An Efficient Optimal Leaf Ordering for Hierarchical Clustering in Microarray Gene Expression Data Analysis

Jianting Zhang

Le Gruenwald

School of Computer Science
The University of Oklahoma

Norman, Oklahoma, 73019, USA
{jianting, ggruenwald}@ou.edu
Outline

- Introduction
- Related Work
- The Proposed Ordering Method
  - The Objective Function
  - The Ordering Algorithm
  - An Example
- Experiment
- Conclusions and Future Work Directions
Introduction

- DNA microarray technologies in recent years produce tremendous amount of gene expression data.
- Clustering is one of the most popular methods to extract patterns hidden in the data and to understand functions of genomes.
- Hierarchical clustering provides users an initial impression of the distribution of data and stimulate thorough inspection of the whole data set.
Hierarchical clustering trees are usually displayed with their leaves in linear order.

Since genes that are adjacent in a linear ordering are often hypothesized to be related in some manner, the ordering of the tree leaves is important for human perception.

The clusters in a hierarchical clustering tree are unordered and there are $2^{n-1}$ possible orderings for a balanced binary clustering tree with $n$ genes.
Thus, additional criteria are needed to generate an ordering and algorithms are needed to generate an optimal ordering based on the criteria.

An example: Assume Gene 1 is more similar to Gene 4:
Introduction

Motivation:

Provide Efficient and Optimal Leaf Ordering method for Hierarchical Clustering In Microarray Gene Expression Data Analysis
Related Work

Heuristics Local Ordering Methods:

- (Alizadeh, et al, 2000): Proposed simple methods of weighting genes, such as average expression level, time of maximal induction, or chromosomal position. It places the element with the lower average weight earlier in the final ordering.

- (Alon, et al, 1999): Each pair of sibling branches is ordered according to the proximity of their centroids to the centroid of their parent’s sibling.
Related Work

Ordering Based Global Optimizations

– (Bar-Joseph, et al, 2001): For an ordering $\pi$ of tree $T$, the objective function is defined as

$$D^\pi(T) = \sum_{i=1}^{n-1} S(\pi_i, \pi_{i+1})$$

S is the gene similarity matrix

The proposed ordering algorithm to maximize has $O(n^4)$ time complexity and $O(n^2)$ space complexity
Related Work

Ordering Based Global Optimizations

– (Ding, 2002): New objective function:

\[ D^\pi (T) = \sum_{l=1}^{n-1} (l^2 \sum_{i=1}^{n-l} S(\pi_i, \pi_{i+l})) \]

They argue that the objective function used in (Bar-Joseph, et al, 2001) ignores the similarity between large distance genes.

Provide an approximate algorithm with \(O(n^2)\) complexity to minimize their objective function.
Related Work

\[ 1^2 \cdot [S(U, V) + S(V, W) + S(W, X) + S(X, Y)] \]

\[ 2^2 \cdot [S(U, W) + S(V, X) + S(W, Y)] \]

\[ 3^2 \cdot [S(U, X) + S(V, Y)] \]

\[ 4^2 \cdot [S(U, Y)] \]
The Proposed Ordering Method

- Our Objective Function
  \[ D^\pi (T) = \sum_{l=1}^{n-1} \left( l \sum_{i=1}^{n-l} S(\pi_i, \pi_{i+l}) \right) \]

- Linear, instead of quadratic as defined in (Ding, 2002)

- Can be rewritten as:
  \[ D^\pi (T) = \sum_{1 \leq i < j \leq n} S(i, j) \cdot |\pi(i) - \pi(j)| \]

- Both of them summarize the weighted distances (weighted by similarity) of all possible edges in a similarity graph under ordering \( \pi \).

- \( \pi_i \) denotes the node (gene) at position i while \( \pi(i) \) denotes the position of the node (gene) i.
The Proposed Ordering Method

\[ D^\pi (T) = \sum_{i,j} S(i, j)^* |\pi(i) - \pi(j)| \]

- Graph Linear Arrangement Problem!
- Ordering leaf nodes of a tree, not nodes of a regular graph
- (Bar-Yehuda, 2001): proposed to approximately solve the regular graph MinLA problem by imposing a global constraint through a Binary Decomposition Tree (BDT)
The Proposed Ordering Method

- We propose to use a binary clustering tree as the BDT and use the algorithm to solve the hierarchical binary clustering tree leaf ordering problem.

- Since (Bar-Yehuda, 2001) can examine the $2^{n-1}$ orderings in $O(n^2)$ time, it will produce the optimal ordering for leaf ordering in $O(n^2)$ time complexity.

- Although the algorithm is approximate for node ordering in a regular graph, it is exact for our leaf ordering in a hierarchical clustering tree for genes.
The Proposed Ordering Method

Number of Possible Orderings $2^{n-1}$ if a BDT is full and balanced

Orientation tree (or_tree): The orientations at each intermediate node of the BDT form a tree that has the same structure as the BDT.
The Proposed Ordering Method

- Recursively test the two possible orientations of a BDT and choose the better one.

- Use position implicitly in computing the cost which is very efficient.

\[
\begin{align*}
\text{cost}_0 &= \text{cost}(\text{left}(0)) + \text{cost}(\text{right}(0)) \\
&\quad + |V(t_2)| \cdot \text{right_cut}(\text{or_tree}(t1)) \\
&\quad + |V(t_1)| \cdot \text{left_cut}(\text{or_tree}(t2)) \\
\text{cost}_1 &= \text{cost}(\text{left}(1)) + \text{cost}(\text{right}(1)) \\
&\quad + |V(t_1)| \cdot \text{right_cut}(\text{or_tree}(t2)) \\
&\quad + |V(t_2)| \cdot \text{left_cut}(\text{or_tree}(t1))
\end{align*}
\]

(1)
The Proposed Ordering Method

\[
\begin{align*}
\text{left_cut}(\text{or_tree}(t)) &= \text{left_cut}(\text{left}(\text{or_tree}(t))) \\
&\quad + \text{left_cut}(\text{right}(\text{or_tree}(t))) - \text{in_cut}(t) \\
\text{right_cut}(\text{or_tree}(t)) &= \text{right_cut}(\text{left}(\text{or_tree}(t))) \\
&\quad + \text{right_cut}(\text{right}(\text{or_tree}(t))) - \text{in_cut}(t)
\end{align*}
\]

(2)

**In_cut**: summation of the weights of edges the beginning and ending nodes of which are within sub-tree t

**Left_cut**: summation of the weights of edges the beginning node of which is not within sub-tree t while the ending node of which is within sub-tree t

**Right_cut**: summation of the weights of edges the beginning node of which is within sub-tree t while the ending node of which is not within sub-tree t
The Proposed Ordering Method

Algorithm of (Bar-Yehuda, 2001)
1. Pre-compute in_cuts for all the non-leaf nodes in the BDT
2. Set tree \( t \) to the root of the BDT. Do the following **recursively**
3. If \( t \) is an intermediate node of the BDT:
   a) Compute the costs and the left_cuts and the right_cuts under both the 0-orientation and the 1-orientation of its two sub-trees \( t_1 \) and \( t_2 \) according to formula (1) and formula (2) respectively
   b) Set the orientation of \( t \) to the one that has less cost as the winner orientation
4. If \( t \) is a leaf node of the BDT, the left_cut is the summation of the weights of the edges with the node as the ending node. The right_cut is the summation of the weights of the edges with the node as the beginning node. The cost of the node is the same as its left_cut.
The Proposed Ordering Method

An Example

Similarity Graph

Clustering Tree (BDT)

In_cuts rooted at $T_0$
In_cuts rooted at $T_{11}$
In_cuts rooted at $T_{12}$
The Proposed Ordering Method

0: left_cut=0.5, right_cut=0.60
1: left_cut=0.30+0.85+0.60=1.75, right_cut=0

$T_{11}$: left_cut=0.50+1.75-0.60=1.65
Right_cut=0.60+0-0.60=0
Cost=0.50+1.75+1*(1.75-0.60)+1*(0.60-0.60)=3.94
The Proposed Ordering Method

Cost = 2.75
✓ Winner

T_{12}: 0-orientation: 1.6 (Winner) 1-Orientation: 1.65
Left_cut = 0, right_cut = 1.65

T_0: 1-orientation
2.75 + 1.6 + 2*(1.65 - 1.65) + 2*(1.65 - 1.65) = 4.35
([2,3],[1,0])
The total cost for \( T_0 \) under 0-orientation happens to be 4.35. By convention, we choose 0-orientation if the costs of both orientations are the same. The optimized ordering is \(([0,1], [3,2])\)

For the optimized ordering \(([0,1], [3,2])\), cost=4.35, Best among the 4!=24 possible orderings

The initial ordering is \(([1,0], [2,3])\), cost=5.05
Experiment

Data Set


- The 800 genes in it have been assigned to five classes termed G1, S, S/G2, G2/M and M/G1 based on domain knowledge in (Spellman, etc, 1998) and they can be used to visually examine the quality of an ordering

- Also used in (Bar-Joseph, et al, 2001) to demonstrate the effectiveness of its leaf ordering technique
Experiment

Preprocessing:

• Remove genes with exceptional missing values and to compute the gene similarity matrix. The final number of genes in our study is 765

• Only consider the gene pairs whose similarities are above a certain threshold (avg+1.5*std_dev): results in 38714 pairs of gene similarity and the average out-degree of the similarity graph is about 50

• Use a graph-partition package called Metis to generate a clustering tree by recursive graph bisection and use it as our BDT
Experiment

• Computation Environment
  • Dell Dimension 4100 866HZ personal computer with 512M memory running Windows 2000 professional
  • It takes about 27 seconds to perform ordering optimization

• Performance Metric:
  • Normalized index
    \[ I = \frac{D^\pi(T)}{\sum_{i,j} S(i,j)} \]
Experiment

Results:

• Original Ordering (81.11), Optimized Ordering (69.69), The average of 1000 times random orderings (255.34)

• The optimized ordering is about 16.2% less than the original ordering. The average of the 1000 times random orderings is 3.14 times of the original ordering and 3.66 times of the optimized ordering

• Both the graph bisection based clustering method and the proposed leaf ordering method are effective
Experiments

visually examination of 22 genes

The optimized ordering is more compact. For example, G1 scatters in 6 segments in the original ordering while it is in 3 segments in the optimized ordering.
Conclusions

- Leaf ordering of hierarchical clusters is significant to presenting gene expression data. Previously proposed methods are either non-efficient or approximate in nature.

- The proposed method is efficient in the sense that the time complexity is $O(n^2)$ when the clustering tree is balanced.

- The proposed method optimal in the sense that it examines all possible $2^{n-1}$ orderings without relying on approximation.

- Preliminary Experiment show good results.
Future Work

• Apply this graph theoretical leaf ordering optimization method to more gene expression data using clustering trees from a variety of hierarchical clustering methods as the BDTs for comparison purposes.

• Develop novel approaches to visually presenting the original and optimized cluster ordering to domain experts and examine the validity of our optimization objective function.
Thanks!

Questions?