International Journal of Advanced Research in

Computer Science Engineering and Information Technology

Volume: 3 Issue: 1 26-Jun-2014,ISSN_NO: 2321-3337

Literature Survey on Constrained Frequent Pattern Mining

P.Subhashini[1] , Dr.Gunasekaran[2]

Research Scholar , Dept.of Information Technology, St.Peter's University, Chennai, India[1].

Professor, Meenakshi College of Engineering, Chennai, India[2]

***Abstract*—Frequent pattern mining is a heavily researched area in the field of data mining with wide range of applications. Mining frequent patterns from large scale databases has emerged as an important problem in data mining and knowledge discovery community. Frequent Pattern Mining often generates a very large number of patterns and rules, which reduces not only the efficiency but also the effectiveness of mining. Recent work has highlighted the importance of constraint based mining paradigm in the context of mining frequent itemsets, associations, correlations, sequential patterns, and many other interesting patterns in large databases. Constraint based frequent pattern mining has been proved to be effective in reducing the search space in the frequent pattern mining task and thus in improving efficiency. We survey frequent pattern mining under various constraints which will give some ideas for the future researchers.**

Index Terms*—patterns, knowledge discovery, associations, correlations.(keywords)*

1. INTRODUCTION

Frequent pattern plays an important role in all data mining task such as clustering, classification, prediction and association analysis. Frequent pattern mining research has substantially broadened the scope of data analysis and will have deep impact on data mining methodologies and applications in the long run. These frequent patterns can be formed either from the precise or uncertain data. Precise data are the data that will be either present or absent. In Uncertain Data the Users are uncertain about the presence or absence of data. Each transaction contains items and their existential probabilities.

1.1 Importance of constraint based Mining:

Frequent pattern mining usually produces too many solution patterns. This situation is harmful for two reasons:

1.Performance: mining is usually inefficient or often simply unfeasible

2.Identification of fragments of interesting knowledge: which is blurred within a huge quantity of small, mostly useless patterns

1.2 Constraints are the solution to these problems:

1.they can be pushed in the frequent pattern computation exploiting them in pruning the search space, thus reducing time and resources requirements.

2.they provide to the user guidance over the mining process and a way of focusing on interesting knowledge.

With constraints we can obtain less patterns which are more interesting.

2. Literature survey

2.1Classification of constraints based on semantics:

Item Constraint:

An item constraint specifies what are the particular individual or group of items that should not be present in pattern. For example a soap company may be interested in patterns containing only soap products, when it mines transactions in a grocery store.

Length Constraint:

A length constraint specifies the requirement on the length of patterns, i.e, the number of items in the patterns. For example when mining classification rules for document, a user may be interested in only frequent patterns with atleast 5 keywords.

Model-based constraint:

A Model-based constraint looks for patterns which are sub or super patterns of some given patterns (models). For example a car dealer may be interested in knowing what are all the other accessory items a purchaser would buy when he buys a car.

Aggregate Constraint:

An Aggregate constraint is on an aggregate of items in a pattern, where the aggregate function can be SUM, AVG, MAX, MIN, etc. For example a marketing analyst may like to find pattern where the average price is over $150.

User Constraint:

User constraints are those in which user can use a rich set of SQL-style constraints to guide the mining process to find only those frequent patterns— containing market basket items—that satisfy the user constraints.

Examples of these constraints include the following:

$C1 \equiv \min(S.Price) \geq \$10$ and $C2 \equiv S.Type = snack$.

Here, constraint C1 says that the minimum price of all items in a pattern/set S is at least $10; constraint C2 says that all items in a pattern S are snack. It is important to note that, besides these market basket items, the set of constraints can also be imposed on individuals, events, or objects in other domains. The following are some examples: $C3 \equiv \max(S.Temperature) \leq 36.8°C$, $C4 \equiv S.Symptom \subseteq \{fever, runny\ nose\}$, $C5 \equiv S.Day \supseteq$

{Saturday, Sunday}, C6 ≡ min(S.Weight) ≤ 23kg, and C7 ≡ avg(S.Weight) ≤ 23kg. Here, constraint C3 says that the maximum (body) temperature of all individuals in a pattern/set S is at most 36.8°C; constraint C4 says that individuals in S suffer only from fever or runny nose. Similarly, constraint C5 say that all the events in a pattern S must span over the weekend (Saturday and Sunday). Constraints C6 and C7, respectively, say that the minimum and the average weights of all the objects in S is at most 23kg.

2.2 Classification of constraints based on properties:

Monotonicity:

When an intemset S satisfies the constraint, so does any of its superset.

sum(S.Price) ≥v is monotone

min(S.Price) ≤v is monotone

Anti-monotonicity:

When an intemset S satisfies the constraint, so does any of its subset

- Frequency is an anti-monotone constraint.

Succinctness:

Given A1, the set of items satisfying a succinct constraint C, then any set S satisfying C is based on A1 , i.e., S contains a subset belonging to A1.

Idea: whether an itemset S satisfies constraint C can be determined based on the singleton items which are in S

min(S.Price)≤v is succinct

sum(S.Price) ≥v is not succinct

Convertible anti-monotone:

If an itemset S violates a constraint C, so does every itemset having S as a prefix w.r.t. R. Let R be the order of items.

Ex. avg(S) ≤ □v w.r.t. item value descending order

Convertible monotone:

If an itemset S satisfies constraint C, so does every itemset having S as a prefix w.r.t. R. Let R be the order of items.

Ex. avg(S) ≥v w.r.t. item value descending order

Antimonotone, monotone and Succinct constraints can be formally defined as:

Definition 2.1. Given an itemset X which is contained in a transaction(tid, Y) if X ⊆ Y. A constraint CAM is antimonotone if

$$\forall Y \subseteq X : CAM(X) \Rightarrow CAM(Y).$$

If CAM holds for X then it holds for any subset of X. The frequency constraint is clearly antimonotone. This property is used by the Apriori algorithm with the following heuristic: if an itemset X does not satisfy Cfreq, then no superset of X can satisfy Cfreq, and hence they can be pruned. This pruning can affect a large part of the search space, since itemsets form a lattice. Therefore the Apriori algorithm operates in a level-wise fashion moving bottom-up on the itemset lattice, and each time it finds an infrequent itemset it prunes away all its supersets.

Definition 2.2. Given an itemsetX, a constraint CM is monotone if:

$$\forall Y \supseteq X : CM(X) \Rightarrow CM(Y).$$

If CM holds for X then it holds for any superset of X.

User constraint can be categorized into several overlapping classes according to the nice properties that they possess. One of these classes is the succinct constraint, and its formal definition is given below.

Definition 2.3: Succinct Constraint.

Let Item be the set of domain items, and let 2Item denote the powerset of Item. Then, the succinct constraint can be defined in several steps, as follows:

(i) *An itemset SSj*⊆Item *is a* succinct set *if SSj can be expressed as a result of selection operation* σ_p(Item),*where* σ *is the usual selection operator and p is a selection predicate.*

(ii) *A powerset of items SP* ⊆2Item *is a* succinct powerset*if there is a fixed number of succinct sets SS*1, . . . , *SSk*⊆Item *such that SP can be expressed in terms of the powersets of SS*1, . . . , *SSk using set union and/or set difference operators.*

(iii) *A constraint C is* succinct *provided that the set of patterns/itemsets satisfying C is a succinct powerset.*

With the above definition, constraints *C*1–*C*6(mentioned above) are *succinct*. For example, the set of patterns/itemsets satisfying *C*1 can be theoretically expressed as 2$\sigma_{Price \geq \$10}$(Item) , which is a succinct powerset. Similarly, the sets of itemsets satisfying the other five succinct constraints (*C*2–*C*6) are also succinct powersets. Practically, one can directly generate precisely all and only those patterns satisfying these constraints by using precise "formulas"—called member generating functions that do not require generating and

excluding patterns not satisfying the constraints. For example, patterns satisfying *C*6 can be precisely generated by combining at least one object of weight ≤ 23kg with some optional objects (of any weight), thereby avoiding the substantial overhead of the generation and exclusion of invalid patterns. It is important to note that a majority of constraints are succinct and many non-succinct constraints can be induced into weaker constraints that are succinct (e.g.,non-succinct constraint *C*7 can be induced into succinct constraint*C*6 because all frequent patterns satisfying *C*7 must satisfy *C*6).

Succinct constraints can be further divided into subclasses—such as succinct anti-monotone (SAM) constraints and succinct non-anti-monotone (SUC) constraints—based on additional properties they possessed. For instance, among the above succinct constraints, *C*1–*C*4 are *SAM constraints* because they possess an additional property of anti-monotonicity. With such a property, supersets of any pattern violating the SAM constraints also violate the constraints (e.g., if a pattern *S* contains an item with price lower than $10, then *S* violates *C*1 and so do any supersets of *S*). In contrast, succinct constraints *C*5–*C*6 are *SUC constraints* because they do not possess such an antimonotonicity property. For instance, if the minimum weight of all objects within a pattern *S* is heavier than 23kg, then *S* violates *C*6 but there is no guarantee that all supersets of *S* would violate *C*6. As an example, let *x Weight* be 30kg and *y Weight* be 20kg. Then, $S \cup \{x\}$ and $S \cup \{y\}$ are both supersets of *S*. Between them, the former violates *C*6 but the latter does not violate the constraint.

2.3 Constraints in sequential pattern mining process:

1. Gap Constraint: A gap constraint is defined in sequence databases where each transaction in every sequence has a timestamp. It requires that the sequential patterns in the sequence database must have the property such that the timestamp difference (difference of days) between every two adjacent transactions in a discovered sequential pattern must not be greater than given gap.

2. Compactness Constraint: A compactness constraint requires that the sequential patterns in the sequence database must have the property such that the time-stamp difference (difference of days) between the first and the last transactions in a discovered sequential pattern must not be greater than given period.

3. Recency Constraint: Recency constraint is specified by giving a recency minimum support (*r_minsup*), which is the number of days away from the starting date of the sequence database. For example, if our sequence database is from 27/12/2007 to 31/12/2008 and if we set *r_minsup*= 200 then the recency constraint ensures that the last transaction of the discovered pattern must occur after 27/12/2007+200 days. In other words, suppose the discovered pattern is < (*a*), (*bc*)>, which means “after buying item a, the customer returns to buy item b and item c”. Then, the transaction in the sequence that buys item b and item c must satisfy recency constraint.

4. Monetary Constraint: Monetary constraint is specified by giving monetary minimum support (*m_minsup*). It ensures that the total value of the discovered pattern must be greater

than *m_minsup*. Suppose the pattern is <(*a*), (*bc*)>. Then we can say that a sequence satisfies this pattern with respect to the monetary constraint, if we can find an occurrence of pattern <(*a*), (*bc*)> in this data sequence whose total value must be greater than *m_minsup*.

5. Frequency Constraint: The frequency constraint is defined as each discovered pattern must satisfy minimum support. Frequency constraint is specified by giving frequency minimum support (*f_minsup*). The frequency of a pattern is the percentage of sequences in database that satisfy the recency constraint and monetary constraint.

6. f-pattern(LIkf), rf-pattern(LIkrf), rfm-pattern(LIkrfm) of length k: Let *B* = <*I*1*I*2...*IS*> be a sequence of itemsets. If the percentage of data sequences in database containing *B* as a subsequence, called *f*-support, is no less than *f_minsup*, *B* is called an *f*-pattern. *B* is called an *rf*-pattern if the percentage of data sequences in database containing *B* as a recent subsequence (which satisfies recency constraint), called *rf*-support, is no less than *f_minsup*. Finally, *B* is called an *rfm*-pattern if the percentage of data sequences in database containing *B* as a recent monetary subsequence (which satisfies recency and monetary constraints), called *rfm*-support, is no less than *f_minsup.*

2.4 Constraints on Pattern growth frequent pattern mining:

*For web log mining e*ach piece of user log information is meant for constructing or updating *consensus tree*. Here level constraint and rule constraints can be used.

Level-wise constraint: Level-wise constraint decides existence of a node in same level of a *tree*. *Consensus tree* is initially constructed with preprocessed log information with above said levels. Traversal is done through the path of the *consensus tree* based on the information like time, type of search engine and keywords used in user query. Each node of *consensus tree* will check the *level constraint* when information traverses based on adaptive threshold. While an article travels through the path of the *consensus tree*, each node's $\theta_{i,j}$ value in consensus path will be incremented. $\theta_{i,j}$ is number of articles visited *ith*node in *jth*level of the *consensus tree* (sometimes referred as *support value* of node). A *ith*node in *jth*level of *consensus tree* is deleted when $conf(\theta_{i,j}) < Li$, referred as *Level-wise constraint*. The *confidence value* ($conf(\theta_{i,j})$) obtained using $\theta_{i,j}$ value of the current *ith node* in *jth*level of the *consensus tree*, and sum of all θ value of the *j+1th* level nodes such as

$$conf(\theta_{ij}) = \theta_{ij} / \Sigma_k \theta,\ k = 1,2,..\ j+1$$

Adaptive Threshold value *L for ith*node is calculated as $Li \approx (conf(\theta_{i,j})/\Sigma_{k(}conf(\theta_{i,j+1}))$. Based on the *support* and *confidence value* a node in the *consensus tree* is removed. Deletion of node happens when there is no user event with particular problems.

Rule Constraint: Rule constraint handles two thus are not regarded as the same, preventing possible generalization. Secondly, many keywords are synonyms however, since they are

spelled differently, they cannot be treated the same for possible generalization. Rule constraint handles both issues using mutation factor along with WordNet (Miller, 1995). Rule constraint are applied on each article whether they exist or not, based on the articles accessed information obtained from preprocessed log. Rule constraint is check on leaf node of the *consensus tree* only. Each leaf node contain pointer, points to which bucket (B1, B2, B3 …), article's detail to be stored. Leaf reference is the index structure of extendible hashing technique. Article's name alone fetched from the URL link, and given to the hash function will specify bucket allocated for that article. Bucket size is decided in such way that a single access load into memory leads to faster access of template. Extendible hashing technique decides when bucket is to be estranged and merged. Bucket is estranged or merged when numbers of articles in it are greater or lesser than some threshold value.

3. SUMMARY

The field of constrained frequent pattern mining has gained importance in recent years because of the necessity to reduce unwanted patterns in many applications. This paper surveys the broad areas of work in this rapidly expanding field. We have presented the important constraints in different types of data. While the field will continue to expand over time, it is hoped that this survey will provide an understanding of the foundational issues and a good starting point to practitioners and researchers in focusing on the important and emerging issues in this field.

REFERENCES

[1] Jian Pei, Jiawei Han "Constrined Frequent Pattern Mining-A pattern growth view", SIGKDD Explorations, June 2002 Carson Kai-Sang Leung∗

[2] Carson Kai-Sang Leung, Dale A. Brajczuk, "Efficient Algorithms for the Mining of Constrained Frequent Patterns from Uncertain Data", SIGKDD Explorations Volume 11, Issue 2.

[3] Francesco Bonchi, "FP-growth Mining of Frequent Itemsets + Constraint-based Mining".

[4] C K Bhensdadia, Y P Kosta, "An Efficient Algorithm for Mining Frequent Sequential Patterns and Emerging Patterns with Various Constraints", International Journal of Soft Computing and EngineeringVolume-1, Issue-6, January 2012.

[5] Ramachandra. V. Pujeri, G.M. Karthik, "Constraint based frequent pattern mining for generalized query templates from web log", International Journal of Engineering, Science and Technology Vol. 2, No. 11, 2010, pp. 17-33

[6] F. Bonchi, F. Giannotti, A. Mazzanti, and D. Pedreschi, " ExAMiner: Optimized level-wise frequent pattern mining with monotone constraints", In Proc. of ICDM, 2003.

[7] Jian Pei, Jiawei Han, Behzad Mortazavi-Asl, Jianyong Wang, Helen Pinto, Qiming Chen, Umeshwar Dayal, and Mei-Chun Hsu, "Mining Sequential Patterns by Pattern-Growth: The PrefixSpan Approach", IEEE Transactions on Knowledge and Data Engineering, Vol. 16, No. 10, October 2004.

[8] J. Pei and J. Han. "Can we push more constraints into frequent pattern mining?", Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'00), pages 350–354, N. Y., Aug. 20–23 2000. ACM Press.

[9] Francesco Bonchiyz, Fosca Giannottiy, Alessio Mazzanti,z and Dino Pedreschiz " ExAnte: Anticipated Data Reduction in Constrained Pattern Mining".

[10] William Cheung , Osmar R. Zaïane, "Incremental Mining of Frequent Patterns Without Candidate Generation or Support Constraint", Proceedings of the Seventh International Database Engineering and Applications Symposium (IDEAS'03)

[11] C. Bucila, J. Gehrke, D. Kifer, and W. White. Dualminer, "A dual-pruning algorithm for itemsets with constraints" in Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2002.

[12] G. Grahne, L. Lakshmanan, and X. Wang, "Efficient mining of constrained correlated sets", In 16th International Conference on Data Engineering (ICDE' 00), pages 512–524. IEEE, 2000.