Mining Coherent Gene Clusters from Gene-Sample-Time Microarray Data*

Daxin Jiang† Jian Pei†‡ Murali Ramanathan† Chun Tang† Aidong Zhang†

† State University of New York at Buffalo, USA ‡ Simon Fraser University, Canada
Email: {djiang3, jianpei}@cse.buffalo.edu, murali@acsu.buffalo.edu, {chuntang, azhang}@cse.buffalo.edu

ABSTRACT

Extensive studies have shown that mining microarray data sets is important in bioinformatics research and biomedical applications. In this paper, we explore a novel type of gene-sample-time microarray data sets, which records the expression levels of various genes under a set of samples during a series of time points. In particular, we propose the mining of *coherent gene clusters* from such data sets. Each cluster contains a subset of genes and a subset of samples such that the genes are coherent on the samples along the time series. The coherent gene clusters may identify the samples corresponding to some phenotypes (e.g., diseases), and suggest the candidate genes correlated to the phenotypes. We present two efficient algorithms, namely the *Sample-Gene Search* and the *Gene-Sample Search*, to mine the complete set of coherent gene clusters. We empirically evaluate the performance of our approaches on both a real microarray data set and synthetic data sets. The test results have shown that our approaches are both efficient and effective to find meaningful coherent gene clusters.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*Data Mining*

General Terms

Algorithms, Experimentation

Keywords

Bioinformatics, clustering, microarray data

*This research is partly supported by NSF grants DBI-0234895, IIS-0308001 and NIH grant 1 P20 GM067650-01A1. All opinions, findings, conclusions and recommendations in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

KDD'04, August 22–25, 2004, Seattle, Washington, USA.

1. INTRODUCTION

The recent microarray technology can measure the expression levels of thousands of genes simultaneously. It is an important research problem in bioinformatics and clinical research to explore the patterns in microarray data sets. For example, in drug development, gene expression patterns may reflect gene-level responses to different drug treatments and provide deep insights into the nature of the diseases.

Most microarray data sets can be divided into two categories. On the one hand, the *gene-time data sets* record the expression levels of various genes during important biological processes over a series of time points. The *gene-sample data sets* account the expression levels of various genes across related samples. Both gene-time data sets and gene-sample data sets can be represented by a $n \times l$ *gene expression matrix* of n genes and l samples/time points. In the gene expression matrix, the rows are the genes and the columns are either samples (in gene-sample data sets) or ordered time points (in gene-time data sets), while each cell represents the expression level of a certain gene on a certain sample or at a certain time point.

With the latest advances in the microarray technology, the expression levels of a set of genes under a set of samples can be monitored synchronically during a series of time points [20]. Different from the previous gene-time or gene-sample microarray data sets, these new data sets have three types of variables: *genes*, *samples* and *time*. We call such data *gene-sample-time microarray data*, or *GST data* for short. Figure 1(a) elaborates the structure of a *GST* microarray data set.

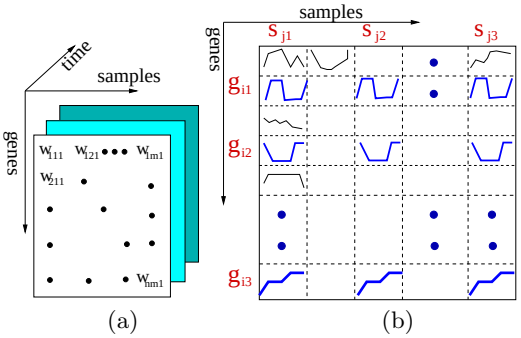


Figure 1: The structure of *GST* microarray data

In general, each cell $m^k_{i,j}$ in a *GST* data set represents the expression level of gene g_i under sample s_j at time point t_k. Interestingly, a *GST* data set can also be viewed as a $n \times l$

matrix, such that each cell $m_{i,j}$ contains the time series with respect to gene g_i under sample s_j, as shown in Figure 1(b).

The previous studies on gene-sample microarray data (e.g., [7, 2, 15, 1]) indicate that high correlations may exist between the gene expression patterns and some diseases. It is natural to extend the similar analysis to *GST* microarray data. That is, it is interesting to identify a subset of genes G and a subset of samples S in a *GST* microarray data set such that each gene $g \in G$ has coherent patterns across the samples in S during the time series. For example, in Figure 1, gene g_{i1}, g_{i2} and g_{i3} show coherent patterns across samples s_{j1}, s_{j2} and s_{j3}, respectively. We call such subsets of genes and samples a *coherent gene cluster*.

What is the biological meaning of the coherent gene clusters? The coherent gene clusters provide valuable hypothesis for biologists. The samples in a cluster may correspond to a phenotype, such as the patients having a disease, while the corresponding set of genes may suggest the candidate genes correlated to the disease.

The functions of genes in an organism are highly complicated. There are typically multiple coherent clusters in a data set. Different clusters may correlate to different phenotypes, such as age and gender. Therefore, to avoid missing any valuable hypothesis, it is necessary to mine *all* the coherent clusters in the data set.

Many previous studies investigate the mining of interesting patterns from microarray matrices. For example, various clustering algorithms can identify the co-expressed genes showing coherent patterns during the time-series (e.g., [17, 12, 15, 8]). Moreover, both supervised or unsupervised approaches are proposed to partition the samples into homogeneous groups (e.g., [4, 14, 9, 16]). Additionally, statistical approaches have been proposed to validate the significance of the mining results (e.g., [3, 18, 5]). However, all previous studies target at conventional gene-time or gene-sample microarray data sets. The models of clusters in those previous studies are different from our coherent gene clusters which disclose the correlation among genes, samples and time points. Therefore, those algorithms cannot be extended directly to solve our problem.

Recently, the *pattern-based* clustering approaches (e.g., [19]) have been developed to discover subsets of objects following similar patterns on subsets of attributes. Conceptually, a pattern-based cluster is a coherent gene cluster. If we treat the *GST* microarray data sets as a $n \times l$ matrix of time series, as shown in Figure 1(b), then pattern-based clusters and coherent gene clusters may have some similarity at the first look. However, in some pattern-based approaches, a cluster requires that each pair of objects in the cluster must be coherent on each pair of attributes. Such a requirement is often too strong in practice. Our coherent gene clustering relaxes the constraints among the objects. Therefore, each traditional pattern-based cluster is a coherent gene cluster, but not necessarily true vice versa.

In this paper, we tackle the problem of *mining coherent patterns from gene-sample-time microarray data sets* and make the following contributions.

First, we propose a model of coherent gene clusters in *GST* microarray data sets. We justify that the model is meaningful for biomedical research.

Second, we identify the computational challenges and conduct a systematic research on mining coherent gene clusters from *GST* microarray data sets. We develop two approaches, namely the *Gene-Sample Search* and the *Sample-Gene Search*, to mine the complete set of coherent gene clusters. We illustrate and compare the efficiency and scalability of both approaches.

Last, we conduct an extensive empirical evaluation on both real data sets and synthetic data sets. Our results show that our proposed methods can find coherent gene clusters interesting to biomedical research from real data sets. The results on synthetic data sets also show that our algorithms are both efficient and scalable.

The rest of the paper is organized as follows. Section 2 defines the problem. Section 3 describes the preprocessing step of computing the maximal coherent samples sets for each individual gene. Section 4 presents two algorithms to mine coherent gene clusters. In Section 5, our methods are evaluated using real and synthetic data sets. Related work is discussed in Section 6. Section 7 concludes the paper.

2. PROBLEM DESCRIPTION

Given a set of n genes *G-Set* $= \{g_1, \ldots, g_n\}$ and a set of l samples *S-Set* $= \{s_1, \ldots, s_l\}$, we can measure the expression levels of the genes on the samples. The results form a conventional $n \times l$ microarray matrix $M = \{m_{i,j}\}$, where $m_{i,j}$ is the expression level of gene g_i $(1 \leq i \leq n)$ on sample s_j $(1 \leq j \leq l)$. If such microarray experiments are conducted synchronically on all genes and all samples at time instants $t_1, \ldots, t_T$, the results form a $n \times l \times T$ *GST microarray matrix* $M = \{m^t_{i,j}\}$, where $(1 \leq t \leq T)$.

A *GST* microarray matrix $M = \{m^t_{i,j}\}$ can also be viewed as a $n \times l$ matrix $M' = \{m'_{i,j}\}$ such that $m'_{i,j}$ is a vector of $\langle m^1_{i,j}, \ldots, m^T_{i,j} \rangle$. Hereafter, we do not strictly distinguish the two notations. Instead, whenever $m_{i,j}$ is written, the vector is referred to.

In this paper, we are interested in finding those genes that are coherent on a subset of samples during the whole time series. There are various methods to measure the correlation between two time series. However, for gene expression data, users are often interested in the overall trends of the expression levels instead of the absolute magnitudes. Therefore, we choose the *Pearson's correlation coefficient* as the coherence measure, since it is robust to shifting and scaling patterns [22]. Specifically, given two vectors m_{i,j_1} and m_{i,j_2} of gene g_i, the coherence $\rho(m_{i,j_1}, m_{i,j_2})$ is defined as

$$\frac{\sum_{t=1}^{T}(m^t_{i,j_1} - \overline{m_{i,j_1}})(m^t_{i,j_2} - \overline{m_{i,j_2}})}{\sqrt{\sum_{t=1}^{T}(m^t_{i,j_1} - \overline{m_{i,j_1}})^2}\sqrt{\sum_{t=1}^{T}(m^t_{i,j_2} - \overline{m_{i,j_2}})^2}},$$

where $\overline{m_{i,j}} = \sum_{t=1}^{T} \frac{m^t_{i,j}}{T}$ is the mean of the expression levels of gene g_i on sample s_j. The correlation coefficient ranges between -1 and 1. The larger the value, the more coherent are the two vectors.

A gene g_i is *coherent* across a subset of samples $S \subseteq$ *S-Set*, if given any pair of samples $s_{j_1}, s_{j_2} \in S, \rho(m_{i,j_1}, m_{i,j_2}) \geq \delta$, where δ is a *minimum coherence threshold* specified by the user. For a subset of genes $G \subseteq$ *G-Set* and a subset of samples $S \subseteq$ *S-Set*, if every gene $g_i \in G$ is coherent across samples in S, we call *gene set* G coherent on *sample set* S. $(G \times S)$ is called a *coherent gene cluster*. A coherent gene cluster having u genes and v samples is said a (u, v)-coherent gene cluster.

Trivially, for any gene g_i and any sample s_j, $(\{g_i\} \times \{s_j\})$ is a $(1, 1)$-coherent gene cluster, and $(G\text{-}Set \times \{s_j\})$ and $(\{g_i\} \times$

S-Set) are trivial ($|G\text{-}Set|, 1$)- and ($1, |S\text{-}Set|$)-coherent gene clusters, respectively. To avoid this triviality, we require that a coherent gene cluster should consist of at least two genes and two samples.

Given a coherent gene cluster ($G \times S$), for any subsets $G' \subseteq G$ and $S' \subseteq S$, ($G' \times S'$) is also a coherent gene cluster. To avoid such redundancy, a coherent gene cluster ($G \times S$) is *maximal* if there exists no any other coherent gene cluster ($G' \times S'$) such that $G \subseteq G'$, $S \subseteq S'$. Moreover, a user may not be interested in very small clusters, which are often formed by chance. Thus, a user can specify the minimum numbers of genes and samples in a cluster. Generally, given min_g and min_s as user defined minimum gene size and sample size thresholds, a cluster ($G \times S$) is called *significant* if $|G| \geq min_g$ and $|S| \geq min_s$.

Problem definition Given a *GST* microarray matrix M, a minimum coherence threshold δ, a minimum gene size threshold min_g and a minimum sample size threshold min_s, the problem of *mining coherent gene clusters* is to find the complete set of significant maximal coherent gene clusters in M with respect to the parameters. Hereafter, a significant maximal coherent gene cluster is called a *coherent gene cluster* for short. ∎

3. MAXIMAL COHERENT SAMPLE SETS

We propose two algorithms computing maximal coherent gene clusters. In both algorithms, to compute coherent gene clusters, we need to check whether a subset of genes are coherent on a subset of samples. To facilitate the tests, for each gene g_k, we compute the sets of samples S such that (1) $|S| \geq min_s$; (2) g_k is coherent on S; and (3) there exists no superset $S' \supset S$ such that g_k is also coherent on S'. S is called a *maximal coherent sample set* of g_k. Please note that, in general, a gene may have more than one maximal coherent sample set.

For a gene g_k, all of its maximal coherent sample sets can be computed efficiently using the following 2-step process.

In the first step, we test whether gene g_k is coherent on each pair of samples (s_i, s_j). A binary triangle matrix $\{c_{i,j}\}$ is populated, where $1 \leq i < j \leq |S\text{-}Set|$. $c_{i,j} = 1$ if gene g_k is coherent on samples s_i and s_j, i.e., $\rho(m_{k,i}, m_{k,j}) \geq \delta$, otherwise, $c_{i,j} = 0$.

Once the matrix $\{c_{i,j}\}$ is populated, the problem of finding g_k's maximal coherent sample sets can be reduced to the problem of finding all maximal cliques of size at least min_s in graph $\mathcal{G}_k = (S\text{-}Set, E)$, where (s_i, s_j) is an edge in the graph if and only if $c_{i,j} = 1$. A clique S is called *maximal* if there exists no any other clique S' such that $S \subset S'$. Please note that there may exist more than one maximal clique in a graph.

Unlike the conventional clique problem where the clique of the maximal size is found, here, we need to find the complete set of maximal cliques in the graph. It is well known that the conventional clique problem is NP-complete. So is the problem of finding the complete set of maximal cliques.

Fortunately, the real *GST* microarray data sets are often sparse and the number of samples is typically below one hundred. For each gene, the number of maximal cliques is quite small and the samples can often be partitioned into exclusive small subsets. Our experimental results show that, with efficient search and pruning techniques, it is still practical to find the complete set of maximal cliques. In the following, we will show how to find the maximal cliques of a sample set by a depth-first search in a sample set enumeration tree.

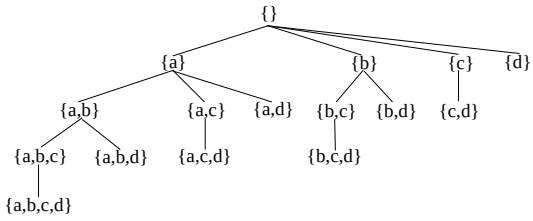


Figure 2: Enumeration of combinations of samples.

Given a set of samples $S = \{s_1, \ldots, s_l\}$, the set 2^S (i.e., all combinations of samples) can be enumerated systematically. For example, consider a set of samples $S = \{a, b, c, d\}$. The complete set of non-empty combinations of samples can be divided into 4 exclusive subsets: (1) the ones having sample a; (2) the ones having sample b but no a; (3) the ones having sample c but no a or b; and (4) $\{d\}$. They are shown as the immediate children of the root in Figure 2.

These subsets can be further partitioned. For example, the first subset can be further divided into three exclusive sub-subsets: (1) the ones having samples a and b; (2) the ones having samples a and c but no b; and (3) $\{a, d\}$.

The tree shown in Figure 2 is called a *set enumeration tree* [13] with respect to $\{a, b, c, d\}$. It provides a conceptual tool to enumerate the complete set of combinations systematically.

We can conduct a *recursive, depth-first search* of the sample set enumeration tree to detect the maximal cliques of the samples. Given a set of samples S, the set enumeration tree has $2^{|S|}$ nodes. However, we never need to materialize such a tree. Instead, we only need to keep a path from the root of the tree to the node we are searching as a working set, which contains at most $(|S| + 1)$ nodes.

Clearly, in the set enumeration tree, each node contains a unique subset of samples. Thus we can use the subset of samples to refer to the node. At node $\{s_{i_1}, \ldots, s_{i_k}\}$ ($1 \leq i_1 < \cdots < i_k \leq l$), we also keep a list $Tail$, which contains the samples that can be used to extend the node to larger subsets of samples in the subtree. We have the following result.

LEMMA 3.1. *At node $v = \{s_{i_1}, \ldots, s_{i_k}\}$ of the sample set enumeration tree, where ($1 \leq i_1 < \cdots < i_k \leq l$), a sample $s_j \notin Tail$ if (1) $j \leq i_k$; or (2) there exists some $1 \leq l \leq k$ such that $c_{i_l,j} = 0$. Moreover, for v's parent node $v' = \{s_{i_1}, \ldots, s_{i_{k-1}}\}$, v's Tail is a subset of that of v'.* ∎

We can prune any subtree that cannot lead to a coherent sample set of at least min_s samples.

PRUNING RULE 3.1 (PRUNING SMALL SAMPLE SETS). *At a node $v = \{s_{i_1}, \ldots, s_{i_k}\}$, the subtree of v can be pruned if $(k + |Tail|) < min_s$.* ∎

For example, for a set of l samples, even the complete set of sample combinations can be divided into l exclusive subsets as shown before, we only need to search the first $(l - min_s + 1)$ subsets, since each of the last $(min_s - 1)$ subsets contains less than min_s samples.

Moreover, if the samples at the current node and its $Tail$ are subsumed by some maximal coherent sample set found

Input: the GST data set, the coherence threshold δ;
minimum sample size threshold min_s;
Output: the maximal coherent sample sets for each gene;
Method:

```
for each gene g_k do
    generate matrix c_{i,j} for g_k;
    // depth-first search
    for i = 1 to (l − min_s + 1) do
        call search-clique({s_i}, {s_{i+1}, ..., s_l});
    end-for
end-for
```

Procedure: *search-clique*$(head, tail)$

```
// head records the samples in the current node
suppose s_i is the last sample in head, remove samples
    s_j from tail such that c_{i,j} = 0; // Lemma 3.1
if (|head ∪ tail| < min_s) // Pruning 3.1
    or (head ∪ tail ⊂ S) s.t. S is a maximal clique
        // Pruning 3.2
then return; //abort the recursive search
if tail = ∅ then output a maximal clique;
else
    do
        let j = min{k|s_k ∈ tail} and tail = tail − {s_j};
        call search-clique(head ∪ {s_j}, tail);
    until tail = ∅;
return;
```

Figure 3: Computing maximal coherent sample sets.

so far, then the recursive search can also be pruned, since it cannot lead to any new maximal coherent sample set.

PRUNING RULE 3.2 (PRUNING SUBSUMED SETS). *At a node $v = \{s_{i_1}, \ldots, s_{i_k}\}$, if $\{s_{i_1}, \ldots, s_{i_k}\} \cup Tail$ is a subset of some maximal coherent sample set, then the subtree of the node can be pruned.* ∎

Based on the above lemma and pruning rules, the preprocessing algorithm is presented in Figure 3. For the readers familiar with the techniques of depth-first mining of maximal/closed frequent patterns, the ideas of pruning here share the similar spirit with the pruning in frequent closed itemset mining (e.g., [10]). However, one key difference is that the frequent pattern mining conducts counting on databases, and we do not need to scan the database for any counting once the triangle matrix $\{c_{i,j}\}$ is materialized. The correctness of the preprocessing algorithm can be shown. Limited by space, we omit the details here.

4. THE MINING ALGORITHMS

A naïve method to find the maximal coherent gene clusters is to test every possible combination of genes and samples thoroughly. After all the coherent gene clusters are found, we can identify and report the maximal ones. The naïve method is very costly and thus infeasible for real data sets. For example, suppose we have $1,000$ genes and 20 samples. The naïve method may have to search up to $(2^{1000} - 1) \times (2^{20} - 1 - 20) \approx 1.12 \times 10^{307}$ combinations!

How can we search the huge space efficiently and prune unpromising subspaces sharply? When computing the maximal coherent sample sets (Figure 3), we systematically enumerate combinations of samples in a recursive depth-first search, and develop techniques to prune unpromising subspaces aggressively. Stimulated by the similar spirit, here we can also systematically enumerate the combinations of genes and samples, and prune unfruitful combinations.

The *Sample-Gene Search*

```
depth-first enumerate subsets of samples
for each subset of samples S do
    find the maximal subsets of genes G
        s.t. G × S is a coherent gene cluster;
    test whether (G × S) is a maximal coherent gene
        cluster;
end-for
```

The *Gene-Sample Search*

```
depth-first enumerate subsets of genes
for each subset of genes G do
    find the maximal subsets of samples S
        s.t. G × S is a coherent gene cluster;
    test whether (G × S) is a maximal coherent gene
        cluster;
end-for
```

Figure 4: The frameworks of the *Sample-Gene Search* and the *Gene-Sample Search*.

Basically, we have two alternatives. On the one hand, we can enumerate all combinations of samples systematically. For each subset of samples, we can find the maximal subsets of genes that form coherent gene clusters on the samples, and check whether the clusters are maximal. This method is called the *Sample-Gene Search*. On the other hand, we can let the gene enumeration go first. For each subset of genes, we find the maximal subsets of samples that form coherent gene clusters with the genes, and check whether the clusters are maximal. The method is called the *Gene-Sample Search*.

The frameworks of the *Sample-Gene Search* and the *Gene-Sample Search* are shown in Figure 4. Proper pruning techniques should be developed to prune unpromising combinations and search branches as early as possible.

A first look at Figure 4 may suggest that the two methods are symmetric. However, since genes and samples are not symmetric in the problem, the technical details are in fact substantially different. We are mining coherent gene clusters on samples. As long as the genes respond coherently on the same subset of samples, they belong to the same cluster. However, the expression patterns of different genes in the same cluster on one sample can be very different!

4.1 Sample-Gene Search

In the *Sample-Gene Search*, we need to address the following issues. First, as we enumerate the combinations of samples systematically, *for each subset of samples, how can we find the maximal sets of genes such that the genes are coherent on the samples?* Second, *during the sample set enumeration, which sample sets can be pruned?* Third, similar to the situation in Pruning rule 3.2, *can we identify and prune the searches that cannot lead to any potential maximal coherent clusters?*. Last, *how can we determine whether a coherent gene cluster is subsumed by the others?* We answer the above questions in this subsection.

4.1.1 Maximal Coherent Gene Sets for Sample Sets

For each combination of samples S, we need to compute the *maximal coherent gene set* G_S such that the genes in G_S

Gene	Maximal coherent sample sets
g_1	$\{s_1, s_2, s_3, s_4, s_5\}$
g_2	$\{s_1, s_2, s_4\}, \{s_1, s_5\}$
g_3	$\{s_1, s_2, s_3, s_4, s_5\}$
g_4	$\{s_1, s_2, s_3\}, \{s_5, s_6\}$
g_5	$\{s_1, s_5, s_6\}$

(a) The maximal coherent sample sets for genes

Sample	The inverted list
s_1	$\{g_1.b_1, g_2.b_1, g_2.b_2, g_3.b_1, g_4.b_1, g_5.b_1\}$
s_2	$\{g_1.b_1, g_2.b_1, g_3.b_1, g_4.b_1\}$
s_3	$\{g_1.b_1, g_3.b_1, g_4.b_1\}$
s_4	$\{g_1.b_1, g_2.b_1, g_3.b_1\}$
s_5	$\{g_1.b_1, g_2.b_2, g_3.b_1, g_4.b_2, g_5.b_1\}$
s_6	$\{g_4.b_2, g_5.b_1\}$

(b) The inverted lists for samples

Figure 5: The maximal coherent sample sets and the inverted lists.

are coherent on S and no proper superset $G' \supset G_S$ also has this property.

Clearly, for a gene g, if there exists a maximal coherent sample set S_g such that $S \subseteq S_g$, then $g \in G_S$. In other words, G_S can be derived by one scan of the maximal coherent sample sets of all genes. If a maximal coherent sample set is a superset of S, then the corresponding gene g is inserted into G_S.

It is expensive to scan the complete list of maximal coherent sample sets of all genes once for every combination of samples. An efficient solution is to use an *inverted list*.

Suppose we have 5 genes and 6 samples. The maximal coherent sample sets for each gene are listed in Figure 5(a).

We label each maximal coherent sample set by the gene g_k, and the set-id, b_j, in the gene. For example, gene g_2 has two maximal coherent sample sets, $g_2.b_1 = \{s_1, s_2, s_4\}$ and $g_2.b_2 = \{s_1, s_5\}$.

For each sample s, we make up the *inverted list* L_s as the list of all maximal coherent sample sets containing s, as shown in Figure 5(b).

Now, when we want to compute the maximal coherent gene sets for a subset of samples, say $\{s_1, s_2, s_3\}$, we do not need to search the complete list in Figure 5(a). Instead, we only need to get the intersection of the inverted lists of the samples s_1, s_2 and s_3, which is $\{g_1.b_1, g_3.b_1, g_4.b_1\}$. By this intersection, we know that $\{g_1, g_3, g_4\}$ is the maximal coherent gene set.

4.1.2 Pruning Irrelevant Samples

For a combination of samples $S = \{s_{i_1}, \ldots, s_{i_k}\}$, where $i_1 < \cdots < i_k$, let S_{tail} be the set of samples that can be used to extend S to a larger set $S' \subset S \cup S_{tail}$ such that there are at least min_g genes coherent on S'. Clearly, a sample $s_j \notin S_{tail}$ if $j \leq i_k$. Moreover, if the maximal coherent gene set of $S \cup \{s_j\}$ contains less than min_g genes, then $s_j \notin S_{tail}$, either. This is in the similar spirit of Lemma 3.1. For example, in our running example (Figure 5), sample s_6 cannot be used to extend sample set $S = \{s_2\}$, since there is no gene coherent on both s_2 and s_6.

Moreover, similar to Pruning rule 3.1, if $|S| + |S_{tail}| < min_s$, then S cannot lead to any coherent gene cluster having min_s or more samples, and thus can be pruned.

4.1.3 Pruning Unpromising Coherent Gene Clusters

Similar to the situation in Pruning rule 3.2, we can prune the unpromising combinations that cannot lead to any new maximal coherent gene cluster. Here, two pruning techniques can be applied.

For example, in our running example (Figure 5), suppose we find the maximal coherent gene cluster $(\{g_1, g_3\} \times \{s_1, s_2, s_3, s_5\})$ before we search sample set $S = \{s_1, s_3\}$. For $S = \{s_1, s_3\}$, $S_{tail} = \{s_5\}$ and $G_{S \cup S_{tail}} = \{g_1, g_3\}$. That is, both $S \cup S_{tail}$ and $G_{S \cup S_{tail}}$ are subsumed by the maximal coherent gene cluster. The recursive search of S cannot lead to any maximal coherent gene cluster and thus can be pruned.

In general, suppose we are searching a sample combination S'. If there exists a maximal coherent gene cluster $(G \times S)$ found before such that $S' \cup S'_{tail} \subseteq S$ and $G_{S' \cup S'_{tail}} \subseteq G$, then any recursive search from S' results in a coherent gene cluster subsumed by $(G \times S)$, and thus can be pruned.

Moreover, if there exists a maximal coherent gene cluster $(G \times S)$ found before such that $S' \subseteq S$ and every maximal coherent sample set containing S' also contains S, then the recursive search of S' cannot lead to any maximal coherent gene cluster either, and thus can be pruned. For example, suppose we search the sample set $S' = \{s_2\}$ after we find the maximal coherent gene cluster $(\{g_1, g_2, g_3, g_4\} \times \{s_1, s_2\})$ in our running example (Figure 5). From Figure 5(a), we can see that every maximal coherent sample set containing s_2 also contains s_1. In other words, there exists no maximal coherent gene cluster containing s_2 but no s_1. Thus, the search of S' can be pruned.

4.1.4 Determining maximal coherent gene clusters

When we search a combination of samples S, we need to check whether $G_S \times S$ is a maximal coherent gene cluster. We look at those maximal coherent gene clusters $(G' \times S')$ such that $S' \supset S$. Clearly, since we conduct depth-first search in the set enumeration tree, such maximal coherent gene clusters should be reported either before S is searched, or in the subtree rooted at S.

Based on the above discussion, we have the *Sample-Gene Search* algorithm in Figure 6.

4.2 Gene-Sample Search

The framework of *Gene-Sample Search* is symmetric to that of *Sample-Gene Search*. That is, we enumerate the combinations of genes systematically. For each combination of genes, we compute the maximal sets of samples that the genes are coherent on. Many pruning techniques in *Sample-Gene Search* have the symmetric versions in *Gene-Sample Search*. Limited by space, we omit the details here. Instead, we only focus on the differences between the two approaches.

4.2.1 Determining Coherent Gene Clusters

The concept of coherent sample sets for a gene can be generalized for a set of genes. Given a set of genes G, a *maximal coherent sample set* with respect to G is a set of samples S_G such that (1) genes in G are coherent on S_G; and (2) there exists no $S' \supset S_G$ that samples in G are also coherent on S'. Please note that there can be more than one maximal coherent sample set for a given set of genes.

How can we compute the maximal coherent sample sets efficiently? Interestingly, S_G can be computed by some simple intersection operations. For example, suppose $S_{\{g_1\}} =$

Input: the maximal coherent samples sets for genes,
// from the algorithm in Figure 3
Output: the complete set of coherent gene clusters
Method:
generate the inverted list for samples
as described in Section 4.1.1;
`for` $i = 1$ `to` $(|S\text{-}Set| - min_s)$ `do`
let $S = \{s_i\}$ and $S_{tail} = \{s_{i+1}, \ldots, s_{|S\text{-}Set|}\}$;
`call` *recursive-search*(S, S_{tail});
`end-for`

Procedure: *recursive-search*(S, S_{tail})
remove irrelevant samples from S_{tail} as described in Section 4.1.2;
`if` $(|S| + |S_{tail}| < min_s)$ `then` return;
derive the intersection of inverted lists for samples in S as described in Section 4.1.1;
`if` S can be pruned by the criteria in Section 4.1.3 `then` return;
`while` $S_{tail} \neq \emptyset$ `do`
let $i = \min\{j | s_j \in S_{tail}\}$;
let $tail = tail - \{s_i\}$;
`call` *recursive-search*$(S \cup \{s_i\}, S_{tail})$;
`end-while`
derive the maximal coherent gene set G_S;
output $(G_S \times S)$ as a maximal coherent gene cluster if it is not subsumed by any maximal coherent gene cluster found before
End

Figure 6: The *Sample-Gene Search* algorithm.

$\{(s_1, s_2, s_3)\}$ and $S_{\{g_2\}} = \{(s_2, s_3, s_4), (s_5, s_7)\}$. Then, $\{s_2, s_3\}$ is the only maximal sample set that both g_1 and g_2 are coherent on. That is, $S_{\{g_1,g_2\}} = \{(s_2, s_3)\}$. In other words, we can derive $S_{\{g_1,g_2\}}$ from $\{s_1, s_2, s_3\} \cap \{s_2, s_3, s_4\}$ and $\{s_1, s_2, s_3\} \cap \{s_5, s_7\}$.

In general, if gene set $G = G_1 \cup G_2$, then S_G can be derived from S_{G_1} and S_{G_2} by the function *find-max-coherent-sample-sets* in Figure 7.

4.2.2 Pruning Irrelevant Genes and Unpromising Coherent Gene Clusters

Similar to the idea in Section 4.1.2, we can prune genes that cannot be used to extend the current combination of genes. Given a set of genes $G = \{g_{i_1}, \ldots, g_{i_k}\}$, where $1 \leq i_i < \cdots < i_k$, a gene g_j cannot be used to extend G to a larger set of genes if $j \leq i_k$ or none of the maximal coherent sample set with respect to $G \cup \{g_j\}$ has at least min_s samples. Moreover, let G_{tail} be the set of genes that can be used to extend G. If $|G| + |G_{tail}| < min_g$, then G should be pruned.

Based on the same idea in Section 4.1.3, we can use the maximal coherent gene clusters to prune the unpromising coherent gene clusters. Suppose we are searching a gene combination G_1. Let S_1 be one maximal coherent sample set with respect to G_1, i.e., $S_1 \in S_{G_1}$. If there exists a maximal coherent gene cluster $(G \times S)$ such that $S_1 \subseteq S$ and $G_1 \subseteq G$, then S_1 should be removed from the list of the maximal coherent sample sets S_{G_1}, since it cannot lead to any maximal coherent gene cluster. Moreover, if S_{G_1} becomes empty after the pruning, then G_1 should be pruned, since any recursive search from G_1 cannot lead to any maximal coherent gene cluster.

Function *find-max-coherent-sample-sets*
Input: gene sets G_1 and G_2, sets of maximal coherent sample sets S_{G_1} and S_{G_2};
Output: the maximal coherent sample sets w.r.t. $G_1 \cup G_2$
Method:
`let` $S_{G_1 \cup G_2} = \emptyset$;
`for` each maximal coherent sample set $S_i \in S_{G_1}$ `do`
`for` each maximal coherent sample set $S_j \in S_{G_2}$ `do`
`let` $S_k = S_i \cap S_j$;
`if` $|S_k| \geq min_s$ `then` insert S_k into $S_{G_1 \cup G_2}$;
`end-for`
`end-for`
`for` each $S_k \in S_{G_1 \cup G_2}$ `do`
`if` S_k is a proper subset of $S' \in S_{G_1 \cup G_2}$
`then` $S_{G_1 \cup G_2} = S_{G_1 \cup G_2} - \{S_k\}$;
`end-for`
`output` $S_{G_1 \cup G_2}$;

Figure 7: Computing maximal coherent sample set for a set of genes.

4.2.3 Merging Coherent Genes in the Tail List

In our running example, the maximal coherent sample sets with respect to gene g_1 and g_3 are identical. Then, for any coherent gene cluster $(G \times S)$ such that $g_1 \in G$, $(G \cup \{g_3\} \times S)$ must also be a coherent gene cluster. Thus, we can search $\{g_1, g_e\}$ in a shoot.

In general, we have the following result.

LEMMA 4.1. *When search a combination of genes G, if there exist genes $\{g_{j_1}, \ldots, g_{j_k}\} \subseteq G_{tail}$ such that they are coherent on every maximal coherent sample set of G, then there exists no maximal coherent gene cluster containing G but no $\{g_{j_1}, \ldots, g_{j_k}\}$.* ∎

Based on Lemma 4.1, we can immediately merge genes $\{g_{j_1}, \ldots, g_{j_k}\}$ to G at the current node, and thus shrink the number of recursions. The computation time is saved as well, since we only need to check the coherent gene clusters, prune the irrelevant genes or unpromising gene clusters for all these genes in one shoot.

In our experiments on real data sets, we observe many genes can be merged by Lemma 4.1. The real-world *GST* microarray data sets are typically sparse and genes are coherent on a quite small number of sample sets. As a consequence, the performance of *Gene-Sample Search* can be improved substantially by this optimization.

One may ask, "*Do we have a symmetric pruning for* Sample-Gene Search*?*" We can apply the similar optimization technique for sample-gene. That is, a sample s_j is merged into current combination of samples S as long as the inverted list of S is a subset of that of s_j. However, such a situation is rare in practice, since it is rare that, for two different sample sets S_1 and S_2, the maximal coherent gene sets with respect to S_1 and S_2 are identical.

Based on the above discussion, we have the *Gene-Sample Search* algorithm as shown in Figure 8.

5. EXPERIMENTAL RESULTS

We implemented and tested our approaches on both a real *GST* microarray data set and synthetic data sets. The system is implemented in Java. The tests are conducted on a Sun Ultra 10 work station with a 440MHz CPU and 256 MB main memory.

Input and output: same as the *Sample-Gene Search* algorithm (Figure 6)
Method:
```
for i = 1 to (|G-Set| − min_g) do
    let G = {g_i} and G_tail = {g_{i+1}, ..., g_{|G-Set|}};
    call recursive-search(G, G_tail);
end-for
```

Procedure: *recursive-search*(G, G_{tail})
```
remove irrelevant genes from G_tail as described in
    Section 4.2.2;
if (|G| + |G_tail| < min_g) then return;
merge coherent genes in G_tail as described in
    Section 4.2.3;
for each maximal coherent sample set S_i ∈ S_G do
    if S_i can be pruned by the second criteria in
        Section 4.2.2 then remove S_i from S_G;
end-for
if (S_G = ∅) then return;
while G_tail ≠ ∅ do
    let i = min{j | g_j ∈ G_tail};
    let G_tail = G_tail − {g_i};
    call recursive-search(G ∪ {g_i}, G_tail);
end-while
for each sample set S_i in S_G do
    output (G × S_i) as a maximal coherent gene
        cluster if it is not subsumed by any maximal
        coherent gene cluster found before
end-for
End
```

Figure 8: The *Gene-Sample Search* algorithm.

5.1 The Data Sets

The real data set. We use the real gene-sample-time microarray data set reported in [20]. It consists of the microarray measurements of 4,324 genes in 13 multiple sclerosis (MS) patients before and at 1, 2, 4, 8, 24, 48, 120 and 168 hours after IFN-β treatments. MS patients show heterogeneous responses to IFN-β treatments. For example, the patients with relapsing MS respond better to IFN-β treatments than the patients with progressive disease do. However, relapsing MS patients also exhibit considerable inter-individual heterogeneity in their clinical responses to IFN-β therapies. So far, the effects of IFN-β treatment at the genomic level in humans are poorly understood. Researchers are interested in distinguishing the heterogeneous clinical response to IFN-β therapy among the patients. On the other hand, characterized gene expression dynamics correlated to the heterogeneous responses potentially help in exploring the causing mechanisms at the molecular level.

Synthetic data. We observe that the preprocessing, i.e., mining the maximal coherent sample sets for each individual gene, is relatively fast. The major bottleneck in mining coherent gene clusters is in the latter part. Therefore, instead of generating synthetic *GST* data sets, we simulate the table of maximal coherent sample sets for genes such as in Figure 5(a). Initially, an empty table is created. Then, a certain number of coherent gene clusters ($G \times S$) are randomly generated. For each $g \in G$, S is inserted into the table as one maximal coherent sample set with respect to g. In addition to the size of the synthetic data set, i.e., the total number of genes in *G-Set* and the number of samples in *S-Set*, the synthetic data generator takes the following parameters: (1) k, the number of coherent gene clusters in the data set; (2) max_{gene} and min_{gene}, the maximal and minimal numbers of genes in a coherent gene cluster, respectively; and (3) max_{sample} and min_{sample}, the maximal and minimal numbers of samples in a cluster, respectively.

We generate the data sets by setting $min_{gene} = 10$ and $min_{sample} = 5$. max_{sample} is set to the same value of |*S-Set*|, and max_{gene} is set to 1000. In practice, only a small number of genes are correlated with a phenotype [7]. When the size of the data set grows, we expect to see more coherent gene clusters. To simulate the situation, we set k to $\frac{|G\text{-}Set| \cdot |S\text{-}Set|}{3000}$.

5.2 Results on the MS Microarray Data

We apply our algorithms on the MS microarray data with $min_g = 15$, $min_s = 3$ and $\delta = 0.8$. In total, 21 coherent gene clusters are reported. To better understand the mining results, we feed the genes in each cluster to *Onto-Express* (http://vortex.cs.wayne.edu/Projects.html) and obtain a hierarchy of functional annotations in terms of *Gene Ontology* for each cluster. An example of gene ontology tree for cluster 17 is shown in Figure 9(a). Then, we further investigate the genes and samples in the clusters. Some interesting observations are obtained as follows.

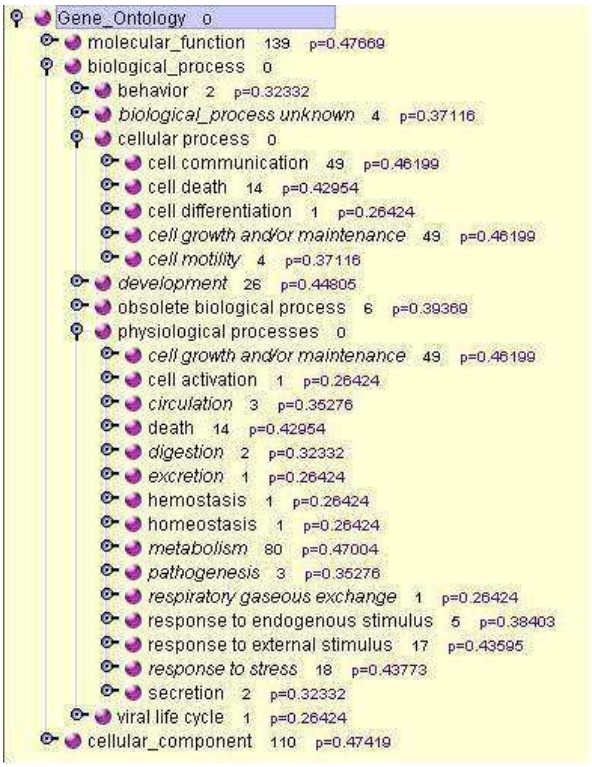
First, as expected, the majority genes in the clusters are involved in cellular processes and physiological processes, while genes involved in other biological processes (e.g., development, behavior and viral life cycle) are not highly represented (Figure 9(b)). Moreover, among the genes involved in cellular process, those involved in cell communication and cell growth and/or maintenance are predominant (Figure 9(c)). Since IFN-β is known to have anti-proliferative activities, the high population of cellular process genes involved in cell growth and/or maintenance is biologically plausible.

Second, a bunch of genes (e.g., in cluster 10, genes $H38522$, $AA704613$, $N92443$, $N75595$, etc.) that are well known for transcriptional signaling and cellular signaling can be identified in the resulted clusters. Those genes, together with the other genes in the same cluster that are unknown or poorly understood, may serve as switches in the genetic network and hence play an essential role in the biological processes. Thus, studying the time-series of the genes in the coherent gene clusters may greatly help people understand the regulatory mechanisms behind the response to IFN-β treatment.

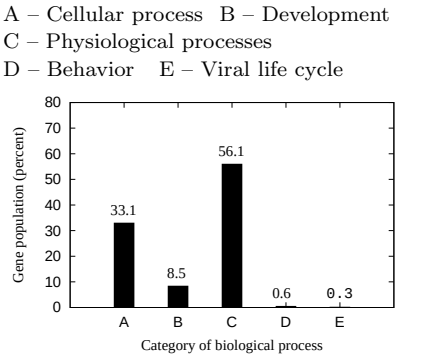
Last, coherent gene clusters also consist of different groupings of samples, which provide promising hypothesis for different phenotypes. For example, in the MS microarray data, the expression data from patients with different responses to IFN-β treatment are collected. Among the 21 reported clusters, only 2 clusters (cluster 10 and 17) consist of 4 samples, while other clusters only consist of the min_s number of samples. Therefore, those two clusters may become good candidates for target phenotype. Based on the clinical records of the patients, and combined with the gene information from the clusters, the interpretation of the sample groups is currently under investigation.

5.3 Effects of the Parameters

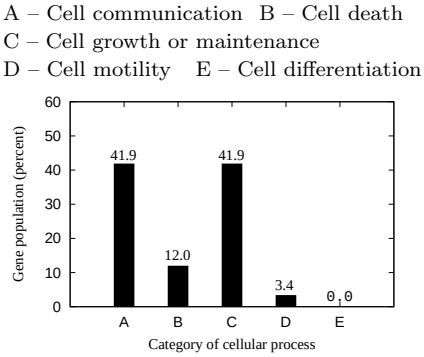
The maximal coherent gene cluster is defined with respect to three parameters, i.e., the minimum number of genes min_g, the minimum number of samples min_s and the coherence threshold δ. We test the effect of the parameters on the real *GST* data set. Figure 10(a) shows the number



(a) The gene ontology tree for genes in cluster 17



(b) The distribution of biological process



(c) The distribution of cellular process

Figure 9: The gene ontology tree and the distribution of functions for genes in cluster 17.

of coherent gene clusters when min_g varies from 5 to 100, $min_s = 3$ and $\delta = 0.8$. Clearly, the number of coherent gene clusters in the data set decreases when min_g increases. The result concurs the intuition: with a lower min_g value, we can catch more clusters with more or less genes. As a matter of fact, with fixed min_s and δ, let $\mathcal{B}_i$ be the complete set of coherent blocks when $min_g = i$. Then, we can show $\mathcal{B}_1 \supseteq \ldots \mathcal{B}_i \supseteq \ldots \mathcal{B}_n$.

Figure 10(b) shows the number of coherent gene clusters with respect to various min_s when min_g and δ are fixed to 10 and 0.8, respectively. This result can be explained in a way similar to the situation of min_g. Figure 10(c) shows the effect of δ on the number of coherent gene clusters in the data set, with $min_g = 10$ and $min_s = 3$. When we lower the coherence threshold, more combinations of samples are "coherent" by chance with respect to a minimum of min_g genes.

Interestingly, the three curves in Figure 10 share similar trends. That is, when the value of the parameter (represented by the X axis) increases, the number of coherent gene clusters (represented by the Y axis) goes down. The curve drops sharply until a "knot" is met, then the curve goes stably to the right. For example, we can see the "knots" of $min_g = 20$ in Figure 10(a), $min_s = 4$ in Figure 10(b) and $\delta = 0.85$ in Figure 10(c). These "knots" indicate that there exist stable and significant coherent gene clusters in the real data set. They are highly correlated, involving a statistically significant number of genes and samples. The "knots" also suggest the best settings of the parameters to avoid the coherent gene clusters formed just by chance.

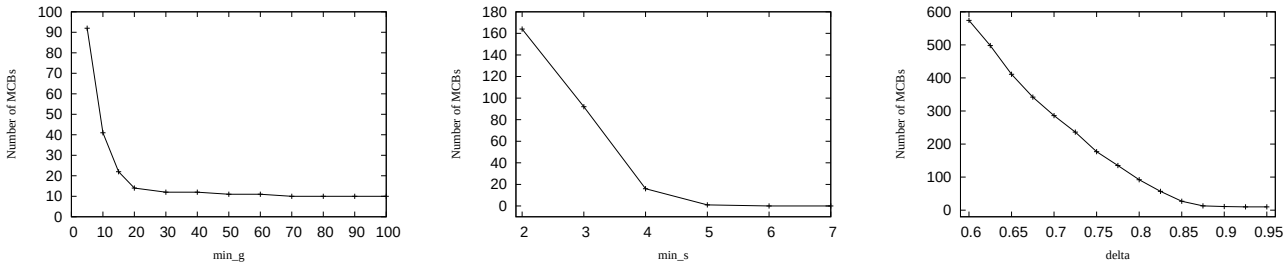
5.4 Scalability

We first test the efficiency of the preprocessing (algorithm in Figure 3) on various random subsets (by sampling) of the real microarray data set. The size of the subsets varies from 500 to 4324 genes, and all the samples are included. For each size, we sampled 30 subsets and calculate the average runtime. Figure 11(a) illustrates the scalability for the preprocessing step. As we discussed in Section 3, the real *GST* microarray data sets are often sparse. With the efficient pruning techniques, the preprocessing algorithm is linearly scalable to the size of the data sets.

We then test the scalability of both *Gene-Sample Search* and *Sample-Gene Search* on synthetic data sets. We set $min_s = 5$, $min_g = 10$, and $\delta = 0.8$. We first fix the number of samples to 30, and report the runtime with respect to number of genes (Figure 11(b)). We can see both approaches show an approximately linear scalability with respect to the number of genes. Figure 11(c) shows the scalability for both approaches under different sizes of sample sets (from 30 to 100), when the number of genes is fixed to 3000. We can see both approaches scale well with respect to the number of samples. Since the number of genes for a microarray data is typically by far larger than that of the samples, the enumeration of genes is much more expensive than the enumeration of samples. This explains why the *Sample-Gene Search* is faster than the *Gene-Sample Search*.

5.5 The Effect of Lemma 4.1

Lemma 4.1 can identify the genes g_i that can be merged into the current combination of genes, and thus can reduce

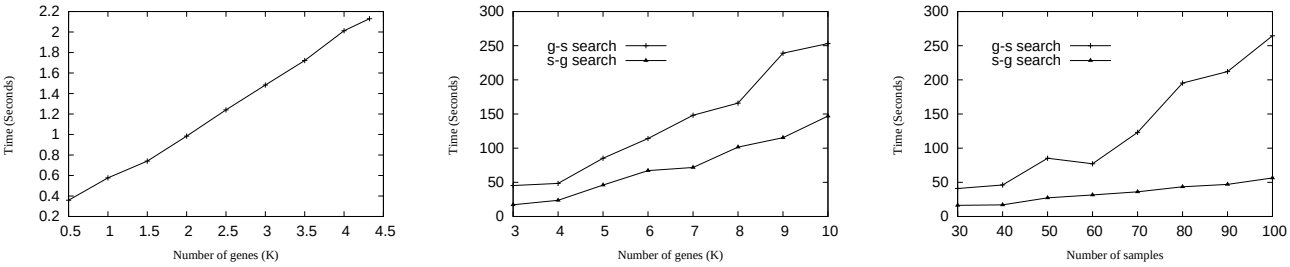


(a) Number of clusters vs. min_g $min_s = 3$, $\delta = 0.8$

(b) Number of clusters vs. min_s $min_g = 10$, $\delta = 0.8$

(c) Number of clusters vs. δ $min_g = 10$, $min_s = 3$

Figure 10: The effects of the parameters on the number of clusters.



(a) Preprocessing

(b) Scalability w.r.t. number of genes (number of samples: 30)

(c) Scalability w.r.t. number of samples (number of genes: 3,000)

Figure 11: Scalability on large data sets.

the number of recursions in the mining. We use some samples of the real microarray data set (each subset contains 100 to 1000 genes and 12 patients) to compare the performance of the *Gene-Sample Search* with and without the optimization. The comparison is conducted in three aspects: (1) the maximal number of recursion levels in the *Gene-Sample Search*; (2) the number of gene combinations in the *Gene-Sample Search*; and (3) the runtime. Figure 12 shows the results. We can clearly see that (1) the maximal number of recursion levels can be reduced substantially (Figure 12(a)); (2) with the optimization, the total number of gene combinations needed to be checked goes down sharply (12(b)); and (3) the runtime is much shorter when the optimization is applied (12(c)). The results strongly confirm that the optimization is effective for *Gene-Sample Search*.

We also apply the spirit of Lemma 4.1 on the *Sample-Gene Search*, and conduct similar tests. However, we can hardly see any significant improvement brought by the optimization. As we discussed in Section 4.2.3, due to the sparsity of the microarray data, many genes can be merged because they are coherent on the same sample set. However, few samples would be merged together since usually the maximal coherent gene sets with respect to two different sample sets are not identical.

6. RELATED WORK

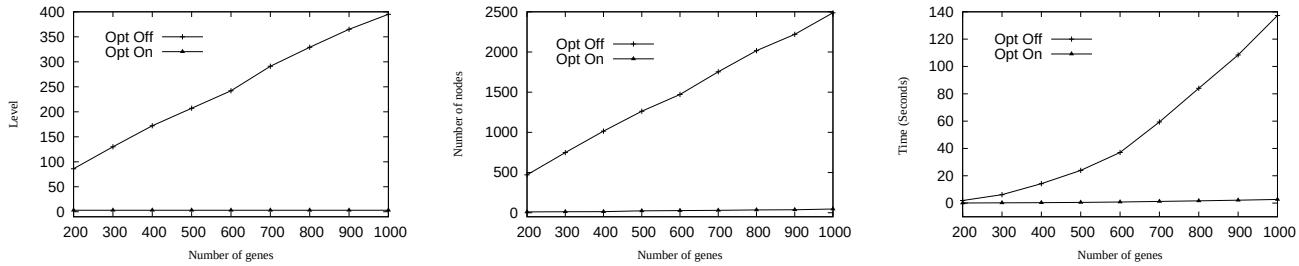
This research is related to previous work on clustering conventional gene-time / gene-sample microarray data and frequent itemset mining.

As we introduced in Section 1, there have been two categories of conventional microarray data sets: gene-time data sets and gene-sample data sets. For gene-time microarray data, various algorithms (e.g. [17, 15, 8]) have focused on clustering the genes. That is, co-expressed genes are grouped based on their expression patterns during the time series. On the other hand, different approaches (e.g. [4, 21, 16]) have been proposed to partition the sample sets to find their macroscopic phenotypes as well as to detect informative genes which manifest the sample partition. However, all the cluster models in those previous studies are substantially different from our coherent gene clusters. As a consequence, those algorithms cannot be extended directly to solve our problem.

In [6], Cheng and Church introduce the concept of bicluster to measure the coherence between genes and conditions (either time series or samples). Given a set of genes and a set of conditions, a bicluster is a subset of genes coherent with a subset of conditions. Yang et al. [22] propose a move-based algorithm to find biclusters more efficiently. Both algorithms in [6] and [22] adopt heuristic search approaches, and thus cannot guarantee to find the complete set of biclusters in the data set.

In [19], Wang et al. propose the model of pattern-based cluster. Given a subset of objects O and a subset of attributes A, pair (O, A) forms a pattern-based cluster if for any pair of objects x, $y \in O$, and any pair of attributes a, $b \in A$, the difference of change of values on attributes a and b between objects x and y is smaller than a threshold δ. In a recent study [11], Pei et al. has proposed an efficient algorithm, *MaPle* to mine the complete set of maximal pattern-based clusters.

We borrow some important ideas from previous studies on frequent itemset mining. First, the framework of our approaches is similar in spirit to that of pattern-growth



(a) Maximal number of recursion levels

(b) Number of gene combinations searched

(c) Runtime

Figure 12: Effect of Lemma 4.1 for *Gene-Sample Search*.

methods for frequent pattern mining. Second, the pruning techniques in our approaches share some interesting similarities with the methods of mining frequent closed itemsets (e.g., [10]). However, there are two essential differences between the frequent pattern mining methods and the approaches developed in this paper. On the one hand, the coherent gene clusters are inherently different from frequent itemsets. Thus, the similarity between the two categories of methods is only at the level of spirit (e.g., set enumeration and pruning). The technical details are dramatically different. On the other hand, new techniques such as the inverted lists are adopted to tackle the particular microarray data.

7. CONCLUSIONS

In this paper, we have investigated a novel type of gene-sample-time microarray data sets and propose a new problem of mining coherent gene clusters from such data sets. We have conducted a systematic study to develop two mining methods: the *Sample-Gene Search* and the *Gene-Sample Search*. Our extensive performance study on both a real microarray data set and synthetic data sets shows that there exist interesting and significant coherent gene clusters in the real data set, and both the algorithms have good performance. Since the number of genes in the microarray data is typically by far larger than the number of samples, the *Sample-Gene Search* usually outperforms the *Gene-Sample Search*.

8. REFERENCES

[1] Alizadeh A.A., et al. Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling. *Nature*, Vol.403:503–511, February 2000.

[2] Alon U., et al. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide array. *Proc. Natl. Acad. Sci. USA*, Vol. 96(12):6745–6750, June 1999.

[3] Black M.A. and Doerge R.W. Calculation of the minimum number of replicate spots required for detection of significant gene expression fold change in microarray experiments *Bioinformatics*, 18:1609-1616, 2002

[4] Ben-Dor A., et al. Class discovery in gene expression data. In *Proc. Fifth Annual Inter. Conf. on Computational Molecular Biology (RECOMB 2001)*.

[5] Kerr M.K. and Churchill G.A. Bootstrapping cluster analysis: Assessing the reliability of conclusions from microarray experiments. *Proc. Natl. Acad. Sci. USA*, Vol. 98:8961-8965.

[6] Cheng Y. and Church G.M. Biclustering of expression data. *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology (ISMB)*, 2000.

[7] Golub T.R., et al. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, Vol. 286(15):531–537, October 1999.

[8] Jiang D., et al. Interactive Exploration of Coherent Patterns in Time-Series Gene Expression Data. In *KDD'03*.

[9] Moler E.J., et al. Analysis of Molecular Profile Data Using Generative and Discriminative Methods. *Physiological Genomics*, Vol. 4(2):109–126, 2000.

[10] Pei J., et al. CLOSET: An efficient algorithm for mining frequent closed itemsets. In *DMKD'00*.

[11] Pei J., et al. MaPle: A Fast Algorithm for Maximal Pattern-based Clusterin. In *IEEE ICDM'03*.

[12] Herwig R., et al. Large-Scale Clustering of cDNA-Fingerprinting Data. *Genome Research*, 9:1093–1105, 1999.

[13] Rymon R. Search through systematic set enumeration. In *Proc. 1992 Int. Conf. Principle of Knowledge Representation and Reasoning (KR'92)*.

[14] Dudoit S., et al. Comparison of discrimination methods for the classification of tumors using gene expression data. *J. of the American Statistical Association*, Vol. 97, No. 457.

[15] Tamayo P., et al. Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. *Proc. Natl. Acad. Sci. USA*, Vol. 96(6):2907–2912, March 1999.

[16] Tang C., et al. Mining phenotypes and informative genes from gene expression data. In *SIGKDD'03*.

[17] Tavazoie S., et al. Systematic determination of genetic network architecture. *Nature Genet*, pages 281–285, 1999.

[18] Tusher V.G., et al. Significance Analysis of Microarrays Applied to the Ionizing Radiation Response. *Proc. Natl. Acad. Sci. USA*, Vol. 98(9):5116–5121, April 2001.

[19] Wang H., et al. Clustering by Pattern Similarity in Large Data Sets. In *SIGMOD 2002*.

[20] Weinstock-Guttman B., et al.. Genomic effects of interferon-b in multiple sclerosis patients. *Accepted pending revisions Journal of Immunology*, 2003.

[21] Xing E.P. and Karp R.M. Cliff: Clustering of high-dimensional microarray data via iterative feature filtering using normalized cuts. *Bioinformatics*, Vol. 17(1):306–315, 2001.

[22] Yang J., et al. δ-cluster: Capturing Subspace Correlation in a Large Data Set. In *ICDE 2002*.