CHAPTER VII

CONCLUSION

7.1 Summary of Dissertation

Recently, Tanbeer et al. proposed an approach for considering the occurrence behavior of patterns (Tanbeer et al., 2009), *i.e.* whether the pattern occur regularly, irregularly or mostly in specific time period of a transactional database. Hence, a pattern is a regular-frequent if it is frequent in terms of the support measure, as defined in (Agrawal and Srikant, 1994), and if it regularly appears (measure of regularity/ periodicity of the pattern which considers the maximum period at which the pattern occurs). To discover a set of regular-frequent itemsets, the authors proposed a highly compact tree structure named *Periodic Frequent patterns tree (PF-tree)* to maintain the database content, and a pattern growth-based algorithm to mine a complete set of regular-frequent itemsets with user-given support and regularity thresholds.

However, it is well-known that the support-based approaches tend to produce a huge number of patterns and it is not easy for end-users to determine a suitable support threshold. Thus, the top-*k* significant patterns mining framework, which allows the users controlling the number of patterns (k) to be mined (which is easy to specify) without a support threshold, is an interesting approach (Han et al., 2002). Therefore, the contributions of this dissertation have focused on the problem of mining top-*k* regular-frequent itemsets as follows.

Chapter 3 introduced the the problem of mining k regular itemsets with the highest supports, called *Top-K Regular-frequent Itemsets Mining*, that allows users to specify the number of regular-frequent itemsets to be mined. From this problem, the users have to specify two parameters: (*i*) a number of desired results (k) (*i.e.* specify the value of k instead of setting the support threshold); and (*ii*) a regularity threshold (*i.e.* to see whether an itemset occurs regularly). Consequently, an efficient algorithm named *Mining Top-K Periodic(Regular)-Frequent Pattern (MTKPP)* was proposed. To mine top-*k* regular-frequent itemsets, the top-*k* list structure (with hash table) and the best-first search strategy were also devised for efficiency reasons. From the experimental results, it can be observed that MTKPP ran faster than PF-tree which exactly mines the same results (with the small and large values of k) and scaled linearly relative to the size of the input database. Thus, the MTKPP algorithm is recommended when the users desire to control the number of outputs.

Subsequently, an efficient algorithm called *Top-K Regular-frequent Itemsets Mining with database Partitioning and support Estimation (TKRIMPE)* to mine a set of k regular-frequent itemsets with the highest supports was presented in Chapter 4. TKRIMPE was devised to improve the performance of MTKPP by trying to reduce the processing time in the intersection process. In TKRIMPE, the database partitioning and support estimation techniques were also introduced to dismiss unnecessary computational costs and to cut down the search space. The experimental study shows that TKRIMPE ran faster than MTKPP when the database is sparse, and also has the similar performance on dense datasets.

Chapter 5 proposed an efficient algorithm, called *Top-K Regular-frequent Itemsets based on Interval Tidset representation (TKRIMIT)* to mine top-k regular-frequent itemsets. In addition, a new concise representation, *Interval Tidset* representation, used to collect the set of tids that each considered itemset occurs was introduced. Based on the interval tidset representation, TKRIMIT can reduce the number of maintained tids that each itemset occurs. This is caused to save the memory usage and runtime to mine the top-k regular-frequent itemsets. As shown by results from the experiments, the use of interval tidset representation gives the better performance from the use of normal tidset representation especially on dense datasets.

In Chapter 6, the combination between the database partitioning technique and interval tidset representation was proposed. The *Hybrid representation* was introduced to maintain the tids that each itemset occurs. It composes of normal tidset representation (*i.e.* sets of normal tids that each itemset occurs) and interval tidset representation (*i.e.* sets of concise (wrap up) tids that each itemset appears). By using this representation, a simple heuristic was devised to choose a proper presentation for the occurrence behavior of each itemset. Consequently, an efficient algorithm based on database partitioning technique and the hybrid representation called *Hybrid representation on Top-K Regular-frequent Itemsets Mining based on database Partitioning (H-TKRIMP)* was introduced. As shown in the experiments, H-TKRIMP can run faster than the other algorithms on both sparse and dense datasets with the small and large values of k.

In summary, this dissertation has studied the regular-frequent itemsets mining problem and then proposed the problem of top-k regular-frequent itemsets mining which allows users to control the number of results to be mined. To mine the top-k regular-frequent itemsets, the efficient and scalable single-pass algorithms based on the top-k list structure have been suggested. They consist of two steps: (*i*) top-k initialization: scan database to construct the top-k list of regular items with their supports, regularities, and tidsets; and (*ii*) top-k mining: merge each pair of entries in the top-k list using the best-first search strategy (*i.e.* first consider the itemsets with the highest supports), then intersect their tidsets to calculate regularity and support of each itemset.

From both steps, it can be observed that the mining process consumes high processing time in the intersection process. Therefore, the partitioning and estimation techniques were invented to reduce the cost of intersection by dismissing some unnecessary computing. From the experiment (as mentioned in Chater 4), it can be seen that applying both techniques can help algorithms to achieve a good performance especially on sparse datasets with the small and large values of k. In addition, to gain a better performance on dense datasets, the number of maintained tids was emphasized. If the number of maintained tids is very few, mining algorithms would spend few time in the intersection process. Thus, a new concise representation was devised to reduce the number of maintained tids of each itemset in the top-*k* list. It uses only one positive and one negative tids to represent a group of consecutive continuous tids. By using this representation, the performance of mining algorithms grow up in terms of time and space especially on dense datasets. Finally, to have a good performance on both sparse and dense datasets without knowing the characteristic of datasets in advance, the combination of database partitioning technique and the concise representation was proposed. Then, the hybrid representation was also devised to maintain tidsets followed by occurrence behavior of each itemset. If the itemset occurs frequent, the concise representation is applied. Otherwise, the original representation is employed. The results show that the use of hybrid representation and partitioning technique can give a good performance on both sparse and dense dataset with the small and large values of k comparing with the other proposed algorithms.

7.2 Discussion

Although, this dissertation introduced the top-*k* regular-frequent itemsets mining and some efficient algorithms that achieve a good performance on both sparse and dense datasets with the small and large values of k, there exists some limitation which can be categorized into several points.

First, the proposed algorithms are based on single scanning and maintaining the tids for each itemset in the top-*k* list. Though, there exists a concise representation which helps to save runtime and memory space, the mining algorithms still spend a lot of time in the intersection process and a lot of memory to maintain tids. Then, the interesting problem is to design a new algorithm that can share the common sets of tids among the itemsets in the top-*k* list. This way of doing may help the mining algorithms to save time to intersect and space to maintain tids during mining process.

Second, to discover the top-*k* regular-frequent itemsets in the presence memory constraint, the proposed algorithms would have a problem on memory consumption because they have to maintain tidset for each itemset the set of results. Thus, a new approach using the secondary storage or using incremental technique to separately consider each partition of database should be discussed in the direction of reducing required memory.

Third, the problem of top-k regular-frequent itemsets mining require two parameters: (*i*) regularity threshold (σ_r) and (*ii*) the number of desired results (k). In some cases, users would suffer from the setting a suitable regularity threshold. Thus, the interesting problem is to automatically specify the regularity threshold to mine the top-k regular-frequent itemsets. To come up with an appropriate regularity threshold, one needs to have detailed knowledge about the mining query and the task-specific data, and be able to estimate, without mining, how many itemsets would be generated with a particular threshold. Unlike a support threshold, the setting of a regularity threshold is quite subtle: a too large threshold may lead to the generation of thousands of itemsets, whereas a too small one may often generate very few or no answers. Therefore, the avoiding of the setting of regularity threshold by using other criteria to find the suitable threshold might become an important task.

Fourth, the problem of top-k regular-frequent itemsets mining works only on static database (*i.e.* no updated record). Therefore, another interesting direction is to study the problem of mining top-k regular-frequent itemsets mining from incremental databases and data streams. In the past few years, research in data streams (also incremental databases) has attracted a lot of researchers. A data stream is a continuous, unbounded, and timely ordered sequence of data elements generated at a rapid rate. Unlike traditional static databases, stream data, in general, has additional processing requirements; *i.e.*, each data element should be examined at most once and processed as fast as possible with the limitation of available memory. Even though mining user-interest based patterns from data stream has become a challenging issue, interests in online stream mining for discovering such patterns dramatically increased. Hence, to find top-k regular-frequent itemsets efficiently from data streams, an efficient algorithm that can capture the stream content with one scan and can competently mine the resultant itemsets is required. Since the proposed algorithms scan database once, then it can be improved the algorithms to directly mine top-k regular-frequent itemsets from data streams.

The author strongly believe that, with the proposed algorithms and the proposed approach, it could seen many interesting, or the ultimate, solution to the mining regular patterns in the near future.