Concise Papers

Mining Multiple-Level Association Rules in Large Databases

Jiawei Han, *Member*, *IEEE Computer Society*, and Yongjian Fu, *Member*, *IEEE Computer Society*

Abstract—A top-down progressive deepening method is developed for efficient mining of multiple-level association rules from large transaction databases based on the Apriori principle. A group of variant algorithms is proposed based on the ways of sharing intermediate results, with the relative performance tested and analyzed. The enforcement of different interestingness measurements to find more interesting rules, and the relaxation of rule conditions for finding "level-crossing" association rules, are also investigated in the paper. Our study shows that efficient algorithms can be developed from large databases for the discovery of interesting and strong multiple-level association rules.

Index Terms—Data mining, knowledge discovery in databases, association rules, multiple-level association rules, algorithms, performance.

◆

1 INTRODUCTION

MINING of association rules from large data sets has been a focused topic in recent data mining research [1], [3], [4], [2], [9], [8], [10], [11], [14], [22], [15], [16], [18], [17], [19], [20], [21], [24]. Many applications at mining associations require that mining be performed at multiple levels of abstraction. For example, besides finding *80 percent of customers that purchase* milk *may also purchase* bread, it is interesting to allow users to "drill-down" and show that *75 percent of people buy* wheat bread *if they buy* 2 percent milk. The association relationship in the latter statement is expressed at a lower level of abstraction but carries more specific and concrete information than that in the former. Therefore, a data mining system should provide efficient methods for mining multiple-level association rules.

To explore multiple-level association rule mining, one needs to provide: 1) data at multiple levels of abstraction, and 2) efficient methods for multiple-level rule mining. The first requirement can be satisfied by providing concept taxonomies from the primitive level concepts to higher levels. In many applications, the taxonomy information is either stored implicitly in the database, such as, "*Wonder wheat bread is a wheat bread which is in turn a bread*," or provided by experts or users, such as, "*Freshman is an undergraduate student*," or computed by applying some cluster analysis methods [12]. With the recent development of data warehousing and OLAP technology, arranging data at multiple levels of abstraction has been a common practice [6]. Therefore, in this study, we assume such concept taxonomies exist, and our study is focused at the second requirement, the efficient methods for multiple-level rule mining.

There are several possible directions to explore efficient mining of multiple-level association rules.

- *J. Han is with the School of Computing Science and the Intelligent Database Systems Research Laboratory, Simon Fraser University, Burnaby, British Columbia, Canada V5A 1S6. E-mail: han@cs.sfu.ca.*
- *Y. Fu is with the Department of Computer Science, University of Missouri–Rolla, Rolla, MO 65409. E-mail: yongjian@umr.edu.*

One choice is the direct application of the existing single-level association rule mining methods to multiple-level association mining. For example, one may apply the Apriori algorithm [3] to examine data items at multiple levels of abstraction under the same minimum support and minimum confidence thresholds. This direction is simple, but it may lead to some undesirable results. First, large support is more likely to exist at high levels of abstraction. If one wants to find strong associations at relatively low levels of abstraction, the minimum support threshold must be reduced substantially, which may lead to the generation of many uninteresting associations at high or intermediate levels. Second, since it is unlikely to find many strong association rules at a primitive concept level, mining strong associations should be performed at a rather high concept level, which is actually the case in many studies [1], [3]. However, mining association rules at high concept levels may often lead to the rules corresponding to prior knowledge and expectations [15], such as "milk ⇒ bread", (which could be common sense), or lead to some uninteresting attribute combinations if the minimum support is allowed to be rather small, such as "toy ⇒ milk", (which may just happen together by chance).

These observations lead us to examine the second choice: applying different minimum support thresholds, and possibly different minimum confidence thresholds as well, at mining associations at different levels of abstraction. Especially, we explore the most typical case: progressively reducing the minimum support thresholds at lower levels of abstraction. This leads to mining interesting association rules at multiple concept levels, which may not only discover rules at different levels, but may also have high potential to find nontrivial, informative association rules because of its flexibility for focusing the attention to different sets of data and applying different thresholds at different levels.

In this study, a progressive deepening method is developed by extension of the Apriori algorithm for mining single-level association rules [3]. The method first finds frequent data items at the topmost level and then progressively deepens the mining process into their frequent descendants at lower concept levels.

One important assumption that we have made in this study is to explore only the descendants of the frequent items, since we consider if an item occurs rarely, its descendants will occur even less frequently and, thus, are uninteresting. Efficient level-shared mining can be explored based on this assumption. One may wonder whether this may miss the associations containing the items which are frequent according to the reduced minimum support threshold at low level but whose ancestors are infrequent. This concern can be addressed with a minor modification to our method as follows. First, we adopt two minimum support thresholds at a high level: one for cutting off infrequent items at the current level and the other (called *level passage threshold*) for passing down relatively frequent items to its lower level. Second, a user may slide down the level passage threshold at high levels to allow the passage of the descendants of the "subfrequent" items down to lower levels. One may even slide it down to the same minimum support threshold as that at the lowest level and thus open the venue for the descendants of all the items. By adding this mechanism, the algorithm presented in this paper will give users the flexibility to control the mining process as well as the chance to reduce the meaningless associations to be generated.

The necessity for mining multiple-level association rules or using taxonomy information at mining association rules has also been observed by other researchers, e.g., [23]. A major difference between our study and theirs is that they use the same support

TABLE 1
A `sales_transaction` Table

transaction_id	bar_code_set
351428	$\{17325, 92108, 55349, 88157, \dots\}$
...	$\{\dots, \dots\}$

threshold across all the levels [23], whereas we use different support thresholds for different levels of abstraction. As discussed above, using a single support threshold will allow many uninteresting rules to be generated together with the interesting ones if the threshold is rather low, but will disallow many interesting rules to be generated at low levels if the threshold is rather high. Therefore, in their study, substantial efforts have been made on how to identify and remove the redundant rules across different levels.

In our study, besides the investigation of several optimization techniques by exploring level-shared mining, two interestingness measures for filtering uninteresting rules are also studied, which subsumes the one used in [23].

The paper is organized as follows. In Section 2, the concepts related to multiple-level association rules are introduced. In Section 3, a method for mining multiple-level association rules in large data sets is proposed and a set of variant algorithms for mining multiple-level association rules are introduced, with their relative efficiency analyzed. In Section 4, a performance study is conducted on different kinds of data distributions which identify the conditions for the selection of algorithms. Section 5 is on mining cross-level association rules. Section 6 discusses the filtering of uninteresting association rules. The study is concluded in Section 7, along with some future work.

2 MULTIPLE LEVEL ASSOCIATION RULES

We assume that the database contains: 1) an item data set which contains the description of each item in $\mathcal{I}$ in the form of $\langle A_i, description_i \rangle$, where $A_i \in \mathcal{I}$, and 2) a transaction data set, $\mathcal{T}$, which consists of a set of transactions $\langle T_i, \{A_p, \dots, A_q\} \rangle$, where T_i is a transaction identifier and $A_i \in \mathcal{I}$ (for $i = p, \dots, q$).

Definition 2.1. *A pattern or an itemset, A, is one item A_i or a set of conjunctive items $A_i \wedge \cdots \wedge A_j$, where $A_i, \dots, A_j \in \mathcal{I}$. The **support** of a pattern A in a set S, $\sigma(A/S)$, is the number of transactions (in S) which contain A versus the total number of transactions in S. The **confidence** of $A \Rightarrow B$ in S, $\varphi(A \Rightarrow B/S)$, is the ratio of $\sigma(A \wedge B/S)$*

versus $\sigma(A/S)$, i.e., the probability that pattern B occurs in S when pattern A occurs in S.

To find relatively frequently occurring patterns and reasonably strong rule implications, a user or an expert may specify two thresholds: *minimum support*, σ', and *minimum confidence*, φ'. Notice that, for finding multiple-level association rules, different minimum support and/or minimum confidence can be specified at different levels.

Definition 2.2. *A pattern A is **frequent** in set S at level l if the support of A is no less than its corresponding minimum support threshold σ'_l. A rule "$A \Rightarrow B/S$" is **strong** if, for a set S, each ancestor (i.e., the corresponding high-level item) of every item in A and B, if any, is frequent at its corresponding level, "$A \wedge B/S$" is frequent (at the current level), and the confidence of "$A \Rightarrow B/S$" is no less than minimum confidence threshold at the current level.*

The definition implies a filtering process which confines the patterns to be examined at lower levels to be only those with large supports at their corresponding high levels (and thus avoids the generation of many meaningless combinations formed by the descendants of the infrequent patterns). For example, in a `sales_transaction` data set, if milk is a frequent pattern, its lower level patterns, such as 2 percent milk, will be examined; whereas if fish is an infrequent pattern, its descendants, such as salmon, will not be examined further.

Based on this definition, the idea of mining multiple-level association rules is illustrated below.

Example 2.1. Let the query be to find multiple-level strong associations in the database in Table 1 for the purchase patterns related to *category, content*, and *brand* of the foods which can only be stored for less than three weeks.

The relevant part of the `sales_item` description relation in Table 2 is fetched and generalized into a generalized `sales_item` description table, as shown in Table 3, in which each tuple represents a generalized item which is the merge of a group of tuples which share the same values in the interested attributes. For example, the tuples with the same *category, content*, and *brand* in Table 2 are merged into one, with their *bar codes* replaced by a bar_code set. Each group is then treated as an atomic item in the generation of the lowest level association rules. For example, the association rule generated regarding milk will be only in relevance to (at the low concept levels) *brand* (such as Dairyland) and *content* (such as 2 percent) but not to *size, producer*, etc.

TABLE 2
A `sales_item` (Description) Relation

bar_code	category	brand	content	size	storage_pd	price
17325	milk	Foremost	2%	1 (ga.)	14 (days)	$3.89
...	...	...	...	...	...	...

TABLE 3
A Generalized `sales_item` Description Table

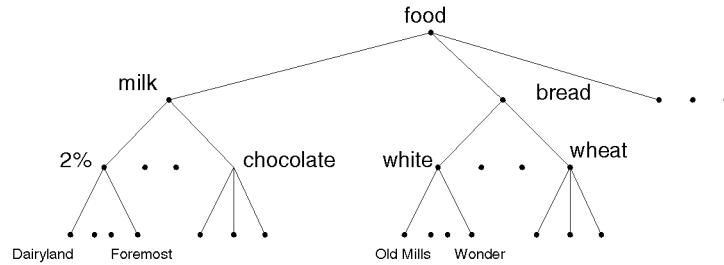| GID	bar_code_set	category	content	brand
112	$\{17325, 31414, 91265\}$	milk	2%	Foremost
...	$\{\dots, \dots\}$	...	...	...

Fig. 1. A taxonomy for the relevant data items.

The taxonomy information is provided implicitly in Table 3. Let *category* (such as "milk") represent the first-level concept, *content* (such as "2 percent") for the second-level one, and *brand* (such as "Foremost") for the third-level one. The table implies a concept tree like Fig. 1.

The process first discovers frequent patterns and strong association rules at the top-most concept level. Let the minimum support at this level be 5 percent and the minimum confidence be 50 percent. One may find: A set of single frequent items (each called a frequent 1-itemset, with the support in parentheses): "bread (25 percent), vegetable (30 percent), ...," a set of pair-wised frequent items (each called a frequent 2-itemset): "⟨vegetable, bread (19 percent)⟩, ...}," etc., and a set of strong association rules, such as, " bread ⇒ vegetable (76 percent), ..."

At the second level, let the minimum support be 2 percent and the minimum confidence be 40 percent. One may find frequent 1-itemsets: "lettuce (10 percent), wheat bread (15 percent), ...," and frequent 2-itemsets: "⟨lettuce, wheat bread (6 percent)⟩, ...," and strong association rules: "lettuce ⇒ wheat bread (60 percent), ...," etc.

The process repeats at even lower concept levels until no frequent patterns can be found.

3 A METHOD FOR MINING MULTIPLE-LEVEL ASSOCIATION RULES

A method for mining multiple-level association rules is introduced in this section, which uses a hierarchy-information encoded transaction table instead of the original transaction table. This is because a data mining query is usually in relevance to only a portion of the transaction database, such as *food*, instead of all the items. It is beneficial to first collect the relevant set of data and then work repeatedly on the task-relevant set. Encoding can be performed during the collection of task-relevant data and, thus, there is no extra "encoding pass" required. Besides, an encoded string, which represents a position in a hierarchy, requires fewer

bits than the corresponding object-identifier or bar-code. Therefore, it is often beneficial to use an encoded table, although our method does not rely on the derivation of such an encoded table because the encoding can always be performed on the fly.

To simplify our discussion, an abstract example which simulates the real life example of Example 2.1 is analyzed as follows.

Example 3.1. As stated above, the taxonomy information for each (grouped) item in Example 2.1 is encoded as a sequence of digits in the transaction table $T[1]$ (Table 4). For example, the item '2 percent Foremost milk' is encoded as '112' in which the first digit, '1', represents 'milk' at level-1, the second, '1', for '2 percent (milk)' at level-2, and the third, '2', for the brand 'Foremost' at level-3. Similar to [3], repeated items (i.e., items with the same encoding) at any level will be treated as one item in one transaction.

The derivation of the frequent item sets at level 1 proceeds as follows. Let the minimum support be 4 transactions (i.e., $minsup[1] = 4$).[1] The level-1 frequent 1-itemset table $\mathcal{L}[1,1]$ can be derived by scanning $T[1]$, registering support of each generalized item, such as $1**, \ldots, 4**$, if a transaction contains such an item, and filtering out those whose accumulated support count is lower than the minimum support. $\mathcal{L}[1,1]$ is then used to filter out: 1) any item which is not frequent in a transaction, and 2) the transactions in $T[1]$ which contain only infrequent items. This results in the filtered transaction table $T[2]$ of Fig. 2. Moreover, since there are only two entries in $\mathcal{L}[1,1]$, the level-1 frequent 2-itemset table $\mathcal{L}[1,2]$ may contain only one candidate item $\{1**, 2**\}$, which is supported by four transactions in $T[2]$.

According to the definition of ML-association rules, only the descendants of the frequent items at level-1 (i.e., in $\mathcal{L}[1,1]$) are considered as candidates in the level-2 frequent 1-itemsets. Let $minsup[2] = 3$. The level-2 frequent 1-itemsets $\mathcal{L}[2,1]$ can be derived from the filtered transaction table $T[2]$ by accumulating the support count and removing those whose support is smaller than the minimum support, which results in $\mathcal{L}[2,1]$ of Fig. 3. Similarly, the frequent 2-itemset table $\mathcal{L}[2,2]$ is formed by the combinations of the entries in $\mathcal{L}[2,1]$, together with the support derived from $T[2]$, filtered using the corresponding threshold. Likewise, the frequent 3-itemset table $\mathcal{L}[2,3]$ is formed by the combinations of the entries in $\mathcal{L}[2,2]$.

Finally, $\mathcal{L}[3,1]$ and $\mathcal{L}[3,2]$ at level 3 are computed in a similar process, with the results shown in Fig. 3. The computation terminates since there is no deeper level requested in the query. Note that the derivation also terminates when an empty frequent 1-itemset table is generated at any level.

TABLE 4
Encoded Transaction Table: $T[1]$

TID	Items
T_1	$\{111, 121, 211, 221\}$
T_2	$\{111, 211, 222, 323\}$
T_3	$\{112, 122, 221, 411\}$
T_4	$\{111, 121\}$
T_5	$\{111, 122, 211, 221, 413\}$
T_6	$\{211, 323, 524\}$
T_7	$\{323, 411, 524, 713\}$

1. For the sake of simplicity, notice that, since the total number of transactions is fixed, the support is expressed in an absolute value rather than a relative percentage.

Level-1 minsup = 4
Level-1 frequent 1-itemsets:
$\mathcal{L}[1,1]$

Itemset	Support
{1**}	5
{2**}	5

Level-1 frequent 2-itemsets:
$\mathcal{L}[1,2]$

Itemset	Support
{1**, 2**}	4

Filtered transaction table:
$\mathcal{T}[2]$

TID	Items
T_1	{111, 121, 211, 221}
T_2	{111, 211, 222}
T_3	{112, 122, 221}
T_4	{111, 121}
T_5	{111, 122, 211, 221}
T_6	{211}

Fig. 2. Large item sets at level 1 and filtered transaction table: $\mathcal{T}[2]$.

3.1 Algorithm ML_T2L1

The above discussion leads to the following algorithm for mining strong ML-association rules.

Algorithm 3.1 (ML_T2L1). *Find multiple-level frequent item sets for mining strong ML association rules in a transaction database.*

Input: 1) $\mathcal{T}[1]$, a hierarchy-information-encoded and task-relevant set of transaction database, in the format of $\langle TID, Itemset \rangle$, in which each item in the *Itemset* contains encoded concept hierarchy information, and 2) the minimum support threshold ($minsup[l]$) for each concept level l.

Output: Multiple-level frequent item sets.

Method: A top-down, progressively deepening process which collects frequent item sets at different concept levels as follows:

Starting at level 1, derive for each level l, the frequent k-items sets, $\mathcal{L}[l,k]$, for each k, and the frequent item set, $\mathcal{LL}[l]$ (for all ks), as follows (in the syntax similar to C and Pascal, which should be self-explanatory):

```
1.   for (l := 1; L[l,1] ≠ ∅ and l < max_level; l++) do {
2.       if l = 1 then {
3.           L[l,1] := get_frequent_1_itemsets(T[1], l);
4.           T[2] := get_filtered_t_table(T[1], L[1,1]);
5.       }
6.       else L[l,1] := get_frequent_1_itemsets(T[2], l);
7.       for (k := 2; L[l,k−1] ≠ ∅; k++) do {
8.           Ck := apriori_gen(L[l,k−1]);
                 // candidate generation algorithm in Apriori [3]
9.           foreach transaction t ∈ T[2] do {
10.              Ct := get_subsets(Ck,t);
11.              foreach candidate c ∈ Ct do c.support++;
12.          }
13.          L[l,k] := {c ∈ Ck|c.support ≥ minsup[l]}
14.      }
15.      LL[l] := ∪k L[l,k];
16.  }  □
```

After finding the frequent itemsets, the set of association rules for each level l can be derived from the frequent itemsets $\mathcal{LL}[l]$ based on the minimum confidence at this level, $minconf[l]$, as in [3].

Level-2 minsup = 3
Level-2 frequent 1-itemsets:
$\mathcal{L}[2,1]$

Itemset	Support
{11*}	5
{12*}	4
{21*}	4
{22*}	4

Level-2 frequent 2-itemsets:
$\mathcal{L}[2,2]$

Itemset	Support
{11*, 12*}	4
{11*, 21*}	3
{11*, 22*}	4
{12*, 22*}	3
{21*, 22*}	3

Level-2 frequent 3-itemsets:
$\mathcal{L}[2,3]$

Itemset	Support
{11*, 12*, 22*}	3
{11*, 21*, 22*}	3

Level-3 minsup = 3
Level-3 frequent 1-itemsets:
$\mathcal{L}[3,1]$

Itemset	Support
{111}	4
{211}	4
{221}	3

Level-3 frequent 2-itemsets:
$\mathcal{L}[3,2]$

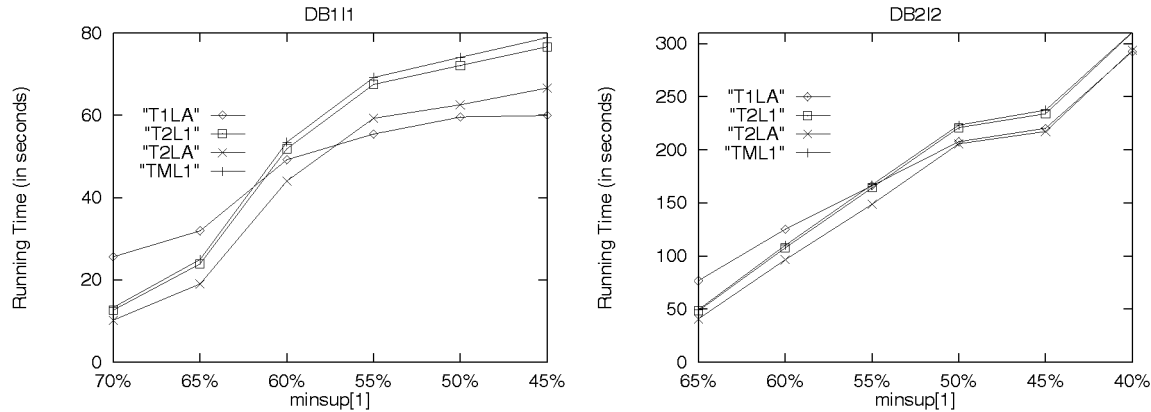| Itemset	Support
{111, 211}	3

Fig. 3. Large item sets at levels 2 and 3.

Fig. 4. Peformance with respect to top-level minimum support.

Potential performance improvements of Algorithm ML_T2L1 are considered by exploration of the sharing of data structures, and intermediate results and maximal generation of results at each database scan, etc.

3.2   Algorithm ML_T1LA

The first variation is to use only one encoded transaction table $\mathcal{T}[1]$, that is, no filtered encoded transaction table $\mathcal{T}[2]$ will be generated in the processing. Moreover, the support for the candidate sets at all the levels are computed at the same time by scanning $\mathcal{T}[1]$ once. This algorithm avoids the generation of a new encoded transaction table. Moreover, it needs to scan $\mathcal{T}[1]$ once for generation of each frequent k-itemset table. Since the total number of scanning of $\mathcal{T}[1]$ will be k times for the largest k-itemsets, it is a potentially efficient algorithm. However, $\mathcal{T}[1]$ may consist of many infrequent items which could be wasteful to be scanned or examined. Also, it needs a large space to keep all $C[l]$, which may cause some page swapping.

3.3   Algorithm ML_TML1

The second variation is to generate multiple encoded transaction tables $\mathcal{T}[1], \mathcal{T}[2], \ldots, \mathcal{T}[max\_l + 1]$, where max_l is the maximal level number to be examined in the processing. A new table $\mathcal{T}[l+1]$ is generated at the processing of each level l, for $l > 1$. This is done by scanning $\mathcal{T}[l]$ to generate the frequent 1-itemsets $\mathcal{L}[l, 1]$, which serves as a filter to filter out from $\mathcal{T}[l]$ any infrequent items or transactions containing only infrequent items, and results in $\mathcal{T}[l+1]$, which will be used for the generation of frequent k-itemsets (for $k > 1$) at level l and table $\mathcal{T}[l+2]$ at the next lower level. Notice that, as an optimization, for each level $l > 1$, $\mathcal{T}[l]$, and $\mathcal{L}[l, 1]$ can be generated in parallel (i.e., at the same scan).

The algorithm derives a new filtered transaction table, $\mathcal{T}[l+1]$, at the processing of each level l. This may save a substantial amount of processing if only a small portion of data are frequent items at each level. However, it may not be so effective if only a small number of the items will be filtered out at the processing of each level.

3.4   Algorithm ML_T2LA

The third variation uses the same two encoded transaction tables $\mathcal{T}[1]$ and $\mathcal{T}[2]$ as in Algorithm ML_T2L1, but it integrates some optimization techniques considered in the algorithm ML_T1LA. That is, the support for each candidate itemset at the level $l(l \geq 1)$ is calculated at the same time by scanning $\mathcal{T}[2]$ once. This algorithm avoids the generation of a group of new filtered transaction tables. It scans $\mathcal{T}[1]$ twice to generate $\mathcal{T}[2]$ and the

frequent 1-itemset tables for all the levels. Then, it scans $\mathcal{T}[2]$ once for the generation of each frequent k-itemset and, thus, scans $\mathcal{T}[2]$ in total $k - 1$ times for the generation of all the k-itemsets, where k is the largest such k-itemsets available. Since k-itemsets generation for $k > 1$ is performed on $\mathcal{T}[2]$, which may consist of much fewer items than $\mathcal{T}[1]$, the algorithm could be a potentially efficient one.

4   Performance Study

To study the performance of the propsoed algoirthms, all four algorithms—ML_T2L1, ML_T1LA, ML_TML1, and ML_T2LA —are implemented and tested on a Sun/Sparcstation20 with 32 MB of main memory running Solaris 2.5.

The testbed consists of a set of synthetic transaction databases generated using a randomized transaction generation algorithm similar to that described in [3]. We select I, the total number of items, to be 1,000; L, the number of potentially frequent itemsets, to be 2,000; and D, the total number of transactions, to be 100,000. The scale-up tests on total number of items, I; average size of transactions, T; and total number of tuples, D, showed satisfactory results as well [13].

Two database settings are used, $DB1$, with average size (the number of items) of potentially frequent itemsets of 4 and average transaction size (the number of items) of 10, and $DB2$, with average size of potentially frequent itemsets of 6 and average transaction size of 20.

Each transaction database is converted into an encoded transaction table, denoted as $\mathcal{T}[1]$, according to the information about the generalized items in the item description (hierarchy) table. The maximal level of the concept hierarchy in the item table is set to 3. The number of the top level nodes keeps increasing until the total number of items reaches I. The fan-outs at the lower levels are selected based on the normal distribution, with mean value being $B2$ and $B3$ for levels 2 and 3, respectively, and with a variance of 2.0. Our experiments show little differences for a variance of 1.0 or 3.0. Two item description tables, $I1$ and $I2$, are used which are generated using $(B2 = 10, B3 = 10)$ and $(B2 = 8, B3 = 5)$, respectively.

4.1   Performance Comparison of Algorithms

The testing results presented in this section are on two synthetic transaction databases: $DB1I1$, which uses the database setting $DB1$ and the item description table $I1$; and $DB2I2$, which uses the database setting $DB2$ and the item description table $I2$. For both databases, we test the algorithms with respect to the minimum
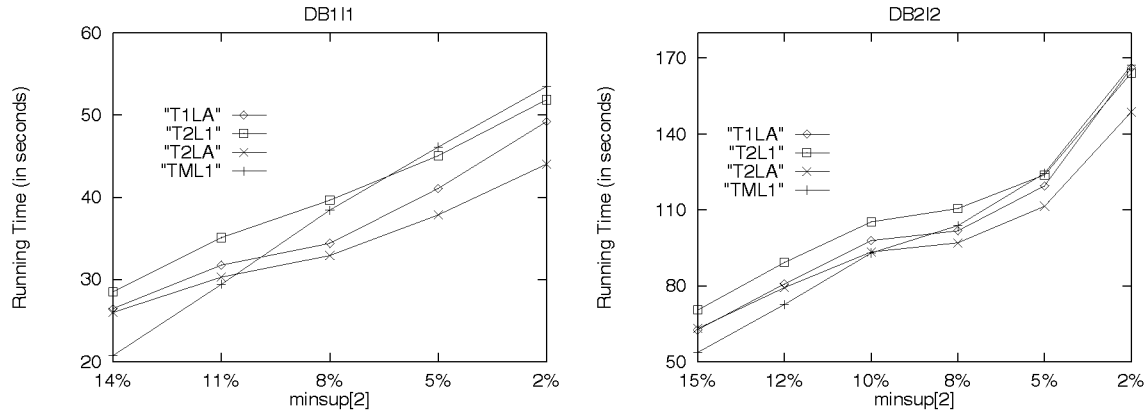
Fig. 5. Minimum support at level 2.

support thresholds at all three levels. Tests on other databases with different database settings and item table settings show similar results.

Fig. 4 shows the running time of the four algorithms on $DB1I1$ and $DB2I2$, with respect to the minimum support threshold at level 1. The minimum supports at level 2 and 3 are fixed to 2 percent and 0.75 percent for $DB1I1$, and 3 percent and 1 percent for $DB2I2$.

The level 2 minimum support threshold is set to 2 percent for $DB1I1$ and 3 percent for $DB2I2$, which means no filtering of items in transactions at level 2. Therefore, $T[3]$ has the same size of $T[2]$ and the derivation of $T[3]$ is a waste. It is obvious that ML_TML1 is always worse than ML_T2LA and ML_T2L1 for these minimum supports.

Fig. 5 shows the running time for the four algorithms on $DB1I1$ and $DB2I21$ with respect to the minimum support threshold at level 2. The minimum supports at levels 1 and 3 are fixed to 60 percent and 0.75 percent for $DB1I1$, and 55 percent and 1 percent for $DB2I2$.

Fig. 6 shows the running time of the four algorithms on $DB1I1$ and $DB2I2$, with respect to the minimum support threshold at level 3. The minimum supports at levels 1 and 2 are fixed to 60 percent and 2 percent for $DB1I1$ and 55 percent and 3 percent for $DB2I2$.

The above figures show two interesting features. First, the relative performance of the four algorithms is highly relevant to the threshold setting (i.e., the power of a filter at each level), especially

the level 1 and level 2 thresholds. Thus, based on the effectiveness of a threshold, a good algorithm can be selected to achieve good performance. Second, the optimization of parallel derivation of $\mathcal{L}[l, k]$ is very useful and the derivation of $T[2]$ is usually beneficial. This can be verified by the experimental results, which show ML_T2LA is always the best or the second best algorithm.

5 MINING CROSS-LEVEL ASSOCIATIONS

The multilevel association rule mining algorithms presented in Section 3 may generate a large number of association rules which are confined to level-by-level relationships in a hierarchy. This can be relaxed to allow the exploration of "*cross-level*" association rules, i.e., the mining is not confined to those associations among the concepts at the *same level* of a hierarchy. This relaxation may lead to the discovery of associations like "2 percent Foremost milk ⇒ Wonder bread" in which the concept at the left-hand side is at a lower level of a hierarchy than the one at the right-hand side. Since such a mining request can be implemented by slightly modifying our algorithms, formal presentation of the extended algorithms is omitted. Only the modifications are outlined below.

First, a single minimum support threshold is used at all levels. Second, when the frequent k-itemsets (for $k > 1$) are generated, items at all levels are considered together, but itemsets which contain an item and its ancestor are excluded.

Two algorithms are implemented to find cross-level rules, $ML\_T2LA\text{-}C$ and $ML\_T1LA\text{-}C$, which are revised versions of
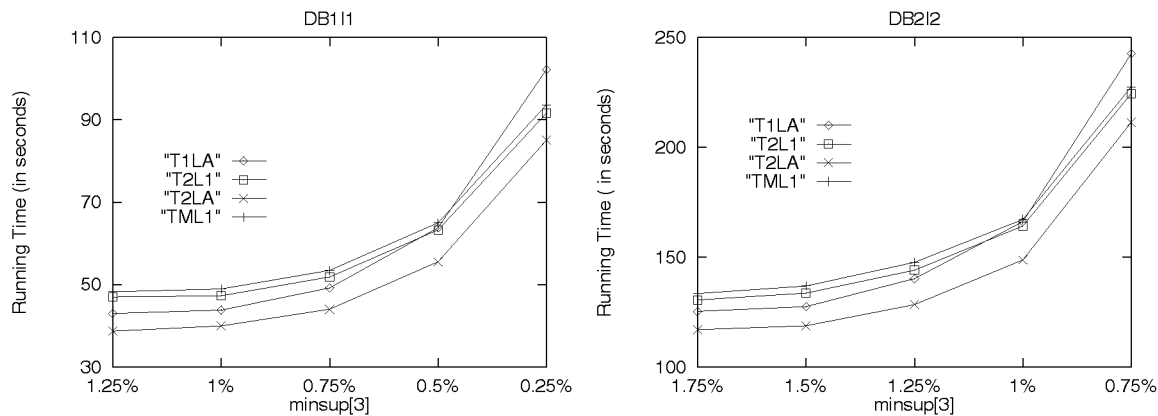

Fig. 6. Minimum support at level 3.

ML_T2LA and ML_T1LA, respectively. Our experiments show both algorithms are an order of magnitude slower than their counterparts for minimum supports from 1 percent to 5 percent [13]. This is because there are many more frequent itemsets at high levels and the support computation is more complex.

6 PRESENTATION OF INTERESTING ASSOCIATION RULES

Not every strong rule so discovered (i.e., passing the minimum support and minimum confidence thresholds) is interesting enough to be presented to users. Two interestingness measures are proposed to filter *redundant rules* and *unnecessary rules*.

6.1 Removal of Redundant Rules

A rule is redundant if it can be derived or computed from a higher level rule (every item is the same or a higher level item) and the simple assumption of a relatively uniform distribution. For examples, suppose there is a rule *"milk → bread (12 percent support, 85 percent confidence)."* Then, another rule *"chocolate milk → bread (1 percent support, 84 percent confidence)"* may not be interesting if 8 percent of milk are chocolate milk since such a rule does not convey additional information and is less general than its corresponding high-level counterpart (if the uniform data distribution is assumed).

Definition 6.1. *A rule R, $A_1 \wedge A_2 \cdots A_n \Rightarrow B_1 \wedge B_2 \cdots B_m$ is* **redundant** *if there is a rule R', $A_1' \wedge A_2' \cdots A_n' \Rightarrow B_1' \wedge B_2' \cdots B_m'$, every item in R is a descendant or the same of the corresponding item in R', and $\varphi(R) \in [exp(\varphi(R)) - \alpha, exp(\varphi(R)) + \alpha]$ where $exp(\varphi(R)) = (\sigma(B_1)/\sigma(B_1')) \times \cdots \times (\sigma(B_n)/\sigma(B_n')) \times \varphi(R')$, and α is a user-defined constant.*

To remove such redundant rules, when a rule R passes the minimum confidence test, it is checked against every strong rule R', of which R is a descendant. If the confidence of R, $\varphi(R)$, falls in the range of the expected confidence with the variation of α, it is not output to user.

From our experiments, it is clear that, by removing redundant rule, we can cut down the number of rule presented to 30 percent to 60 percent of all strong rules [13]. In addition, the reduction ratio decreases slowly when α increases because more rules are removed.

6.2 Removal of Unnecessary Rules

If we have a rule "80 percent of customers who buy milk also buy bread," a rule "80 percent of customers who buy milk and butter also buy bread" provides little extra information than the previous one. Such a rule is unnecessary because it is not significantly different from a simpler rule and, thus, is uninteresting [7].

Definition 6.2. *A rule R, $A \wedge C \Rightarrow B$ is* **unnecessary** *if there is a rule R', $A \Rightarrow B$, and $\varphi(R) \in [\varphi(R') - \beta, \varphi(R') + \beta]$, where β is a user given constant, A, B, C are itemsets, and C is not empty.*

To filter out such kinds of unnecessary association rules, for each strong rule $R : A \Rightarrow B$, we test every such rule $R' : A - C \Rightarrow B$, where $C \subseteq A$. If the confidence of R, $\varphi(R)$, is not significantly different (specified by β) from that of R', $\varphi(R')$, it is not presented to user.

Our experiments reveal that the removal of unnecessary rules can reduce the number of presented rules to 50 percent to 80 percent of all strong rules [13].

7 CONCLUSIONS AND FUTURE WORK

We have extended the scope of the study of mining association rules from single level to multiple concept levels and studied methods for mining multiple-level association rules from large transaction databases. Mining multiple-level association rules may lead to progressive mining of refined knowledge from data and have interesting applications for knowledge discovery in transaction databases, as well as other business or engineering databases. A top-down progressive deepening technique is developed for mining multiple-level association rules. Based on different sharing techniques, a group of algorithms have been developed. Our performance study shows that different algorithms may have the best performance for different distributions of data. Methods for mining cross-level association rules and interestingness measures for association rules are also investigated. Our study shows that mining multiple-level association rules from databases has wide applications, and efficient algorithms can be developed for discovery of interesting and strong such rules in large databases.

Extension of methods for mining single-level knowledge rules to multiple-level ones poses many interesting issues for further investigation. For example, with the studies on mining single-level sequential patterns [4], it is interesting to develop efficient algorithms for mining multiple-level sequential patterns. Also, with the generalization of mining associations to mining correlations in large databases [5], mining multiple-level correlations in databases is another interesting issue to be studied in the future.

REFERENCES

[1] R. Agrawal, T. Imielinski, and A. Swami, "Mining Association Rules Between Sets of Items in Large Databases," *Proc. 1993 ACM SIGMOD Int'l Conf. Management of Data,* pp. 207–216, Washington, D.C., May 1993.
[2] R. Agrawal and J.C. Shafer, "Parallel Mining of Association Rules: Design, Implementation, and Experience," *IEEE Trans. Knowledge and Data Eng.,* vol. 8, pp. 962–969, 1996.
[3] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," *Proc. 1994 Int'l Conf. Very Large Data Bases,* pp. 487–499, Santiago, Chile, Sept. 1994.
[4] R. Agrawal and R. Srikant, "Mining Sequential Patterns," *Proc. 1995 Int'l Conf. Data Eng.,* pp. 3–14, Taipei, Taiwan, Mar. 1995.
[5] S. Brin, R. Motwani, and C. Silverstein, "Beyond Market Basket: Generalizing Association Rules to Correlations," *Proc. 1997 ACM SIGMOD Int'l Conf. Management of Data,* pp. 265–276, Tucson, Ariz., May 1997.
[6] S. Chaudhuri and U. Dayal, "An Overview of Data Warehousing and OLAP Technology," *ACM SIGMOD Record,* vol. 26, pp. 65–74, 1997.
[7] M.S. Chen, J. Han, and P.S. Yu, "Data Mining: An Overview from a Database Perspective," *IEEE Trans. Knowledge and Data Eng.,* vol. 8, pp. 866–883, 1996.
[8] D.W. Cheung, J. Han, V. Ng, A. Fu, and Y. Fu, "A Fast Distributed Algorithm for Mining Association Rules," *Proc. 1996 Int'l Conf. Parallel and Distributed Information Systems,* pp. 31–44, Miami Beach, Fla., Dec. 1996.
[9] D.W. Cheung, J. Han, V. Ng, and C.Y. Wong, "Maintenance of Discovered Association Rules in Large Databases: An Incremental Updating Technique," *Proc. 1996 Int'l Conf. Data Eng.,* pp. 106–114, New Orleans, Feb. 1996.
[10] Y. Fu and J. Han, "Meta-Rule-Guided Mining of Association Rules in Relational Databases," *Proc. First Int'l Workshop Integration Knowledge Discovery with Deductive and Object-Oriented Databases (KDOOD '95),* pp. 39–46, Singapore, Dec. 1995.
[11] T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama, "Data Mining Using Two-Dimensional Optimized Association Rules: Scheme, Algorithms, and Visualization," *Proc. 1996 ACM SIGMOD Int'l Conf. Management of Data,* pp. 13–23, Montreal, June 1996.

[12] J. Han, Y. Cai, and N. Cercone, "Data-Driven Discovery of Quantitative Rules in Relational Databases," *IEEE Trans. Knowledge and Data Eng.,* vol. 5, pp. 29–40, 1993.

[13] J. Han and Y. Fu, "Mining Multiple-Level Association Rules in Large Databases," technical report, Univ. of Missouri–Rolla, URL: http://www.umr.edu/~yongjian/pub/ml.ps, 1997.

[14] M. Houtsma and A. Swami, "Set-Oriented Mining for Association Rules in Relational Databases," *Proc. 1995 Int'l Conf. Data Eng.,* pp. 25–34, Taipei, Taiwan, Mar. 1995.

[15] M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, and A.I Verkamo, "Finding Interesting Rules from Large Sets of Discovered Association Rules," *Proc. Third Int'l Conf. Information and Knowledge Management,* pp. 401–408, Gaithersburg, Md., Nov. 1994.

[16] K. Koperski and J. Han, "Discovery of Spatial Association Rules in Geographic Information Databases," *Proc. Fourth Int'l Symp. Large Spatial Databases (SSD '95),* pp. 47–66, Portland, Maine, Aug. 1995.

[17] H. Mannila, H. Toivonen, and A.I. Verkamo, "Efficient Algorithms for Discovering Association Rules," *Proc. AAAI '94 Workshop Knowledge Discovery in Databases (KDD '94),* pp. 181–192, Seattle, July 1994.

[18] R. Meo, G. Psaila, and S. Ceri, "A New SQL-Like Operator for Mining Association Rules," *Proc. 1996 Int'l Conf. Very Large Data Bases,* pp. 122–133, Bombay, India, Sept. 1996.

[19] J.S. Park M.S. Chen, and P.S. Yu, "An Effective Hash-Based Algorithm for Mining Association Rules," *Proc. 1995 ACM SIGMOD Int'l Conf. Management of Data,* pp. 175–186, San Jose, Calif., May 1995.

[20] G. Piatetsky-Shapiro, "Discovery, Analysis, and Presentation of Strong Rules," *Knowledge Discovery in Databases,* G. Piatetsky-Shapiro and W.J. Frawley, eds., pp. 229–238, AAAI/MIT Press, 1991.

[21] A. Savasere, E. Omiecinski, and S. Navathe, "An Efficient Algorithm for Mining Association Rules in Large Databases," *Proc. 1995 Int'l Conf. Very Large Data Bases,* pp. 432–443, Zurich, Sept. 1995.

[22] *Proc. Second Int'l Conf. Knowledge Discovery and Data Mining (KDD '96),* E. Simoudis, J. Han, and U. Fayyad, eds., AAAI Press, Aug. 1996.

[23] R. Srikant and R. Agrawal, "Mining Generalized Association Rules," *Proc. 1995 Int'l Conf. Very Large Data Bases,* pp. 407–419, Zurich, Sept. 1995.

[24] R. Srikant and R. Agrawal, "Mining Quantitative Association Rules in Large Relational Tables," *Proc. 1996 ACM SIGMOD Int'l Conf. Management of Data,* pp. 1–12, Montreal, June 1996.