CCMine: Efficient Mining of Confidence-Closed Correlated Patterns

Won-Young Kim*, Young-Koo Lee**, and Jiawei Han

Department of Computer Science
University of Illinois at Urbana-Champaign, Illinois, U.S.A.
{wykim, yklee, hanj}@uiuc.edu

Abstract. Correlated pattern mining has become increasingly important recently as an alternative or an augmentation of association rule mining. Though correlated pattern mining discloses the correlation relationships among data objects and reduces significantly the number of patterns produced by the association mining, it still generates quite a large number of patterns. In this paper, we propose **closed correlated pattern mining** to reduce the number of the correlated patterns produced without information loss. We first propose a new notion of the *confidence-closed* correlated patterns, and then present an efficient algorithm, called CCMine, for mining those patterns. Our performance study shows that confidence-closed pattern mining reduces the number of patterns by at least an order of magnitude. It also shows that CCMine outperforms a simple method making use of the the traditional closed pattern miner. We conclude that confidence-closed pattern mining is a valuable approach to condensing correlated patterns.

1 Introduction

Though association rule mining has been extensively studied in data mining research, its popular adoption and successful industry application has been hindered by a major obstacle: *association mining often generates a huge number of rules, but a majority of them either are redundant or do not reflect the true correlation relationship among data objects*. To overcome this difficulty, interesting pattern mining has become increasingly important recently and many alternative interestingness measures have been proposed [1,2,3,4].

While there is still no universally accepted best measure for judging interesting patterns, *all_confidence* [5] is emerging as a measure that can disclose true correlation relationships among data objects [5,6,7,8]. One of important properties of all_confidence is that it is not influenced by the co-absence of object pairs in the transactions—such an important property is called *null-invariance* [8]. The

* Current address: On Demand Service Team, Digital Home Research Division, ETRI, Korea

** Current address: School of Electronics and Information, Kyung Hee University, Korea

H. Dai, R. Srikant, and C. Zhang (Eds.): PAKDD 2004, LNAI 3056, pp. 569–579, 2004.
© Springer-Verlag Berlin Heidelberg 2004

co-absence of a set of objects, which is normal in large databases, may have unexpected impact on the computation of many correlation measures. All_confidence can disclose genuine correlation relationships without being influenced by object co-absence in a database while many other measures cannot. In addition, all_confidence mining can be performed efficiently using its downward closure property [5].

Although the all_confidence measure reduces significantly the number of patterns mined, it still generates quite a large number of patterns, some of which are redundant. This is because mining a long pattern may generate an exponential number of sub-patterns due to the downward closure property of the measure. For frequent itemset mining, there have been several studies proposed to reduce the number of items mined, including mining closed [9], maximal [10], and compressed (approximate) [11] itemsets. Among them, the closed itemset mining, which mines only those frequent itemsets having no proper superset with the same support, limits the number of patterns produced without information loss. It has been shown in [12] that the closed itemset mining generates orders of magnitude smaller result set than frequent itemset mining.

In this paper, we introduce the concept of *confidence closed correlated pattern*, which plays the role of reducing the number of the correlated patterns produced without information loss. All_confidence is used as our correlation measure. However, the result can be easily extended to several other correlation measures, such as coherence [6]. First, we propose the notion of the *confidence-closed* correlated pattern. Previous studies use the concept of *support-closed* pattern, i.e., the closed pattern based on the notion of support. However, support-closed pattern mining fails to distinguish the patterns with different confidence values. In order to overcome this difficulty, we introduce *confidence-closed* correlated pattern which encompasses both confidence and support. Then we propose an efficient algorithm, called CCMine, for mining confidence-closed patterns. Our experimental and performance study shows that confidence-closed pattern mining reduces the number of patterns by at least an order of magnitude. It also shows that superiority of the proposed algorithm over a simple method that mines the confidence-closed patterns using the patterns generated by the support-closed pattern miner.

The rest of the paper is organized as follows. Section 2 introduce basic concepts of frequent itemset mining and all_confidence. Section 3 defines the notion of the confidence-closed patterns and discusses its properties. Section 4 presents an efficient algorithm for mining confidence-closed correlated patterns. Section 5 reports our experimental and performance results. Finally, Section 6 concludes the paper.

2 Background

We first introduce the basic concepts of frequent itemset mining, and then present a brief review of the all_confidence measure.

Let $I = \{i_1, i_2, \ldots, i_m\}$ be a set of items, and DB be a database that consists of a set of transactions. Each transaction T consists of a set of items such that $T \subseteq I$. Each transaction is associated with an identifier, called *TID*. Let A be a set of items, referred to as an *itemset*. An itemset that contains k items is a k-*itemset*. A transaction T is said to *contain* A if and only if $A \subseteq T$. The *support* of an itemset X in DB, denoted as $sup(X)$, is the number of transactions in DB containing X. An itemset X is *frequent* if it occurs no less frequent than a user-defined minimum support threshold. In most data mining studies, only the frequent itemsets are considered as significant and will be mined.

The all_confidence of an itemset X is the minimal confidence among the set of association rules $i_j \rightarrow X - i_j$, where $i_j \in X$. Its formal definition is given as follows. Here, the max_item_sup of an itemset X means the maximum (single) item support in DB of all the items in X.

Definition 1. *(all-confidence of an itemset) Given an itemset* $X = \{i_1, i_2, \ldots, i_k\}$*, the all_confidence of* X *is defined as,*

$$max\_item\_sup(X) = max\{sup(i_j) | \forall i_j \in X\} \tag{1}$$

$$all\_conf(X) = \frac{sup(X)}{max\_item\_sup(X)} \tag{2}$$

Given a transaction database DB, a minimum support threshold min_sup and a minimum all_confidence threshold min_α, a frequent itemset X is ***all_confident*** or ***correlated*** if $all\_conf(X) \geq min\_\alpha$ and $sup(X) \geq min\_sup$.

3 Confidence-Closed Correlated Patterns

It is well known that closed pattern mining has served as an effective method to reduce the number of patterns produced without information loss in frequent itemset mining. Motivated by such practice, we extend the notion of closed pattern so that it can be used in the domain of correlated pattern mining. We present the formal definitions of the original and extended ones in Definitions 2 and 3, respectively. In this paper, we call the former *support-closed* and the latter *confidence-closed*.

Definition 2. *(Support-Closed Itemset) An itemset* Y *is a* ***support-closed (correlated) itemset*** *if it is frequent and correlated and there exists no proper superset* $Y' \supset Y$ *such that* $sup(Y') = sup(Y)$.

Since the support-closed itemset is based on support, it cannot retain the confidence information—notice that in this paper *confidence* means *the value of all_confidence*. In other words, support-closed causes information loss.

Example 1. Let itemset $ABCDE$ be a correlated pattern with support 30 and confidence 30% and itemset CDE be one with support 30 and confidence 80%. Suppose that we want to get a set of non-redundant correlated patterns when

$min\_sup = 20$ and $min\_\alpha = 20\%$. Support-closed pattern mining generates $ABCDE$ only eliminating CDE since $ABCDE$ is superset of CDE with the same support. We thus lose the pattern CDE. However, CDE might be more interesting than $ABCDE$ since the former has higher confidence that the latter.

We thus extend the support-closed itemset to encompass the confidence so that it can retain the confidence information as well as support information.

Definition 3. *(Confidence-Closed Itemset) An itemset Y is a* ***confidence-closed itemset*** *if it is correlated and there exists no proper superset $Y' \supset Y$ such that $sup(Y') = sup(Y)$ and $all\_conf(Y') = all\_conf(Y)$.*

By applying mining of confidence-closed itemsets to Example 1, we can obtain not only itemset $ABCDE$ but also CDE as confidence-closed itemsets since they have different confidence values and therefore no information loss occurs. In the rest of our paper, we call the support-closed pattern as SCP and the confidence-closed pattern as CCP, respectively.

4 Mining Confidence-Closed Correlated Patterns

In this section, we propose two algorithms for mining CCPs: CCFilter and CCMine. CCFilter is a simple algorithm that makes use of the existing support-closed pattern generator. CCFilter consists of the following two steps: First, get the complete set of SCPs using the previous proposed algorithms [13]. Second, check each itemset and its all possible subsets in the resulting set whether it is confidence-closed. If its confidence satisfies min_α and it has no proper superset with the same confidence, it is generated as a confidence-closed itemset. CCFilter is used as a baseline algorithm for comparison in Section 5.

CCFilter has a shortcoming: It generates SCPs with less confidence than min_α during the mining process. At the end, these patterns are removed. In order to solve this problem, CCMine integrates the two steps of CCFilter into one. Since all_confidence has the downward closure property, we can push down the confidence condition into the process of the confidence-closed pattern mining.

CCMine adopts a pattern-growth methodology proposed in [14]. In the previous studies (e.g., CLOSET+ [13] and CHARM [15]) for mining SCPs, two search space pruning techniques, *item merging* and *sub-itemset merging*, have been mainly used. However, if we apply these techniques directly into confidence-closed pattern mining, we cannot obtain a complete set of CCPs. This is because if there exists a pattern, these techniques remove all of its sub-patterns with the same support without considering confidence. We modify these optimization techniques so that they can be used in confidence-closed pattern mining.

Lemma 1. *(confidence-closed item merging) Let X be a correlated itemset. If every transaction containing itemset X also contains itemset Y but not any proper superset of Y, and $all\_conf(XY) = all\_conf(X)$, then XY forms a confidence-closed itemset and there is no need to search any itemset containing X but no Y.* ∎

Lemma 2. *(confidence-closed sub-itemset pruning) Let X be a correlated itemset currently under construction. If X is a proper subset of an already found confidence-closed itemset Y and $all\_conf(X) = all\_conf(Y)$ then X and all of X's descendants in the set enumeration tree cannot be confidence-closed itemsets and thus can be pruned.*

Lemma 1 means that we have to mine the X-conditional database and the XY-conditional database separately if $all\_conf(X) \neq all\_conf(XY)$. However, though $all\_conf(X)$ and $all\_conf(XY)$ are different, the X- and XY- conditional databases are exactly the same if $sup(X) = sup(XY)$. Using this property, we can avoid the overhead of building conditional databases for the prefix itemsets with the same support but different confidence. We maintain a list *candidateList* of the items that have the same support with the size of the X-conditional database but are not included in the item merging because of their confidence. The list is constructed as follows. For X-conditional database, let Y be the set of items in f_list such that they appear in every transaction. Do the following: for each item Y_i in Y, if $sup(Y_i) \leq max\_item\_sup(X)$, $X = X \cup Y_i$; otherwise insert Y_i to *candidateList*. When we check whether an itemset Z containing $X (Z \supset X)$ is confidence-closed, we also check whether the itemset $Z \cup Y' (Y' = Y_1 \ldots Y_k, Y_i \in candidateList)$ could be confidence-closed. Using this method, we compute CCPs without generating the two conditional databases of X and of XY when $all\_conf(X) > all\_conf(XY)$ and $sup(X) = sup(XY)$.

Algorithm 4 shows the CCMine algorithm, which is based on the extension of CLOSET+ [13] and integrates the above discussions into the CLOSET+. Among a lot of studies for support-closed pattern mining, CLOSET+ is the fastest algorithm for a wide range of applications.

CCMine uses another optimization technique to reduce the search space by taking advantage of the property of the all_confidence measure. Lemma 3 describes the pruning rule.

Lemma 3. *(counting space pruning rule) Let $\alpha = i_1 i_2 \ldots i_k$. In the α-conditional database, for item x to be included in an all_confident pattern, the support of x should be less than $sup(\alpha)/min\_\alpha$.* ∎

Proof. In order for αx to be an all_confident pattern, $max\_item\_sup(\alpha x) \leq sup(\alpha x)/min\_\alpha$. Moreover, $|sup(\alpha)| \geq |sup(\alpha x)|$. Thus, $max\_item\_sup(\alpha x) \leq sup(\alpha)/min\_\alpha$. Hence the lemma. ∎

With this pruning rule, we can reduce the set of items I_β to be counted and, thus, reduce the number of nodes visited when we traverse the FP-tree to count each item in I_β.

Example 2. Let us illustrate the confidence-closed mining process using an example. Figure 1 shows our running example of the transaction database DB. Let $min\_sup = 2$ and $min\_\alpha = 40\%$. Scan DB once. We find and sort the list of frequent items in support descending order. This leads to f_list = $\langle a{:}9, b{:}7, c{:}6, e{:}6, g{:}5, f{:}4, d{:}3, i{:}3, k{:}3, j{:}2, h{:}1\rangle$. Figure 2 shows the global FP-tree. For lack of space, we only show two representative cases: mining for prefix $j{:}2$ and $eg{:}5$.

Algorithm 41 CCMine: Mining confidence-closed correlated patterns

Input: a transaction database DB; a support threshold min_sup
a minimum all_confidence threshold min_α

Output: The complete set of confidence-closed correlated patterns.

Method:

1. Let CCP be the set of confidence-closed patterns. Initialize $CCP \leftarrow \emptyset$
2. Scan DB once to find frequent items and compute frequent list $f\_list(=\langle f_0, f_1, \dots \rangle)$.
3. Call CCMine($\emptyset$, DB, f_list, CCP, $\emptyset$).

ProcedureCCMine(α, CDB, f_list, CCP, $candidateList$)

1: For each item Y in f_list such that it appears in every transaction of CDB, delete Y from f_list and set $\alpha \leftarrow Y \cup \alpha$ if $all\_conf(Y\alpha) \geq all\_conf(\alpha)$, otherwise insert Y into $candidateList$ in the support increasing order; {confidence-closed item merging}
2: call GenerateCCP(α, $candidateList$, CCP);
3: build FP-tree for CDB using f_list, which excludes all the items Ys in the previous step;
4: **for** each a_i in f_list (in reverse descending support order) **do**
5: set $\beta = \alpha \cup a_i$;
6: call GenerateCCP(β, $candidateList$, CCP);
7: get a set I_β of items to be included in β-projected database; {counting space pruning rule}
8: for each item in I_β, compute its count in β-projected database;
9: **for** each b_j in I_β **do**
10: if $sup(\beta b_j) < min\_sup$, delete b_j from I_β; {pruning based on min_sup}
11: if $all_conf(\beta b_j) < min_\alpha$o, delete b_j from I_β;{pruning based on min_α}
12: **end for**
13: call FP-mine(β, CDB, f_list, CCP, $candidateList$);
14: delete the items that was inserted in step 1 from $candidateList$;
15: **end for**

Procedure GenerateCCP(α, $candidateList$, CCP)
for k-itemset $Y = Y_1 \dots Y_k (Y_i \in candidateList)$ **do**
add $\alpha \cup Y$ into CCP if $all\_conf(\alpha \cup Y) \geq min\_\alpha$ if $\alpha \cup Y$ is not a subset of X (in CCP) with the same support and confidence; {confidence-closed sub-itemset pruning}
end for

1. After building the FP-tree we mine the confidence-closed patterns with prefix j:2. **Computing counts:** We compute the counts for items a, c, e, f, and i to be included in the j-projected database by traversing the FP-tree shown in Fig. 2. First, we use Lemma 3 to reduce items to be counted. The support of item $z(z \in \{a, c, e, f, i\})$ should be less than or equal to $sup(j)/min\_\alpha = 2/0.4 = 5$. With this pruning, items a, c and e are eliminated. Now, we compute counts of items f and i and construct j-projected database. They are 2 and 1, respectively. **Pruning:** We conduct pruning based on min_sup and min_α. Item i is pruned since its support is less than min_sup. Item

TID	items bought
10	a, b, c, d, e, g
20	a, b, c, e, g, k
30	a, b, c, e, g
40	a, b, d, f, h
50	a, e, f, i, j
60	a, b, e, g, k
70	a, i, k
80	a, c
90	b, c, f
100	a, d, e, g
110	c, f, j
120	b, i

Fig. 1. An transaction database DB.

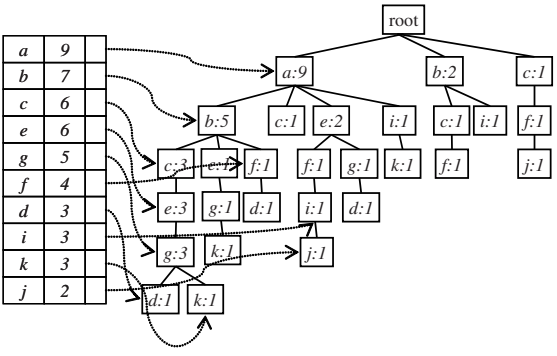


Fig. 2. FP-tree for the transaction database DB.

f is not pruned since and its confidence(2/4) is not less than min_α. Since f is the only item in j-conditional database, we do not need to build the corresponding FP-tree. And fj:2 is a CCP.

2. After building conditional FP-tree for prefix g:5 and we mine g:5-conditional FPtree with $f_list = \langle$a:5, e:5, b:4, c:3$\rangle$. **Confidence Item Merging:** We try confidence-closed item merging of a and e. We delete a and e from f_list. Since $all\_conf(ag) < all\_conf(g)$, we insert a into *candidateList*. Then, we extend the prefix from g to eg by the confidence-closed item merging. **GenerateCCP:** we generate eg:5 as a CCP. In addition, we also generate aeg:5, in which item a come from *candidateList*. Now, in f_list, only two item b:4 and c:3 left. We mine the CCPs with prefix ceg:3. First, we generate ceg as a CCP. However we can not generate $aceg$ as CCP since $all\_conf(aceg) < min\_\alpha$. Since items b the only item in f_list, $bceg$ is a CCP. Again, $abceg$ can not be CCP, since it also does not satisfy min_α. In this way, we mine the beg:4-conditional database and generate beg and $abeg$ as a CCP. After returning mining beg:4-conditonal FP-tree, item a is removed from *candidateList*.

5 Experiments

In this section, we report out experimental results on the performance of CCMine in comparison with CCFilter algorithm. The result shows that CCMine always outperforms CCFilter especially at low min_sup. Experiments were performed on a 2.2GHz Pentium IV PC with 512MB of memory, running Windows 2000. Algorithms were coded with Visual C++.

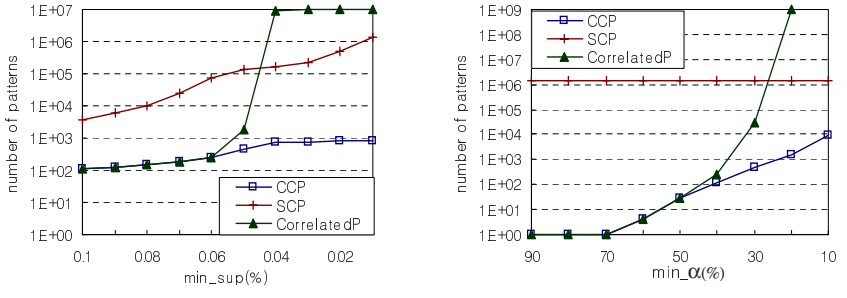
Our experiments were performed on two real datasets, as shown in Table 1. Pumsb dataset contains census data for population and housing and is obtained from http://www.almaden.ibm.com/software/quest. Gazelle, a transactional data set comes from click-stream data from Gazelle.com. In the table, ATL/MTL represent average/maximum transaction length. The gazelle dataset

is rather sparse in comparison with pumsb datasets, which is very dense so that it produce many long frequent itemsets even for very high values of support.

Table 1. Characteristics of Real Datasets.

Dataset	#Tuples	#Items	ATL/MTL
gazelle	59602	497	2.5/267
pumsb	49046	2113	74/74

We first show that the complete set of CCPs is much smaller in comparison with both that of correlated patterns and that of SCPs. Figure 3 shows the number of CCPs, correlated patterns, and SCPs generated from the gazelle data set. In this figure, the number of patterns is plotted on a log scale. Figure 3(a) shows the number of patterns generated when *min_sup* varies and *min_α* is fixed while Figure 3(b) shows those when *min_α* varies and *min_sup* is fixed. We first describe how many we can reduce the number of correlated patterns with the notion of CCPs. Figures 3(a) and 3(b) show that CCP mining generates a much smaller set than that of correlated patterns as the support threshold or the confidence threshold decreases, respectively. It is a desirable phenomenon since the number of correlated patterns increases dramatically as either of the thresholds decreases. These figures also show that the number of SCPs is quite bigger than that of CCPs over the entire range of the support and confidence threshold. These results indicate that CCP mining generates quite a smaller set of patterns even at the low minimum support threshold and low minimum confidence threshold.



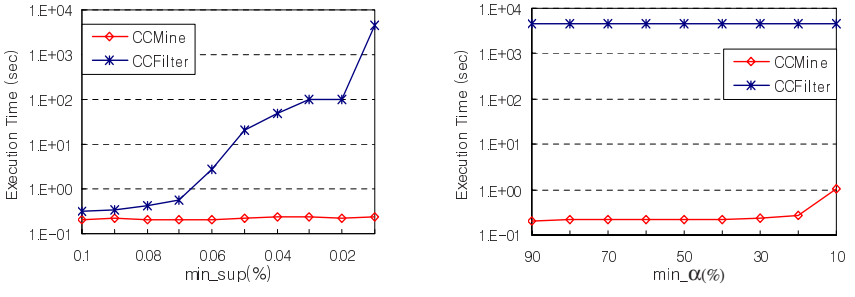
(a) $min\_\alpha = 25\%$ (b) $min\_sup = 0.01\%$

Fig. 3. Number of patterns generated from the gazelle data set.

Let us then compare the relative efficiency and effectiveness of the CCMine and CCFilter methods. Figure 4 (a) shows the execution time of the two methods on the gazelle dataset using different minimum support threshold while *min_α* is fixed at 25%. Figure 4(a) shows that CCMine always outperforms CCFilter over the entire supports of experiments. When the support threshold is low, CCMine

is faster more than 100 times compared with CCFilter, e.g., with *min_sup* 0.05%, CCFilter uses 20 seconds to finish while CCMine only uses 0.2. The reason why CCMine is superior to CCFilter is that CCFilter has to find all of the support closed patterns although many of them do not satisfy the minimum confidence threshold and the number of these patterns increases a lot as the minimum support threshold decreases. Figure 4(b) shows the performance on the gazelle dataset when *min_sup* is fixed at 0.01% and *min_α* varies. As shown in the figure, CCMine always outperforms CCFilter and the execution times of CCMine increases very slowly while *min_α* decreases. CCFilter almost does not change while *min_α* varies, which means it does not take any advantage from *min_α*. This is because it spends most of processing time on mining SCP.

Now, we conduct the experiments on the pumsb dataset, which is a dense dataset. Figure 5(a) shows the execution time on the pumsb dataset when *min_α* varies while *min_sup* is fixed at 60%. Figure 5(a) shows that CCMine method outperforms CCFilter method when *min_sup* is less than 60%. When *min_sup* becomes less then 50%, CCFilter run out of memory and cannot finish. Figure 5(b) shows that CCMine method always outperforms CCFilter method over entire range of *min_α*.



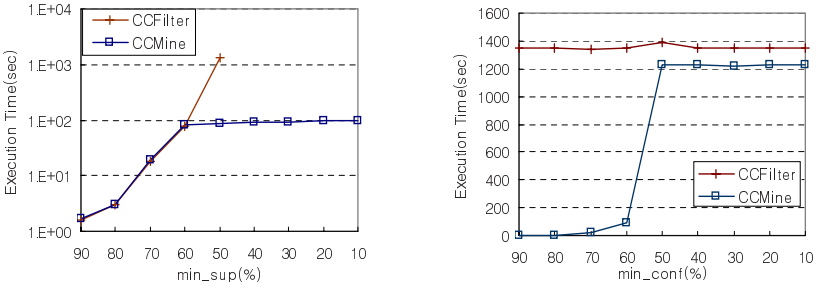
(a) *min_α* = 25%. (b) *min_sup* = 0.01%.

Fig. 4. Execution time on gazelle data set

In summary, experimental results show that the number of confidence closed correlated patterns are quite small in comparison with that of the support-closed patterns. The CCMine method outperforms CCFilter especially when the support threshold is low or the confidence threshold is high.

6 Conclusions

In this paper, we have presented an approach that can effectively reduce the number of correlated patterns to be mined without information loss. We proposed a new notion of confidence-closed correlated patterns. Confidence-closed correlated patterns are those that have no proper superset with the same support



(a) when $min\_\alpha = 60\%$. (b) when $min\_sup = 50\%$.

Fig. 5. Execution time on the pumsb dataset.

and the same confidence. For efficient mining of those patterns, we presented the CCMine algorithm. Several pruning methods have been developed that reduce the search space. Our performance study shows that confidence-closed, correlated pattern mining reduces the number of patterns by at least an order of magnitude in comparison with correlated (non-closed) pattern mining. It also shows that CCMine outperforms CCFilter in terms of runtime and scalability. Overall, it indicates that confidence-closed pattern mining is a valuable approach to condensing correlated patterns.

As indicated in the previous studies of mining correlated patterns, such as [6,5,8], all_confidence is one of several favorable correlation measures, with null-invariance property. Based on our examination, CCMine can be easily extended to mining some correlation measures, such as coherence or bond [6,5,8]. It is an interesting research issue to systematically develop other mining methodologies, such as constraint-based mining, approximate pattern mining, etc. under the framework of mining confidence-closed correlated patterns.

References

1. C. C. Aggarwal and P. S. Yu. A new framework for itemset generation. In *Proc. 1998 ACM Symp. Principles of Database Systems (PODS'98)*, pages 18–24, Seattle, WA, June 1999
2. S. Brin, R. Motwani, and C. Silverstein. Beyond market basket: Generalizing association rules to correlations. In *Proc. 1997 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'97)*, pages 265–276, Tucson, Arizona, May 1997.
3. S. Morishita and J. Sese. Traversing itemset lattice with statistical metric pruning. In *Proc. 2000 ACM SIGMOD-SIGACT-SIGART Symp. Principles of Database Systems (PODS'00)*, pages 226–236, Dallas, TX, May 2001.
4. P.-N. Tan, V. Kumar, and J. Srivastava. Selecting the right interestingness measure for association patterns. In *Proc. 2002 ACM SIGKDD Int. Conf. Knowledge Discovery in Databases (KDD'02)*, pages 32–41, Edmonton, Canada, July 2002.
5. E. Omiecinski. Alternative interest measures for mining associations. *IEEE Trans. Knowledge and Data Engineering*, 15:57–69, 2003.

6. Y.-K. Lee, W.-Y. Kim, D. Cai, and J. Han. CoMine: Efficient Mining of Correlated Patterns. In *Proc. 2003 Int. Conf. Data Mining (ICDM'03)*, pages 581–584, Melbourne, FL, Nov. 2003.
7. S. Ma and J. L. Hellerstein. Mining mutually dependent patterns. In *Proc. 2001 Int. Conf. Data Mining (ICDM'01)*, pages 409–416, San Jose, CA, Nov. 2001.
8. H. Xiong, P.-N. Tan, and V. Kumar. Mining Strong Affinity Association Patterns in Data Sets with Skewed Support Distribution. In *Proc. 2003 Int. Conf. Data Mining (ICDM'03)*, pages 387–394, Melbourne, FL, Nov. 2003.
9. N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. In *Proc. 7th Int. Conf. Database Theory (ICDT'99)*, pages 398–416, Jerusalem, Israel, Jan. 1999.
10. R. J. Bayardo. Efficiently mining long patterns from databases. In *Proc. 1998 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'98)*, pages 85–93, Seattle, WA, June 1998.
11. J. Pei, G. Dong, W. Zou, and J. Han. On computing condensed frequent pattern bases. In *Proc. 2002 Int. Conf. on Data Mining (ICDM'02)*, pages 378–385, Maebashi, Japan, Dec. 2002.
12. M. Zaki. Generating Non-redundant Association Rules. In *Proc. 2000 ACM SIGKDD Int. Conf. Knowledge Discovery in Databases (KDD'00)*, Aug. 2000.
13. J. Wang, J. Han, and J. Pei. Closet+: Searching for the best strategies for mining frequent closed itemsets. In *Proc. 2003 ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD'03)*, Washington, D.C., Aug. 2003.
14. J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *Proc. 2000 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'00)*, pages 1–12, Dallas, TX, May 2000.
15. M. Zaki and C. Hsiao. CHARM: An Efficient Algorithm for Closed Itemset Mining, In *Proc. SDM'02*, Apr. 2002.