SUGGETIONS TO IMPROVE THE EFFICIENCY OF ASSOCIATION RULESTECHNIQUES IN DATA MINING [+]

مقترح لتحسين كفاءة تقنيات قوانين الارتباط في تنقيب البيانات

Dr. Hillal Hadi Salih [*] **Dr. Soukaena Hassan Hashem** [**]

Shaimaa Akram Hassan [***]

Abstract:

Data mining is a process that uses a variety of data analysis tools to discover patterns and relationships that can be hidden among vast amount of data. From these businesses and organizations can make valid predictions about future trends in all areas of business. Association rule mining is a typical approach used in data mining domain for uncovering interesting trends, patterns and rules in large datasets

This research concentrates on one particular aspect to improve the efficiency of the association rules technique in data mining by the following:

1. **With databases have large set of items, it suggested to find the frequent itemsets by using depth search. In detail that done by finding the largest frequent itemset and then finding all the sub frequent itemsets from it. This proposal aim will speed up the process of finding frequent itemsets.**
2. **Classify the frequent itemsets to three classes closed frequent, maximal frequent and normal frequent. This proposal classification is important in the analysis process to support and strength the prediction with association rules.**

Keywords: association rules, frequent itemsets, closed frequent itemsets and maximal frequent itemsets

المستخلص:

تنقيب البيانات هي عملية استخدام أنواع مختلفة من طرق تحليل البيانات لاكتشاف الانماط والعلاقات المخبؤة وسط كم هائل من البيانات والتي من خلالها تتمكن الشركات والتعاملات المالية من التنبؤ المستقبلي المقبول في مختلف الجوانب.قوانين الارتباط هي احدى التقنيات المستخدمة في تنقيب البيانات لاكتشاف الصيغ والقوانين المهمة في قواعد البيانات الكبيرة في هذا البحث تم التركيز على تحسين كفاءة قوانين الارتباط من خلال:

[+] Received on 10/6/2008 , Accepted on 1/11/2009
[*] Prof. University of Technology
[**] Lec. University of Technology
[***] Medical -Technical Institute

- تم اقتراح ايجاد مجموعة العناصر المتكررة في قواعد البيانات التي تحتوي على عدد كبير من العناصر باستخدام تقنية البحث بالعمق لايجاد اطول مجموعة عناصر متكررة ومن خلالها يتم ايجاد كل العناصر المتكررة الجزئية بهدف زيادة السرعة.
- تصنيف مجموعة العناصر المتكررة الى ثلاث مجاميع وهي: المتكررة المغلقة والمتكررة العظمى والمتكررة الطبيعية. هذا التصنيف المقترح ذو اهمية في عملية التحليل لدعم وتقوية التنبؤ بواسطة قوانين الارتباط.

Introduction:

With the enormous amount of data stored in files, databases, and other repositories, it is increasingly important, if not necessary, to develop powerful means for analysis and perhaps interpretation of such data and for the extraction of interesting knowledge that could help in decision-making. *Data Mining*, also popularly known as *Knowledge Discovery in Databases* (KDD), refers to the nontrivial extraction of implicit, previously unknown and potentially useful information from data in databases. While data mining and knowledge discovery in databases (or KDD) are frequently treated as synonyms, data mining is actually part of the knowledge discovery process. In principle, data mining is not specific to one type of media or data. Data mining should be applicable to any kind of information repository. However, algorithms and approaches may differ when applied to different types of data. Indeed, the challenges presented by different types of data vary significantly. Data mining is being put into use and studied for databases, including relational databases, object-relational databases and object oriented databases, data warehouses, transactional databases, unstructured and semi structured repositories such as the World Wide Web, advanced databases such as spatial databases, multimedia databases, time-series databases and textual databases, and even flat files [1].

The kinds of patterns that can be discovered depend upon the data mining tasks employed. There are two types of data mining tasks: *descriptive data mining* tasks that describe the general properties of the existing data, and *predictive data mining* tasks that attempt to do predictions based on inference on available data. The data mining functionalities and the variety of knowledge they discover are briefly presented in the following list:

- **Characterization**: Data characterization is a summarization of general features of objects in a target class, and produces what is called *characteristic rules*.
- **Discrimination**: Data discrimination produces what are called *discriminate rules* and is basically the comparison of the general features of objects between two classes referred to as the *target class* and the *contrasting class*.
- **Association analysis**: Association analysis is the discovery of what are commonly called *association rules*. It studies the frequency of items occurring together in transactional databases, and based on a threshold called *support*, identifies the frequent item sets. Another threshold, *confidence*, which is the conditional probability that an item appears in a transaction when another item appears, is used to pinpoint association rules.

- **Classification**: Classification analysis is the organization of data in given classes. Also known as *supervised classification*, the classification uses given class labels to order the objects in the data collection. Classification approaches normally use a *training set* where all objects are already associated with known class labels. The classification algorithm learns from the training set and builds a model. The model is used to classify new objects.

- **Clustering**: Similar to classification, clustering is the organization of data in classes. However, unlike classification, in clustering, class labels are unknown and it is up to the clustering algorithm to discover acceptable classes [2, 3].

Association Rules Algorithm:

The efficient discovery of such rules has been a major focus in the data mining research community. Many algorithms and approaches have been proposed to deal with the discovery of different types of association rules discovered from a variety of databases. However, typically, the databases relied upon are alphanumerical and often transaction-based. The problem of discovering association rules is to find relationships between the existence of an object (or characteristic) and the existence of other objects (or characteristics) in a large repetitive collection. Such a repetitive collection can be a set of transactions for example, also known as the market basket. Typically, association rules are found from sets of transactions, each transaction being a different assortment of items, like in a shopping store ({milk, bread, etc}). Association rules would give the probability that some items appear with others based on the processed transactions, for example milk →bread [50%], meaning that there is a probability 0.5 that bread is bought when milk is bought. Essentially, the problem consists of finding items that frequently appear together, known as frequent or large item-sets.

The problem is stated as follows, see table 1, Let $I = \{i1, i2, ...im\}$ be a set of literals, called items. Let D be a set of transactions, where each transaction T is a set of items such that $T \subseteq I$. A unique identifier *TID* is given to each transaction. A transaction T is said to contain X, a set of items in I, if $X \subseteq T$. An *association rule* is an implication of the form "$X \Rightarrow Y$", where $X \subseteq I$, $Y \subseteq I$, and $X \cap Y = \varnothing$. The rule $X \Rightarrow Y$ has a *support s* in the transaction set D is s% of the transactions in D contain $X \cup Y$. In other words, the support of the rule is the probability that X and Y hold together among all the possible presented cases. It is said that the rule $X \Rightarrow Y$ holds in the transaction set D with *confidence c* if c% of transactions in D that contain X also contain Y . In other words, the confidence of the rule is the conditional probability that the consequent Y is true under the condition of the antecedent X. The problem of discovering all association rules from a set of transactions D consists of generating the rules that have a *support* and *confidence* greater that given thresholds. These rules are called *strong rules* [4, 5].

Table 1: A model of a simple transaction database

TID	**Items**

001	A C D
002	B C E
003	A B C E
004	B E

The Proposed System:

To have a clear understanding of the present study it will be organized to the following points:

1 A new strategy for finding frequent itemsets from a transactional database has been proposed. The algorithm represents that strategy deals efficiently with very large itemsets in database by representing the itemsets in a tree, traversing this tree in depth-search manner, getting the last most left itemset and considering it the largest frequent itemset then checking if its support pass the minimum support threshold considered then all of its children itemsts will be generated to constitute the set of all frequent itemsets, otherwise one item will be deleted each time until the largest frequent itemset will be found. This will minimize the amount of times that needed to scan the databases. For example if the set of itemsets are A,B,C,D,E and the largest frequent itemset is A,B,C,D. in current study the set of itemsets are represented in a tree then this tree is traversed in depth manner getting the itemset in the left most side (ABCDE) checking if it is frequent or not which it is not frequent in this example see figure[1]. So, one item will be deleted, and then checking the remaining items (ABCD) which it is frequent. In this case, the database is scanned two times. All the largest frequent itemset children are generated which they are also frequent without checking. In the current example, the children of the largest frequent itemset are (A, B,C,D,AB,AC,AD,BC,BD,CD,ABC,ABD,ACD,BCD).

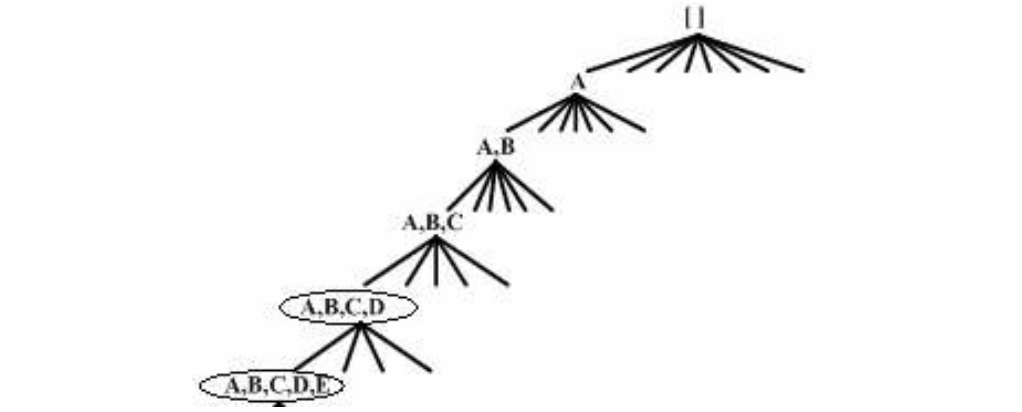


Figure: (1) the depth search manner.

In previous algorithms such as Apriori the tree is scanned in breadth manner, so in the first time the set of 1-itemset is checked so the item A is checked and then B and so on, while in the second phase, the set of 2-itemset is checked so the itemsets AB,AC,AD,AE,…etc, is checked and so on until reaching the set of n-itemset (n is the number of items). In that case, the numbers of database scans equal to 2^N .[6]

2 After finding the set of all frequent itemsets the support of each frequent itemsets will be found.
3 Then the discovered frequent itemset will be classified to three classes: the first class called closed frequent itemsets which means that the itemsets was found in the other frequent itemsets but not with the same support , the second class called maximal frequent itemsets which means that the itemsets was not found in the other frequent itemsets and the third class called normal frequent where the itemsets were neither closed nor maximal.
4 After the classification step the association rules will be found using the known association rules procedure [6].
5 Finally, analyzing the resulting association rules according to the types of frequents itemsets classes.

1- The Algorithms of the Proposed System:

Here in this section the preposed system will be explained the in three phases the first one explain all the algorithms of the proposed system to find the largest frequent itemsets, all frequent itemsets, closed frequent itemsets and maximal frequent itemsets. The second one displays the algorithm that used to generate the association rules. While the third phase explains the analysis of the generated association rules.

1.1- The First Phase:
The General Proposed Algorithm*:*

Find the set of association rules in database according to the set of **MFI**, **CFI**, and **FI**.

Input: The database **DB**, Minimum support **Min_sup**, Minimum confidence **Min_conf**.

***Output*:** The set of association rules support the database with concern maximal and closed itemsets.

First step: Find all frequent itemsets by using depth_first search strategy.

Second step: Count the support of each frequent itemset.

Third step: Find the set of closed frequent itemsets.

Fourth step: Find the set of maximal frequent itemsets.

Fifth step: Find the set of association rules.

The Depth Search Algorithm*:*

Find the largest frequent itemset by using depth_first search strategy.

Input: The set of initial items in database **Initial**, minimum support **Min_sup**.

Output: The largest frequent itemset.

```
{
    initialize Temp to empty string
    initialize Current_item to the first item in Initial
    While not end of Initial
    {
        Temp = Temp U Current_item
        Current_item = next item in Initial
    }
    label:
   count the support of Temp
    If Temp.support >= Min_sup Then     the largest frequent itemset = Temp
     Else
        If Initial <> [ ] Then
        {
            largest frequent = largest frequent after delete most right item.
            Go to label  ( Temp= large frequent, Min_sup)
        }
}
```

The Frequent Itemsets Finding Algorithm*:*

Find the set of all frequent itemsets in database.

***Input*:** The largest frequent itemset in database.

***Output*:** The set of all frequent itemsets FI from the largest one in database.

```
{
    Initialize L to the length of largest frequent
    Find the maximum number of the frequent itemsets that must not be
    exceeded  M=2^L
    initialize I to 1
```

```
        While L>0
        {
                find from largest frequent  all children of level I and store them in FI
                I=I+1; L=L-1
        }
}
```

The Frequent Itemsets support Algorithm:

Count the support of each frequent itemset in FI.

***Input*:** The database **DB**, the set of frequent itemsets **FI**.

***Output*:** The set of frequent itemsets associated with their supports.

```
{
        initialize L to the length of FI
        For I: 1 to L
        {
                Pass the DB count the Support of FI [I]
                FI_sup[I] = [FI[I], Support]
        }
}
```

The Closed Frequent Itemsets Finding Algorithm*:*

Find the set of closed frequent itemsets from the set of frequent itemsets which associated with their supports.

***Input*:** The set of frequent itemsets associated with their supports.

***Output*:** The set of closed frequent itemsets.

```
{
        initialize L to the length of FI_sup
        For I: 1 to L
        {
                Current_item= FI_sup[I,Support]
                If Current_item was found in the other frequent itemsets but not with
                 the same Support Then add Current_item to CFI
        }
}
```

The Maximal Frequent Itemsets Finding Algorithm:

Find the set of maximal frequent itemsets from the set of frequent itemsets.

***Input*:** The set of frequent itemsets **FI**.

***Output*:** The set of maximal frequent itemsets.

```
{
```

```
        initialize L to the length of FI
        For I:= 1 to L
        {
                Current_item= FI[I]
                If Current_item was not found in any other frequent itemset Then add
                Current-item into MFT
        }
}
```

3.1.2- The Second Phase:

The Association Rules Finding Algorithm:

The second phase in discovering association rules based on all frequent i-itemsets, which have been found in the first phase using the *proposed algorithms* or some other similar algorithm, is relatively simple and straightforward. For a rule that implies $\{x_1, x_2, x_3\} \rightarrow x_4$, it is necessary that both itemset $\{x_1, x_2, x_3, x_4\}$ and $\{x_1, x_2, x_3\}$ are frequent. Then, the confidence of the rule is computed as the quotient of supports for the itemsets *confidence* = *support* $(x_1, x_2, x_3, x_4\}$ / *support* $(x_1\ x_2, x_3)$. Strong association rules are rules with a confidence value *c* above a given threshold. In this proposed system we use the traditional association rules generation algorithm used in apriori algorithm. Which is written as follows [6]:

```
BUILDASSOCIATIONS(f, mincon):
 1  rules ← NIL
 2  for k ∈ fis, itemset ∈ f_k, i ← 1 to length(itemset):
 3          for c ∈ COMBINATIONS(itemset, i):
 4                  lhs ← c
 5                  rhs ← NIL
 6                  for k ∈ itemset: if k ∉ c: rhs.append(k)
 7                  confidence ← 100 × SUPPORT(itemset, f) / SUPPORT(lhs, f)
 8                  if confidence > mincon:
 9                          support ← SUPPORT(itemset, f)
10                          rules.append(lhs → rhs(support, confidence))
11  return rules
```

1.3- The Third Phase:

This is the last phase in the proposed system. A new procedure that describes the results of the second phase has been preposed: by analyzing the obtained association rules and introducing the most important groups of rules to support the problem then determining the frequent itemsets that building the association rules if they were normal, closed or maximal frequent itemsets. For example the following association rule,

ABC ^ BCD ⟶ ABCEF (90%).

This means that the existane of closed frequent itemsets ABC and maximal frequent itemsets BCD will predicate a 90% normal frequent one.

4- The Implementation:

To explain the proposed system and prove it is quality it will be implemented using visual basic programming language as follows:

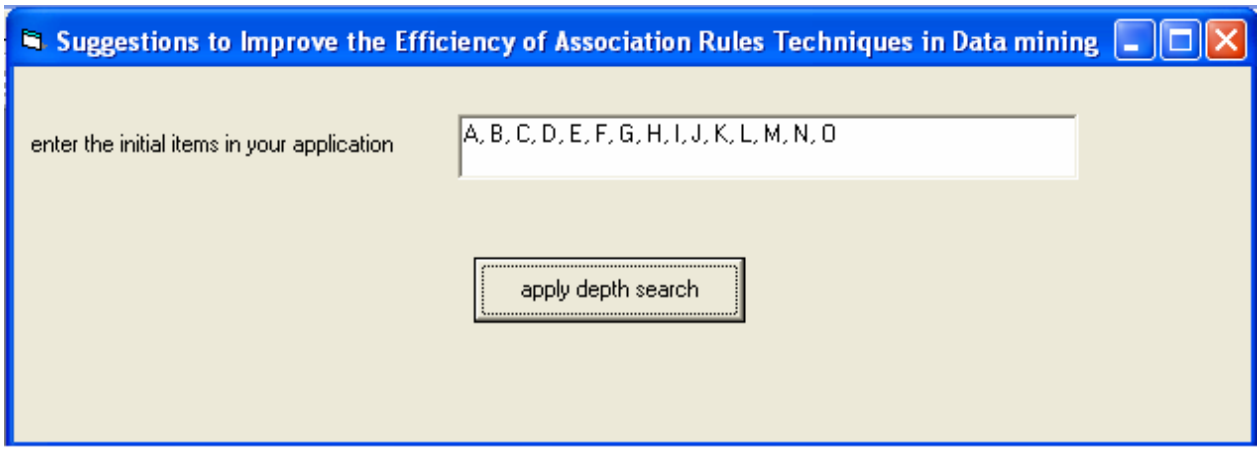


Figure (2): the main window which is display the initial items of the association rules.

Figure (2) displays the initial items and by clicking the button called apply depth search a new window will be appeared see figure (3).

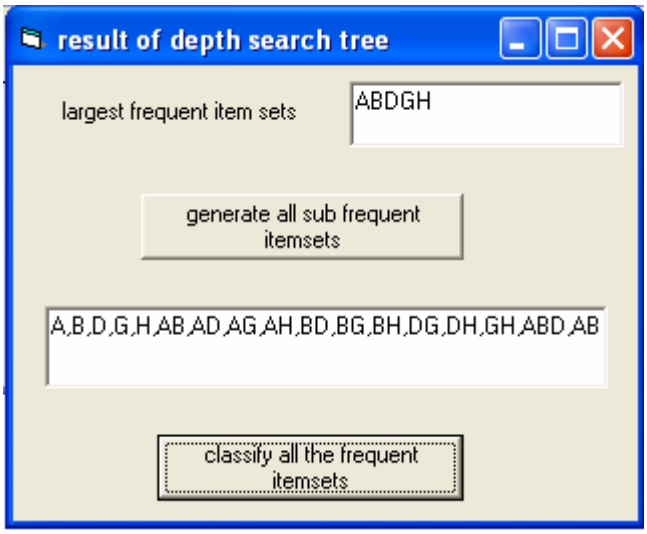


Figure (3): the window which is display the largest frequent itemsets and all subfrequent itemsets.

Figure (3) displays the largest itemsets and by clicking the button called generate all subfrequent itemsets the second textbox will display all the generated frequents itemsets. The second button called classify the frequent itemsets when it is clicked new window will appear see figure (4).

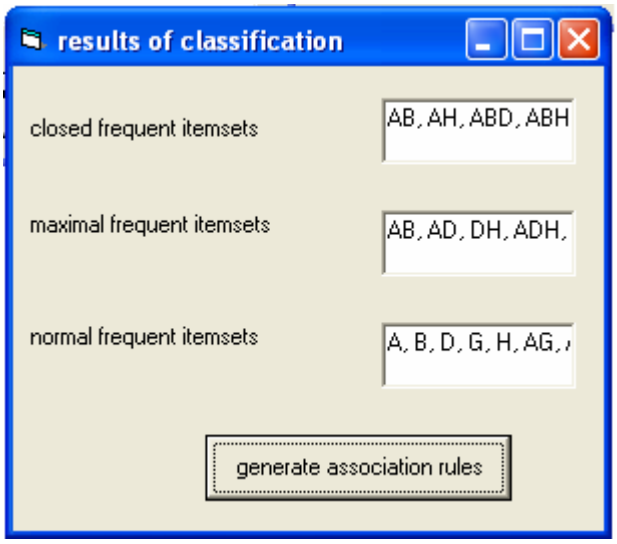
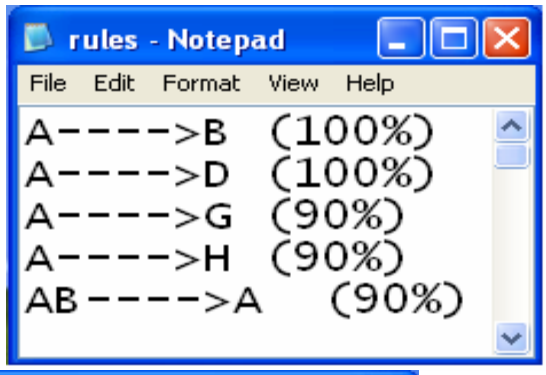


Figure (4): the window which is display the three classes of frequent itemsets.

Figure (4) displays the three classes of the frequent itemsets and by clicking the button called generate association rules, the rules will be generated and stored in a file called rule.txt. and the analysis.txt file will contain the analysis of the rules generated in rule.txt file. See figure (5) and figure (6).



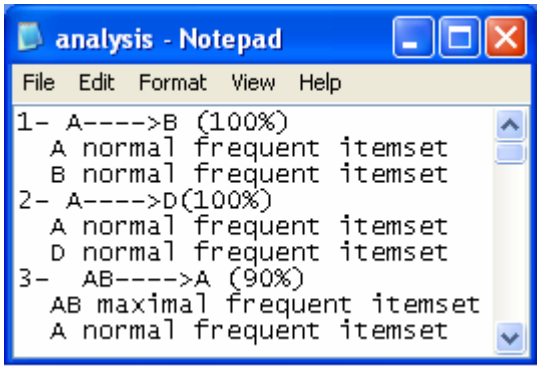


Figure (4): rule file.

Figure (5): analysis file.

5- Conclusions:

From the present study the following conclusion will be summarized:

1- Using depth search technique to find the largest frequent itemset is more efficient than other techniques that have been used in other algorithm such as

apriori to keep time. This is done by passing the database only one time to check if the largest one is frequent or not.

2- Generating all the frequent itemsets from the largest itemset is also basic factor to keep the time. This is done by extracting the frequent itemsets without made any pass to the database but in aprior algorithm the database is scanned for each itemset to check if it is frequent or not.

3- The classification process to classify the frequent itemsets to three classes normal, maximal and closed, is the base stone to strength the analysis process. Since we know the analysis is an important stage in all the applications depended on the association rules.

6- References:

1- M. S. Chen, J. Han, and P. S. Yu. **"Data mining: An overview from a database perspective"**. *IEEE Trans. Knowledge and Data Engineering, 8:866-883*, 1996.

2- U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy. **"Advances in Knowledge Discovery and Data Mining"**. *AAAI/MIT Press*, 1996.

3- J. Han and M. Kamber. **"Data Mining: Concepts and Techniques"**. *Morgan Kaufmann*, 2000.

4- Mitra S., and Ahharya T., **"Data Mining Multimedia, Soft Computing, and Bioinformatics"**, *John Wiley and Sons, Inc*., 2003.

5- Mohammadian M., **"Intelligent Agent for DM and Information Retrieval"**, *Idea Group Publisher*, 2004.

6- Agrawal R., Mannila H., Srikant R., Toivenen H., and Verkamo A., **"Fast Discovery of Association Rules"**, Santiago de Chile, 1996.