Research Article Impact Factor: 0.621 ISSN: 2319-507X
Vishal Rathod, IJPRET, 2014; Volume 2 (8): 387-398 IJPRET

INTERNATIONAL JOURNAL OF PURE AND APPLIED RESEARCH IN ENGINEERING AND TECHNOLOGY

A PATH FOR HORIZING YOUR INNOVATIVE WORK

MINING SEQUENTIAL PATTERN WITH DELTA CLOSED PATTERNS AND NONINDUCED PATTERNS FROM SEQUENCES USING SUFFIX TREE

VISHAL B. RATHOD[1], PROF. A. V. DEORANKAR[2], DR. P. N. CHATUR[3]

1. M. Tech CSE student, Government College Of Engineering, Amravati, India.
2. HOD Information Technology Department, Government College Of Engineering, Amravati, India.
3. HOD Computer science Department, Government College Of Engineering, Amravati, India.

Accepted Date: 15/02/2014 ; Published Date: 01/04/2014

Abstract: The suffix tree data structure plays an important role in the efficient implementations of some querying algorithms. In genome study identifying the repeats in DNA sequence is typical problem. There are various approaches designed to find patterns in the DNA sequence. As genome data becomes large, we require fast and efficient algorithm to identify such patterns. In this paper we are using ukkonens algorithm to construct suffix tree. Ukkonen is the method of choice for most problems requiring the construction of a suffix tree. In the real world, usually a large set of patterns could be discovered yet many of them are redundant, thus degrading the output quality. By using two algorithm proposed by Andrew K.C. Wong & Dennis Zhuang we can remove redundant pattern, by delta tolerance closed item set to remove redundant patterns that are not delta closed. The concept of statistically induced patterns is proposed to capture redundant patterns which seem to be statistically significant.

Keywords: Closed frequent item set, delta closed patterns, statistically induced patterns, suffix tree

PAPER-QR CODE

Corresponding Author: MR. VISHAL B. RATHOD

Access Online On:

www.ijpret.com

How to Cite This Article:

Vishal Rathod, IJPRET, 2014; Volume 2 (8): 387-398

INTRODUCTION

Sequence data are a very significant type of data in many forms: biological sequence, web click stream, custom purchase history, event sequence, etc. Today, a vast amount of such data from the genomic, proteomic, and business arenas has been acquired. The discovery of new interesting knowledge from these enormous sequence data has important applications and great value in many sectors but the discovery process needs to be carried out in an efficient and effective way. We take multiple strings (i.e., DNA and protein sequences) as input sequences and substrings as patterns, which are referred to as contiguous patterns. Over the years, many algorithms have been proposed to discover frequent patterns. However, most of them overlook the output quality, producing a very large set of patterns, many of which are highly redundant.

In item set mining, it has been observed that frequent item sets (FIs) often contain redundancies. Thus, closed frequent item sets (CFIs) are proposed as a concise representation of FIs. Although CFIs contain fewer redundancies than FIs, sometimes the definition of the closure is too restrictive and the compression is fairly low as they retain all the information. Hence, CFIs are further extended to the delta-tolerance closed item sets aiming at giving a more concise representation. Delta closed item sets provide a controllable tight lossy approximation to the closed item sets. By allowing tunable tolerance in delta closed item sets, a great number of redundant item sets are pruned while important information is retained. We employ the notion of delta closed item set to discover delta closed patterns from sequences.

To further shrink the output size, many methods attempt to extract statistically significant patterns from frequent patterns. The motivation is that frequent expected patterns might not be as interesting as statistically unexpected or significant patterns. Statistical significance is evaluated through statistical hypothesis test which measures how much the frequency of a pattern deviates from the expected one given the random model. It is hoped that patterns occurring with significantly higher frequency will correspond to the functional units inherent in the sequences. Furthermore, the assessment of statistical significance can help in ranking output patterns, enabling experts to assess the result. However, among the statistically significant patterns some are actually statistically redundant. They are considered as significant merely because they contain very strong significant sub patterns. We introduce the notion of statistically induced patterns to capture such redundant patterns.

2. RELATED WORK

Discovering Sequential patterns

Discovering sequence pattern is important as it has impact on the Science and Society, mainly on studies in biology in which DNA sequences are studied . For discovering these patterns sequential pattern mining is important. This problem of sequential pattern mining was introduced by Srikant and Agrawal.

A. Sequential Pattern Mining

Sequential Pattern Mining addresses the problem of discovering frequent subsequences as a pattern in a sequence database. It extracts patterns that appear more frequently than a user-specified minimum support while maintaining their item occurrence order [16]. It has been extensively studied in data mining communities since the first research work in [16]. A variety of algorithms have been proposed over the last decade. In general, there are Apriori-based algorithms and Pattern-growth algorithms. Most of algorithms such as AprioriAll, BIDE[18], GAP-BIDE[20]. For sequential pattern mining are based on the Apriori property proposed in association rule mining. Pattern-growth algorithms such as FreeSpan, PrefixSpan [18]. AprioriAll algorithm has two steps. The first stage is to generate the candidate sequences and those sequences may be frequent. Next, the sequential database is scanned to check the support of each candidate to determine the frequent sequential patterns according to the minimal support. The main drawback of AprioriAll is that it is not so efficient because of too many passes over the database are required and too many candidates are generated.

B. Prefix Span

Prefix span is based on recursively constructing the patterns, and simultaneously, restricting the search to projected databases. An a-projected database is the set of subsequences in the database that are suffixes of the sequences that have prefix . At each step, the algorithm looks for the frequent sequences with prefix a in the corresponding projected database. In this way, the search space is reduced at each step, allowing for better performances in the presence of small support thresholds. In general, pattern growth methods can be seen as depth-first traversal algorithms because they construct each pattern separately in a recursive way. Prefix Span is a more efficient algorithm for mining sequential patterns compared to GSP and Apriori. Prefix Span is also capable of dealing with a very large database. Prefix Span mainly employs the method of database projection to make the database for the next pass much smaller and can make the algorithm faster; also in Prefix Span there is no need for candidates generation only recursively project the database according to their prefix. In PrefixSpan, there are three different projection methods, level-by-level projection, bi-level projection and pseudo-projection. In recent years, mining sequential patterns from a multidimensional sequential database is proposed in [18]. An algorithm for mining approximate sequential patterns has been developed.

C. Frequent Closed Sequence

Mining frequent sets generates the large set of output and i contains redundant pattern. So, concept of closed frequent set was introduced. Given a support threshold t , a sequence S is a frequent sequence on if the occurs not less than t times. If sequence is frequent and there exists no proper super sequence of with the same support, we call a frequent closed sequence.

D. BIDE

In [20] this paper, they present BIDE an algorithm for mining frequent closed sequences without candidate maintenance. It adopts a novel sequence closure checking scheme called BI-Directional Extension, and prunes the search space more deeply compared to the previous algorithms by using the Back Scan pruning method and the Scan Skip optimization technique.

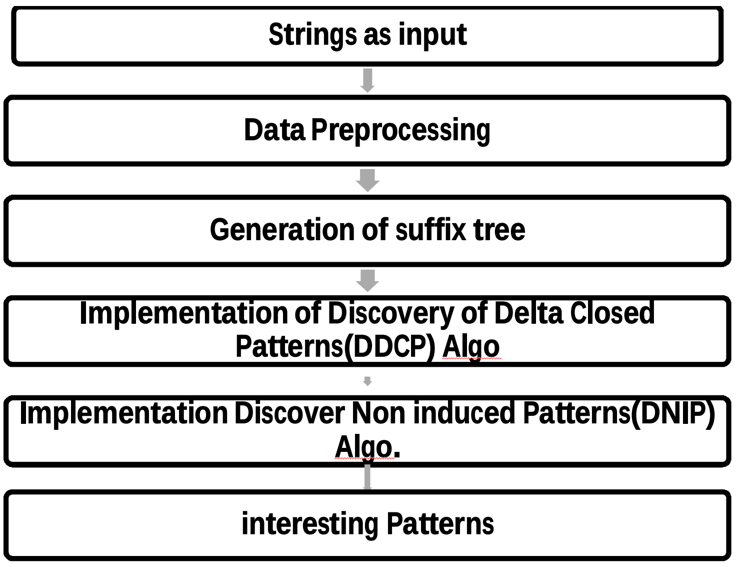
E. Gap-BIDE

In Gap-BIDE we can specify the gap constraint and use it to get contiguous sequential patterns. As Gap-BIDE inherits the same design as BIDE algorithm, it shares the same merit, it does not need to maintain a candidate pattern set, which saves space consumption. This algorithm is used where we have to mine data with certain Gap constraints, for example, in case of Biological Studies. This algorithm mainly works on theorem that a gap constrained of pattern P is closed, if and only if there is no forward and backward extension for the pattern in given gap constrain.

3. PRPOSED METHOD

In this proposed approach strings are taken as input for sequences to find interesting patterns. Here “Pride and Prejudice” novel is taken as dataset from which punctuation & space between words are removed to get long string over the alphabet of 26 english letters. The purpose of removing the

FIG. 1. DECISION SUPPORT SYSTEM


Strings as input
Data Preprocessing
Generation of suffix tree
Implementation of Discovery of Delta Closed Patterns(DDCP) Algo
Implementation Discover Non induced Patterns(DNIP) Algo.
interesting Patterns

delimiter information is to explicitly show that no prior information is used in the pattern discovery process and the segmentation procedure presented later. After this by using generalized suffix tree calculate path length and according to that two algorithm are applied to it to get interesting patterns. Fig. 1 presents the steps described in proposed approach. Two algorithm used in proposed approach are Discovery of delta closed patterns (DDCP) & Discovery of noninduced patterns (DNIP)

3.1 SUFFIX TREE:

Let s' denote a string over the alphabet Σ. Let $ ∈ Σ be a unique termination character and s = s' $. Then , we use s for constructing suffix tree , which has following properties:

1.The tree has n leaves, labelled 1 . . . n, one corresponding to each suffix of s.

2.Each internal node has at least 2 children.

3.Each edge in the tree is labelled with a substring of s.

4.The concatenation of edge labels from the root to the leaf label gives suffix

5.The labels of the edges connecting a node with its children start with different characters .

Algorithm for Discovery of delta closed patterns is given in algorithm 1 as follows[7]

ALGORITHM 1: DDCP

```
Discovery of delta closed patterns (DDCP)
•Construct generalized suffix tree T for the input sequences.
•Annotate k(x) the number of positions under each node x of T.
•Extract set of nodes whose k(x) >= min_occ
•Sort  the above nodesin descending order according to the length of pl(x) using counting sort.
•For each node x
a) Let y be the cover node of x.
b) Let xs and xp be the suffix node and parent node of x, respectively.
c) If x has no cover node
i.   Annotate x as delta closed
d) Else
i.   If  pl(x) is delta closed $(k(y)< \delta.k(x))$
ii.  Else
Annotate x as delta closed
iii.  End If
e) End If
```

• If x is delta closed End for

•Output the path labels of nodes annotated as delta closed

•Set the cover node ys of xs tox if ys does not

•exist or k(x)>k(y).

•Set the cover node yp of xp to x if ypdoes not

•exist or k(x)>k(yp)

•g) Else

•Set the cover node ys of xs to y if ys does not

•exist or k(y)>k(ys)

•Set the cover node yp of xp yo y if yp does not

•exist or k(y)>k(yp)

Algorithm for Discovery of noninduced patterns is given in algorithm 2 as follows[7]

ALGORITHM 2: DNIP

Discovery of noninduced patterns (DNIP)

•Use algorithm 1 to anotate each node v to be delta

closed or not and extract nodes with k(x)>= min_occ

•Sort the nodes in ascending order according to the

length of their path lables.

•For each node x

a) Find the valid node w for x using Procedure 1.

b) If pl(x) is delta closed , statistically significant and is

not statistically induced by pl(w)

```
i. Mark x as noniduced and output pl(x)

•End for

 Procedure 1 Find the valid node for x

• Let xs and xp be the suffix node and parent node of x,

 respectively

• Let ws and wp be the valid nodes of xs and xp

• If xs is noninduced and  k(xs)/pr(pl(xs)) >= k(ws)/pr(pl(ws))[**]

let w1 be the xs

     4.   Else

       Let w1 be the ws

     5.  End if

     6.  If xp is noninduced and k(xp)/pr(pl(xp)) >= k(wp)/pr(pl(wp))

       Let w1 be the xp

     7.  Else

          Let w2 be the wp

     8.  End If

     9.  If  k(w1)/pr(pl(w1)) >= k(w2)/pr(pl(w2))

     10. Else

           w2 is the valid node for x

     11. End If
```

CONSTRUCTION OF SUFFIX TREE

Suffix tree is important data structure for processing the strings. Ukkonen's algorithm constructs an implicit suffix tree Ii for each prefix S[1..i] of S, starting from I1, and incrementing i by one until Im. is built. The true suffix tree for S is constructed from Im and the time for the entire algorithm is O(m). Ukkonen's algorithm first presented for O(m)-time method to build all trees Ii and then optimizing its implementation to obtain the claimed time bound.

ALGORITHM 3: HIGH-LEVEL UKKONEN ALGORITHM

High-level Ukkonen algorithm

```
Construct tree Ii.
    For i from 1 to m — 1 do
    begin [phase i + 1}
    For j from 1 to i + 1
    begin {extension j}
    Find the end of the path from the root labeled S[j..i] in the
    current tree, If needed, extend that path by adding character S[i + 1),
    thus assuring that string S[j. .i + 1] is in the tree.
    end;
end;
```

This algorithm above will use small tricks to obtain the linear time construction which are:

1. Trick 1: Suffix Links

If there is node v with label p$,where p is character and $ is string then the suffix link of v points to s(v),which has label $.

2. Trick 2: Skip / Count

This is known as the skip/count trick. This helps us to find the place for adding the new S[i+1] in the phase (i+1) very quickly.

3. Trick 3: Once a leaf, always a leaf.

The observation of the fact that, if we create leaf node in suffix tree it will be leaf node forever. This helps to build suffix tree fastly. In this case, at leaf node we just require to change the staring position of the edge lable comming towards it from the parent of the leaf node.

4. Trick 4:

If at any phase i, the character S[i] is found then that character will be found in further all the suffixes of that phase , so we do not require to search in further suffixes of that phase. This trick makes algorithm faster.

4.Conclusion And Future Scope

In this paper we review the existing techniques of pattern mining . We discussed a variety of pattern mining method techniques such as Prefix Span Approach , Delta Closed Patterns and Noninduced Patterns from Sequences. For each technique we have provided a detailed explanation of the techniques which are used for finding the patterns. It is observed that for finding frequent pattern there is problem of fake pattern & redundant. So to solve this problem there is Delta Closed Patterns and Noninduced Patterns algo which gives good output in minimum time. They produce a relatively small set of patterns which reveal interesting information in the sequences.

Mining of patterns using Discovery of Delta Closed Patterns and Noninduced Patterns using suffix tree gives patterns and proposes the notion of statistically induced patterns to capture redundant patterns. Here efficient algorithms for discovering delta closed patterns and noninduced patterns from large sequence data given. Two algorithms that use a generalized suffix tree in an innovative manner, assisting the identification of these patterns effectively in linear time. The proposed approach is very useful to give interesting patterns at the end.

REFERENCES

1. Hongwei Huo ,Xiaowu Wang, Stojkovic, “An Adaptive Suffix Tree Based Algorithm for Repeats Recognition in a DNA” International joint conference Bioinformatics, Systems Biology and Intelligent Computing, 2009. IJCBS '09, pp. 181 – 184, Aug. 2009.

2. J. Zheng and S. Lonardi. “Discovery of Repetitive Patterns in DNA with Accurate Boundaries” Proc. of Fifth IEEE Symposium on Bioinformatics and Bioengineering (BIBE’05), Minneapolis, Minnesota, pp. 105-112, 2005.

3. Dan He, “Using Suffix Tree to Discover Complex Repetitive Patterns in DNA Sequences,” IEEE International Conference of the Engineering in Medicine and Biology Society (EMBS 06), IEEE Press, pp. 3474-3477, Aug. 2006.

4. Neil C. Jones and Pavel A. Pevzner, Introduction to Bioinformatics Algorithms, 1st ed., MIT Press: Cambridge, pp.311-337, 2004.

5. Kurtz S, Choudhuri J. V, Ohlebusch E, Schleiermacher C, Stoye J, and Giegerich R, “REPuter: The Manifold Applications of Repeat Analysis on a Genomic Scale,” Nucleic Acids Res., vol. 29, no.22, pp. 4633–4642, 2001.

6. Kurtz S, Choudhuri J. V, Ohlebusch E, Schleiermacher C, Stoye J, and Giegerich R, “REPuter: The Manifold Applications of Repeat Analysis on a Genomic Scale,” Nucleic Acids Res., vol. 29, no.22, pp. 4633–4642, 2001.

7. Gusfield D, Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology, 1st ed, Cambridge University Press:Cambridge, 1997, pp87-168.

8. Lander E.S, Linton L.M, Birren B, et al., “Initial Sequencing and Analysis of the Human Genome,” Nature, vol. 409, pp. 860-921,Feb. 2001.

9. R. Giegerich and S. Kurtz, “From Ukkonen to McCreight and Weiner: A Unifying View of Linear-Time Suffix Tree Construction,” Algorithmica, vol. 19, no.22, 1997,pp. 331-353.

10. Esko Ukkonen, “On-line construction of suffix tree,” Algorithmica, vol. 14, no.3, pp.249-260, 1995.

11. Lefebvre A, Lecroq T, Dauchel H. and Alexandre J., “FORRepeats: Detects Repeats on Entire Chromosomes and between Genomes,”Bioinformatics, vol. 19, no. 3, pp. 319-326, , 2003.

12. Wong A.K.C. , Zhuang, D., Li G.C.L, Lee E.-S.A.“Discovery of Delta Closed Patterns and Noninduced Patterns from Sequences “ ,IEEE Transactions Knowledge and Data Engineering, vol. 24 , no.8. ,pp.1408 - 1421 ,Aug. 2012.

13. N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal, “Discovering Frequent Closed Itemsets for Association Rules,” Proc. Seventh Int’l Conf. Database Theory, pp. 398-416, 1999.

14. J. Cheng, Y. Ke, and W. Ng, “#-Tolerance Closed Frequent Itemsets,” Proc. Sixth Int’l Conf. Data Mining, pp. 139-148, 2006. S.C. Chan and A.K.C Wong, “Synthesis and Recognition of

Sequences," IEEE Trans. Pattern Analysis Machine Intelligence, vol. 13, no. 12, pp. 1245-1255, Dec. 1991.

15. A.K.C. Wong, D.K.Y. Chiu, and S.C. Chan, "Pattern Detection in Biomolecules Using Synthesis Random Sequence," J. Pattern Recognition, vol. 29, no. 9, pp. 1581-1586, 1995.

16. R. Agrawal and R. Srikant, "Mining Sequential Patterns," Proc. 11th Int'l Conf. Data Eng., pp. 3-14, 1995.

17. R. Srikant and R. Agrawal, "Mining Sequential Patterns: Generalizations and Performance Improvements," Proc. Fifth Int'l Conf. Extending Database Technology, pp. 3-17, 1996.

18. J. Pei and J. Han, "PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth," Proc. 17th Int'l Conf. Data Eng., pp. 215-224, 2001.

19. C. Antunes and A.L. Oliveira, "Generalization of Pattern-Growth Methods for Sequential Pattern Mining with Gap Constraints," Proc. Int'l Conf. Machine Learning and Data Mining, pp. 239-251,2003.

20. Jianyong Wang , Jiawei Han ,"BIDE: efficient mining of frequent closed sequences", Proc. 20th International Conference on Data Engineering,pp 79-90, 2004.

21. Floratou. A, Tata. S, Patel J.M., "Efficient and accurate discovery of patterns in sequence datasets",IEEE 26th International Conference on Data Engineering (ICDE), pp. 461 – 472, 2010.

22. J. Chen, "Contiguous Item Sequential Pattern Mining using UpDown Tree," J. Intelligent Data Analysis, vol. 12, no. 1, pp. 25-49, Jan. 2008.

23. X. Yan, J. Han, and R. Afshar, "CloSpan: Mining Closed Sequential Patterns in Large Databases," Proc. Third SIAM Int'l Conf. Data Mining, pp. 166-177, 2003.

24. J. Wang and J. Han, "BIDE: Efficient Mining of Frequent Closed Sequences," Proc. 20th Int'l Conf. Data Eng., pp. 79-90, 2004.

25. C. Li and J. Wang, "Efficiently Mining Closed Subsequences with Gap Constraints," Proc. Eighth SIAM Int'l Conf. Data Mining, pp. 313-322, 2008.

26. Syed Khairuzzaman T., Chowdhury F.,"Discovering Periodic-Frequent Patterns in Transactional Databases",13th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining ,pp. 242-253, 2009.