

Chapter 2

RELATED WORK

Sequential pattern discovery has been an active research topic in past several years. The application of sequential pattern mining spans over a wide range, from analyzing user access patterns of a web site to the protein motif discovery, from studying the workload of a large computer system to the child-abuse cases, etc. The diversity of the applications suggest that it may not be possible to apply a single sequential pattern model to all these problems. Each application may require a unique model and solution. A number of research projects were established in recent years to develop meaningful sequential pattern models and efficient algorithms for mining these patterns. Most of these models belong to one of the following four categories, frequent patterns, periodic patterns, statistically significant patterns, and approximate patterns. We will describe some of the state of art achievement within these fields in this chapter.

1. Frequent Patterns

In many applications, the frequency can be viewed as a very useful metric to indicate the importance of a pattern. If a pattern occurs a large number of times in a data set, then this pattern may be important in characterizing or analyzing the data set. For instance, in a long genome (DNA) sequence, a pattern that occurs a large number of times may correspond to a tandem repeat, which can be very interesting to a molecular biologist. In the application of intrusion detection, there may be a set of sequences each of which represents the system calls of an intrusion process. A pattern that occurs in a large percentage of these sequences may be a signature for the intrusion.

The frequent pattern is one of earliest problems studied in the realm of the sequential pattern mining. It was first studied in [2, 19]. The input data is a set of sequences, where each sequence is a list of sets. For instance, a sequence can be in the form of $\{a\}\{b, c\}\{d\}$ where the first occurred set of items is a , the

secondly occurred set of items are b and c and etc. b and c can be considered as occurring simultaneously. We say that a sequence S supports a pattern P if S contains P (or P appears in S). In the above example, $\{a\}\{b, c\}\{d\}$ supports pattern $\{a\}\{b\}$, $\{b, c\}\{d\}$, and $\{a\}\{d\}$. The support of a pattern P in a data set is the percentage of the sequences that support P . These patterns whose support is above a user-defined threshold are called frequent sequential patterns. The goal of frequent sequential pattern mining is to find all frequent sequential patterns from a given data set with a user specific support threshold.

The authors in [1] used a technique similar to that of level-wise mining algorithm of the frequent itemset mining for discovering frequent sequential patterns. The main idea behind the Apriori algorithm is that if a longer pattern P is qualified as a frequent pattern, any of P 's sub-patterns should also be qualified as a frequent pattern, which is known as the Apriori property. Formally, the Apriori property can be described as the following.

PROPERTY 1.1 Apriori Property. *Let Π be a set of sequences, P be a sequential pattern, and $\text{supp}(P)$ be the support of P in Π . We have, $\text{supp}(P) \leq \min_{P' \subset P}(\text{supp}(P'))$ where P' is a sub-pattern (subsequence) of P .*

The Apriori property can be directly derived from the definition of support. In order for a pattern P to appear in a sequence S , every sub-pattern of P has to appear in S . This means that S supports every sub-pattern of P , and in turn, the support of P is less than or equal to that of every sub-pattern of P .

Based on the Apriori property, the authors in [1] devise a level-wise search algorithm. The algorithm proceeds one level at a time. In level i , it finds i -patterns (i.e., patterns with i positions or sets). In the first level, the algorithm finds the frequent patterns with only one set. From these 1-patterns, it constructs the candidate set of 2-patterns in the following manner. If $\{a\}$ and $\{b, c\}$ are the frequent 1-patterns, then it is possible that $\{a\}\{b, c\}$ and $\{b, c\}\{a\}$ could be frequent 2-patterns. Next, the actual support of the 2-patterns are computed based on the entire data set. The algorithm terminates at level j when there is no any frequent pattern with j sets found.

However, this method may not be efficient if the pattern is very long, i.e., consists of a large number of positions. To further improve the performance, projection-based algorithms such as PrefixSpan [16] was introduced to reduce the candidate patterns and hence reduce the number of scans of the data. The main technique of projection-based pattern mining is to search the patterns in a depth-first fashion. After discovering pattern P as a frequent pattern, the algorithm searches the patterns with P as the prefix. For instance, if pattern a is deemed frequent, next, pattern $\{a\}\{b\}$, $\{a\}\{c\}, \dots$ will be searched. Due to the order of pattern search, the data set can be partitioned in the following manner. To compute the support of patterns having P as the prefix, we only need to search in the sequences that contain P as a subsequence. Therefore,

during the depth-first search, we can recursively partition (project) the data sets according to the prefixes. As a result, the search time can be significantly reduced.

2. Regular Patterns

The above frequent sequential pattern mining algorithms discover the sequential patterns that occur many times in a set of sequences. However, it does not care the position where a pattern occurs. In some application, a user may not be interested in the frequent patterns but rather be interested in the patterns that occur in some regularity. For instance, if we can find sequential patterns that occur in some periodicity, then we can predict the occurrences of some event in the future and better understand the inherent characteristics of the underlying data set.

There are two kinds of regularity. One is the cyclic patterns and the other is periodic patterns. The cyclic pattern was proposed in [13], which is also called cyclic association rules. The input data to [13] is a set of transactions, each of which consists of a set of items. In addition, each transaction is tagged with an execution time. The goal is to find association rules that repeat themselves throughout the input data. An association rule may exhibit a cyclic behavior which can be represented as (l, o) . An association rule has a cycle (l, o) if the association rule holds in every l th time unit starting at time unit o . For instance, if the unit of time is an hour and “coffee \rightarrow dough-nuts” holds during the interval 7AM to 8AM every day (i.e., every 24 hours), the rule “coffee \rightarrow dough-nuts” has a cycle $(24, 7)$. The authors proposed two methods for discovering the cyclic patterns. The main ideas behind these approaches is to use some pruning techniques to reduce the computation. For instance, if itemset A and B have different non-overlapped cycles, then it is impossible for the rule $A \rightarrow B$ to exhibit cyclic behavior.

For the periodic patterns, the input is a long sequence of sets of items. The goal is to find the subsequences that exhibit the periodicity in the input sequence. Assume that $\{a\}\{b\}\{c\}\{a\}\{b\}\{c\}\{a\}\{b\}\{c\}$ is the input sequence. The pattern $\{a\}\{b\}\{c\}$ is called periodic pattern because it repeats itself with period 3. This is also called full periodic pattern because every position in the pattern exhibits the periodicity. However, in many applications, not every position may exhibit the periodic behavior. For example, in a set of custom purchase transactions, a periodic trend may be only observed at certain time of a day. For instance, let $\{a\}\{a\}\{c\}\{a\}\{b\}\{c\}\{a\}\{c\}\{c\}$ be the input sequence, then there is no full periodic pattern with length 3. However, when the constraint is relaxed, e.g., some position can be don't care, then we can find the pattern $\{a\} * \{c\}$ where $*$ is a wide card and can represent any set of items. This is called *partial periodic pattern*.

Han et. al. [9] presented algorithms for efficiently mining these partial periodic patterns by exploring some interesting properties related to partial periodicity such as the Apriori property and the max-sub-pattern hit set property. In essence, to discover patterns with periodicity of l , the algorithm divides input sequence into a set of contiguous segments with length l , each of which can be viewed as an independent sequence, then a frequent sequential pattern mining algorithm is used. Although the partial periodic pattern model can capture the patterns where some position is non-specified, it could not represent the patterns whose occurrences are asynchronous. For instance, if there is some noise in the input sequence and some sets are missed or extra sets are added (which is a common phenomenon in many real applications), e.g., the input sequence may become $\{a\}\{a\}\{c\}\{a\}\{c\}\{a\}\{c\}\{c\}$ due to noises, then there does not exist any partial periodic pattern with periodicity of 3.

3. Statistically Significant Patterns

In many applications, the symbols in a sequence have very skewed distribution. For instance, in a trace of messages of some router, some message type occurs very rarely, e.g., “the link connected to the router is saturated” while other type of messages may occur commonly, e.g., “I am alive” message. As a result, the patterns with common messages should be expected to have a higher support than that of these with the rare messages. If we use the uniform support (number of occurrences) as a measure of importance, then we would miss these patterns with rare events. Researchers have been investigating this problem in various data mining applications.

Brin et al. [4] first introduced the concept of correlation and it was shown that in many applications the correlation measurement can reveal some very important patterns. The Chi-squared test was used to test the correlation among items. For example, by analyzing the words in the *clari.world.africa* news article, Brin et. al. found that there exist some very strong correlation among some words combination, e.g., “nelson” and “mandela”, but their support is relatively low. In this approach, the itemsets form a lattice based on the superset-subset relationship. Instead of explicitly enumerating all correlated itemsets, the border comprising the set of minimal correlated itemsets¹ is identified, and no further distinction is made on the degree of correlation of itemsets above the border (i.e., supersets of some itemset on the border). This model sometimes becomes sub-optimal. As shown in Figure 2.1, itemsets A and B are highly correlated but C is independent of them². In addition, $\{A, B, C, D\}$ is also highly correlated. We can view that the degree of correlation of $\{A, B, C\}$ is not as strong as that of $\{A, B\}$ and $\{A, B, C, D\}$. This observation can also be confirmed by the Chi-squared test³. In many applications, users are only interested in the itemsets such as $\{A, B\}$ and $\{A, B, C, D\}$, but not $\{A, B, C\}$. However, [4] cannot distinguish between $\{A, B, C\}$ and $\{A, B, C, D\}$ once $\{A, B\}$ is

Transaction ID	Items
1	ABCD
2	ABFG
3	CEGF
4	ABCD
5	CEGH
6	CEFH

Figure 2.1. An Example of Transaction Set

identified as a correlated itemset. Furthermore, if a user is interested in finding k itemsets with the highest correlation, then all itemsets in the lattice have to be examined before k highest ones can be determined. Another potential drawback of this model is the expensive computation required by this model. The running time of all patterns with i -correlated items is $O(n \times |CAND| \times \min\{n, 2^i\})$ where n and $|CAND|$ are the number of transactions and the number of candidates at the i th level, respectively. To overcome these drawbacks, Oates et al. [11, 12] proposed models for statistical dependencies using G statistic and devised randomized algorithms to produce approximate results.

More recently, Cohen et al. [6] and Fujiwara et al. [8] address the problem of identifying pairs of attributes with high confidence or similarity (in terms of probabilistic correlations in the database) in the absence of support requirement. Hashing based algorithms [6] are proposed to tackle the problem, which consist of three general phases: computing hash signature, generating candidates, and pruning candidates. To avoid both false negatives and false positives that may be yielded with the hashing based scheme, a family of so called *dynamic miss counting* algorithms are proposed in [8]. Instead of counting the number of hits (as most other algorithms do), the number of transactions where the given pair of attributes disagree is counted and this counter is deleted as soon as the number of misses exceeds the maximum number of allowed misses for that pair. This strategy is proved to be able to reduce the memory size significantly.

Another important advance is accomplished in mining so-called *unexpected patterns*. Berger et al. [3] proposed a probabilistic measure of interestingness based on unexpectedness in the context of temporal logic, whereby a pattern is deemed interesting if the ratio of the actual number of occurrences of the pattern exceeds the expected one by some user defined threshold. Solving the problem in its general frame is in nature NP-hard and hence some heuristics are proposed to produce an approximate answer. Padmanabhan et al. [14, 15, 18] define the unexpectedness of association rules relative to a system of prior

beliefs. Specifically, the belief is of the form $X \rightarrow Y$ and a rule is said to be unexpected if it contradicts the belief. The set of beliefs (given by the user) are used to conduct the mining process efficiently so that an exhaustive search is avoided. The primary advantage of this model is that it can customize the mining process for the users who have fairly good prior knowledge and specific interests, and is particularly useful in refinements of user's beliefs.

A formal study of surprising patterns is furnished in [5], focusing on the analysis of variation of inter-item correlations along time. The surprise is defined in terms of the coding length in a carefully chosen encoding scheme and has solid theoretic foundation, but requires much more expensive computation comparing to other models.

4. Approximate Patterns

Approximate frequent itemset mining is studied in [17, 20]. Although the two methods are quite different in techniques, they both explored approximate matching among itemsets. Instead of perfect matches, the model in [17, 20] allow imperfect matches. A transaction supports an itemset I if a large portion of the items in I , e.g., more than 95% occur in the transaction. Algorithms similar to mining perfect itemsets are devised to mine the approximate itemsets.

In [7], Chudova and Smyth used a Bayes error rate framework under a Markov assumption to analyze different factors that influence string pattern mining in computational biology. Extending the theoretical framework to mining sequences of sets could shed more light to the future research direction.

Notes

- 1 A minimal correlated itemset is a correlated itemset whose subsets are all independent.
- 2 $Prob(AB) \times Prob(C) = \frac{1}{2} \times \frac{2}{3} = Prob(ABC)$.
- 3 In general, the chi-squared test requires a large sample. For the demonstration purpose only, we assume that the chi-squared test is valid in this example.

References

- [1] Agrawal, R., and Srikant, R. (1994). Fast algorithms for mining association rules in large databases. *Proc. of the Int'l Conference on Very Large Databases (VLDB)*. pp. 487-499.
- [2] Agrawal, R., and Srikant, R. (1995). Mining sequential patterns. *Proc. of the Int'l Conference on Data Engineering (ICDE)*. pp. 3-14.
- [3] Berger, G., and Tuzhilin, A. (1998). Discovering unexpected patterns in temporal data using temporal logic. *Temporal Databases - Research and Practice, Lecture Notes on Computer Sciences*. vol. (1399) pp. 281-309.

- [4] Brin, S., Motwani, R., and Silverstein, C. (1997). Beyond market baskets: generalizing association rules to correlations. *Proc. ACM SIGMOD Int'l. Conference on Management of Data (SIGMOD)*. pp. 265-276.
- [5] Chakrabarti, S., Sarawagi, S., and Dom, B. (1998). Mining surprising patterns using temporal description length. *Proc. Int. Conf. on Very Large Data Bases (VLDB)*. pp. 606-617.
- [6] Cohen, E., Datar, M., Fujiwara, S., Cionis, A., Indyk, P., Motwani, R., Ullman, J., and Yang, C. (2000). Finding interesting associations without support pruning. *Proc. 16th IEEE Int'l. Conference on Data Engineering (ICDE)*. pp. 489-499.
- [7] Chudova, D., and Smyth, P. (2002). Pattern discovery in sequences under a markov assumption. *Proc. of the Eighth ACM Int'l Conference on Knowledge Discover and Data Mining (KDD)*. pp. 153-162.
- [8] Fujiwara, S., Ullman, J., and Motwani, R. (2000) Dynamic miss-counting algorithms: finding implication and similarity rules with confidence pruning. *Proc. 16th IEEE Int'l. Conference on Data Engineering (ICDE)*. pp. 501-511.
- [9] Han, J., Dong, G., and Yin, Y. (1999) Efficient mining partial periodic patterns in time series database. *Proc. of IEEE Int'l. Conf. on Data Engineering*. pp. 106-115.
- [10] Han, J., Pei, J., Mortazavi-Asl, B., Chen, Q., Dayal, U., and Hsu, M. (2000). FreeSpan: frequent pattern-projected sequential pattern mining. *Proc. of the Sixth ACM Int'l Conference on Knowledge Discover and Data Mining (KDD)*. pp. 355-359.
- [11] Oates, T. (1999) Identifying distinctive subsequences in multivariate time series by clustering. *Proc. of the Fifth ACM Int'l Conference on Knowledge Discover and Data Mining (KDD)*. pp. 322-326.
- [12] T. Oates, M. D. Schmill, and P. R. Cohen. (1999) Efficient mining of statistical dependencies. *Proc. 16th Int'l. Joint Conf. on Artificial Intelligence (JCAI)*. pp. 794-799.
- [13] Ozden, B., Ramaswamy, S., and Silberschatz, A. (1998). Cyclic association rules. *Proc. 14th Int'l. Conference on Data Engineering (ICDE)*. pp. 412-421.
- [14] Padmanabhan B., and Tuzhilin, A. (1998) A belief-driven method for discovering unexpected patterns. *Proc. of ACM Int'l Conference on Knowledge Discover and Data Mining (KDD)*. pp. 94-100.
- [15] Padmanabhan, B., and Tuzhilin, A. (2000). Small is beautiful: discovering the minimal set of unexpected patterns. *Proc. of ACM Int'l Conference on Knowledge Discvoer and Data Mining (KDD)*. pp. 54-63.
- [16] Pei, J., Han, J., Mortazavi-Asl, B., Pinto, H., Chen, Q., and Dayal, U., and Hsu, M. (2001). PrefixSpan: mining sequential patterns by prefix-projected growth. *IEEE Int'l Conference on Data Engineering (ICDE)*. pp. 215-224.
- [17] Pei, J., Tung, A., and Han, J. (2001). Fault-tolerant frequent pattern mining: problems and challenges. *Proc. ACM SIGMOD Int'l Workshop on Data Mining and Knowledge Discovery (DMKD)*. pp. 7-12.

- [18] Silberschatz, A., and Tuzhilin, A. (1996). What makes patterns interesting in knowledge discover systems. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, vol. 8 no. 6, pp. 970-974.
- [19] Srikant, R., and Agrawal, R. (1996). Mining sequential patterns: generalizations and performance improvements. *Proc. of Int'l Conference on Extended Database Technologies (EDBT)*, pp. 3-17.
- [20] Yang, C., Fayyad, U., and Bradley, P. (2001). Efficient discovery of error-tolerant frequent itemsets in high dimensions. *Proc. of ACM Int'l Conference on Knowledge Discover and Data Mining (KDD)*, pp. 194-203.

Mining Sequential Patterns from Large Data Sets

Wang, W.; Yang, J.

2005, XV, 163 p., Hardcover

ISBN: 978-0-387-24246-0