IJCAT - International Journal of Computing and Technology, Volume 1, Issue 10, November 2014
ISSN : 2348 - 6090
www.IJCAT.org

A Birds Eye View on Mining Sequential Patterns from Traditional Databases

[1] **Gurram Sunitha,** [2] **Dr. A. Rama Mohan Reddy**

[1] Research Scholar, Department of CSE, S.V.U. College of Engineering, Tirupati

[2] Professor, Department of CSE, S.V.U. College of Engineering, Tirupati

Abstract - Sequential pattern mining is the process of mining relationships amid elements. The relationships like causal, blocking etc leads to analogue of various kinds of sequential patterns. Association rule mining allows to abundance associations between elements without the concrete notion of time. Sequential pattern mining incorporates temporal ordering into the mining process. This allows accretion of knowledge of associations among elements in the course of time. This paper is intended to give a brief insight into the landmark research that has brought the area of sequential pattern mining into limelight.

***Keywords* - Data Mining, Frequent Patterns, Sequential Patterns, Transactional Databases, Sequence Databases.**

1. Introduction

The area of data mining emerged with the analysis of market-basket data. It is most generally called as a classical database or a traditional database in the field of data mining. It is a simple transactional database which contains a transaction id, customer id and a set of items purchased by the customer. For the purpose of mining sequential patterns, a temporal ordering is imposed on the transactions and the notion of time may be concrete or relative. A transactional database is transformed into a sequence database to support mining sequential patterns. Example transactional database and sequence database are shown in Fig 1 and Fig 2 respectively. The problem of sequential pattern mining can be formulated as follows.

An *item* is an product purchased by the customer and an *itemset* is the non-empty set of items. Let $i_1, i_2 \ldots i_n$ denote items and $(i_1, i_2, \ldots i_m)$ denote an itemset such that $m \leq n$. A *sequence* can be defined as an temporally ordered list of itemsets denoted as $S = <e_1, e_2, \ldots e_k>$ where $e_1, e_2, \ldots e_k$ denote elements of the sequence. Each element of a sequence is an itemset. Let $S_1 = <a_1, a_2, \ldots a_p>$ and $S_2 = <b_1, b_2, \ldots b_q>$ be any two sequences. The sequence S_1 is said to be a sub-sequence of S_2 if there exist integers $x_1 < x_2 < \ldots < x_p$ such that $a_1 \subseteq b_{x1}, a_2 \subseteq b_{x2}, \ldots, a_p \subseteq b_{xp}$.

CID	TID	Items Purchased
1	T1	i_1, i_6, i_8
2	T2	i_3, i_4, i_7
3	T3	i_2, i_5
2	T4	i_3, i_4, i_5, i_9
3	T5	i_2
4	T6	i_5, i_6, i_8
1	T7	i_1, i_4, i_7
1	T8	i_3, i_6, i_7, i_8, i_9
4	T9	i_1, i_2, i_7
3	T10	i_9
3	T11	i_9, i_{10}

Fig 1. Example transactional database

CID	Sequence
1	$<(i_1, i_6, i_8), (i_1, i_4, i_7), (i_3, i_6, i_7, i_8, i_9)>$
2	$<(i_3, i_4, i_7), (i_3, i_4, i_5, i_9)>$
3	$<(i_2, i_5), (i_3, i_4, i_5, i_9), (i_9), (i_9, i_{10})>$
4	$<(i_5, i_6, i_8), (i_1, i_2, i_7)>$

Fig 2. Example sequence database

A sequence with k elements is said to be a *k-sequence*. A transaction *T* is said to support a sequence *S* if it contains *S* The support of a sequence *S* is defined as the average number of customers supporting *S*. A sequence *S* is said to be a *Sequential Pattern* if it has minimum support in the given sequence database. Sequential pattern mining is the problem of finding all sequential patterns from the given sequence database. Association rules are intra-transaction patterns whereas sequential patterns are inter-transaction patterns.

Further, the paper is organized as follows. Section 2 discusses the approaches advised and developed for sequential pattern mining. Section 3 and 4 gives a perspective on techniques for constrained sequential pattern mining and incremental sequential mining respectively.

2. Mining Sequential Patterns

Agrawal et al. [1] coined the concept of sequential patterns and have also proposed a way to convert customer (transactional) database into a sequence database. Sequential pattern discovery process is suggested to be a 5-phase approach – sort, litemset, transformation, sequence and maximal phases. Three algorithms AprioriAll, AprioriSome and DynamicSome are proposed. AprioriAll is based on the Apriori algorithm and generates all the maximal sequences. The idea behind AprioriSome and DynamicSome algorithms is to generate longer sequences quickly so that counting support for subsequences of longer sequences can be skipped. Hit ratio is used as the factor to determine the number of steps to be skipped.

Srikant et al. [2] incorporated the idea of time constraints, sliding time windows and taxonomies into the approach proposed in [1] and has presented GSP algorithm to discover generalized sequential patterns. A new approach more appropriate for candidate generation in sequential pattern mining is discussed that reduces the number of candidates for counting. Use of hash tree for maintaining candidates to reduce the number of candidates for support counting as well as for reducing the traversing time through candidate set is proven. Also, hash tree provides an efficient representation to check whether a candidate sequence is subsequence of another sequence.

Han et al. [3] proposed FreeSpan method for efficient mining of sequential patterns from sequence databases. This method has changed the view on the approach of discovering sequential patterns in a deserving way. In the traditional Apriori-based methods, the number of candidates may be exponential. Hence support counting is time-consuming and number of database scans needed will be more. In order to avoid such scenario, FreeSpan develops candidates based on the projected sequence databases. Initially, the 1-frequent itemsets are gathered by scanning the original sequence database once. Then, using these 1-frequent itemsets, frequent item matrix is constructed. The matrix is then used to generate 2-length sequential patterns as well as a set of annotations on repeating items and projected databases. An α-projected database is defined as the set of sequences having α as their subsequence. Database is then scanned to find item-repeating patterns and projected databases. Matrix projection mining is performed on projected databases recursively to generate sequential patterns of length greater than 2. FreeSpan is better than GSP as number of candidates generated and checked are not exponential. FreeSpan needs only 3 database scans irrespective of the length of the sequential patterns to be generated.

Pei et al. [4] enhanced the idea of FreeSpan [3] and developed PrefixSpan algorithm to further speed up the process of sequential pattern mining. GSP joins globally frequent items to generate potential candidates and hence the number of candidates that satisfy minimum support may be least. FreeSpan and PrefixSpan propose the idea of joining locally frequent items to generate potential candidates. Hence, the chance of candidates satisfying the minimum support will be high. Also, FreeSpan and PrefixSpan are pattern-growth methods based on the idea of FP-growth approach used for generating association rules. FreeSpan has the following disadvantages as stated in [4]:

- ✓ It may generate many non-trivial projected databases.
- ✓ The projected databases may not shrink through the recursive mining process, if a pattern appears in each sequence of the sequence database.
- ✓ The k-length subsequence may grow at any position according to the definition of the α-projected database in FreeSpan. Hence, the counting process of a (k+1)-length candidate sequence will need to check every possible combination which is costly.

PrefixSpan reduces the size of projected databases as the recursive process continues and also reduces the cost of checking at every possible position of a potential candidate sequence by fixing the order of items within each element. It also changes the definition of the α-projected database. In PrefixSpan, a α -projected database is defined as the set of sequences having α as their prefix. This definition allows the projected databases to contain only the suffix sequences for the prefix sequence α. This allows an orderly manner of examining the potential sequences and the projected databases. PrefixSpan completely eliminates the idea of candidate generation. It also uses the pseudoprojection technique. Advantages of PrefixSpan can be stated as follows –

- ✓ Suitable for larger databases that do not entirely fit into the main memory.
- ✓ No candidate generation and counting.
- ✓ As the length of the sequential patterns being generated increases, the projected databases size decreases.
- ✓ Pseudoprojection technique reduces the number and size of the projected databases.
- ✓ 2 to 3 times faster than SPADE, GSP, FreeSpan algorithms.

Zaki [5] presented SPADE algorithm for quick discovery of sequential patterns. It needs 3 database scans to discover all the sequential patterns of any length. Use of vertical data format, lattice-theoretic approach and pattern search strategies played a key role in the increased speed of the algorithm. SPADE is efficient in both reduced I/O costs and computational costs. Use of temporal join which is a simple join operation and avoiding use of complex candidate generation techniques and complex data structures like hash tree makes it ideal for integration into the database management system. SPADE fails when applied on databases containing consecutive repetitions of items. In such cases, GO-SPADE developed by Leleu et al. [6] is proven to remain efficient when compared to SPADE.

Many algorithms does not scale linearly while discovering longer sequential patterns. Ayres et al. [7] riveted on developing an algorithm to suit needs of generating longer sequential patterns from huge databases. The SPAM algorithm proposed by the authors assumes that the sequence database fits in its entirety in the memory. It uses depth-first strategy through the lexicographic sequence tree to generate the sequential patterns. The vertical bit-map representation of data helps in efficient support counting. Use of I-step and S-step for extending the length and size of a sequence respectively is suggested in line with the lexicographic sequence tree. Pruning methods used in I-step and S-step aimed at reducing the potential candidates efficiently. Lexicographic order of items within the elements is preserved to make search and comparisons of sequences easier and faster. It can be stated that SPADE is efficient in terms of memory usage whereas SPAM is efficient in terms of time whereas uses larger memory than SPADE and PrefixSpan.

Kum et al. [8] proposed to generate consensus patterns when the sequences in the sequence database are long and noisy. A consensus pattern is an approximation of a group of sequential patterns. Sometimes it may be useful to generate approximate patterns shared by many sequential patterns instead of generating complete and exact list of large number of sequential patterns. In such cases where summarized sequential patterns are enough ApproxMAP algorithm is best suitable. Such patterns are generated by clustering the sequences in the database based on their similarity and then a consensus pattern is generated for each cluster. Hierarchical edit distance is used to measure similarity of any two sequences based on the cost of insert, delete and replace operations to convert one sequence into another. The implementation of the ApproxMAP uses k-NN clustering. Multiple alignment of sequences within a cluster is performed to generate a long approximate pattern for the cluster called as consensus pattern. Consensus patterns provide concise, compressed and interesting knowledge that may lead to view the patterns in the data with a different proposition.

Mining from interval-based event databases was uncovered by Kempe et al. [9]. Considering the database to be an interval sequence database, temporal patterns are defined based on the Allen's 7 temporal relationships [10]. Based on Apriori algorithm [11], a new algorithm has been developed by modifying the candidate generation step and defining a new support measure. Hirate et al. [12] created the idea of having a time interval extended sequence database to extract more meaningful sequential patterns by imposing maximum time interval constraint between any two transactions involved in generating a sequential pattern. The proposed algorithm is based on the PrefixSpan algorithm.

Temporally-annotated sequential patterns as defined by Giannotti et al. [13] are sequential patterns where each transition in the pattern is annotated by a transition time. For example,

$$A \xrightarrow{[t1,t2]} B$$

says that if A occurs then there is a probability that B occurs after a time period t1 to t2. The sequence database is considered to be a temporally annotated one.

Discovering quantitative sequential patterns as opposed to the traditional qualitative patterns was the idea of Kim et al. [14]. The work has been done on the traditional market-basket database which can be extended to work on other kinds of databases. Two naïve algorithms Apriori-QSP and PrefixSpan-QSP are developed based on the Apriori and PrefixSpan algorithms. The same are extended with two optimization techniques – hash filtering and quantity sampling to present more efficient algorithms Apriori-All and PrefixSpan-All respectively. This research tried to incorporate context-level data into the sequential mining process. Further research may be thought of, such as including more than one context-level attribute into the mining process.

Even on using interestingness measures for reducing spurious patterns, the number of resultant patterns produced are large and redundant. Many resultant patterns are small in length which represent common sense knowledge. To mine more useful and informative patterns, correlation between data has to be considered. Work on the same thought has been done by Lin et al. [15]. Complications of correlation measures such as not satisfying Apriori property leading to computational complexities are discussed. PSBSpan algorithm designed in this paper is based on the PrefixSpan algorithm. A number of correlation measures like all-confidence, lift etc have been studied.

Redundancy is the major problem of the resultant patterns generated using the mining algorithms based on anti-monotone property. The same property can be used to avoid redundancy of information from the discovered patterns. The solution is to mine maximal sequential patterns or closed sequential patterns. A closed sequential pattern P, is a frequent sequential pattern such that there does not exist a super pattern of P with the same support as P. A maximal sequential pattern P, is a frequent sequential pattern such that none of the super patterns of P are frequent. Based on the vast work done on mining frequent closed itemsets, Yan et al. [16] proposed to apply the concept of mining closed patterns to sequential pattern mining to achieve the same advantages. They have proposed CloSpan which is the extension of PrefixSpan algorithm that produces only closed sequential patterns as the final result. It has been shown experimentally that CloSpan performs better than PrefixSpan and does not fail even when the minimum support is low, thus providing scalable mining.

Generally, the task of setting the value of minimum support to extract meaningful and useful patterns, is tedious. The users must play by varying the minimum support threshold value between a range and try to aggregate the results that are consistent with varying support threshold. The same problem aroused with mining frequent itemsets for which the solution found was to find top-k frequent itemsets of pre-defined minimum length *minl*. The solution to this problem with regard to the sequential pattern mining has been provided by Tzvetkov et al. [17] by modifying the existing technique for finding top-k patterns as suitable for sequential pattern mining. A closed sequential pattern P is said to be a top-k closed sequential pattern if it is at least of length *minl* and if there exists no more than (k-1) closed sequential patterns whose length is at least *minl* and whose support is greater than the support of P. The TSP algorithm designed for the purpose is based on the PrefixSpan approach and uses multi-pass approach to extract top-k closed sequential patterns. As progressing through passes, the minimum support value is raised through each pass.

Many algorithms may have been developed to mine sequential patterns, but customizing algorithms to integrate them into real-life systems and proving their efficacy to support quick and intelligent decision making is tough. Zaki et al. [18] and Wu et al. [142] have tried to integrate sequential pattern mining algorithms into real-world systems to show the use of mining algorithms in real-world applications. [18] presented PLANMINE algorithm to mine patterns from plan execution database for the purpose of predicting plan failures. [19] gathered alarm data available in network databases and extracted knowledge to predict network failures.

3. Constrained Sequential Pattern Mining

There have been developed many measures to measure strength and interestingness of the patterns, but the huge set of extracted patterns may not be of much use for a focused user. Users generally want to view only results containing their elements of interest. Domain experts may want to experiment and compare the results by posing varying constraints. One way is to produce all the potential patterns and then filter them. But, this is computationally expensive and fails in producing timely results to the inquisitive user. To help directed mining process, [20], [21], [22], [23] paved way to induce constraints deep into the mining process. These works have helped in finding popular patterns that share interests of the user.

Garofalakis et al. [20] unraveled the process of pushing user-constraints into the mining process at different levels. User can specify constraints using regular expressions. Concept of regular expressions is chosen because of the two important factors as mentioned by the authors. The first is the simple and yet powerful syntax of regular expressions to specify user constraints in an compendious manner. A set of four algorithms – SPIRIT(N), SPIRIT(L), SPIRIT(V) and SPIRIT(R). SPIRIT(N) is the naïve approach and handles the weakest relaxation of the user constraints and the consecutive algorithms push constraints deeper into the mining process with SPIRIT(R) able to handle the strongest relaxations of the constraints. The validity and legality of the sequences is proposed to enable to define the type of patterns extracted by each of the four SPIRIT algorithms. Experiments showed that SPIRIT(V) is a consistently good performer and SPIRIT(R) provided consistent support for highly directive mining process.

Given a reference pattern, the problem of finding sequential patterns as defined by Capelle et al. [21] is to discover all patterns that are frequent as well as similar to the reference pattern. It is said by the authors that the use of frequency constraint in combination with the similarity constraint provides pruning efficiency globally in the candidate generation process that uses level-wise approach. Frequency constraint is an anti-monotone constraint whereas similarity constraint is an convertible anti-monotone constraint and the combination of these constraints is neither anti-monotone nor convertible anti-monotone. Similarity of two patterns is measured by computing the cost of insert, delete and substitute operations for converting one pattern into another. The similarity ranges from 0 to a maximum of 1.

[20] has shown a way to define the structure, limit the type and number of items and limit the length of the resultant patterns. [21] has shown how to extract only patterns that are similar to the user-defined reference pattern. The authors of these papers have done the initiative work and their effort remained on instigating the idea of highly selective data mining. Hence they have handled only simple constraints and did not define the criteria for pushing varying, multiple and complex constraints into the mining process.

Pei et al. [22] have performed a thick study on categorizing the constraints with respect to the sequential pattern mining. The constraints have been categorized based on application point of view and in view of pushing constraints into the mining process. Based on the application point of view, the constraints have been classified into 7 categories.

a) Item Constraint – constrains the items and their groups that can be present in the sequential patterns.

b) Length Constraint – defines limit on the length of the patterns.

c) Super-pattern Constraint – delimits the result to the set of patterns that have a user-specified pattern as sub-pattern.

d) Aggregate Constraint – checks that the resultant patterns satisfy the restrictions placed on the aggregate of items in the pattern.

e) Regular Expression Constraint – constraints are defined in the form of regular expressions as explained in [20].

f) Duration Constraint – restricts the result to the patterns which have occurred completely within a given duration i.e., the time gap between the occurrence of the first and last elements in the pattern must be greater than(or less than) a user-defined duration.

g) Gap Constraint – The pattern to be considered as part of final resultant set should satisfy the rule that time gap between occurrences of any two consecutive elements in the pattern must be greater than(or less than) a user-defined time period.

Also, the constraints have been studied extensively in the view of pushing them deep into the mining process. The characterization is based on the monotonicity, anti-monotonicity and succinctness properties of the constraints. According to the study all the simple constraints are succinct and all complex constraints are not.

The problem of constraint-based sequential pattern mining is conceived by the authors and prefix-growth algorithm has been framed for the purpose. The prefix-growth algorithm is based on the PrefixSpan algorithm and the prefix-monotone property. Prefix-growth by its design outperformed SPIRIT in pushing regular expression constraints into the mining process.

Some users may be interested in only one event type and may want to investigate the cause-effect relationships of the interested event type on the others and vice-a-versa. Sun et al. [23] have defined for the same purpose patterns called as event-oriented patterns, which define the patterns leading to the cause of a given target event type. It focused on directing the sequential pattern mining process to derive patterns of form

$$\left\{P \overset{T^{(P)}}{\Longrightarrow} E\right\}$$

where P is the event-oriented pattern causing events of target event type E. $T^{(P)}$ is the minimum size of temporal interval within which P can cause E, also called as 'Temporal Feature'. Temporal Feature of an event-oriented pattern defines its interestingness. Deriving event-oriented patterns are useful in situations where the focus is on finding the causes of an event type of interest such as fraudulent analysis.

4. Incremental Sequential Pattern Mining

The problem with real-life datasets is that as the events or transactions occur their details are accumulated and are updated in the database. Sensors, buoys, satellites, location-aware services etc continuously collect data through years. Hence, the database will be in continuously being updated. As new data adds on, some patterns of the old mining results may become invalid and some new patterns may become valid. As mining process is expensive in terms of I/O and computational power, it is not recommended to repeat complete mining process whenever the database is updated, specifically with the databases that store streaming data. Extracting knowledge from such databases need incremental mining methodologies and techniques. [24], [25], [26], [27], [28], [29], [30], [31] have done appreciable work on incremental sequential pattern mining.

5. Conclusion

In this paper, the problem of sequential pattern mining is underlined. Algorithms and techniques for the purpose of extracting knowledge in the form of sequential patterns have been discussed. An effort is made to debrief the process of focused mining and incremental mining.

References

[1] Agrawal R. and Srikant R., "Mining Sequential Patterns," In Proceedings International Conference on Data Engineering (ICDE'95), pp. 3-14, 1995.

[2] Srikant R. and Agrawal R., "Mining Sequential Patterns: Generalizations and Performance Improvements," In Advances in Database Technology (EDBT'96), 1996, pp.1-17.

[3] Han J., Pei J., Mortazavi-Asl B., Chen Q., Dayal U., and Hsu M. C., "FreeSpan: Frequent Pattern-Projected Sequential Pattern Mining," In Proceedings of the sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 355-359, Aug. 2000.

[4] Pei J., Han J., Mortazavi-Asl B., Wang J., Pinto H., and Chen Q., "Mining Sequential Patterns by Pattern-Growth: The PrefixSpan Approach," IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 11, pp. 1424-1440, 2004.

[5] Zaki M., "SPADE: An Efficient Algorithm for Mining Frequent Sequences," Machine Learning, vol. 42, no. (1/2), pp. 31-60, 2001.

[6] Leleu M., Rigotti C., Boulicaut J. F., and Euvrard G., "GO-SPADE: Mining Sequential Patterns over Datasets with Consecutive Repetitions," In Machine Learning and Data Mining in Pattern Recognition, Springer Berlin Heidelberg, pp. 293-306, 2003.

[7] Ayres J., Flannick J., Gehrke J., and Yiu T., "Sequential Pattern Mining Using Bitmap Representation," In Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining(KDD'02), pp. 429-435, 2002.

[8] Kum H.C., Pei J., Wang W., and Duncan D., "ApproxMAP: Approximate Mining of Consensus Sequential Patterns," In Proceedings of the 3rd SIAM International Conference on Data Mining(SDM '03), pp. 311-315, 2003.

[9] Kempe S. and Hipp J., "Mining sequences of temporal intervals," In Knowledge Discovery in Databases: PKDD 2006, pp. 569-576, 2006.

[10] Allen J. F., "Maintaining knowledge about temporal intervals," Communications of the ACM, vol. 26, no. 11, pp. 832-843, 1983.

[11] Agrawal R. and Srikant R., "Fast algorithms for mining association rules," In Proc. 1994 International Conference on Very Large Data Bases (VLDB'94), pp. 487–499, Sept. 1994.

[12] Hirate Y. and Yamana H., "Sequential Pattern Mining with Time Intervals," In Advances in Knowledge Discovery and Data Mining, Springer Berlin Heidelberg, pp. 775-779, 2006.

[13] Giannotti F., Nanni M., and Pedreschi D., "Efficient Mining of Temporally Annotated Sequences," In Proceedings of the 6th SIAM International Conference on Data Mining, pp. 346-357, Apr. 2006.

[14] Kim C., Lim J. H., Ng R. T., and Shim K., "SQUIRE: Sequential pattern mining with quantities," Journal of Systems and Software, vol. 80, no. 10, pp. 1726-1745, 2007.

[15] Lin C. X., Ji M., Danilevsky M., and Han J., "Efficient mining of correlated sequential patterns based on null hypothesis," In Proceedings of the 2012 international workshop on Web-scale knowledge representation, retrieval and reasoning, pp. 17-24, Oct. 2012.

[16] Yan X., Han J., and Afshar R., "CloSpan: Mining Closed Sequential Patterns in Large Datasets," In Proceedings of the 3rd SIAM International Conference on Data Mining(SDM '03), pp. 166-177, May 2003.

[17] Tzvetkov P., Yan X., and Han J., "TSP: Mining Top-K Closed Sequential Patterns," Knowledge and Information Systems, vol. 7, no. 4, pp. 438-457, 2005.

[18] Zaki M. J., Lesh N., Ogihara M., "PLANMINE: Sequence mining for plan failures," Artificial Intelligence Review, vol. 14, no. 6, pp. 421–446, 2000.

[19] Wu P. H., Peng W. C., and Chen, M. S., "Mining sequential alarm patterns in a telecommunication database," In Databases in Telecommunications II, Springer Berlin Heidelberg, pp. 37-51, 2001.

[20] Garofalakis M. N., Rastogi R., and Shim K., "SPIRIT: Sequential Pattern Mining with Regular Expression Constraints," In Proceedings 1999 International Conference Very Large Data Bases(VLDB'99), vol. 99, pp. 7-10, Sep. 1999.

[21] Capelle M., Masson C., and Boulicaut J. F., "Mining Frequent Sequential Patterns under a Similarity Constraint," In Intelligent Data Engineering and Automated Learning—IDEAL 2002, Springer Berlin Heidelberg, pp. 1-6, 2002.

[22] Pei J., Han J., and Wang W., "Mining Sequential Patterns with Constraints in Large Databases," In Proceedings of the Eleventh International Conference on Information and Knowledge Management(CIKM'02), pp. 18-25, Nov. 2002.

[23] Sun X., Orlowska M. E., and Li X., "Finding Temporal Features of Event-Oriented Patterns," In Advances in Knowledge Discovery and Data Mining, Springer Berlin Heidelberg, pp. 778-784, 2005.

[24] Parthasarathy S., Zaki M., Ogihara M., and Dwarkadas S., "Incremental and Interactive Sequence Mining," In Proceedings of the 8th International Conference on Information and Knowledge Management(CIKM'99), pp. 251-258, 1999.

[25] Zhang M., Kao B., Cheung D. W., and Yip C. L., "Efficient Algorithms for Incremental Update of Frequent Sequences," Advances in Knowledge Discovery and Data Mining, Springer Berlin Heidelberg, pp. 186-197, 2002.

[26] Lin M. and Lee S., "Improving the Efficiency of Interactive Sequential Pattern Mining by Incremental Pattern Discovery," In Proceedings of the 36th Annual Hawaii International Conference on System Sciences, pp. 8-pp, Jan. 2003.

[27] Masseglia F., Poncelet P., and Teisseire M., "Incremental Mining of Sequential Patterns in Large Databases," Data and Knowledge Engineering, vol. 46, no. 1, pp. 97-121, 2003.

[28] Lin M. Y. and Lee S. Y., "Incremental update on sequential patterns in large databases by implicit merging and efficient counting," Information Systems, vol. 29, no. 5, pp. 385-404, 2004.

[29] Cheng H., Yan X., and Han J., "IncSpan: Incremental Mining of Sequential Patterns in Large Database," In Proceedings of the tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining(KDD'04), pp. 527-532, Aug. 2004.

[30] Nguyen S. N., Sun X., and Orlowska M. E., "Improvements of IncSpan: Incremental Mining of Sequential Patterns in Large Database," In Advances in Knowledge Discovery and Data Mining, Springer Berlin Heidelberg, pp. 442-451, 2005.

[31] Chen Y., Guo J., Wang Y., Xiong Y., and Zhu Y., "Incremental Mining of Sequential Patterns Using Prefix Tree," In Advances in Knowledge Discovery and Data Mining, Springer Berlin Heidelberg, pp. 433-440, 2007.

Gurram Sunitha has completed B.E. in Electronics & Communications Engineering from Gulbarga University in 1999 and M.Tech in Computer Sciences from JNT University, Anantapur in 2005. Currently she is pursuing Ph.D. in Computer Science and Engineering at S.V.University, Tirupati. Her research interests include Data Mining, Automata Theory and Database Systems.

Dr. A. Rama Mohan Reddy received his B.Tech. degree from JNT University, Anantapur in 1986, M. Tech degree in Computer Science from NIT, Warangal in 2000 and Ph.D. in Computer Science and Engineering in 2008 from S. V. University, Tirupathi. He is presently working as Professor in Department of Computer Science and Engineering, S. V. University College of Engineering, Tirupathi, A.P. India. His research interests include Software Architecture, Software Engineering and Data Mining. He is life member of ISTE and IE.