RESEARCHERID
THOMSON REUTERS

ISSN: 2277-9655
[Gurusamy* *et al.*, 6(11): November, 2017] Impact Factor: 4.116
IC™ Value: 3.00 CODEN: IJESS7

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

MINING MAXIMAL PERIODIC PATTERNS IN FEWER LEVELS OF RECURSION

Vairaprakash Gurusamy[*1] and K.Nandhini[2]

[*1]Department of Computer Applications, School of IT, Madurai Kamaraj University, Madurai
[2]Technical Support Engineer, Concentrix India Pvt Ltd, Chennai

DOI: 10.5281/zenodo.1042104

ABSTRACT

To reduce the number of levels of recursion for mining maximal periodic patterns a novel data structure, UpDown Directed Acyclic Graph (UDDAG) is invented. UDDAG allows bidirectional pattern growth along both ends of detected patterns which result in fewer levels of recursion than the traditional. To mine k+1 length patterns it uses $\log_2 k$ +1 levels of recursion at best instead of k levels. With UDDAG, at level i recursion, we grow the length of patterns by 2i-1 at most. Thus, a length-k pattern can be detected in $\log_2 k+1$ levels of recursion at minimum. Traditionally periodic patterns of length k+1 are mined based on the projected database of length k patterns. At each level of recursion they unidirectional grow the length of detected patterns by one along the suffix of detected patterns i.e. the new patterns are obtained by adding one item to the previous level pattern. The UDDAG approach grows patterns from both ends (prefixes and suffixes) of detected patterns, which results in faster pattern growth because of less levels of database projection compared to traditional approaches. This special feature of UDDAG enables its extension toward applications involving searching in large spaces.

KEYWORDS: Pattern Mining, UDDAG Approach.

I. INTRODUCTION

Data mining is the process of extracting patterns from data. It is the process of applying the methods like neural networks, clustering, genetic algorithms (1950s), decision trees (1960s) and support vector machines (1980s) to data with the intention of uncovering hidden patterns. It has been used for many years by businesses, scientists and governments to sift through volumes of data such as airline passenger trip records, census data and supermarket scanner data to produce market research reports. Data mining, "The extraction of hidden predictive information from large databases", is a powerful new technology with great potential to help companies focus on the most important information in their data warehouses. Data mining tools predict future trends and behaviors, allowing businesses to make proactive, knowledge-driven decisions. Data mining tools can answer business questions that traditionally were too time consuming to resolve. They scour databases for hidden patterns, finding predictiveinformation that experts may miss because it lies outside their expectations.

There are various Steps that are involved in Knowledge discovery are:

1. **Data Cleaning**: The data collected are notclean and may contain errors, missing values, noisy or inconsistent data, so that differenttechniques nedded to get rid of such anomalies.
2. **Data Integration:** Data are collected andintegrated from all the different sources.
3. **Data Selection:**In this step, select onlythose data useful for data mining.
4. **Data Transformation:** The data even after cleaning are not ready for mining so transform theminto forms appropriate for mining. The techniquesused to accomplish this are smoothing, aggregation, normalization.
5. **Data Mining:**In this step, apply data mining techniques on the data to discover the interesting patterns. Various techniques like clustering and associationanalysis are usedfor data mining.
6. **Pattern Evaluation and Knowledge Presentation:** This step involves visualization, transformation, removing redundant patterns from the generated patterns .
7. **Decisions / Use of Discovered Knowledge**: This step helps user to make use of the knowledge acquired to take better decisions.

A. UDDAG

Sequential pattern mining is an important data mining problem, which detects frequent subsequences in a sequence database. A major technique for sequential pattern mining is pattern growth. Traditional pattern growth-based approaches (e.g., PrefixSpan) derive length-(k+1) patterns based on the projected databases of a length-k pattern recursively. At each level of recursion, the length of detected patterns is grown by 1, and patterns are grown unidirectionally along the suffix direction. Consequently, we need k levels of recursion to mine a length-k pattern, which is expensive due to the large number of recursive database projections. In this paper, a new approach based on UpDown Directed Acyclic Graph (UDDAG) is proposed for fast pattern growth. UDDAG is a novel data structure, which supports bidirectional pattern growth from both ends of detected patterns. With UDDAG, at level i recursion, we may grow the length of patterns by 2i-1 at most. Thus, a length-k pattern can be detected in $[\log_2 k]+1$ levels of recursion at minimum, which results in better scale-up property for UDDAG compared to PrefixSpan. Our extensive experiments clearly demonstrated the strength of UDDAG with its bidirectional pattern growth strategy. When minSup is very large such that the average length of patterns is very small (close to 1), UDDAG and PrefixSpan have similar performance because in this case, the problem degrades into a basic frequent item counting problem. However, UDDAG scales up much better compared to PrefixSpan. It often outperforms PrefixSpan by one order of magnitude in our scalability tests. UDDAG is also considerably faster than two other representative algorithms, Spade and LapinSpam. Except for some extreme cases, the memory usage of UDDAG is comparable to that of PrefixSpan. UDDAG generally uses less memory than Spade and LapinSpam. UDDAG may be extended to other areas where efficient searching in large searching spaces is necessary.

B. Sequential Pattern Mining

Sequential pattern mining is an important data mining problem, which detects frequent subsequences in a sequence database. The major techniques for sequential pattern mining are

- Apriori-based Approaches
  - – GSP
  - – SPADE
- Pattern-Growth-based Approaches
  - – FreeSpan
  - – PrefixSpan

The problem of sequential pattern mining was introduced by Agrawal and Srikant [1]. Among the many algorithms proposed to solve the problem, GSP [17] and PrefixSpan [13], [14] represent two major types of approaches: Apriori based and pattern growth-based.

C. Apiori based Sequential Pattern Mining

Apriori principle states that any super sequence of a non-frequent sequence must not be frequent. Apriori-based approaches can be considered as breadth-first traversal algorithms because they construct all length-k patterns before constructing length-(k+1) patterns. The AprioriAll algorithm [1] is one of the earliest a prioribased approaches. It first finds all frequent item sets, transforms the database so that each transaction is replaced by all frequent item sets it contains, and then finds patterns. The GSP algorithm [16] is an improvement over AprioriAll. To reduce candidates, GSP only creates a new length-k candidate when there are two frequent length-(k-1) sequences with the prefix of one equal to the suffix of the other. To test whether a candidate is a frequent length-k pattern, the support of each length-k candidate is counted by examining all the sequences.

The PSP algorithm [12] is similar to GSP except that the placement of candidates is improved through a prefix tree arrangement to speed up pattern discovery. The SPIRIT algorithm [9] uses regular expressions as constraints and developed a family of algorithms for pattern mining under constraints based on a priori rule. The Sparse Algorithm [3] improves GSP by using both candidate generation and projected databases to achieve higher efficiency for high pattern density conditions. The approaches above represent databases horizontally. In [4] and [19], databases are transformed into vertical layout consisting of items' id-lists. The Spade algorithm [19] joins id-list pairs to form sequence lattices to group candidate sequences such that each group can be stored in the memory. Spade then searches patterns across each sequence lattice. In Spade, candidates are generated and tested on the fly to avoid storing candidates, which costs a lot to merge the id-lists of frequent sequences for a large number of candidates. To reduce this cost, The SPAM algorithm [4] adopts the lattice concept but represents each id-list as a vertical bitmap. SPAM is more efficient than Spade for mining long patterns if all the

bitmaps can be stored in the memory. However, it generally consumes more memory. LapinSpam [20] improves SPAM by using last position information of items to avoid the ANDing operation or comparison at each iteration in the support counting process. One major problem of Apriori-based approaches is that a combinatorial explosive number of candidate sequences may be generated in a large sequence database, especially when long patterns exist.

D. Pattern growth based Sequential Pattern Mining

Pattern growth approaches can be considered as depth-first traversal algorithms as they recursively generate the projected database for each length-k pattern to find length- (k+1) patterns. They focus the search on a restricted portion of the initial database to avoid the expensive candidate generation and test step. The FreeSpan algorithm [10] first projects a database into multiple smaller databases based on frequent items. Patterns are found by recursively growing subsequence fragments in each projected database. Based on a similar projection technique, the same authors proposed the PrefixSpan algorithm [13], [14], which outperforms Free- Span by projecting only effective postfixes. One major concern of PrefixSpan is that it may generate multiple projected databases, which is expensive when long patterns exist. The MEMISP [11] algorithm uses memory indexes instead of projected databases to detect patterns. It uses the find-then-index technique to recursively find the items that constitute a frequent sequence and constructs a compact index set that indicates the set of data sequences for further exploration. As a result of effective index advancing, fewer and shorter data sequences need to be processed as the discovered patterns become longer. MEMISP is faster than the basic PrefixSpan algorithm but slower when pseudoprojection technique is used in PrefixSpan. Among the various approaches, PrefixSpan was one of the most influential and efficient ones in terms of both time and space. Some approaches may achieve betterperformance under special circumstances; however, the overall performance of PrefixSpan is among the best. For example, LAPIN [21] is more efficient for dense data sets with long patterns but less efficient in other cases. Besides, it consumes much more memory than PrefixSpan.FSPM [18] declares to be faster than PrefixSpaninmanycases. However, the sequences thatFSPM mines contain only a single item in each item set. In this sense, FSPM is not a pattern mining algorithm as we discuss here. SPAM outperforms the basic PrefixSpan but is much slower than PrefixSpan with pseudoprojection technique [17].

E. PrefixSpan

In prefixSpan instead of projecting sequence databases by considering all the possible occurrences of frequent subsequences, the projection is based only on frequent prefixes because any frequent subsequence can always be found by growing a frequent prefix.

<a>, <aa>, <a (ab)> and <a (abc)> are prefixes of sequence <a (abc) (ac) d (cf)>.
Procedure:
•Step 1: Find length-1 sequential patterns
– <a>, <b>, <c>, <d>, <e>, <f>

•Step 2: Divide search space.

The complete set ofsequential patterns can be partitioned into 6 subsets: The ones havingprefix <a>, the ones having prefix <b>… the oneshaving prefix <f>.

SID sequence
10 <a (abc)(ac)d(cf)>
20< (ad) c (bc) (ae)>
30<(ef)(ab)(df)cb>
40<eg (af) cbc>

Table 1.1: Sequence Database
• Step 3: Now only need to consider projections with
respect to <a>

<a>-Projected Database:
<(abc)(ac)d(cf)>, <(_d)c(bc)(ae)>, <(_b)(df)cb>, <(_f)cbc>.Find all the length-2 seq. pat. Having prefix <a>:<aa>, <ab>, <(ab)>, <ac>, <ad>, <af>. Further partition into 6 subsets having prefix<aa>…having prefix<af>.

ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

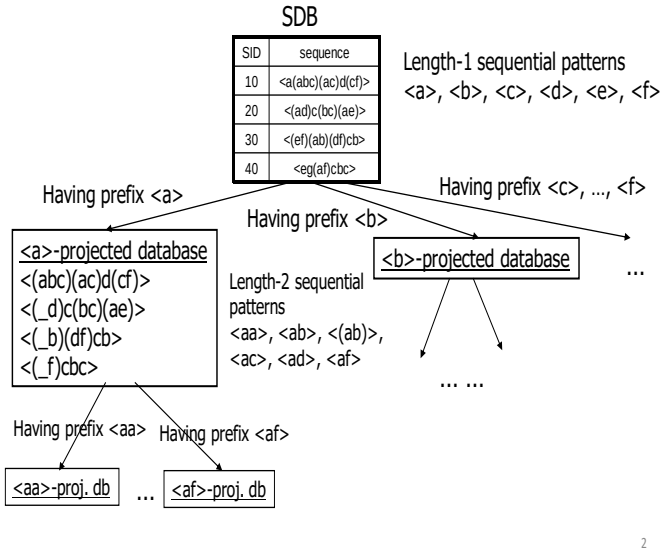


Fig 2.1:Prefixspan

F. UDDAG-based pattern mining:

UDDAG-based pattern mining approach, which first transforms a database based on frequent item sets, then partitions the problem, and finally, detects each subset using UDDAG.

II. DATABASE TRANSFORMATION

Based on frequent item sets, transform each sequence in a database D into an alternative representation. First, assign a unique id to each FI in D. Then replace each item set in each sequence with the ids of all the FIs contained in the item set.

`The FIs are: (1),(2), (3), (4), (5), (6), (1,2), (2,3). By assigning a unique id toeach FI, e.g., (1)-1, (1,2)-2, (2)-3, (2,3)-4, (3)-5, (4)-6, (5)-7, (6)-8, we can transform the database as shown in Table.

Seq. Id	Sequence
1	<1 (1,2,3,4,5) (1,5) 6 (5,8)>
2	<(1,6) 5 (3,4,5) (1,7)>
3	<(7,8) (1,2,3) (6,8) 5 3>
4	<7 (1,8) 5 3 5>

III. PROBLEM PARTITIONING

Let $\{x_1, x_2. . .x_t\}$ be the frequent item sets in a database D, x1 < x2<….<xt, the complete set of patterns (P) in D can be divided into t disjoint subsets. The i_{th} subset (denoted by P_{xi} ,$1 \leq i \leq t$) is the set of patterns that contains xi and FIs smaller than xi.

ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

A. Finding subsets of patterns

To detect Px, first detect patterns in Pre(^{x}d) and Suf(^{x}d) and then combine them to derive Px. This is a recursive process because for Pre(^{x}d) and Suf(^{x}d), perform the same action until reaching the base case, where the projected database has no frequent item set.

Finding P8: First, we build Pre(8D) andSuf(8D), which are, Pre(8D):

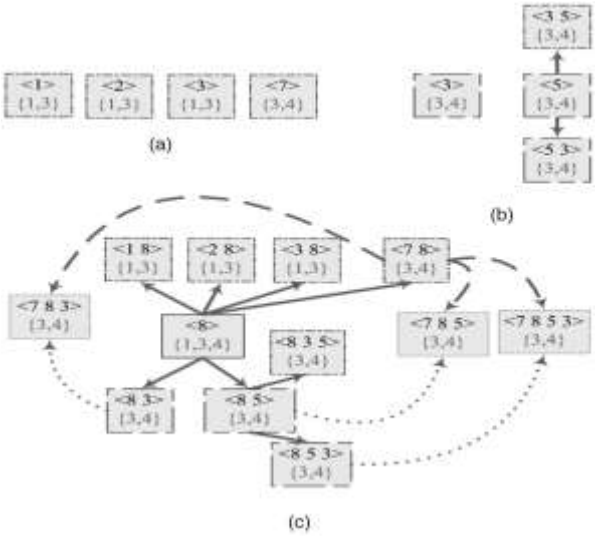
1) <1(1,2,3,4,5) (1,5) 6>,
3) <(7,8) (1,2,3) >,
4)<7>;

Suf(8D):

1) <>,
3) < (1,2,3) (6,8) 5 3>,
4) <5 3 5>.

Let PP be all the patterns in Pre (8D), since
the FIs in Pre(8D) are (1), (2), (3), and (7) and partition PP into four subsets, PP7;PP3; PP2, and PP1. First, detect PP7.Since the prefix-projected database of 7 in Pre(8D) is empty, and the suffix-projecteddatabase of 7 is: 3) <(1, 2, 3)>, 4) <>, the onlypattern in PP7 is <7>. Similarly, PP3 ¼f<3>g; PP2 ¼ f<2>g, and PP1 ¼ f<1>g.

Thus, PP ¼ f<1>;<2>; <3>;<7>g.Let PS be all the patterns in Suf (8D), since the FIs in Suf (8D) are (3) and (5), we canpartition PS into two subsets, PS5 and PS3.First, we detect PS5. The prefix-projecteddatabase of 5 in Suf (8D) is: 3) < (1, 2, 3) (6, 8)>, 4) <5 3>, which contains a pattern <3>. Thesuffix-projected database of 5 in Suf (8D) is: 3) < 3>, 4) <3 5>, which also contains a pattern<3>. Since both databases have patterns, weneed consider case 3, i.e., whether concatenating<3> with root 5 and <3> is also a pattern Here, the occurrence set of <3> in the PrefixProjectedDatabase of 5 is {3, 4}, and theoccurrence set of <3> in the suffix-projecteddatabase of 5 is also {3, 4}. Thus, theirintersection set is {3, 4}, which means that thesupport of <3 5 3> is at most 2. However, since5 occurs twice in tuple 4, we need checkwhether it really contains <3 5 3>, which is nottrue by verification. Thus, the support of<3 5 3> is 1, and it is not a pattern. Therefore,PS5 ¼ f<3 5>;<5 3>;<5>g. S i m i l a r l y ,PS3 ¼ f<3>g. All together, PS ¼ f<3 5>;<5 3>;<5>;<3>g.Next, we detect P8 based on the Up andDown DAGs of 8 (Figs. 2a and 2b) byevaluating each candidate vertex pair. First, detect the VDVSs for length-1 pattern inPre (8D), i.e., up vertexes 1, 2, 3, and 7. Forvertex 1, first, we check its combination withdown vertex 3, the intersection of the occurrencesets is {3}. Thus, the correspondingsupport is at most 1, which is not a validcombination. Similarly, up vertex 1 and downvertex 5 are also invalid combination. Basedon Lemma 5, all the children of down vertex 5are not valid for up vertex 1. Therefore,VDVS1 ¼ _. Similarly,VDVS3 ¼ _; VDVS7¼ fov (<3>); ov (<5>); ov (<5 3>) g:Since no length-2 pattern exists in Preð8DÞ,the detection stops. Eventually, we haveP8={<8>,<1 8>,<2 8>,<3 8>,<7 8>,<8 3>,<8 5>,<8 3 5>,<8 5 3>,<7 8 3>,<7 8 5>,<7 8 5 3>} 8-UDDAG based on detected patterns in P8is shown in Figure below.



ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

Similarly,

P7={<7>,<7 1>,<7 3>,<7 1 3>,<7 5>,
<7 1 5>,<7 3 5>,<7 5 3>}

P6={<6>,<1 6>,<2 6>,<3 6>,<6 3>,<6 5>,
<6 5 3>,<3 6 5>,<2 6 5>,<1 6 5>}

P5={<5>,<1 5>,<2 5>,<3 5>,<5 5>,<1 3 5>,<1
5 5>,<5 1>,<5 3>,<5 5>,<1 5 1>,<1 5
3>}

P4={<4>,<1 4>,<4 1>,<1 4 1>}

P3={<3>,<1 3>,<3 1>,<1 3 1>}

P2={<2>}

P1={<1>,<1 1>}

IV. PATTERN COLLECTION

The complete set of patterns is the union of all the subsets of patterns detected above.

A. Advantages of Proposed System:

- Faster
- Efficient
- Less levels of recursion
- Less memory
- Bidirectional

V. CONCLUSION

In this paper, a novel data structure UDDAG is invented for efficient pattern mining. The new approach grows patterns from both ends (prefixes and suffixes) of detected patterns, which results in faster pattern growth because of less levels of database projection compared to traditional approaches. The extension of this approach to other types of sequential pattern mining problems, e.g., mining with constraints, closed patterns mining, approximate pattern mining, and domain-specific pattern mining, etc. there is a need to extend UDDAG-based approach to other areas, where large searching spaces are involved and pruning of searching spaces is necessary

VI. REFERENCES

[1] R. Agrawal and R. Srikant, "Mining SequentialPatterns," Proc.Int'l Conf. Data Eng. (ICDE'95), pp. 3-14, 1995.
[2] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," Proc. 20th Int'l Conf. Very Large Data Bases (VLDB), pp. 487-499, 1994.
[3] C .Antunes and A.L .Oliveira, "Generalization ofPattern-Growth Methods for Sequential PatternMining with Gap Constraints," Proc. Int'l Conf. Machine Learning and Data Mining 2003, pp. 239-251, 2003.
[4] J. Ayres, J. Gehrke, T. Yu, and J. Flannick, "Sequential Pattern Mining Using a Bitmap Representation," Proc. Int'l Conf. Knowledge Discovery and Data Mining 2002, pp. 429-435, 2002.
[5] S. Berkovich, G. Lapir, and M. Mack, "A Bit-Counting Algorithm Using the Frequency Division Principle," Software: Practice and Experience, vol. 30, no. 14, pp. 1531-1540, 2000.
[6] J. Chen and T. Cook, "Mining Contiguous Sequential Patterns from Web Logs," Proc. World Wide Web Conf. (WWW '07) Poster Session, May 2007.
[7] J. Chen and K. Xiao, "BISC: A Binary Itemset Support Counting Approach Towards Efficient Frequent Itemset Mining," to be published in ACM Trans. Knowledge Discovery in Data.
[8] G. Grahne and J. Zhu, "Efficiently Using Prefix-Trees in Mining Frequent Itemsets"Proc.Workshop Frequent ItemsetMiningImplementations,2003.
[9] M. Garofalakis, R. Rastogi, and K. Shim, "SPIRIT: Sequential Pattern Mining with Regular Expression Constraints," Proc. Int'l Conf. Very Large Data Bases (VLDB '99), pp. 223-234, 1999.
[10] J. Han, J. Pei, B. Mortazavi-Asl, Q. Chen, U. Dayal, and M.C. Hsu, "FreeSpan: Frequent Pattern-Projected Sequential Pattern Mining," Proc. ACM SIGKDD, pp. 355-359, 2000.
[11] M.Y. Lin and S.Y. Lee, "Fast Discovery ofSequential Patterns through Memory Indexing andDatabase Partitioning," J. Information Science andEng., vol. 21, pp. 109-128, 2005.

[12] F. Masseglia, F. Cathala, and P. Poncelet, "The PSP Approach for Mining Sequential Patterns," Proc. European Symp. Principle of Data Mining and Knowledge Discovery, pp. 176-184, 1998.
[13] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M.C. Hsu, "PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth," Proc. 2001 Int'l Conf. Data Eng.(ICDE '01), pp. 215-224, 2001.
[14] J. Pei, J. Han, B. Mortazavi-Asl, J. Wang, H. Pinto, Q. Chen, U.Dayal, and M.C. Hsu, "Mining Sequential Patterns by Pattern- Growth: The PrefixSpan Approach," IEEE Trans. Knowledge and Data Eng., vol. 16, no. 11, pp. 1424-1440, Nov. 2004.
[15] E.M. Reingold, J. Nievergelt, and N. Deo,Combinatorial Algorithms—Theory and Practice. Prentice-Hall, Inc., 1977.
[16] R. Srikant and R. Agrawal, "Mining Sequential Patterns: Generalizations and Performance Improvements," Proc. Int'l Conf. Extending Database Technology 1996, pp. 3-17, 1996.
[17] K. Wang, Y. Xu, and J.X. Yu, "Scalable Sequential Pattern Mining for Biological Sequences," Proc. 2004 ACM Int'l Conf. Information and Knowledge Management, pp. 178-187, 2004.
[18] J. Wang, Y. Asanuma, E. Kodama, T. Takata, and J. Li, "Mining Sequential Patterns MoreEfficiently by Reducing the Cost of Scanning Sequence Databases," IPSJ Trans. Database, vol. 47,no. 12, pp. 3365-3379, 2006.
[19] M. Zaki, "Spade: An Efficient Algorithm for Mining Frequent Sequences," Machine Learning, vol. 40, pp. 31-60, 2001.
[20] Z. Zhang and M. Kitsuregawa, "LAPIN-SPAM: An Improved Algorithm for Mining Sequential Pattern," Proc. Int'l Special Workshop Databases for Next Generation Researchers, pp. 8-11, Apr.2005.
[21] Z. Zhang, Y. Wang, and M. Kitsuregawa, "Effective Sequential Pattern Mining Algorithms for Dense Database," Proc. Japanese Nat'l Data Eng. Workshop (DEWS 06), 2006..

CITE AN ARTICLE

Gurusamy, V., & Nandhini, K. `. (2017). MINING MAXIMAL PERIODIC PATTERNS IN FEWER LEVELS OF RECURSION. *INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY, 6*(11), 151-154.