a counter-block) indicate that
   the block of the MASP ending at $Imax$ will not be t-mined.
```

In step B.1, the first data set from the candidate blocks is picked for finding the most frequent item *Imax* if any. In step B.3, the number of row counts *maxC* of *Imax* within the candidate block *Can* is calculated and then this count is used to test if two conditions are satisfied; first, the support $P(Imax) = maxC/|D|$ is tested against the minimum support τ_S followed by the confidence test $P(Imax|\text{parent MASP}) = maxC/|Can| \geqslant \tau_C$. This second test confirms existence of *adequate* association between the most frequent item and its parent MASP that has a block of $|Can|$ records. Notice that imposing the highest association is performed by find the *Imax*. After passing both these tests, the new most frequent item *Imax* is added to its parent node in the *MTree* in step B.4.1.1. In step B.4.1.2, two new blocks *Bmax* and *CrBmax* are formed; the search will continue through these new candidates. A similar process is applied to the counter-block in steps of B.4.1.4. In some steps, we indicate whether a block of a MASP is to be mined or not. As we mentioned earlier, after finding all MASPs, our approach enables us to find the ARM rules using a traditional approach within the data set retrieved by the query of each MASP. Not all blocks of MASPs need to be mined; if a MASP produces a block whose row counts do not satisfy the minimum support threshold, then its flag 't-mined' is set to 'false'.

Let's illustrate the algorithm with a sample generic data set given in Fig. 5. We set the minimum support = 20% ≡ 2 records, and association strength threshold = 30% ≡ 3 records. The initial *CanB* = {*D*} and *Can* = *CanB*(1) = *D*. According to step B.3, the distribution of each item in the candidate block is obtained as shown in Fig. 6. It is found that *Imax* = ($A_1 = V_{11}$); the candidate blocks formed at this level is shown in Fig. 7. The query to retrieve data for the block is Q_B = {SELECT A_2, A_3, A_4, A_5; FROM *D*;WHERE $A_1 = V_{11}$} and that of the counter-block is Q_{CB} = {SELECT A_1, A_2, A_3, A_4, A_5; FROM*D*; WHERE $A_1 \neq V_{11}$}. Consequently, the MASP and the candidate block will be updated as $M = (V_{11})$ and $CanB = \{B_{11}, CB_{11}\}$, respectively. In the next iteration, *Can* = B_{11} is processed. Similarly, *Imax* = (V_{32}) is found for the block B_{11} of MASP $M = (V_{11})$; consequently, $M = (V_{11}, V_{32})$ and $CanB = \{CB_{11}, B_{32}, CB_{32}\}$ are updated. The new candidate blocks are given in Fig. 8. The queries will be

$$Q_B = \{\text{SELECT } A_2, A_4, A_5; \text{FROM } D; \text{WHERE } A_1 = V_{11} \text{ and } A_3 = V_{32}\} \text{ and}$$
$$Q_{CB} = \{\text{SELECT } A_2, A_3, A_4, A_5; \text{FROM } D; \text{WHERE } A_1 = V_{11} \textit{ and } A_3 \neq V_{32}\}.$$

A1	A2	A3	A4	A5
V11	V21	V31	V41	V51
V11	V21	V32	V41	V53
V11	V22	V31	V42	V52
V11	V21	V32	V42	V52
V11	V22	V32	V43	V52
V11	V22	V32	V43	V52
V12	V22	V31	V44	V51
V12	V22	V31	V44	V51
V12	V23	V31	V45	V51
V12	V23	V32	V46	V51

Fig. 5. A sample data table (Donepudi, 2013).

Item	Frequency	Item	Frequency
A1=V11	6	A4=V42	2
A1=V12	4	A4=V43	2
A2=V21	3	A4=V44	2
A2=V22	5	A4=V45	1
A2=V23	2	A4=V46	1
A3=V31	5	A5=V51	5
A3=V32	5	A5=V52	4
A4=V41	2	A5=V53	1

Fig. 6. Item distribution of the candidate block (Donepudi, 2013).

A1	A2	A3	A4	A5
V11	V21	V31	V41	V51
V11	V21	V32	V41	V53
V11	V22	V31	V42	V52
V11	V21	V32	V42	V52
V11	V22	V32	V43	V52
V11	V22	V32	V43	V52
V12	V22	V31	V44	V51
V12	V22	V31	V44	V51
V12	V23	V31	V45	V51
V12	V23	V32	V46	V51

Fig. 7. Parent block, block B_{11} (upper right block) and counter-block CB_{11} (lower block) obtained at the initial step.

A3	A2	A4	A5
V32	V21	V41	V53
V32	V21	V42	V52
V32	V22	V43	V52
V32	V22	V43	V52
V31	V21	V41	V51
V31	V22	V42	V52

Fig. 8. Parent block, block B_{32} (upper right block) and counter-block CB_{32} (lower block) obtained by processing *Can* = B_{11}.

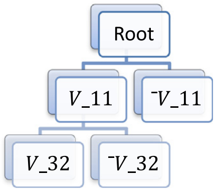

Fig. 9. *MTree* after processing $Can = B_{11}$.

At this stage, the *MTree* will have the children of ($A_1 = V_{11}$) and ($A_1 \neq V_{11}$) for the root and the children of ($A_3 = V_{32}$) and $A_3 \neq V_{32}$ to the parent node ($A_1 = V_{11}$) as shown Fig. 9.

3.2. Generating rules

In the realm of T-ARM, a rule is generated from a frequent pattern by testing association strength of all its possible combinations.

Lemma 3 (*Generating rules*). *Given a MASP M = (I_1, I_2, . . ., I_K), its rule set will be* $\{(I_1, I_2, \ldots, I_{K-1}) \to I_K; (I_1, I_2, \ldots, I_{K-2}) \to I_{K-1}; I_1 \to I_2\}$. *Simply, given a MASP* (V_{11}, V_{41}, V_{31}), *the corresponding rule set will be* $\{(V_{11}, V_{41}) \to V_{31}; (V_{11}) \to V_{41}\}$. *On the other hand, it is worth testing the forms (I_1,I_2, . . ., I_{k-1}) → (I_k, . . ., I_K) for association strength since they satisfy the minimum support condition; if the test passes, then the forms will be included in the rule set of the MASP.*

Proof. Let M_{K-1} = (I_1, I_2, . . ., I_{K-1}) be the MASP of the length (K – 1). By the algorithm, an item I_K is added to M_{K-1} if and only if P((M_{K-1}, I_K)) satisfies the minimum support (see the step B.3 and B.4.1.4) and P((I_K |M_{K-1})) satisfies the minimum confidence (see the step B.4.1 and B.4.1.4.1). Therefore, by definition, $M_{K-1} \to I_K$ will be a rule. Similarly, $M_{K-2} \to I_{K-1}$, and so on. □

The order of the items in the rules obtained by the MASP approach has significance whereas this is not the case for T-ARM. As an example, V_{11}, $V_{41} \to V_{31} \not\equiv V_{41}$, $V_{11} \to$ V_{31} in MASP realm, but V_{11}, $V_{41} \to V_{31} \equiv V_{41}$, $V_{11} \to V_{31}$ in the T-ARM realm.

A MASP may have negations such as $(V_{11}, V_{41}, \bar{V}_{52}, V_{31})$ and its corresponding rule would be $(V_{11}, V_{41}) \to (\bar{V}_{52}, V_{31})$. The negated item, here $\bar{V}_{52}$, serves as a pointer to the block where V_{31} is found as the *Imax*.

Lemma 4 (*MASP+ patterns*). *The combination of the MASP and the rules obtained from its block is also a rule. Let a MASP* (V_{11}, V_{41}, V_{31}) *and the rules* $\{V_{21}, V_{61} \to V_{51}; V_{22} \to V_{53}\}$ *from its block are given; then, the combined rule set will be* $\{((V_{11}, V_{41}, V_{31}), V_{21}, V_{61}) \to V_{51}; ((V_{11}, V_{41}, V_{31}), V_{22}) \to V_{53}\}$.

Proof. Let $M_K = (I_1, I_2, \ldots, I_K)$ be a MASP and the rule set $R(M_K) = \{A_1 \to C_1, \ldots, A_j \to C_j, \ldots\}$ obtained from the block of M_K; that is, the itemset (A_j, C_j) is preceded by the itemset M_K. Therefore, $R(M_K)$ can be written as $R(M_K) = \{(M_K, A_1) \to C_1, \ldots, (M_K, A_j) \to C_j, \ldots\}$. Each rule of $(M_K, A_j) \to C_j$ is a named as *MASP+* rule. □

The proposed approach provides the maximally associated items in an efficient way. In addition, the MASP tree can be utilized to generate rules from the power set of the items included in each MASP; we call this set 'sub-lattice of a MASP'. Theorem 1 in sequel emphasizes this property.

Theorem 1. *A sub-lattice of a MASP is composed of frequent items. This can be proven informally; let a MASP (A, B, C, D) be given. Using the closure property of the frequent items set from top to down, every 3-items sub-set of the pattern "ABCD" must be frequent; this gives us the sub-set {ABC, ABD, ACD, BCD}. Similarly, we can obtain 2-items sub sets as {AB, AC, AD, BC, BD, CD}, in which all items must be frequent. From these sub-sets, one can generate 3- and 2-items rules. This property can speed up the mining interesting patterns at the vicinity of the most associated sequential patterns.*

3.3. Complexity of the algorithm

Mainly, the algorithm generates 2 candidate nodes from a parent node at each iteration and finds the item that has the highest frequency in the data table of each node. At the worst case scenario in each iteration, the data table of a node will be split into almost half and the number of operations to find the *Imax* will be $M \times N = |D|$, where M and N denote the number of attributes (columns) and the number of records, respectively, and D denotes the initial data table. Therefore, considering the height of the tree to be $\log_2|D|$, the time complexity of creating a MASP Tree will be $O(|D| \log_2 |D|)$.

4. Experiments and results

In this section, we presented results of the experiments conducted on 5 datasets. These datasets are our own traffic accident data, Pumsb and Connect from the UCI datasets and Pumsb dataset (Bayardo, 2014), and Blog (Buza, 2014) and Diabetes (Strack et al., 2014). Outcome of proposed scheme is compared with the T-ARM scheme in terms of the size of the rules detected, the size of the data employed, and the elapsed time; we utilized publicly available software Rattle (version 3.1.0) (Williams, 2009) for preprocessing and R (version 3.1.1) (Team, 2008) for extraction of T-ARM rules; R employs Apriori for rule discovery. Two samples of the MASP tree and some rules obtained from these trees are provided.

Table 1 summarizes the comparison between MASP scheme and T-ARM scheme using 5 datasets of having 8000 records. We first fix the number of records to 8000 records and find the max number of attributes that both our MASP application and R can mine by reducing the number of attributes starting from the maximum number of attributes until the computer stops responding, which we consider as 'out of memory exception error' (OoMEE). As an example with the min-support of 10% and the min-confidence of 80%, R raised OoMEE until the number of attributes was reduced

Table 1
Comparison of MASP scheme and T-ARM scheme with smaller dataset.

	Min-support (%) →		10			0.1		
	Min-confidence (%) →		80	50	25	80	50	25
Traffic crash	MASP	Max # attributes	33			33		
		Longest rule size	19+	19+	19+	19+	32+	31+
		Time in sec	1.5	1.59	1.48	1.46	1.69	6.75
	T-ARM	Max # attributes	28	28	28	19	19	19
		Longest rule size	17	17	17	17	19	19
		Time in sec	28.79	31.78	32.3	30.86	36.03	38.92
Pumsb	MASP	Max # attributes	50			50		
		Longest rule size	20+	32+	32+	20+	43+	46+
		Time in sec	2.17	3.57	3.51	2.17	4.34	34.57
	T-ARM	Max # attributes	26	26	26	19	19	19
		Longest rule size	18	18	18	18	18	18
		Time in sec	24.89	25.75	26.25	42.18	46.15	48.82
Connect	MASP	Max # attributes	50			50		
		Longest rule size	28+	32+	32+	28+	36+	38+
		Time in sec	3.57	4.17	4.17	3.59	4.14	43.47
	T-ARM	Max # attributes	24	24	24	19	19	19
		Longest rule size	19	19	19	19	19	19
		Time in sec	110.99	120.57	122.61	76.08	100.5	108.05
Blog	MASP	Max # attributes	50			50		
		Longest rule size	6+	17+	17+	6+	47+	50+
		Time in sec	3.53	7.54	8.65	3.36	18.37	504.8
	T-ARM	Max # attributes	21	21	21	20	20	20
		Longest rule size	20	20	20	19	19	19
		Time in sec	24.91	25.27	26.8	122.02	129.6	132.46
Diabetes	MASP	Max # attributes	36			36		
		Longest rule size	21+	26+	26+	21+	28+	31+
		Time in sec	2.35	2.78	2.85	2.21	2.86	11.56
	T-ARM	Max # attributes	32	32	32	26	26	26
		Longest rule size	18	18	18	17	17	17
		Time in sec	5.21	5.69	6.05	10.08	10.67	11.6

to 28 when we use the traffic data. The MASP is capable of mining patterns up to the full size of 33 attributes. The longest rule size/max # attributes for MASP (R) was 19+/33, 19+/33, and 19+/33 (17/28, 17/28, and 17/28) items for 80%, 50%, and 25% confidence, respectively, when min-support of 10% is used. With 0.1% min-support, MASP's (R's) longest rule sizes were 19+/33, 32+/33, and 31+/33 (17/19, 19/19, and 19/19), respectively. In the table, the sign '+', as in "19+", is used next to the longest rule size for MASP; this means that the MASP tree generated a rule up to size 19 and some additional items of a pattern can be found when a T-ARM method is used to generate MASP+ patterns. Therefore, if a T-ARM finds rules of up to 5-items, then the longest rule size would be 19 + 5 = 24. As the table reads, the MASP rules are able to provide more detail associations (longer rule size) than the T-ARM scheme can provide in significantly less computation time. As an example, with a minimum support of 1% and minimum confidence of 25% the longest rule size detected by the MASP scheme from the traffic data is 32+ while the T-ARM extracts rules up to the size of 19. Similarly, it is 46+/50 (18/19) in the Pumsb, 38+/50 (19/19) in the Connect, 50+/50 (19/20) in the Blog, and 31+/36 (17/26) in the Diabetes, respectively. As the second observation, the MASP scheme is capable of finding rules from a bigger set of attributes compared to the T-ARM scheme. As an example with the same minimum thresholds of min-support = 0.1 and min-confidence = 0.25, the MASP scheme is able to mine rules out of 33, 50, 50, 50, and 36 attributes from data of the Traffic, the Pumsb, the Connect, the Blog, and the Diabetes while the T-ARM is capable of finding rules out of 19, 19, 19, 20, and 26 attributes using the same number of records for both schemes, respectively. An interesting case for the longest rule size of the MASP aroused with 80% min-confidence while discovering patterns from Blog data; it was 6+ for MASP versus 20 for T-ARM. The corresponding MASP was (A45, A44, A47, A46, A3, A49). Recall that the MASP scheme discovers the 'most associated' items in 'sequential' manner; that is, the item A45 was the most frequent item within the transactions of the dataset and the next most frequent one was A44 'given' A45, and so on. This property of the MASP scheme is important to

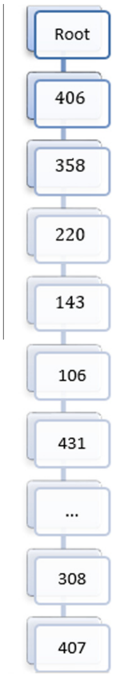

Fig. 10. A sample MASP tree obtained with min-support = 10% and min-confidence = 80% (the tree is rotated).

Table 2
Comparison of MASP and T-ARM schemes with larger dataset; # records are given as in (174436).

	Min-support (%) →		10			0.1		
	Min-confidence (%) →		80	50	25	80	50	25
TRAFFIC CRASH	MASP (174,436)	Max # attributes	33			33		
		Longest rule size	18+	19+	19+	18+	27+	28+
		Time in sec	25.99	28.58	30.03	28.58	35.97	140.59
	T-ARM (55,000)	Max # attributes	28	28	28	19	19	19
		Longest rule size	19	19	19	16	16	16
		Time in sec	67.85	72.55	76.87	120.56	144.48	154.77
Pumsb	MASP (49,046)	Max # attributes	50			50		
		Longest rule size	19+	29+	29+	19+	43+	45+
		Time in sec	11.34	18.47	18.98	11.5	24.67	168.2
	T-ARM (29,046)	Max # attributes	26	26	26	19	19	19
		Longest rule size	18	18	18	18	18	18
		Time in sec	55.69	57.16	57.19	65.03	70.17	74.34
Connect	MASP (67,558)	Max # attributes	50			50		
		Longest rule size	26+	28+	28+	26+	36+	36+
		Time in sec	31.45	32.72	27.21	30.81	46.41	201.5
	T-ARM (15,000)	Max # attributes	24	24	24	19	19	19
		Longest rule size	19	19	19	19	19	19
		Time in sec	127.55	142.33	162.95	76.81	94.44	103.38
Blog	MASP (52,397)	Max # attributes	50			50		
		Longest rule size	16+	16+	33+	16+	50+	50+
		Time in sec	11.34	18.47	18.98	11.5	24.67	168.42
	T-ARM (10,000)	Max # attributes	21	21	21	20	20	20
		Longest rule size	20	20	20	20	20	20
		Time in sec	98.06	103.87	110.88	178.36	298.87	332.33
Diabetes	MASP (37,770)	Max # attributes	36			36		
		Longest rule size	22+	23+	23+	22+	28+	30+
		Time in sec	12.82	15.18	16.27	12.62	14.57	95.31
	T-ARM (9740)	Max # attributes	32	32	32	26	26	26
		Longest rule size	18	18	18	17	17	17
		Time in sec	105.14	107.83	109.36	162.96	174.68	185.32

reach directly to the potentially interesting patterns. On the other hand, as a future work, the search may continue through the items which are within the p% of the most frequent one. This is another property of the MASP scheme that allows constraints (C-Constraints) at the time of searching for the frequent itemsets. Another property of the MASP scheme is that potentially interesting patterns with negations, as in “406, NOT 358, 220, NOT 143, …, 308, 407”, can be discovered. We presented the longest rule size with negations in the table (See Table 1).

A similar experiments were conducted with larger number of records for the same datasets. The results are provided in Table 2. For these experiments, we used the same attributes and find the largest sub-datasets from the available datasets that the MASP and T-ARM applications are capable of processing before giving OoMEE. As the table reads, the MASP scheme is capable of handling larger dataset than T-ARM scheme can handle; the sizes of the datasets from which patterns are discovered by the MASP (T-ARM) scheme were 174,436 (55,000), 49,046 (29,046), 67,558 (15,000), 52,397 (37,770), respectively. The length of the longest patterns discovered by the MASP scheme is bigger than that of the T-ARM scheme. The MASP patterns are extracted in significantly less computation time.

Fig. 10 shows a sample MASP tree obtained with min support = 10% and min-confidence = 80%. The description of the items is listed in Table 3 with their statistics. The tree has a single terminal branch from which 18 sequential rules can be derived. These rules are $\{(406) \rightarrow 358; (406, 358) \rightarrow 220; \ldots; (406, 358, \ldots, 308) \rightarrow 407\}$.

A second sample of the MASP tree which was obtained using min-support = 0.1% (8 crashes) and min-confidence = 25% is provided in Fig. 11. Since the size of its text gets too small when we

Table 3
Description of items for the sample MASP tree shown in Fig. 10 (L: Level, LF: Lift, C: Confidence in %, S: Support in %).

L	ID	Item name	LF	C	S
1	406	HighwayTypeCode = A			
2	358	ConstructionMaintenanceZone = 0	1	96	96
3	220	AlcoholPresent = Alcohol No	1	96	93
4	143	FourthHarmfulEvent = NOT REPORTED	1	94	87
5	106	ThirdHarmfulEvent = NOT REPORTED	1.06	93	81
6	431	HolidayType = Non-Holiday	1	92	74
7	360	RoadwayRelation = ON ROADWAY	1.06	91	68
8	405	RoadwayDeparture = no	1.15	92	62
9	23	FirstHarmfulEvent = MOTOR VEHICLE IN TRANSPORT	1.18	94	59
10	179	MostHarmfulEvent = MOTOR VEHICLE IN TRANSPORT	1.27	96	56
11	63	SecondHarmfulEvent = NOT REPORTED	1.27	91	51
12	321	VisionObscurements = NO OBSCUREMENTS	1.02	90	46
13	377	SurfaceCondition = DRY	1.07	89	41
14	335	Weather = CLEAR	1.18	87	36
15	238	DriverDistraction = NOT DISTRACTED	1.11	84	30
16	295	PrimaryContributingFactor = VIOLATIONS	1.11	84	25
17	2	Severity = COMPLAINT	1.10	84	21
18	308	SecondaryContributingFactor = MOVEMENT PRIOR TO CRASH	1.24	84	18
19	407	Intersection = no	0.97	80	14

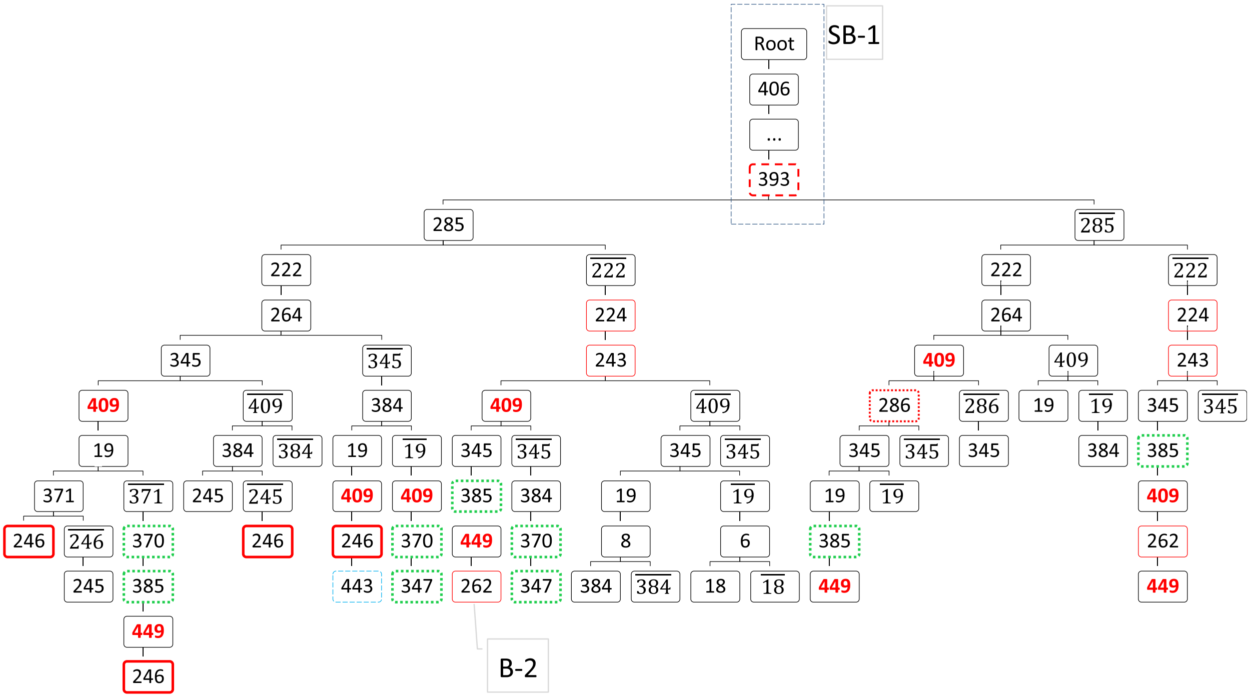
Root
SB-1
406
...
393
285 $\overline{285}$
222 $\overline{222}$ 222 $\overline{222}$
264 224 264 224
345 $\overline{345}$ 243 409 $\overline{409}$ 243
409 $\overline{409}$ 384 409 $\overline{409}$ $\overline{286}$ $\overline{286}$ 19 $\overline{19}$ 345 $\overline{345}$
19 384 $\overline{384}$ 19 $\overline{19}$ 345 $\overline{345}$ 345 $\overline{345}$ 345 $\overline{345}$ 345 384 385
371 $\overline{371}$ 245 $\overline{245}$ 409 409 385 384 19 $\overline{19}$ 19 $\overline{19}$ 409
246 $\overline{246}$ 370 246 246 370 449 370 8 6 385 262
245 385 443 347 262 347 384 $\overline{384}$ 18 $\overline{18}$ 449 449
449
246
B-2

Fig. 11. A sample MASP tree obtained with min-support = 0.1% (8 crashes) and min-confidence = 25% (not showing whole tree for clarity).

Table 4
Description of the visual formats for the MASP tree shown in Fig. 11.

Purpose	Color	Thickness/Font	Dash style	Sample
Driver behavior	Red	Thin	None	224 (Driver Condition = Inattentive)
Traffic condition	Red	**Thick**	None	246 (Movement Reason = Due to congestion)
Time	Blue	Thin	–	443 (Time of day = late afternoon)
Road	Green	Thick	Dot	385 (Surface type = black top)
Lighting	Red	Thick	Dot	286 (Dark continuous street light)
Location	Red	Bold	n/a	409 (Road number = 10,000)
Manner of collision	Red	Thick	–	393 (Manner collision = rear end)

add too many nodes, we did not show all branches for clarity. We format the nodes and their text for a better visual analysis; Table 4 gives the list of these visual formats. As an example, the MASP tree tells us that rear-end collision (with the node ID 393) is observed at the level of 19. The rule statistics obtained from the sub-branch SB-1 (406, …, 407, 393) are $S(406, \ldots, 407, 393) = 11\%$, $C(406, \ldots 407 \rightarrow 393) = P(393|406, \ldots, 407) = 79\%$ and $L(406, \ldots 407 \rightarrow 393) = 1.46$. The lift tells us that the association degree of rear-end collisions (393) with the factors (or conditions) listed within the antecedent (406, …, 407, 393) is higher than being independent from its antecedent by 0.46; notice that in the case of independency the Lift = 1.

As another example, the visual analysis of the branch B-2 in the same tree shows that the driver inattentive behavior (224), driver violation (243), and careless operation (262) seem effective at the locations 409 and 449 where the surface type is black-top (385). The lift, for example, at 449 is 6.4 and $P(449) = 10.33\%$; this means that the confidence $P(449|385, \ldots, 406) = 66.04\%$ is lifted by 5.4 compared to its frequency in the whole data set. That is, under the existence of the conditions listed on the sub-branch (385, …, 406), the frequency of observing 449 is elevated 6.4 times.

5. Conclusion and future work

Searching interesting patterns from huge data is one of the big challenges of this century. Association rule mining is the major field in attacking this challenge. In this field, consuming less resources in the detection of patterns is one of the major problems. The traditional exhaustive search approaches attack this problem by proposing efficient data structures to reduce the detection time while trading off the memory allocation complexity. Another major challenge in dealing with big data is extraction of useful information that can be used in decision making. Traditionally, mining useful information (or interesting patterns), involve two main steps: The first step is detection of frequent patterns and then pruning them to generate interesting ones in the second step. In addition, some constraints are applied for fine-tuning of search results.

In this paper, we attempted to address these two major challenges. The proposed approach searches the lattice utilizing a heuristic approach by imposing an interestingness constraint to detect the most associated item sequences. The proposed approach does not require a pruning step. The patterns detected by the proposed algorithm come in the form of 'most associated sequential pattern'. The MASPs can be combined with the patterns obtained by a traditional method to generate 'MASP+' patterns. An important feature of the MASPs is that less computational resources are required to produce long sequence of items that conveys potentially useful patterns. The proposed algorithm also generates a MASP tree as an outcome; this tree can be used for visual exploration of patterns. In addition, the proposed approach can lead to detection of frequent items under the sub-lattice of the MASP tree. This sub-lattice further is utilized to generate more *k*-items rules. The sub-lattice property of the MASP tree speeds up the mining of interesting patterns. Our experimental results confirms that the MASP scheme is capable of providing more detail, and potentially interesting, associations (longer rule size) than the T-ARM scheme can provide in significantly less computation time as well as finding rules from a bigger set of attributes and larger datasets compared to the T-ARM scheme.

As a future work, the proposed algorithm can be modified to find the patterns that are 'favorable' at some degree and to discover the rare rules as well. In addition, as an attempt to discover rare rules, the author plans to study the weakest associated pattern tree (WASP tree) and its sub-lattice; note that the rare rules are prone to have low statistical occurrence in nature. The WASPs would help discovering useful patterns that may occur rare in the nature. Our work can be extended to find interesting patterns utilizing different constraints on selection of items at each level. These constraints would be utility based, similarity distance based, mutual entropy, fuzzy relations, and so on. In addition, the interestingness measures can be applied to the MASP+ rules for further discovery of useful rules. Furthermore, the utilization of the proposed approach can be studied under the big data analytics. The current implementation requires the data to reside in the memory; this limits the amount of the data to be processed. This problem can be addressed by utilizing disk memory trading the computation time. On the other hand, rule mining tasks can naturally be performed in parallel; therefore, the algorithms can be modified for multi-thread computation or high-performance computation. Currently, we are working on multi-thread implementation of MASP, Apriori-TID, and Fp-Growth algorithms. We will share our results with the community soon. Another research direction would be the assessment of patterns obtained from MASP+ and its sub-lattice by the domain experts. The author has been working on discovering useful traffic accident patterns utilizing MASP+ and its sub-lattice together with the visual pattern analyzer of MASP tree.

Acknowledgments

The author would like to thank LA DOTD for continuous support in research and Harisha Donepudi and Eera Gupta for assisting in implementation of the algorithms proposed in this paper and in preparation of data for testing.

References

Agrawal, A., Srikant R. (1994). "Fast algorithms for mining association rules." In *Proceedings of the 20th VLDB conference*. Santiago, Chile.

Ashrafi, M. Z., Taniar, D., & Smith, K. (2004). A new approach of eliminating redundant association rules, database and expert systems applications. *Lecture Notes in Computer Science*, 465–474.

Ashrafi, M. F., Taniar, D., & Smith, K. (2005). Redundant association rules reduction techniques, AI 2005: Advances in artificial intelligence. *Lecture Notes in Computer Science*, 254–263.

Bayardo, R. Frequent itemset mining dataset repository. <http://www.cs.rpi.edu/~zaki/Workshops/FIMI/data/> (accessed October 2014).

Burdick, D., Calimlim, M., Gehrke J. (2001). "MAFIA: A maximal frequent itemset algorithm for transactional databases." In *Proceedings of the 17th international conference on data engineering*. Heidelberg, Germany (pp. 443–452).

Buza, K. (2014). Feedback prediction for blogs. In *Data analysis, machine learning and knowledge discovery* (pp. 145–152). Springer International Publishing.

Cheng, J., Ke, Y., & Ng, W. (2008). Effective elimination of redundant association rules. *Data Mining and Knowledge Discovery, 16*, 221–249.

Chen, C.-H., Lan, G.-C., Hong, Z.-P., & Lin, Y.-K. (2013). Mining high coherent association rules with consideration of support measure. *Expert Systems with Applications, 40*, 6531–6537.

Chen, C. M., & Tsai, P. (2004). Mining interesting association rules from customer databases and transaction databases. *Information Systems, 29*(8), 685–696.

Das, A., Ng, W., & Woon, Y. (2001). *Rapid association rule mining, CIKM'01*. Atlanta, Georgia, USA: ACM.

Deng, Z.-H. (2014). Fast mining top-rank-k frequent patterns by using node-lists. *Expert Systems with Applications, 41*, 1763–1768.

Denwattana, N., Getta, J. R. (2001). A parameterised algorithm for mining association rules, In *The 12th Australasian dataset conference* (pp. 45–51).

Do, T. D., Hui, S. C., Fong A. (2003). Mining frequent itemsets with category-based constraints." In *6th International conference discovery science, lecture notes in computer science*. Sapporo, Japan (pp. 76–86).

Donepudi, H. (2013). *"Detection of interesting traffic accident patterns by association rule mining"* (thesis). Baton Rouge, LA, USA: Computer Science and Engineering, Louisiana State University.

Guillet, F., & Hamilton, H. J. (2007). *Choosing the right lens: Finding what is interesting in data mining. Quality measures in data mining*. Springer.

Han, J., & Fu, y. (1999). Mining multiple-level association rules in large databases. *IEEE Transactions on Knowledge and Data Engineering, 11*(5), 798–805.

Han, J., Pei, J., Yin, Y., & MAO, R. (2004). Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data Mining and Knowledge Discovery, 8*, 53–87.

Heravi, M. J., Zaïane, O. R. (2010). A study on interestingness measures for associative classifiers. In *2010 ACM symposium on applied computing (SAC)*. Sierre, Switzerland.

Herrera, F., Carmona, C. J., González, P., & Jesus, M. J. (2011). An overview on subgroup discovery: Foundations and applications. *Knowledge and Information Systems, 29*, 495–525.

Hong, T., Horng, C., Wu, C., & Wang, S. (2009). An improved data mining approach using predictive itemsets. *Expert Systems with Applications, 36*, 72–80.

Jin, Z., Wang, R., Huang, H., & Hu, Y. (2014). Efficient interesting association rule mining based on causal criterion using feature selection. *Journal of Information & Computational Science, 11*(12), 4393–4403.

Király, A., Laiho, A., Abonyi, J., & Gyenesei, A. (2014). Novel techniques and an efficient algorithm for closed pattern mining. *Expert Systems with Applications, 41*, 5105–5114.

Klbsgen, W. (1996). Explora: a multipattern and multistrategy discovery assistant. *Advances in Knowledge Discovery and Data Mining*, 249–271.

Klemettinen, M., Heikki, M., Ronkainen, P., Toivonen, H. Verkamo, I. (1994). Finding interesting rules from large sets of discovered association rules. *CIKM'94 Proceedings of the third international conference on Information and knowledge management*. Gaithersburg, MD, USA (pp. 401–407).

Kotsiantis, S., & Kanellopoulos, D. (2006). Association rules mining: A recent overview. *GESTS International Transactions on Computer Science and Engineering, 32*(1), 71–82.

Lo, D., Khoo, S. C., & Wong, L. (2009). Non-redundant sequential rules — Theory and algorithm. *Information Systems, 34*, 438–453.

Marukatat, R. (2006). Structure-based rule selection framework for association rule mining of traffic accident data. In *Computational Intelligence and Security* (pp. 231–239). Springer.

Omiecinski, E. R. (2003). Alternative interest measures for mining associations in databases. *IEEE Transactions on Knowledge and Data Engineering, 15*(1), 57–69.

Pasquier, N., Bastide, Y., Taouil, R. Lakhal, L. (1998). Discovering frequent closed itemsets for association rules. In *ICDT'99, International conference on database theory, lecture notes in computer science* (pp. 398–416).

Pyun, Gwangbum., & Yun, Unil (2014). Mining top-k frequent patterns with combination reducing techniques. *Applied Intelligence, 41*, 76–98.

Rodríguez-González, A. Y., Martínez-Trinidad, J. F., & Carrasco-Ochoa, J. A. (2013). Mining frequent patterns and association rules using similarities. *Expert Systems with Applications, 40*, 6823–6836.

Sahar, Sigal. (2010). Interestingness measures – On determining what is. In *Data mining and knowledge discovery handbook*, pp. 603–612). Springer.

Strack, B., DeShazo, J. P., Gennings, C., Olmo, J. L., Ventura, S., Cios, K. J., et al. (2014). Impact of HbA1c measurement on hospital readmission rates: Analysis of 70,000 clinical database patient records. *BioMed Research International, 2014*.

Tan, P. N., Kumar, V., Srivastava J. (2002). Selecting the right interestingness measure for association patterns. In: *KDD'02 Proceedings of the eighth ACM SIGKDD international conference on knowledge discovery and data mining*. New York.

Team, R. (2008). Development core, R: A language and environment for, Vienna: R foundation for statistical computing.

Tseng, F.-C. (2013). Mining frequent itemsets in large databases: The hierarchical partitioning approach. *Expert Systems with Applications, 40*, 1654–1661.

Vo, B., Coenen, F., & Le, B. (2013). A new method for mining frequent weighted itemsets based on WIT-trees. *Expert Systems with Applications, 40*, 1256–1264.

Webb, G. I., & Yu, X. (2004). Discovering interesting association rules by clustering. In *Advances in artificial intelligence* (pp. 1055–1061). Springer.

Williams, G. (2009). Rattle: A data mining GUI for R. *The R Journal, 1*(2), 45–55.

Wojciechowski, M., & Zakrzewicz, M. (2002). Dataset filtering techniques in constraint-based frequent pattern mining. In *ESF exploratory workshop on pattern detection and discovery, lecture notes in computer science* (pp. 77–91). Springer.

Xu, Y., Li, Y., & Shaw, G. (2011). Reliable representations for association rules. *Data & Knowledge Engineering, 70*, 555–575.