Improving Performance of Bicluster Discovery in a Large Data Set

Jiong Yang [1], Wei Wang [2], Haixun Wang [3], Philip Yu [4]

Keywords: gene expression analysis, microarray, bicluster.

Microarrays are one of the latest breakthroughs in experimental molecular biology, which provide a powerful tool by which the expression patterns of thousands of genes can be monitored simultaneously and are already producing huge amount of valuable data. Analysis of such data is becoming one of the major bottlenecks in the utilization of the technology. The gene expression data are organized as matrices — tables where rows represent genes, columns represent various samples such as tissues or experimental conditions, and numbers in each cell characterize the expression level of the particular gene in the particular sample. Investigations show that more often than not, several genes contribute to a disease, which motivates researchers to identify a subset of genes whose expression levels rise and fall coherently under a subset of conditions, that is, they exhibit fluctuation of a similar shape when conditions change. Discovery of such clusters of genes is essential in revealing the significant connections in gene regulatory networks.

Although many clustering algorithms have been proposed to efficiently cluster data in multi-dimensional space, they are based on distance measure instead of coherence. In [1], a so-called *bicluster* model is proposed for coherence based clustering of the gene expressions in the microarray data. Let X be a set of genes while Y the set of conditions. Let a_{ij} be the element of logarithm of the relative abundance of the mRNA of ith gene under jth condition. Let $I \subset X$ and $J \subset Y$. The submatrix is a bicluster if the mean squared residue score, $H(I, J)$ is less than a small threshold δ.

$$H(I,J) = \frac{1}{|I||J|} \sum_{i \in I, j \in J} (a_{ij} - a_{iJ} - a_{Ij} + a_{IJ})^2$$

where

$$a_{iJ} = \frac{1}{|J|} \sum_{j \in J} a_{ij}, a_{Ij} = \frac{1}{|I|} \sum_{i \in I} a_{ij}, a_{IJ} = \frac{1}{|I||J|} \sum_{i \in I, j \in J} a_{ij}$$

The algorithms presented in [1] identify biclusters in a data matrix in a greedy manner. After identifying the first cluster, each successive bicluster is mined on an altered matrix by replacing entries in the discovered biclusters with random data (noise). It is possible that the second, third clusters discovered by the above approach are corrupted by the random injected noise and leads to inaccurate results.

In this paper, we propose a move based algorithm (called $FLOC^5$) which can discover multiple bicluster simultaneously. The FLOC algorithm starts from a set of seeds (initial clusters) and carries out an iterative process to improve the overall quality of the clustering. At each iteration, each row and column is moved among clusters to produce a better clustering in terms of a lower average residue. The best clustering obtained during each iteration will serve as the initial clustering for

[1]IBM T. J. Watson Research Center, 19 Skyline Drive, Hawthorne, NY 10532. E-mail: `jiyang@us.ibm.com`
[2]IBM T. J. Watson Research Center, 19 Skyline Drive, Hawthorne, NY 10532. E-mail: `ww1@us.ibm.com`
[3]IBM T. J. Watson Research Center, 19 Skyline Drive, Hawthorne, NY 10532. E-mail: `haixun@us.ibm.com`
[4]IBM T. J. Watson Research Center, 19 Skyline Drive, Hawthorne, NY 10532. E-mail: `psyu@us.ibm.com`
[5]FLOC stands for FLexible Overlapped Clustering.

the next iteration. The algorithm terminates when the current iteration fails to improve the overall clustering quality. The detail of the FLOC algorithm can be found in [2].

The FLOC algorithm has two phases (Figure 1). In the first phase, k initial clusters are constructed. As we presented in the previous section, a bicluster contains a set of objects (rows) and a set of attributes (columns). For each initial cluster, a random switch is employed to determine whether a row or column should be included. Each row and column is included in the cluster with some probability ρ. Consequently, each initial cluster is expected to contain $M \times \rho$ rows and $N \times \rho$ columns.

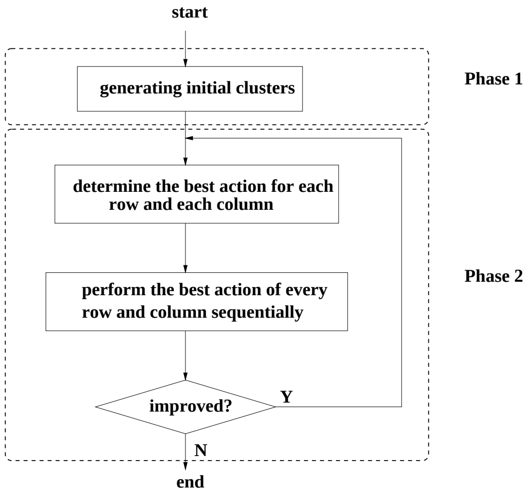

Figure 1: The Flowchart of the FLOC Algorithm

The second phase is an iterative process to improve the quality of the clusters continuously. During each iteration in the second phase, each row and each column is examined to determine its best action towards reducing the average residue. These actions are then performed successively to improve the clustering. An **action** is defined with respect to a row (or column) and a cluster. There are k potential actions associated with each row (or column), one for each cluster. For a given row (or column) x and a cluster c, the **action** $Action(x, c)$ is defined as the change of membership of x with respect to c. Note that this action is uniquely defined at any stage. If x is already included in c, then $Action(x, c)$ represents the removal x from the cluster c. Otherwise, $Action(x, c)$ denotes the addition of x to the cluster c. Among k possible actions, the best one is determined according to its gain (the reduction of residue). Then, the best action for each column and row is performed successively. The best clustering from one iteration serves as the initial clustering for the next round. This phase terminates when no improvement can be made during an iteration.

We experimented the FLOC algorithm on many several data set, including the yeast data set used in [1]. We found that our FLOC algorithm is able to discover multiple biclusters with lower residue at a much less time.

References

[1] Y. Cheng and G Church. Biclustering of expression data. *Proc. of 8th Intern. Conf. on Intelligent System for Molecular Biology*, 2000.

[2] J. Yang, W. Wang, H. Wang, and P. Yu. δ-cluster: capturing subspace correlation in a large data set. *Proc. of 18th IEEE Intern. Conf. on Data Engineering*, 2002.