Volume 2, No. 2, February 2011
Journal of Global Research in Computer Science
ISSN-2229-371X
REVIEW ARTICLE
Available Online at www.jgrcs.info

A Framework for associated pattern mining over Microarray database

Nilamadhab Mishra
Department Of Computer Science &Application
Krupajal Group of Institutions, Orissa, India.
nilamadhab.mishra@rediffmail.com

Abstract**:** - Microarray database is a typical Relational database ,which contains a large number of columns and a small number of rows, and it poses a great challenge for existing associated pattern mining algorithms that discover patterns in item enumeration space. Here I want to Review some algorithms which helps to explore the row enumeration space to mine associated patterns. The row enumeration algorithms are used to avoid searching the large number of columns /items enumeration space, but those algorithms can try to search the associated patterns in the row enumeration space. The column enumeration algorithms can not be scaled to microarray database, where as it is possible to scale the row enumeration algorithms to microarray database. So I can right to say that the associated patterns /rules can be the better search substitutes, which can minimize the search time and complexcity. So instead of searching the large number of columns in a microarray database (bioinformatics database), its associated framing patterns should be searched.

Keywords**:** Associated pattern, CHARM, RERII, CLOSET, CARPENTER, CLOSET+, REPT.

INTRODUCTION

Microarray database may contain up to thousands or tens of thousands of columns (genes) but only tens or hundreds of rows. Discovering frequent patterns from microarray database is very important and useful, especially in the following:
1) To discover association rules, which can not only reveal biological relevant associations between genes and environments/categories to identify gene regulation pathways but also help to uncover gene networks [1].
2) To discover bi-clustering of gene expression as shown in [8].However, these high-dimensional microarray database pose a great challenge for existing frequent pattern discovery algorithms. While there are a large number of algorithms that have been developed for frequent pattern Discovery and associated pattern mining [3, 4, 7], their basic approaches are based on item enumeration in which combinations of items are tested systematically to search for frequent associated patterns. As a result, their running time increases exponentially with increasing average length of the records. The high dimensional microarray database render most of these algorithms impractical. It was first shown in [2] that the complete frequent associated patterns can also be obtained by searching in the row enumeration space, which was also observed in [5]. Moreover, [9] proposed an algorithm, CARPENTER, to explore the row enumeration search space by constructing projected transposed database recursively. Considering that many algorithms have been proposed to mine frequent associated patterns by item enumeration, it would be interesting to investigate whether some ideas can be borrowed from these algorithms to search row enumeration space more efficiently. In this paper, two new efficient algorithms, RERII and REPT are reviewed to explore the row enumeration space to discover frequent associated patterns. Algorithm RERII is inspired by algorithms that mine patterns from vertical layout data [7], while algorithm REPT is inspired by algorithms that are based on FP-tree [4]. But RERII and REPT are very different from them in that both of them adopt row enumeration. Compared with CAR-PENTER, RERII and REPT use different implementation methods and employ more powerful pruning methods. Several experiments are performed on real-life microarray database to show that the new algorithms are much faster than the existing algorithms, including CLOSET [4], CHARM [7], CLOSET+[6]andCARPENTER[2].
CARPENTER [3] is developed to perform row enumeration on bioinformatics database.

EX: Table (Tab-1)

i	ri
1	a, b, c, l, o, s
2	a, d, e, h, p, l, r
3	a, c, e, h, o, q, t
4	a, e, f, h, p, r
5	b, d, f, g, l, q, s, t

Transposed Table, TT (Tab-2)

Fj	R (fj)
a	1,2,3,4
b	1, 5
c	1, 3
d	2, 5
e	2, 3,4
f	4, 5
g	5
h	2, 3,4
l	1,2,5
o	1,3
p	2, 4
q	3, 5
r	2,4
s	1,5
t	3,5

Row enumeration tree

1. Start ( ) As root node.
2. Place i value 1 2 3 4 5 as child nodes under this root .
3. For next child nodes expand each i value. for i=1,place (1,2)(1,3)(1,4)(1,5) as one one child node and take the common Ri .
4. Create further child nodes by combine three i values (123,124,125 etc.)and take common Ri.
5. Finally create the leaf nodes by combine 4 i values and take common Ri.

The row enumeration algorithm uses the row enumeration tree to find out the closest associated patterns.

CARPENTER is a row enumeration algorithm which looks for frequent associated patterns by testing various combinations of rows. Since the bioinformatics database has small number of rows and large number of features, the number of row combinations will be much smaller than the number of feature combinations. As such, row enumeration algorithms like CARPENTER will be more efficient than feature enumeration algorithms on these kinds of database. From the above, it is natural to make two observations.

First, we can conclude that different database will have different characteristics and thus require a different enumeration method in order to make associated pattern mining efficient. Furthermore, since these algorithms typically focus on processing different subset of the data during the mining, the characteristics of the data subset being handled will change from one subset to another. For example, a dataset that has much more rows than features may be partitioned into sub-database with more features than rows. Therefore a single feature enumeration method or a single row enumeration method may become inefficient in some phases of the enumeration even if they are the better choice at the start of the algorithm. As such, it makes sense to try to switch the enumeration method dynamically as different subsets of the data are being processed. Second, both classes of algorithms will have problem handling database with large number of features and large number of rows. This can be seen if we understand the basic philosophy of these algorithms. In both classes of algorithms, the aim is to reduce the amount of data being considered by searching in the smaller enumeration space.

For Example, when performing feature enumeration, the number of rows being considered will decrease as the number of features in a feature set grows. It is thus possible to partition the large number of rows into smaller subset for efficient mining. However, for database with large number of rows and large number of features, adopting only one single enumeration

method will make it difficult to reduce the data being considered in another dimension. Motivated by these observations, we derived a new algorithm called COBBLER.

COBBLER is designed to automatically switch between feature enumeration and row enumeration during the mining process based on the characteristics of the data subset being considered. This approach will produce good results when handling different kinds of database.

PRELIMINARY

Let I = {i1, i2... im} be a set of items. Let D be the dataset (or table) which consists of a set of rows R= {r1... rn} with each row ri consisting of a set of items in I, i.e. RI.

Here, I want to introduce two concepts called feature support set and row support set.

Definition 1
Feature Support Set, R (F'). Given a set of features F' which is either subset or equal to F .We use R(F') is subset of equal to R to denote the maximum set of rows that contains F' .

Definition 2
Row Support Set, F (R'). Given a set of rows R' which is either subset or equal to R .We use F(R') is subset of equal to F to denote the large set of features that are common amount the rows in R' .

PROBLEM DEFINITION

Given a dataset D which contains records that are subset of a set of items I , the problem is to discover all frequent associated patterns with respect to a user given support threshold minus. In addition, we assume that the database satisfies the condition |R| << |I|. To solve this problem, the CARPENTER is designed based on two basic concepts. One is projected transposed table and the other is row enumeration. In the microarray database the required rows has to be enumerated but the columns do not require any further enumerations. The COBBLER: Combining Column and Row Enumeration. It is the Extension of CARPENTER to handle database with both large number of columns and rows and also it switches dynamically between column and row enumeration based on the estimated cost of processing. The Switching Conditions are (1) Naïve idea of switching based on row number and feature number does not work well.
(2) To estimate the required computation of an enumeration Sub-tree, i.e., row enumeration sub-tree or feature enumeration sub-tree. Estimate the maximal level of enumeration for each children sub tree.
As we can see, the basic characteristic of a row enumeration tree or a feature enumeration tree is that the tree is static. The current solution is to make a selection between these approaches based on the characteristic of the enumeration algorithm.
For database with many rows and few features, algorithms like CHARM [11] and CLOSET+ [10] that search in the feature enumeration tree will be more efficient since the number of possible feature combinations will be small.
However, when the number of features is much larger than the number of rows, a row enumeration algorithm like CARPENTER [9] was shown to be much more efficient. There are two motivations for adopting a more dynamic approach.
First, the characteristics of the conditional tables could be different from the original table. Since the number of rows (or tuples) can be reduced as we move down the enumeration tree, it is possible that a table which has more rows than features initially, could have the characteristic reversed for it's conditional tables (i.e. more features than rows). As such, it makes sense to adopt a different enumeration approach as the data characteristic changes.
Second, for database with large number of rows and also large number of features, a combination of row and feature enumeration could help to reduce both the number of rows and features being considered in the conditional tables thus enhancing the efficiency of mining.

SOME EFFECTIVE ALGORITHMS

Algorithm REPT

Like CARPENTER, algorithm REPT traverses the row enumeration tree with the help of projected transposed table. Its first main difference from CARPENTER is that REPT represents (projected) transposed table with prefix trees, which can help in saving memory and saving computation in counting frequency. The second main difference of REPT from CARPENTER lies in pruning method. The prefix tree used to represent transposed table is similar to the FP-tree used in [4] to represent original table. In FP-tree, each node represents an item while the node of prefix tree used in REPT represents a row.

4.2 Algorithm RERII (D, minsup)
1. Scan database D to find the set of frequent items F
2. Remove the infrequent items in each row ri of D
3. Each ri forms a node in the first level of row enumeration tree and let N be the set of nodes
4. RERIIdepthfirst (N, F CP)
5. Let CF be the set of closed items in F, F CP= F CP and CF
Return F CP

RERII has already discovered the set of frequent single items by scanning the database once, we need to discover those frequent associated closed patterns .
Hence various algorithms are there like CHARM, RERII, CLOSET, CARPENTER, CLOSET+, REPT are already proposed to mine the associated patterns. If we analyze the memory usage of various algorithms we observe that REPT consumes least memory space while CHARM consumes most memory space. Also if we make further analysis we observe that Tree based schemes (e.g., CLOSET, REPT using FP-tree) generally consume less memory, while non-tree- based algorithms(e.g., CHARM, RERII) are typically more efficient on the data that we use .

CONCLUSION

The associated pattern mining is a vital topic which has drawn the attention during the past decade.

The number of associated patterns in a large data set can be very large and many of these associated patterns may be redundant. To reduce the frequent associated patterns to a compact size, mining frequent closed associated patterns has been already proposed.

Another algorithm for mining frequent closed associated pattern is CARPENTER. CARPENTER is a pure row enumeration algorithm. CARPENTER discovers frequent closed associated patterns by performing depth-first, row enumeration combined with efficient search pruning techniques. CARPENTER is especially designed to mine frequent closed associated patterns in database Containing large number of columns and small number of rows. So this algorithm can be effectively used to frame the associated closed Patterns over microarray database.

FUTURE WORK

In my future work, I want to implement the associated pattern mining through the genetic algorithm. Before finding out the associated pattern, the large dataset is to be normalized .To minimize the enumerated space, the genetic algorithm can be implemented successfully. So here I can use the genetic algorithm as enumerated space optimizer and it will help me to find out the optimal solutions. I have also keen interest to implement the features of Associative memory (soft computing) over the microarray database to extract the associated pattern pairs.

REFERENCES

[1]C. Creighton and S. Hanash. Mining gene expression datasets for association rules. Bioinformatics, 19, 2003.

[2] F. Pan, G. Cong, A. K. H. Tung, J. Yang, and M. J. Zaki. CAR- PENTER: Finding closed patterns in long biological datasets. In Proc. ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD), 2003.

[3] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. In Proc. 7th Int'l Conf. Database Theory (ICDT), 1999.

[4] J. Pei, J. Han, and R. Mao. CLOSET:An efficient algorithm for mining frequent closed itemsets. InProc. ACM- SIGMOD Int'l Workshop Data Mining and Knowledge Discovery (DMKD), 2000.

[5] F. Rioult, J.-F. Boulicaut, B. Cremileux, and J. Besson. Using Transposition for pattern discovery from microarray data. In Proc. ACM-SIGMOD Int'l Workshop Data Mining and Knowledge Discovery (DMKD), 2003.

[6] J. Wang, J. Han, and J. Pei. CLOSET+: Searching for the best strategies for mining frequent closed itemsets. In Proc. ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining (KDD), 2003.

[7] M. J. Zaki and C. Hsiao. CHARM: An efficient algorithm for closed association rule mining. In Proc. SIAM Int'l Conf. on Data Mining (SDM), 2002.

[8] Z. Zhang, A. Teo, B. Ooi, and K.-L. Tan. Mining deterministic biclusters in gene expression data. In 4th Symposium on Bioinformatics and Bioengineering, 2004.

[9] F. Pan, G. Cong, and A. K. H. Tung. Carpenter: Finding Closed patterns in long biological database. In Proc.Of ACM-SIGKDD Int'l Conference on Knowledge Discovery and Data Mining, 2003.

[10] J. Wang, J. Han, and J. Pei. Closet+: Searching for the best strategies for mining frequent closed item sets. In Proc. 2003ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD'03), Washington, D.C., Aug 2003.

[11] M. Zaki and C. Hsiao. Charm: An efficient algorithm for closed association rule mining. In Proc. of SDM 2002, 2002.

AUTHORS PROFILE

Nialamdhab Mishra: Presently working with Krupajal Group as Asst. Professor of Computer Science & Application and is actively engaged in conducting Academic, Research and development programs in the field of Computer Science and IT Engineering. Contributed various research level papers to many national and International journals. He is an associate life member of computer society of India and also a member of International Association of Engineering and Management Education. Having research interests include Adhoc Network, Sensor Network, Evolutionary Computation and Data Mining.