Analysis of DNA Microarray Data using Association Rules: A Selective Study

M. Anandhavalli Gauthaman

Abstract—DNA microarrays allow the measurement of expression levels for a large number of genes, perhaps all genes of an organism, within a number of different experimental samples. It is very much important to extract biologically meaningful information from this huge amount of expression data to know the current state of the cell because most cellular processes are regulated by changes in gene expression. Association rule mining techniques are helpful to find association relationship between genes. Numerous association rule mining algorithms have been developed to analyze and associate this huge amount of gene expression data. This paper focuses on some of the popular association rule mining algorithms developed to analyze gene expression data.

Keywords—DNA Microarray, Gene expression, Association rule mining.

I. INTRODUCTION

GENE is a segment of DNA, which contains the formula for the chemical composition of one particular protein. Genes serve as the blueprints for proteins and some additional products, and mRNA is the first intermediate during the production of any genetically encoded molecule. The concentration of a specific mRNA molecule is usually called the expression level of the respective gene, and it serves as an indicator of the amount of end product that is currently being produced. Nowadays, the expression levels of thousands of genes, possibly all genes in an organism, can be measured simultaneously in a single experiment using microarrays. This new technology gives rise to a challenge: to interpret the meaning of this immense amount of biological information formatted in numerical matrices. To meet the challenge, various methods have been developed using both traditional and innovative techniques to extract, analyze and visualize gene expression data generated from **DNA microarrays**. A key step in the analysis of gene expression data is to find association and correlation relationship between gene expression patterns.

II. MICROARRAY TECHNOLOGY

Microarray is a technology which enables the researchers to investigate and address issues which were once thought to be non traceable. Microarray technology has empowered the scientific community to understand the fundamental aspects underlining the growth and development of life as well as to explore the genetic causes of anomalies occurring in the functioning of the human body.

The basic principle underlying microarray technology is that complementary nucleic acids will hybridize. This is also the basis for traditional gene expression analyses, such as Southern and Northern blotting. Hybridization provides exquisite selectivity of complementary stranded nucleic acids, with high sensitivity and specificity. In the traditional techniques, in which radioactive labeling materials are usually used, the simultaneous hybridization of test and reference samples is impossible.

III. DNA MICROARRAY EXPERIMENT

A **DNA chip** is the instrument that measures simultaneously the concentration of thousands of mRNA molecules. It also refers to as a DNA microarray. They can measure simultaneously the expression levels of up to 20,000 genes. The DNA microarrays are produced as follows:

Divide a glass or silica plate of 1 cm across (the chip) into pixels. Here each pixel will be dedicated to one gene. Millions of 25 base pair long single strand DNA, copied from a particular segment of gene, is synthesized on the dedicated pixel. These are called **probs**. The mRNA molecules are extracted from the cell taken from tissue of interest (such as cancer tissue). They are reverse transcribed from RNA to DNA and their concentration is enhanced. Then the resulting DNA is transcribed back into fluorescently marked single strand RNA. The solution of marked and mRNA molecules (copies of the mRNA molecules that were originally extracted from the tissue) is placed on the chip and labeled RNA diffuse over the dense forest of single strand DNA probes. When such an mRNA encounters a bit of the probe, of which the RNA is the perfect copy, it attaches to it with high affinity which is called hybridization. After the mRNA solution is washed off, only those molecules that found their perfect match remain fixed to the chip. Now the chip is illuminated with a laser, and those fixed targets fluoresce. By measuring the light radiating from each pixel, one obtains a measure of targets that stuck. It is proportional to the concentration of those mRNA in the investigated tissue.

M. Anandhavalli Gauthaman is with Department of Computer Science & Engineering, Sikkim Manipal Institute of Technology, Majitar, Rangpo, East Sikkim-737132, India (e-mail: anandhigautham@gmail.com).
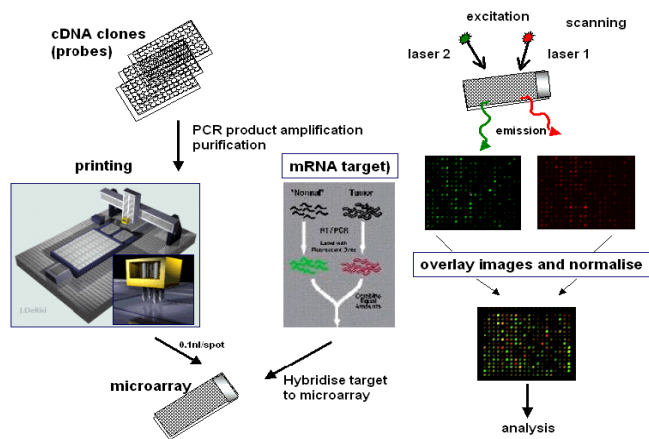
World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:2, No:6, 2008

Fig. 1 DNA Microarray Experiment

IV. GENE EXPRESSION DATA

A typical DNA microarray experiment provides the expression profiles of several tens of samples (say Ns % 100), over several thousand (Ng) genes. These results are summarized in an Ng×Ns expression table; each row corresponds to one particular gene and each column to a sample. Entry Egs of such an expression table stands for the expression level of gene g in sample s. The original gene expression matrix obtained from a scanning process contains noise, missing values, and systematic variations arising from the experimental procedure. Data pre-processing is indispensable before any association rules analysis can be performed.

A. Analysis of Gene Expression Data

Analysis of gene expression data helps the molecular biologists in many aspects, like, gathering information about different cell states, functioning of genes, identifying genes that reflect biological process of interest etc. Several obvious meanings of gene expression data analysis are the following:

- Identify genes whose expression levels reflect biological processes of interest (such as development of cancers).
- Group the tumors into classes that can be differentiated on the basis of their expression profiles, possibly in a way that can be interpreted in terms of clinical classification. If one can partition tumors, on the basis of their expression levels into relevant classes (such as e.g. positive vs. negative responders to a particular treatment), the classification obtained from expression analysis can be used as a diagnostic and therapeutic tool.
- Finally the analysis can provide clues and guesses for the function of genes (proteins) of yet unknown role.

V. ASSOCIATION RULES

Association rule mining finds interesting association and correlation relationships among a large set of data items [8]. The rules are considered interesting if they satisfy both a minimum support threshold and minimum confidence threshold [3]. The most common approach to finding association rules is to break up the problem into two parts [6].

1. Find frequent item sets: By definition, each of these item sets will occur at least as frequently as a pre-determined minimum support count [8].

2. Generate strong association rules from the frequent item sets: By definition, these rules must satisfy minimum support and minimum confidence [8].

The second step is easier of the two. The overall performance of mining association rules is determined by the first step. As shown in [2], the performance, for large databases, is most influenced by the combinatorial explosion of the number of possible frequent itemsets that must be considered and also by the number of database scans that has to be performed. Many conventional association rule mining algorithms (such as A priori [1], FP-growth [2], DynFP-growth [11], Partitioning (Savasere et al., 1999), Dynamic Item set counting (DIC) (Aggarwal, 1998), Direct Hashing and Pruning DHP (Park et al., 1995) etc.) have been adapted or directly applied to gene expression data These association rules mining algorithms have been proven useful for identifying biologically relevant association among the genes.

A. Importance of association rule mining Techniques in Gene Expression

Using association rule mining approach, we can analyze: 1.The expression of one gene leads to the induction of a serial of target gene expressions. This expression pattern is denoted regulation of gene expression. The relationship between one gene and the other target genes can be viewed as an associative relation. 2. Several gene expressions lead to the expression of one target gene. Transcription factors and their target gene is one of many examples in this category (Morishita, 1999). 3. Gene expression leads to the induction of new biological function (Nakaya et al., 2000).

VI. ASSOCIATION RULES MINING ALGORITHMS USED IN GENE EXPRESSION DATA

In this section some of the popular association rule mining techniques used to find the association relationship between gene expression data are reviewed. There are several association rule mining algorithms on gene expression analysis. But in this paper only a few numbers of them are briefly discussed.

A. Apriori

Apriori algorithm (Agrrawal and Srikant 1994) used prior knowledge of frequent itemset properties for mining frequent itemsets for Boolean association rules. It employs an iterative approach known as a level-wise search, where k-itemsets are used to explore (k+1) items. The first pass of the algorithm simply counts item occurrences to determine the large 1-itemsets. A subsequent pass, say pass k, consists of two phases. First, the large itemsets L_{k-1} found in the (k-1)th pass are used to generate the candidate itemsets C_k, using the Apriori candidate generation function . Next the database is scanned and the support of the candidates in C_k is counted. For fast counting, an efficient determination if the candidates in C_k that are contained in a given transaction t is needed. A hash

World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:2, No:6, 2008

tree structure [12] is used for this purpose. The Apriori-gen function takes as argument L_{k-1}, the set of all large (k-1) itemsets. It returns a superset of the set of all large k-itemsets and is described in [12]. If the dataset is huge, the multiple database scan makes the execution of the Apriori algorithm very long. Therefore several algorithms were developed to speed up the Apriori algorithm. These improved algorithms reduce the I/O cost in different ways. This is a level-wise algorithm thus it accesses the database (DB) as many times as the length of the frequent itemset.

B. FP growth

FP-tree growth (Han et al.) adopts a divide-and-conquer strategy that mines the complete set of frequent itemsets without candidate generation. FP-growth algorithm constructs the conditional frequent pattern (FP)-tree and performs the mining on this tree. FP-tree is an extended prefix tree structure, storing crucial and quantitative information about frequent sets. The tree nodes are frequent items and are arranged in a such a way that more frequently occurring nodes will have better chances of sharing nodes than the less frequently occurring ones. The method starts from frequent 1-itemsets as an initial suffix pattern and examines only its conditional pattern base (a subset of the database), which consists of set of frequent items co-occurring with the suffix pattern. The algorithm involves two phases. In phase I, it constructs the FP-tree with respect to a given support factor σ. The construction of this tree requires two passes over the whole database. In phase II, the algorithm dose not use the transaction database anymore, but it uses the FP-tree. Interestingly, the FP-tree contains all the information about frequent itemsets with respect to the given σ. The FP-tree growth method transforms the problem of finding long frequent patterns to searching for shorter ones recursively and then concatenating the suffix. It uses the least frequent items as a suffix, offering good selectivity. The method substantially reduces the search costs [2].

When the database is large, it is sometimes unrealistic to construct a main memory-base FP-tree. A study on the performance of the F-method shows that it is efficient and scalable for mining both long and short frequent patterns.

C. DynFP-growth

An interesting alternative to the FP-growth is **Dynamic Frequent Pattern** algorithm (Gyorodi C et al). In FP-growth, some observations on the way FP-tree constructed a) the resulting FP-tree is not unique for the same "logical" database b) the process needs two complete scans of the database. A solution to these observations were given by Gyorodi C., et al (2003) [11], for first observation by using a support descending order together with the lexicographic order ensuring in this way the uniqueness of the resulting FP-tree for different "logically equivalent" databases and for the second observation by devising a dynamic FP-tree reordering algorithm and employing this algorithm whenever a "promotion" to a higher order of at least one item is detected. Although the resulting FP-tree could be too large to be stored

in its entirety in the main memory, because of its properties, and for a relatively high number of queries with different minimum supports, it would be more practical, form time consuming point of view, to it on disk in its full form and using only the portions that are required from it. Using the dynamic reordering one doesn't have to rebuild the FP-tree even if the actual database is updated. In this case the algorithm has to be performed taking into consideration only the new transactions and the stored FP-tree. This approach can provide a very quick response to any queries even on databases that are being continuously updated fact that is true in many cases. Because the dynamic reordering process, Gyorodi C., et al (2003) [11] proposed a modification of the original structures, by replacing the singly linked list with a doubly linked list for linking the tree nodes to the header and adding a master-table to the same header. All these modifications are presented in more details in [11].

It can be observed that the execution time of DynFP-growth does not depend on support but only on the database size, this is because the tree construction technique does not need the support information. In this way the tree will contain all the database transactions and depending on the required support the results will be refined so that they will contain only the itemsets that have their frequency greater than the required support.

D. Partition

Partition algorithm (Savasere [SON95], 1995) is based on the observation that the frequent sets are normally very few in number to the set of all itemsets. As a result, it divides the database into partitions such that each partition can be placed into main memory. This algorithm reduces the number of database scans to two and when it scans the database it brings that partition into memory and counts the items in that partition alone. The algorithm executes in two phases. In the first phase, it logically divides the data base into a number of non-overlapping partitions. The partitions are considered one at a time and all frequent itemsets for that partition are generated. Thus if there are n partitions, Phase I of the algorithm takes n iterations. At the end of phase I, these frequent itemsets are merged to generate the set of all potential frequent itemsets. In this step, the local frequent itemsets of same lengths from all n partitions are combined to generate the global candidate itemsets. In phase II, the actual support of these itemsets is generated and the frequent itemsets are identified. That is, during the first database scan, it finds all frequent itemsets in each partition. During the second scan, only those itemsets that are frequent in at least in one partition are used as candidates and counted to determine if they are large across the entire database.

These algorithms may be able to adopt better to limited main memory. In addition, it would be expected that the number of itemsets to be counted per partition would be smaller than those need for the entire database. Incremental generation of association rules may be easier to perform by treating the current state of the database as one partition and treating the new entries as a second partition.

World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:2, No:6, 2008

If the itemsets are uniformly distributed across the partitions, then a large fraction of the itemsets will be large. However, if the data are not uniform, there may be a large percentage of false itemsets.

E. Dynamic Itemset Counting (DIC)

The basic idea of **DIC** algorithm (Brin et al. in 1997) is to generate new candidates as early as possible. It scans the database and increments the counters of the candidates when an itemset became frequent so it is possible to generate new candidates based on the new frequent itemset. The candidate generation is a complex task; therefore the new candidates are generated only at checkpoints. The distance of the checkpoints is an important in this algorithm; the optimum is reached at about 10,000 read transactions as described in [4]. It tries to reduce the I/O cost via early candidate generation. The working mechanism of DIC is as follows:

1. Mark empty set with a solid box. All the 1 – itemset are marked with dashed circles & others unmarked.
2. Read M transactions. For each transaction increment the counter marked with dashes.
3. If a dashed circle count exceeds threshold, turn it into a dashed square. If any of the superset has all its subsets as solid or dashed square add counter and make dashed circle to superset.
4. If a dashed itemset has been counted thro' all transactions make it solid & stop counting.
5. If end of file then rewind to beginning.
6. If any more dashed items then goto step 2.

The number of passes is less in DIC if the data is homogenous. Because of its dynamic nature, it is flexible and can be adapted to parallel and incremental mining.

VII. Conclusion

Microarrays have become a standard research tool for today's laboratory. Microarray analysis has been used successfully to define transcriptional signatures to allow for patient-tailored therapy strategy in breast cancer or to classify better tumors having no histological counterparts in normal tissues. It can be a useful tool to identify genes directly activated or repressed by expression of a transcription factor. Subsequently, primary response genes can be identified by computational searching of factor-specific responsive elements in a DNA region located upstream of genes found to be differentially expressed in microarray experiments. But, all these things and possibilities depend on efficient and proper analysis of gene expression data.

Basically, data mining is an application-dependent issue and different applications may require different mining techniques to copy up with. To apply mining association rules in gene expression pattern analysis, we need to understand the properties of gene data. The data in gene expression database are very large, and are divided into different tissues or organs. Most genes are duplicated in different tissues. To study the associations between genes, we need to eliminate these duplicated data since they are present in every tissue. Thus to apply an existing algorithms to the gene analysis, we will need

to modify it to filter the data. Also, we have to consider the negative implication of the results. Furthermore, unlike data mining in business applications, the size of a transaction in gene analysis is relatively small since the number of tissues present in an organism (e.g. human tissues) is limited. However, the number of items (genes) in one single transaction is very large. When we select an algorithm to facilitate this analysis, the number of passes is not a major factor to be considered

From the above study first of all, the FP-growth algorithm needs at most two scans of the database, while the number of database scans for Apriori increases with the dimension of the candidate itemsets. Also, the performance of the FP-growth is not influenced by the support factor, while the performance of the Apriori algorithm decreases with the support factor.

Thus, the candidate generating algorithms (derived from Apriori) behave well only for small databases (max.50, 000 transactions) with a large support factor (at least 30%). In other cases the algorithm without candidate generation DynFP-growth and FP-growth much better. DIC algorithm considerably faster than Apriori which must make as many passes as the maximum size of the candidate itemsets but it is sensitive to the homogenous data and depends on the data location.

From the above study we have seen that there is no such single association rule mining algorithm that can able to handle all the issues like requirement of domain knowledge, large data sets, large dimensionality, different types of data efficiently and filtering of duplicate data removal. When the size of the data set is small to medium and its dimensionality is low, then there are a sufficient number of algorithms that achieve good and fast association rules.

Nevertheless when the size of data set is vary large, it has many dimensions and a high level of duplication of data itemsets then no good and fast algorithm exists. Lately there are some algorithms that try to issue these problems with promising ideas and results, but still a general solution is far away. A solution is further hardened by the fact that most association rule mining algorithms are sensitive to their input parameters. For different data sets different input parameter settings will give a satisfactory result. Finding the correct one is not at all an easy task.

References

[1] R.Agrawal, R.Srikant, "Fast algorithms for mining association rules in large databases". Proc. of 20th Int'l conf. on VLDB: 487-499, 1994.
[2] J.Han, J.Pei, Y.Yin, "Mining Frequent Patterns without candidate generation". Proc. Of ACM-SIGMOD, 2000.
[3] C.Gyorodi, R.Gyorodi. "Mining Association rules in Large Databases". Proc. of Oradea EMES'02: 45-50, Oradea, Romania, 2002.
[4] S.Brin, R.Motawani, J.D.Ullman and S. Tsur, "Dynamic Itemset counting and implication rules for market basket data" in Proc. of the ACM SIGMOD Int'l Conf. on Management of data, Tucson, Arizona, USA, 1997, pp. 255-264.

World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:2, No:6, 2008

[5] Aggarwal, Charu, Yu, Philip: Bulletin of the IEEE Technical Committee on Data Engineering, Vol 21, No.1, Page 23-31, March 1998.
[6] M.H. Dunham. "Data Mining – Introductory and Advanced Topics". Prentice Hall, 2003, ISBN 0-13-088892-3.
[7] Morishita, Shinichi, Hishiki, eruyoshi and Okubo, Kousaku: Proc. 1999 ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD), pages 21-25, June 1999.
[8] J. Han M.Kamber, "Data Mining Concepts and Techniques". Morgan Kaufmann Publishers, San Francisco, USA, 2001, ISBN 1558604898.
[9] Nayaka, Akihiro, Hishigaki, Harutsugu and Morishita, Shinichi: In Proc. of Pacific Symposium on Biocomputing, pages 367-379, January 4-9, 2000.
[10] Park, J-S., Chen, M-S., and Yu P.S: Proc. ACM SIGMOD, May 1995, pp.175-186.
[11] C.Gyorodi, R.Gyorodi, T.Cofeey & S.Holban – "Mining association rules using Dynamic FP-Trees" – in Proc. of The Irish signal and Systems Conference, University of Limerick, Limerick, Ireland, 30th June- 2nd July 2003, ISBN 0-9542973-1-8, page 76-82.
[12] R.Agrawal, T.Imielinki and A.Swami, "Mining association rules between set of item of large databases" in Proc. Of the ACM SIGMOD Int'l Conf. on Management of data, Washington, D.C.,USA, 1993, pp 207-216.

TABLE I
COMPARISON TABLE

	Apriori	**FP growth**	**Dyn-FP growth**	**Partition**	**DIC**
Category	Level wise iterative method	Two-phase mining (Divide and conquer) method	Two-phase mining (Divide and conquer) method	Level wise iterative method	Level wise iterative method
User-defined Parameter	L and support factor (σ)	Set of transactions D and σ and FP-tree	Set of transactions D and σ	L , σ and Partitions of database transactions	Set of itemsets and σ
Candidate generation	Yes	No	No	Yes	Yes
Completeness of patterns to be mined	Complete set of Frequent patterns	Complete set of Frequent patterns	Complete set of Frequent patterns	Complete set of Frequent patterns	Complete set of Frequent patterns
Kinds of patterns to be mined	Frequent patterns	Frequent patterns	FP-tree	Frequent patterns	Frequent patterns
Scans	Number of items (m) +1	2	2	2	2
Data structure used	Hash tree	Frequent Pattern Tree	Dynamic frequent Pattern Tree	Hash table	Hash tree
Kinds of rules to be mined	Association rules	Association rules	Association rules	Association rules	Association rules
Types of values handled in the rule	Numerical	Numerical	Numerical	Numerical	Numerical
Limitation	Large number of DB scans	Needs a large amount of memory.	Execution time depends on DB size and not on support factor	If data are not uniform, large % of false itemsets	Sensitive to homogenous data and dependence on the data location