http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 5 (6): 1043-1047, 2006
ISSN 1812-5638

An Efficient Frequent Temporal Pattern Mining Algorithm

[1]B. Sivaselvan and [2]N.P. Gopalan
[1]Department of Computer Science and Engineering, [2]Department of Computer Applications,
National Institute of Technology, Tiruchirapalli, India 620 015

Abstract: Frequent item-set generation, a key step in association mining, is the process of generating item-sets that satisfy a minimum support threshold. EFTP, a new frequent temporal pattern mining algorithm proposed in this study, requires lesser number of repeated scans of original input in comparison to apriori principle based algorithms. Experimental results demonstrate the significant betterment in execution time due to reduced number of input scans and support independence of the proposed algorithm as compared to existing Apriori principle based algorithms.

Key words: Apriori principle, association mining, frequent pattern mining, multimedia data mining, temporal support

INTRODUCTION

Data Mining, an active research field since the 90's has been exhaustively explored with respect to transactional and relational databases. With the growth of internet and massive accumulation of multimedia data (non-transactional), there is an imminent need to extract useful information from them. One such emerging trend is Multimedia Data Mining (MDM) (Zaiane *et al*., 2002a; 1998, 1999). Multimedia data such as audio and video are bound by temporal properties that distinguish them from conventional data. Conventional data mining algorithms lacking these properties are not suited for MDM and there is a research requirement for efficient algorithms in such data domains.

Association mining, classification, clustering and prediction (Han *et al*., 2001), a few of the existing data mining techniques find immediate application in the domain of multimedia data. Images and videos could be subject to classification and clustering to determine their appropriate class or cluster (Zaiane *et al*., 2002b; Wijesekara *et al*., 2000). Association mining could be employed to discover relationship between entities of images or video inputs and hence in classification and prediction (Zhu *et al*., 2003a, b; Tesic *et al*., 2003; Zaiane *et al*., 2002a; 1999). Association Mining, though exhaustively explored with respect to conventional data, is relatively new in multimedia data domain. A key step of association mining, frequent item-set construction, has been a major research area over the years. Frequent set construction in temporal data domain, an emerging research trend is the focus of this study.

Existing frequent pattern mining algorithms in temporal domain are all based on Apriori's (Agrawal *et al*., 1994) candidate generation logic and hence suffer from repeated input scans setback. PrefixSpan (Han *et al*., 2004), a variant of FP-Growth (Frequent Pattern) (Han *et al*., 2000b), proposed for efficient sequential pattern mining requires input to be in the form of transactions or records and does not maintain temporal continuity across transactions. As a result PrefixSpan and other algorithms such as SPADE (Zaki, 2001), do not suit the frequent temporal pattern mining problem discussed. This study proposes an alternate efficient frequent temporal pattern methodology that requires only $|L_1|$ number of repeated scans of original input.

ASSOCIATION RULES

Association Rules were first proposed by (Agrawal *et al*., 1994) as a means of identifying frequently related items in a market basket database consisting of several transactions. An Association Rule is basically an expression of the form X→Y, where X and Y are item-sets. Two statistical measures of significance for association mining are support and confidence (Han *et al*., 2001). Traditionally association mining is viewed as a two step process namely (i) Frequent item-set construction and (ii) Rules generation. The second step is relatively less complex and straightforward in comparison to the first step. Frequent item-set construction aims at generating all possible item-sets that satisfy the minimum support threshold condition. Various algorithms that have evolved

Corresponding Author: B. Sivaselvan, Department of Computer Science and Engineering, National Institute of Technology, Tiruchirapalli, India 620 015

over the years differ in the number of scans required, data structures employed and the methodology used in the construction of frequent item-sets.

Apriori, one of the first algorithms to be proposed is based on a candidate set generation logic. It is based on the anti-monotonic property of set theory which states that "Every subset of a frequent item-set is also frequent". It is a level wise algorithm and first generates frequent 1 item-set or L_1 from candidate 1 item-set or C_1 subject to the support factor. Then it generates higher level candidate item-set from previous level frequent item-set. That is C_2 is generated from L_1. This process is repeated until no further new candidate sets can be generated. Though this approach has an inherent pruning advantage, it suffers from the setback of repeated scans of original input in the process of frequent item-set construction. It requires one overall scan of the original input to decide if a candidate set is frequent or infrequent. Expressing mathematically, it requires $\Sigma_i | C_i |$ scans, where i ranges from 1 to maximum length of frequent item-sets possible and $| C_i |$ represents the cardinality of a candidate item-set at level i. Thus the number of scans with respect to the entire frequent item-set construction process becomes huge and has been a major area of research.

FP-Growth is a major improvement proposed to overcome the repeated scan limitation of Apriori. It is a pattern growth based approach and requires only two overall scans of the original input for the frequent item-set construction process. Dynamic Item-set counting (Brin *et al*., 1997), another algorithm proposed to overcome the repeated scan limitation also reduces the number of scans by constructing frequent item-sets in a simultaneous fashion. Both these algorithms and other algorithms discussed by Goethals (2003) inherently lack the temporal aspect in them. They are suited to transactional database mining that is not bound by temporal properties. Frequent temporal pattern mining requires ordered item-sets or patterns to be generated, a property lacking in the algorithms discussed so far. That is it differentiates a pattern AB from BA.

PROBLEM DEFINITION - FREQUENT TEMPORAL PATTERN MINING

Here we discuss the problem of mining frequent temporal patterns and the existing algorithms for the same. As discussed earlier, the variants of Apriori are not suited for FTP mining as they lack the temporal property and cannot be incorporated in the existing version. The problem is to mine all possible frequent temporal patterns or sequences from an input sequence. Let S be an input sequence consisting of items ranging from i_1 to i_n. The task is to generate the counts of various possible patterns from this input sequence that satisfy the support threshold.

The problem differs from its non-temporal counterpart in two key factors namely; temporal support and temporal distance threshold (Zhu *et al*., 2003a). An application of this problem is in (Zhu *et al*., 2003a, b, 2005), which proposes a video association mining technique involving two phases of (i) Transformation and (ii) Mining. The transformation phase transforms the input video to a sequence database, which is basically a collection of clusters of the various shots that constitute the video. Once this sequence is generated, the problem is to mine all possible frequent sequences. Sequence generation as said earlier is governed by temporal support and distance factors.

Apriori can be easily incorporated for the above problem but as explained earlier and shown in Table 1 it requires repeated scans of the input. Let us assume that the input sequence S is A B C B C D C A C B C A B D A B C D B E and the temporal support threshold is 3. Here we are not interested in repeated items or clusters within a pattern. But the algorithms, Apriori and proposed can incorporate this if required. Table 1 shows the various stages of the Apriori execution, with the temporal support counts of the patterns mentioned in parenthesis.

It can be observed that the problem can be viewed as a sequential pattern mining process. PrefixSpan (Han *et al*., 2004), an efficient algorithm based on pattern growth is a variant of FP Growth and FreeSpan (Han *et al*., 2000a) for constructing frequent sequences avoiding the repeated scans setback of Apriori. It is based on generating projected databases and constructing only locally frequent items. It offers considerable improvement over Apriori in terms of reducing the number of scans. But however it does require limited repeated scans as a result of the projection logic, which when compared to Apriori is significantly small. PrefixSpan, FreeSpan and other algorithms for sequential pattern mining require the input to be in transactional or records format and does not maintain temporal continuity across transactions. Thus in our problem, if the entire sequence is viewed as a single transaction, these algorithms consider the occurrence of any specific pattern within a transaction only once and thus our objective is not met.

The proposed algorithm is similar to PrefixSpan in the fact that it requires limited number of repeated scans but differs from the projected database logic that also adds up to the number of scans required. EFTP requires only $|L_1|$ repeated scans of the input sequence for the entire frequent temporal pattern mining process.

Table 1: Level wise execution of apriori algorithm

Sequence	L_1	C_2	L_2	C_3	L_3	C_4	L_4
				ABC(3)			
				ABD(3)			
		AB(4)		ACB(3)			
		AC(3)		ACD(3)			
ABCBC	A(4)	AD(3)	AB(4)	ADB(2)			
DCACB	B(6)	BA(3)	AC(3)	ADC(2)	ABC(3)	ABCD(3)	
CABDA	C(6)	BC(4)	AD(3)	BAC(2)	ABD(3)	ABDC(2)	
BCDBE	D(3)	BD(3)	BA(3)	BAD(1)	ACB(3)	ACBD(2)	ABCD(3)
		CA(2)	BC(4)	BCA(2)	ACD(3)	ACDB(2)	
		CB(4)	BD(3)	BCD(3)	BCD(3)	BCDA(2)	
		CD(3)	CB(4)	BDA(2)	CDB(3)	CDBA(1)	
		DA(2)	CD(3)	BDC(2)			
		DB(2)		CBA(2)			
		DC(2)		CBD(2)			
				CDA(2)			
				CDB(3)			

PROPOSED EFFICIENT FREQUENT TEMPORAL PATTERN MINING ALGORITHM

EFTP is similar to FP-Growth algorithm in the fact that it constructs a pattern tree. It requires number of frequent 1 items + 1 scans of the original input sequence. The first scan identifies the frequent and infrequent 1 items. The input sequence is then scanned once for every member of the L_1 set. The algorithm then constructs $|L_1|$ pattern tree's, one for each member of L_1, having the respective member of L_1 as the root of the tree. Figure 1 details the algorithm and the illustration for the input sequence considered is provided in Fig. 2.

- Scan the original input sequence to identify frequent 1 items that constitutes the set L_1.
- For every member of L_1, now construct a pattern tree as follows:
    - Make the element of L_1 the root of the tree.
    - Start scanning the original input sequence from the point where the considered element appears.
    - Add paths from the root of the tree to the item encountered in the sequence in a cumulative fashion updating both the item count as well as the path count. This is done till the considered element appears again in the input. If an element that has already been added as a node to the tree appears again then, maintain suitable reverse links as well.
    - Repeat the earlier step for other sequences that commence with the considered element, for which either traverse the existing paths updating the counters suitably or add new paths from the root of the tree depending on the items encountered in the sequence.
- From each of the pattern tree's constructed, temporal patterns and hence frequent temporal patterns are generated as follows:
    - Traverse the tree from the root retrieving patterns whose temporal count is the path count.
    - Once a branch is traversed, suitably decrement the path count as well as the item count to reflect that the pattern has been counted once.
    - To generate patterns that do not appear as a branch in the tree, consider pending nodes that appear to the right of some specific node in the tree as either destination or intermediate items in the pattern.
    - Integrate such patterns that match with the tree branch pattern to generate the net temporal pattern set.
- From each of the pattern tree's temporal pattern set, retain only those patterns that satisfy the temporal support threshold specified.

Fig. 1: Proposed Efficient Frequent Temporal Pattern Mining (EFTP) algorithm

The various temporal patterns that are constructed from the pattern tree for A in Fig. 2 is as follows:

Pattern	Count	Pattern	Count
A	4	AB	4(3+1)
AC	3(2+1)	AD	3(2+1)
ABC	3(2+1)	ACB	3(1+2)
ACD	3(2+1)	ADC	2(1+1)
ADB	2(1+1)	ABD	3(1+2)
ABCD	3(2+1)	ABDC	2(1+1)
ADBC	2(1+1)	ADCB	2(1+1)
ACBD	2(1+1)	ACDB	2(1+1)

Similarly pattern trees can be constructed for the remaining frequent 1 items and hence the final temporal pattern set and frequent temporal patterns that satisfy the support threshold can be generated. It can be verified that the frequent temporal patterns generated by our algorithm and Apriori based algorithm is the same.

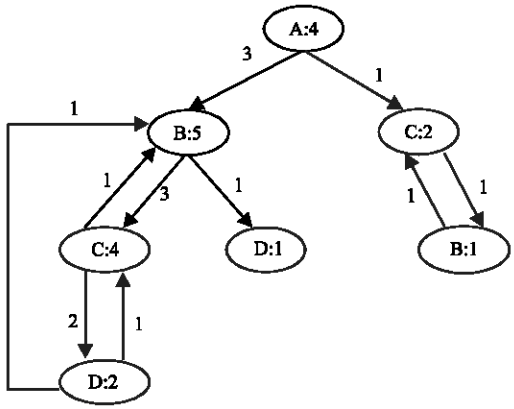

Fig. 2: Pattern tree for item A

RESULTS AND DISCUSSION

This section deals with performance analysis of the proposed algorithm in comparison to Apriori based algorithm. All the experiments have been carried out on an 8GB RAM system supporting LINUX 9 under a multi-user setup. The entire performance analysis is concentrated on the execution times of the algorithms. The algorithms are analyzed for the effect of input sequence length and support factor.

Figure 3 depicts the performance improvement of the proposed algorithm due to reduced number of repeated input scans. Results for varied input sequences establish a similar qualitative but different quantitative growth. The overall average improvement in performance of the proposed algorithm in relation to Apriori algorithm is 88.08%.

Having demonstrated the significant performance improvement achieved by the proposed algorithm, we shall now establish the support factor independence. With respect to Apriori, execution time is inversely proportional to support factor. This is a result of the level wise principle of Apriori, wherein as a result of the lowered support values, more patterns become frequent at a particular level and hence the candidate set generation in the next level and the subsequent frequent set construction in the same level consumes more time.

Our algorithm requires the support value only during the initial scan to decide on frequent 1 patterns and finally to generate frequent patterns of all lengths. The effect of support value over the performance of proposed and apriori principle based algorithm can be

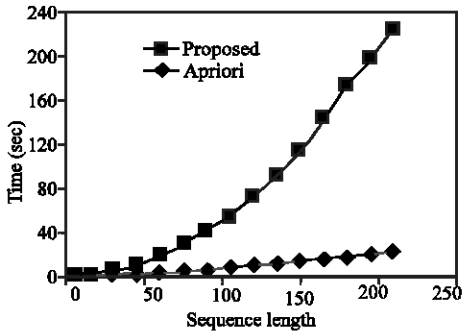

Fig. 3: Effect of sequence length

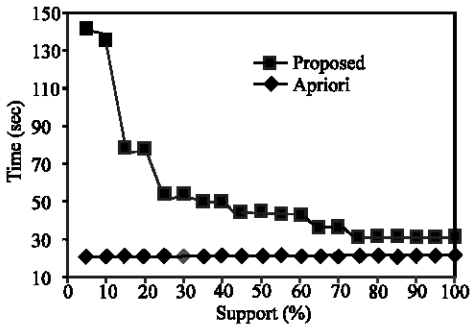

Fig. 4: Effect of support

observed from Fig. 4. Experimental results for varied input sequences establish a similar qualitative but different quantitative pattern of growth. Apriori's execution time varies significantly with varied support factor values, whereas execution time of the proposed algorithm remains stable.

Variation in execution times of Apriori algorithm is due to the increased or decreased number of candidate and frequent sets as a result of support variation. This variation in execution time is quite significant and is a result of Apriori's support dependent logic. Fig. 4 shows that there are slight variations in execution time of the proposed algorithm at varied support values due to run time resource allocation constraints. Thus neglecting the run time variations, our algorithm's performance is completely support independent and hence can be uniformly applied over all situations, whereas Apriori is entirely support dependent, thereby favoring less number of frequent patterns situations. Apriori algorithm's performance deteriorates drastically with reduced support values when the number of frequent patterns will be more as opposed to higher support values, wherein the number of frequent patterns will be less.

CONCLUSIONS

We have explored the research area of frequent pattern generation in temporal data domain, coming up with a new and scan efficient frequent temporal pattern

mining algorithm. This algorithm evolved from the need for an efficient technique for frequent set construction during the association mining phase for generating video associations. Frequent pattern generation in temporal domain is entirely different from its non temporal counterpart. Though the algorithm reduces the number of scans considerably, we feel there is still some overhead as a result of the limited repeated scans of the original input and the traversal of the tree. Further research will concentrate in this direction and results will be reported in the future.

REFERENCES

Agrawal, R. and R. Srikant, 1994. Fast algorithms for mining association rules. Proceedings of VLDB, pp: 487-499.

Brin, S., J. Rajeev Motwani, D. Ullmann and S. Tsur, 1997. Dynamic Itemset Counting. Proceedings of ACM SIGMOD Conference, pp: 255-264.

Goethals, B., 2003. Survey on frequent pattern mining. HelSiniki, 2004: 43.

Han, J.J., B. Pei, Mortazavi-Asl, Q. Chen, U. Dayal and M.C. Hsu, 2000a. FreeSpan: Frequent pattern-projected sequential pattern mining. Proceedings of ACM SIGKDD International Conference on Knowledge Discovery in Databases, pp: 355-359.

Han, J., J. Pei and Y. Yin, 2000b. Mining frequent patterns without candidate generation. Proceedings of ACM SIGMOD., pp: 1-12.

Han, J. and M. Kamber, 2001. Data Mining: Concepts and Techniques. Morgan Kauffmann Publishers, Ch. 6-8.

Han, J.J., B. Pei, J. Mortazavi-Asl, H. Wang, Pinto, Q. Chen, U. Dayal and M-C. Hsu, 2004. Mining sequential patterns by pattern growth: The prefixspan approach. IEEE Transactions of Knowledge and Data Engineering, 16: 11.

Tesic, J., S. Newsam and B.S. Manjunath, 2003. Mining image datasets using perceptual association rules. Proceedings of SIAM 6th International Workshop on Mining Scientific and Engineering Datasets in conjunction with 3rd SIAM International Conference on SDM., pp: 71-77.

Wijesekara D. and D. Barbara, 2000. Mining cinematic knowledge-work in progress. Proceedings of International Workshop on MDM/KDD, pp: 98-103.

Zaiane, O.R. J. Han, Z. Li and J. Hou, 1998. Mining multimedia data. Proceedings of CASCON., pp: 83-96.

Zaiane, O.R., 1999. Resource and knowledge discovery from the internet multimedia repositories. Ph.D Thesis, School of Computing Science, Simon Fraser University.

Zaiane, O.R., J. Han and H. Zhu, 2000. Mining recurrent items in multimedia with progressive resolution refinement. Proceedings of ICDE., pp: 461-470.

Zaiane, O.R., J. Han, Z. Li and J. Chiang, 2002a. Multimedia miner- a system prototype for multimedia data mining. Proceedings of ACM SIGMOD, pp: 581-583.

Zaiane, O.R., M.L. Antonie and A. Coman, 2002b. Mammography classification by an association rule based classifier. Proceedings of International Workshop on MDM with ACM SIGKDD., pp: 62-69.

Zaki, M., 2001. SPADE: An efficient algorithm for mining frequent sequences. Proceedings of Machine Learning, 40: 31-60.

Zhu X. and X. Wu, 2003a. Mining video associations for efficient database management. Proceedings of 18th IJCAI., pp: 1422-1432.

Zhu X. and X. Wu, 2003b. Sequential association mining for video summarization. Proceedings of ICME, Baltimore, pp: 333-336.

Zhu, X., X. Wu, A.K. Elmagarmid, Z. Feng and L. Wu, 2005. Video data mining: Semantic indexing and event detection from the association perspective. IEEE Transactions on Knowledge and Data Engineering, 17: No. 5.