Learning Phenotype Structure Using Sequence Model

Yuhai Zhao [#1], Guoren Wang [#2], Xiang Zhang [†3], Jeffrey Xu Yu [‡4], Zhanghui Wang [#1]

[#] *Information Science and Engineering, Northeastern University, China*

[†] *Electrical Engineering and Computer Science, Case Western Reserve University, USA*

[‡] *Systems Engineering and Engineering Management, Chinese University of Hong Kong, Hong Kong*

[1]zhaoyuhai@ise.neu.edu.cn [2]wanggr@mail.neu.edu.cn [3]xiang.zhang@case.edu [4]yu@se.cuhk.edu.hk

Abstract—Advanced microarray technologies have enabled to simultaneously monitor the expression levels of all genes. An important problem in microarray data analysis is to discover phenotype structures. The goal is to (1) find groups of samples corresponding to different phenotypes (such as disease or normal), and (2) for each group of samples, find the representative expression pattern or *signature* that distinguishes this group from others. Some methods have been proposed for this issue, however, a common drawback is that the identified signatures often include a large number of genes but with low discriminative power.

In this paper, we propose a g^*-sequence model to address this limitation, where the ordered expression values among genes are profitably utilized. Compared with the existing methods, the proposed sequence model is more robust to noise and allows to discover the signatures with more discriminative power using fewer genes. This is important for the subsequent analysis by the biologists. We prove that the problem of phenotype structure discovery is NP-complete. An efficient algorithm, FINDER, is developed, which includes three steps: (1) trivial g^*-sequences identifying, (2) phenotype structure discovery, and (3) refinement. Effective pruning strategies are developed to further improve the efficiency. We evaluate the performance of FINDER and the existing methods using both synthetic and real gene expression datasets. Extensive experimental results show that FINDER dramatically improves the accuracy of the phenotype structures discovered (in terms of both statistical and biological significance) and detects signatures with high discriminative power. Moreover, it is orders of magnitude faster than other alternatives.

Index Terms—Data mining, bioinformatics, Microarray data.

1 Introduction

Advanced microarray technologies have made large amounts of gene expression profiles available. Analyzing microarray data is essential for understanding the gene functions, gene regulation, cellular process, and subtypes of cells [1]–[3].

An important task in microarray data analysis is phenotype structure discovery [4]. Given a microarray dataset of m samples and n genes, a phenotype structure refers to a group of "blocks" (or submatrices), each of which consists of a subset of samples and a subset of genes such that: (1) the samples from all the blocks make up a partition of m samples, and the samples in a block correspond to a phenotype (such as a disease subtype); and (2) the gene expression pattern within a block can be used as the signature to distinguish this group of samples from others [5]. The genes in a signature may suggest the potential biomarkers related to the disease. In particular, phenotype structure discovery is an unsupervised learning problem. It is more challenging than the problem of biomaker selection with known class labels [4], [6].

A simplistic example could be used to help understand what we mentioned above. As in Fig. 1, suppose "◯", "△" and "▱" denote three different phenotype styles hidden in the original expression matrix. After rearranging the rows and columns, three submatrices are clearly outlined in the rearranged matrix. The samples from the three submatrices make up a partition of all given samples and each group of samples match a real phenotype. Moreover, if the genes in a group exhibit a specific pattern distinguishing this group of samples from others, they may suggest the potential biomarkers related to the phenotype.

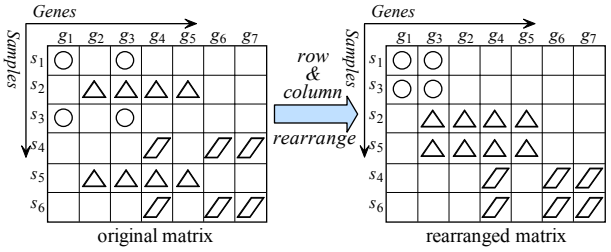


Fig. 1. A simplistic example of the phenotype structure

The existing methods for phenotype structure discovery can be classified into two categories: singleton discriminability-based approach and combination discriminability-based approach [4], [6]. The singleton approach evaluates individual genes by their discriminative power for the current sample partition and selects top-ranked genes. This approach assumes that the genes are mutually independent. It does not utilize any relationship among genes, although genes usually act and coordinate with each other to achieve certain biological function [7]. The combination approach focuses on finding a subset of genes that have strong discriminant power when considered together. However, it does not yet explicitly model the relationship among genes except the co-occurrence of the selected genes. This often leads to a large number of selected genes. The large number of genes poses crucial challenge for the domain experts to interpret and validate the results.

In this paper, we model the discriminative genes from a new

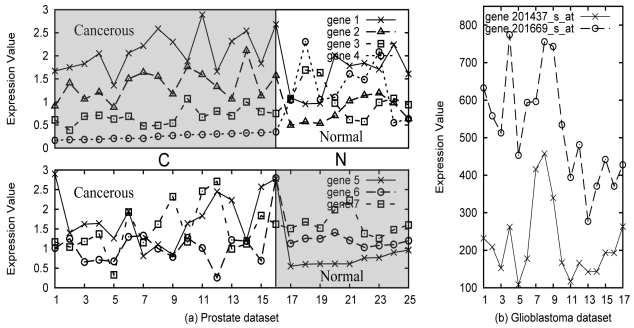


Fig. 2. (a) discriminative gene sets from the Prostate cancer dataset (b) lack of discriminative power of the OPSM model in the Glioblastoma cancer dataset

perspective by exploiting their ordered gene expression values. Our model considers the interrelationship in the expression patterns and is more robust to noise compared with the existing models. Fig. 2(a) shows an example of discriminative gene sets using the Prostate cancer gene expression dataset [8].

Example 1: Fig. 2(a) consists of two subfigures. In the top subfigure, 4 genes are expressed over 25 samples. Samples $1\sim16$ are cancerous (labeled as 'C') and samples $17\sim25$ are normal (labeled as 'N'). In the bottom subfigure, another set of 3 genes are expressed over the same set of samples. The existing singleton or combination discriminability-based methods cannot distinguish the two phenotypes. Since most genes are of similar average expression values in the two phenotypes, they will not be selected by the singleton approach. Moreover, all genes are expressed in both phenotypes. Thus, the combination approach based on the co-occurrence of genes will not select them either. Both of the methods ignore the hidden interrelation among genes. In the top subfigure, the gene order over the samples of cancerous phenotype 'C' is always $gene_4 \prec gene_3 \prec gene_2 \prec gene_1$. Such order is disturbed in normal phenotype 'N'. In the bottom subfigure, the gene order in normal phenotype 'N' is $gene_5 \prec gene_6 \prec gene_7$, while in cancerous phenotype 'C' such order does not exist. Based on the ordered expression values, a perfect phenotype structure (consisting of the two shadowed "blocks") is identified.

In biology community, discriminative sequential patterns involving the ordered gene expression values have been shown effective in distinguishing phenotypes [7], [9]. Such patterns have an intuitive biological interpretation. Complex diseases often involve the cooperation of multiples genes. These genes work together as a system to keep the cell in a specific state, e.g., disease or normal. In such a state, some special interrelationship among genes will exhibit. Once such relationship is disrupted, the state may change, e.g., from normal to disease.

Another advantage of the sequence model is that only a small number of genes are needed to achieve high phenotype discriminability. Intuitively, this is because it exploits more information ignored by other models, i.e., the interrelation among the genes beyond the co-occurrence. Finding fewer but more powerful discriminative genes is crucial for interpretation and validation in the subsequent wet-lab experiments [10].

Biclustering algorithms have been studied to analyze gene expression data [11]–[13]. Among the existing biclustering algorithms, the order-preserving submatrix (OPSM) model also incorporates the order information of the gene expression values [11], [14], [15]. An OPSM consists of a subset of genes and a subset of experimental conditions such that the expression profiles of the genes show the same tendency, e.g., strictly ascending or descending. Its goal is to capture the pattern coherence of genes, not the homogeneity of conditions. However, our goal is to partition the samples into groups corresponding to the real phenotypes and discover the discriminative genes. The two are not necessarily associated, and the OPSM method cannot be directly applied to the problem of phenotype structure discovery. An example is given below.

Example 2: *(1) A phenotype structure may not be modeled by an OPSM:* In Figure 2(a), the two shadowed blocks constitute a perfect phenotype structure. However, in the top block, although $gene_{208413}$ shows a strictly ascending profile over samples $1\sim16$, the other genes do not follow. The similar case also occurs in the bottom block, where the three genes also show different profiles over samples $17\sim25$. Thus, no OPSM will be discovered even though the phenotype structure does exist. *(2) An OPSM may not correspond to a phenotype structure:* In Figure 2(b), genes 201437_s_at and 201669_s_at show the similar rising and falling tendency over 17 samples in the Glioblastoma dataset [16]. The OPSM method will report a pattern involving these two genes. However, this pattern cannot be used for phenotype structure discovery, since among the 17 sample, the first 9 samples are tumor cells and the last 8 samples are pseudopalisading cells. The two coexpressed order preserving genes cannot help to distinguish these two classes.

In this paper, we develop a sequence model incorporating the ordering information of gene expression values into phenotype structure discovery. Our contributions are summarized as follows.

(1) We propose a g^*-sequence model to characterize the phenotype structure. It introduces the concept of *significant chain* to ensure that there is a significant difference between the expression values of any pair of genes. This property helps to improve the robustness of the proposed model, and enables to identify highly discriminative signatures with only a small number of genes.

(2) To measure the quality of a candidate phenotype structure, we propose a novel sequence dissimilarity metric, namely *projection divergence*. Based on this metric, the difference between a pair of blocks (submatrices) can be quantified based on the discriminative power of the signatures within the blocks.

(3) We show that the problem of phenotype structure discovery is NP-complete. Given n genes, the total number of subsequences (candidate signatures) is $\sum_{i=1}^{m}(C_m^i \cdot i!)$. We prove that the prohibitively large search space can be reduced to a much smaller scale.

(4) An efficient algorithm, FINDER, is developed to find the optimal phenotype structure. By incorporating the cross projection into a progressive exploring framework, candidate phenotype structures are searched in a quality-guaranteed way.

(5) We conduct extensive experiments on both real and synthetic datasets. The results show that FINDER dramatically improves the efficiency of the mining process. With very few

genes, the discovered signatures are able to unravel phenotype structures that are both statistically and biologically significant.

The rest of this paper is organized as follows. In Section 2, we introduce some preliminaries and give the problem description. Section 3 details our solution. Experimental analysis is given in Section 4. Section 5 reviews some related work. Finally, Section 6 concludes this paper.

2 THE PROBLEM

2.1 g^*-sequence

A microarray dataset D is an $m \times n$ matrix, with m samples $S=\{s_1, s_2, \cdots, s_m\}$ and n genes $G=\{g_1, g_2, \cdots, g_n\}$. A real value d_{ij} in D represents the expression value of gene g_j on sample s_i. An example microarray dataset with 9 genes and 4 samples is shown in Table 1. Microarray data are often noisy. We introduce the concept of equivalent dimension group which represents a set of genes with similar expression values.

TABLE 1
An Example Microarray Dataset

Sample	g_1	g_2	g_3	g_4	g_5	g_6	g_7	g_8	g_9
s_1	103	68	76	48	71	101	55	50	83
s_2	35.5	20.1	28.7	17.2	13.2	23.8	13.5	15.8	30
s_3	5.7	6.7	9	5	10.3	10	15.2	5.2	8.7
s_4	32	53	79	43	35	72	105	38	68

Definition 1: For a sample $s_i \in S$, a subset of genes $G' \subseteq G$ is an **equivalent dimension group**, or an *EDG*, if G' satisfies the following two criteria:

$$\max_{g_j, g_{j'} \in G'} |d_{ij} - d_{ij'}| < \delta \times \min_{g_j \in G'} d_{ij}, \quad (1)$$

$$\forall g_j \in G', \min_{g_{j'} \in G'} |d_{ij'} - d_{ij}| < \min_{g_{j''} \in (G-G')} |d_{ij} - d_{ij''}|. \quad (2)$$

Criterion (1) limits the maximum difference between any pair of expression values in an EDG. Criterion (2) guarantees that a gene is always grouped with its closest neighbor. If a gene satisfies criterion (1) but is excluded from an EDG by criterion (2), we call this gene a *breakpoint*.

Due to the highly noisy, considering close values as ordered is impractical in the context of microarray data analysis. An EDG encloses a group of genes with the similar expression values together. The sequences of genes in which any pair of genes are not contained by the same EDG is robust to noise w.r.t the group threshold δ. Moreover, only considering such genes makes the maximum size of the sequences is far less than that of the original ones. Thus, the time taken by sequence mining is greatly reduced while keeping the significant results.

For a sample s_i, a sliding window approach can be applied to find all EDGs. First, all genes are sorted by their expression values in ascending order. Second, we slide a window from left to right. The size of every window is initially determined by criterion (1), and then refined by criterion (2). If there is a breakpoint, the next window starts from the first breakpoint. Otherwise, the next window starts from the position immediately right to the current left-end of the window.

Suppose that $\delta=0.5$. Applying the sliding window method on every sample in Table 1, we can obtain the corresponding sequences of EDGs, as shown in Fig. 3. Note that different brackets are used to distinguish different EDGs in the same sequence. Within an EDG, no order is considered. For example, $(g_4 g_8 g_2)$ and $(g_8 g_4 g_2)$ are identical. We use EDG_i to denote the i-th EDG in a sample.

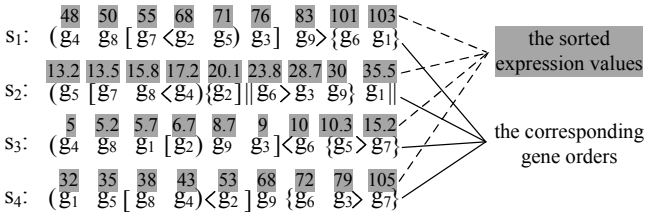


Fig. 3. g^*-sequences for the samples in Table 1, $\delta=0.5$

In Fig. 3, each sample can be treated as a sequence of EDGs. Note that the EDGs may overlap with each other. For example, in s_1, g_5 belongs to EDG_1, EDG_2 and EDG_3. The overlapping EDGs can be treated as allowing *negative* gap in a sequence pattern. This is different from the traditional sequence pattern definitions which only allow zero and/or positive gaps, i.e., non-overlaping events. In this sense, we refer to a sequence of EDGs as a ***g*-sequence***, where $g*$ means any gaps. For sample s_i, its g^*-sequence is denoted as $\mathbb{S}_i$.

Given a g^*-sequence $\mathbb{S}_i$, $\mathcal{R}(x, y)$ is a binary relation for a pair of genes x and y. $\mathcal{R}(x, y)$ is TRUE if there exists an EDG in $\mathbb{S}_i$ containing both x and y. Otherwise, $\mathcal{R}(x, y)$ is FALSE.

Definition 2: Given two g^*-sequences, $\mathbb{S}_i$ and $\mathbb{S}_j$, if $\forall x, y \in \mathbb{S}_i$, $\mathcal{R}(x, y)$ always holds the same value for both $\mathbb{S}_i$ and $\mathbb{S}_j$, we say $\mathbb{S}_i$ is a **subsequence** of $\mathbb{S}_j$, denoted as $\mathbb{S}_i \sqsubseteq \mathbb{S}_j$. In particular, if $\forall x,\ y \in \mathbb{S}_i$, $\mathcal{R}(x, y)$ is always FALSE in $\mathbb{S}_i$ and $\mathbb{S}_j$, we say $\mathbb{S}_i$ is a **significant chain** of $\mathbb{S}_j$. Further, $\mathbb{S}_i$ is **closed** if there is no $\mathbb{S}_i'$ s.t. $\forall \mathbb{S}_j$, $\mathbb{S}_i \sqsubseteq \mathbb{S}_i' \sqsubseteq \mathbb{S}_j$.

Consider $\mathbb{S}_1$ in Figure 3, and two other g^*-sequences, $\mathbb{S}_i = (g_8 \langle g_2 g_5 \rangle g_3 \rangle g_6$ and $\mathbb{S}_j = (g_8 g_2 \langle g_5 \rangle g_3 \rangle g_6$. We have that $\mathbb{S}_i \sqsubseteq \mathbb{S}_1$, since $\forall x, y \in \mathbb{S}_i$, $\mathcal{R}(x, y)$ always holds the same value for both $\mathbb{S}_i$ and $\mathbb{S}_1$. However, $\mathbb{S}_j \not\sqsubseteq \mathbb{S}_1$ since $\mathcal{R}(g_2, g_3)$ is FALSE in $\mathbb{S}_j$, but TRUE in $\mathbb{S}_1$. Moreover, $g_8 g_3 g_6$ is a significant chain of $\mathbb{S}_1$. A significant chain ensures that there is a significant difference between the expression values of any pair of genes within it. In particular, $g_8 g_3 g_6$ is a closed significant chain, since no other significant chain $\mathbb{S}_i'$ s.t. $g_8 g_3 g_6 \sqsubseteq \mathbb{S}_i' \sqsubseteq \mathbb{S}_1$.

2.2 Phenotype Structure

Next we quantify the quality of a phenotype structure based on the g^*-sequence model. Note that a sample s_i can be represented by its g^*-sequence $\mathbb{S}_i$.

Definition 3: Suppose that m g^*-sequences $\mathbb{S}_i$ ($i \in [1, m]$) are partitioned into k disjoint subsets $\mathbb{S}_1, \mathbb{S}_2, \cdots, \mathbb{S}_k$. A subsequence $\mathbb{S}$ is a **signature** of subset $\mathbb{S}_l$ ($l \in [1, k]$), iff: (1) $\forall \mathbb{S}_x \in \mathbb{S}_l$, $\mathbb{S} \sqsubseteq \mathbb{S}_x$; and (2) $\forall \mathbb{S}_y \notin \mathbb{S}_l$, $\mathbb{S} \not\sqsubseteq \mathbb{S}_y$. In particular, if $\forall \mathbb{S}_x \in \mathbb{S}_l$, $\mathbb{S}$ is a significant chain of $\mathbb{S}_x$, we call $\mathbb{S}$ a **p-signature** of $\mathbb{S}_l$.

Consider the example in Fig. 3. Suppose that the four g^*-sequences are grouped into two disjoint subsets, $\mathbb{S}_1 = \{\mathbb{S}_1, \mathbb{S}_2\}$ and $\mathbb{S}_2 = \{\mathbb{S}_3, \mathbb{S}_4\}$. According to Definition 3, $\mathbb{S} = g_7(g_6 g_1)$ is a signature of $\mathbb{S}_1$. Moreover, two p-signatures can be derived

from it, i.e. g_7g_6 and g_7g_1. A p-signature is used as the fingerprint of $\mathbb{S}_l$.

Given a p-signature p_i and a sample s, the *projection* of p_i on s, denoted as $p_i|_s$, refers to the sequence of all genes in p_i permuted according to their relative orders in $\mathbb{S}$. If a pair of genes in p_i has a reverse relative order in $p_i|_s$, we call it a *reverse pair*. Given p_i and $p_i|_s$, for a gene x, if it is at the k-th locus in p_i and at the j-th locus in $p_i|_s$, we call $|k-j|$ the *distortion* of x between p_i and $p_i|_s$, denoted as $dist_x(p_i,s)$.

For example, suppose that p_i is $g_3g_4g_6$ and the sample s is s_1 in Table 1. Then, we have that $p_i|_s=g_4g_3g_6$. Specially, (g_3, g_4) is a reverse pair since it has the reverse relative order in p_i and $p_i|_s$. The locus of g_3 in p_i is 1, and in $p_i|_s$ is 2. Therefore, $dist_{g_3}(p_i, s)=2-1=1$. Likewise, we have that $dist_{g_4}(p_i, s)=1$.

Definition 4: Given a p-signature p_i and a sample s, the **projection divergence** of p_i and $p_i|_s$, denoted as $PD(p_i, p_i|_s)$, is

$$PD(p_i, p_i|_s) = \sum_{\substack{x,y\in p_i \\ x\neq y}} \psi(x,y)[dist_x(p_i,s) + dist_y(p_i,s)], \tag{3}$$

$$\text{where } \psi(x,y) = \begin{cases} 1 & \text{if } (x,y) \text{ is a reverse pair,} \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

Different from some commonly used sequence distance metrics, such as edit distance ED [17], which only accumulates the difference on individual items, PD takes the interrelationship among genes (items) into consideration when computing the difference. Continuing previous example where p_i is $g_3g_4g_6$ and the sample s is s_1 in Table 1, since there is only one reverse pair in p_i, i.e., (g_3, g_4), we have that $PD(p_i, p_i|_s)= 1\times[1+1]=2$.

Based on PD, a quality measure for a candidate phenotype structure can be defined as follows.

Definition 5: For a microarray dataset D, let $\mathbb{S}=\{\mathbb{S}_1, \mathbb{S}_2, \ldots, \mathbb{S}_k\}$ be a partition of the m samples and $\mathbb{G}=\{p_1, p_2, \ldots, p_k\}$ be a set of p-signatures, where p_i is a p-signature of $\mathbb{S}_i$ ($1\leq i\leq k$). A **phenotype structure** refers to the collection of all submatrices $\{(\mathbb{S}_i, p_i)\}$. Its **quality function** is defined as

$$Q(\mathbb{S},\mathbb{G}) = \frac{1}{C_k^2}\sum_{i=1}^{k}\sum_{j=i+1}^{k}\mathfrak{B}(i,j), \tag{5}$$

$$\text{where } \mathfrak{B}(i,j) = \frac{\sum_{\forall s\in\mathbb{S}_j} PD(p_i,p_i|_s) + \sum_{\forall s\in\mathbb{S}_i} PD(p_j,p_j|_s)}{|\mathbb{S}_i|+|\mathbb{S}_j|}, \tag{6}$$

$|\mathbb{S}_i|$ (or $|\mathbb{S}_j|$) denotes the number of samples in $\mathbb{S}_i$ (or $\mathbb{S}_j$).

Let $D_i=\{d_{x,y}|s_x\in\mathbb{S}_i,\ g_y\in p_i\}$ be the submatrix of $\mathbb{S}_i$ projected on p_i. $\mathfrak{B}(i,j)$ evaluates the mutual difference between two submatrixes D_i and D_j by crossly projecting p_i on $\mathbb{S}_j$ and p_j on $\mathbb{S}_i$. Larger $\mathfrak{B}(i,j)$ indicates larger mutual difference between D_i and D_j. The intuition behind *cross-projection* is that there should be significant difference between any pair of submatrices within a real phenotype structure. $Q(\mathbb{S},\mathbb{G})$ measures the average pairwise difference between submatrices.

Consider the example in Table 1. Suppose that the samples are partitioned into $\mathbb{S}_1=\{s_1, s_2\}$ and $\mathbb{S}_2=\{s_3, s_4\}$ with p-signatures $p_1=g_7g_1$ and $p_2=g_1g_6$, respectively. The corresponding $Q(\mathbb{S},\mathbb{G})$ can be calculated as follows. First, the projection of p_1 (resp. p_2) on every sample in $\mathbb{S}_2$ (resp. $\mathbb{S}_1$) are derived, i.e., $p_1|_{s_3}=p_1|_{s_4}=g_1g_7$, and $p_2|_{s_1}=p_2|_{s_2}=g_6g_1$. Then, according to Definition 4, we have that $\sum_{\forall s\in\mathbb{S}_2} PD(p_1,p_1|_s)=2+2=4$, and $\sum_{\forall s\in\mathbb{S}_1} PD(p_2,p_2|_s)=0+0=0$. Next, since $|\mathbb{S}_1|=|\mathbb{S}_2|=2$, we have that $\mathfrak{B}(1,2)=\frac{4+0}{2+2}=1$. Finally, we have that $Q(\mathbb{S},\mathbb{G})=\mathfrak{B}(1,2)=1$, where $k=2$. Note that $\sum_{\forall s\in\mathbb{S}_1} PD(p_2,p_2|_s)=0$, since $\{g_6g_1\}$ is an EDG, in which order should not be considered.

2.3 The Computational Problem

Given an expression matrix D of m samples and n genes, and a grouping threshold δ, our goal is to find the phenotype structure with the largest quality score $Q(\mathbb{S},\mathbb{G})$. To filter out the blocks with too few or too many samples, we introduce Min_s and Max_s to limit the minimum and the maximum number of samples in a block.

Theorem 1: The problem stated above is $\mathcal{NP}$-complete.

Proof: The hardness proof is by reduction from EXACT COVER problem, where the instance is a collection $\mathbb{U}$ of subsets of a set X, and the question is whether there exists a subcollection $\mathbb{U}^*$ of $\mathbb{U}$, such that each element $x_i\in X$ is contained by one and only one subset in $\mathbb{U}^*$. EXACT COVER is known to be one of Karp's 21 $\mathcal{NP}$-complete problems [18].

First, we prove the problem is in $\mathcal{NP}$. This is provable because the validation of whether a given $(\mathbb{S},\mathbb{G})$ is a solution can be decided in polynomial time according to Def. 5.

Second, we prove the problem is $\mathcal{NP}$-complete by reduction from EXACT COVER problem in the two following steps.

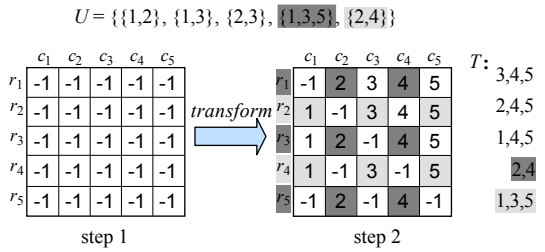


Fig. 4. An illustration of the proof

Step 1. Given any instance of EXACT COVER problem, $(X,\mathbb{U})$, w.l.o.g. $X=\{x_1, x_2, x_3, x_4, x_5\}$ and $\mathbb{U}=\{\{x_1,x_2\}, \{x_1,x_3\}, \{x_2,x_3\}, \{x_1,x_3,x_5\}, \{x_2,x_4\}\}$. We construct a matrix M of n rows and n columns, and initially set all entries in M to '-1' (as shown at the left hand of Fig.4).

Step 2. For each element in $\mathbb{U}$, say $U_l=\{x_i, x_j, \ldots, x_k\}$, we do a polynomial time operation. That is, for each x_r in U_l, hold the values of row r on columns $i, j, \ldots, k$, turn the values of row r on the rest columns to the corresponding column ids and record the set of ids in T. The transformed result of the given instance is shown at the right hand of Fig.4.

From the above steps, we can see that: (1) every entry in M will never turn from a positive column id to '-1', and (2) all entries of row i on column i always fix their values as '-1'. Thus, in $\mathbb{U}$, any pair of elements will not have the same set

of column ids. As such, there is a one-to-one correspondence between the elements in $\mathbb{U}$ and the submatrices, each of an element in $\mathbb{U}$ and the corresponding set of column ids. That is, there is an exact cover of X iff we can find a solution of M satisfying the mentioned problem. Hence the proof. □

3 THE FINDER ALGORITHM

FINDER consists of three major steps: (1) trivial g^*-sequences identifying, where the genes of less interests are filtered out; (2) phenotype structure discovery. The p-signatures and the corresponding samples are first derived to form the candidate phenotype blocks. A progressive block combination test are then used to find the phenotype structure with maximum average pairwise block difference; and (3) refinement, which further refines the quality of the results.

3.1 Trivial g^*-sequence identifying

A subsequence $\mathcal{S}$ is *trivial* if it is common to all m samples. Clearly, a trivial sequence cannot be selected as a p-signature to distinguish a specific phenotype from the others, and the genes involved in the trivial subsequences can be ignored. However, it is intractable to exhaustively enumerate all trivial subsequences. The following theorem states that the search space of trivial subsequences can be dramatically reduced.

Theorem 2: The genes covered by all trivial g^*-sequences are just as that covered by all closed trivial significant chains.

Proof: Let $\mathcal{S}$ be a trivial g^*-sequence and $x \in \mathcal{S}$ be a gene not covered by any significant chain of $\mathcal{S}$. According to the sliding window approach discussed in Section 2, there must be another gene $y \in \mathcal{S}$ such that either xy or yx form a significant chain, which contradicts the assumption. Moveover, Definition 2 indicates any significant chain must be contained in a closed significant chain. This completes the proof. □

Theorem 2 indicates that, instead of testing all trivial g^*-sequences, we only need to consider the closed trivial significant chains, the lengths of which are usually much shorter than that of the original g^*-sequences. As a result, the search space is greatly reduced.

In the next, we show that a Head-Tail matrix and a template-driven pattern growth method can be used to further improve the efficiency.

3.1.1 *Head-Tail Matrix*

The Head-Tail matrix M is a data structure that can be utilized to determine if a sequence is a significant chain. Given a g^*-sequence $\mathcal{S}_i$, and a gene $x \in \mathcal{S}_i$, we refer to the index of the first and the last EDG containing x as the *head position*, denoted as $H_i(x)$, and the *tail position*, denoted as $T_i(x)$, of x w.r.t. $\mathcal{S}_i$, respectively.

Table 2 shows the Head-Tail matrix M corresponding to the example in Fig. 3. Every entry $M_{(i,j)}$ records a vector (x, y), where x is the head position of g_j, $H_i(g_j)$, in $\mathcal{S}_i$, and y is the tail position of g_j, $T_i(g_j)$, in $\mathcal{S}_i$.

For a pair of genes g_j and g_k, we can efficiently decide whether $g_j g_k$ is a significant chain of $\mathcal{S}_i$ using M: If $H_i(g_k) > T_i(g_j)$, $g_j g_k$ must be a significant chain. Otherwise, it is not.

TABLE 2
The Head-Tail Matrix Corresponding to Fig. 3

Sample	g_1	g_2	g_3	g_4	g_5	g_6	g_7	g_8	g_9
s_1	4, 4	1, 3	2, 3	1, 1	1, 3	4, 4	1, 2	1, 1	3, 3
s_2	5, 5	2, 4	4, 5	1, 3	1, 1	3, 5	1, 2	1, 2	4, 5
s_3	1, 1	1, 2	2, 2	1, 1	3, 4	3, 3	4, 4	1, 1	2, 2
s_4	1, 1	2, 3	3, 4	1, 2	1, 1	3, 4	4, 4	1, 2	3, 3

For example, consider $g_4 g_9$ and s_2 in Fig. 3. We have that $g_4 g_9$ is a significant chain, since $H_2(g_9)=4>T_2(g_4)=3$. The process can be generalized to a sequence of any length as shown in the following theorem.

Theorem 3: Let $\mathcal{S} \sqsubseteq \mathcal{S}_i$ be a g^*-sequence. It is a significant chain of $\mathcal{S}_i$ iff $\forall x, y \in \mathcal{S}$ s.t. $x \prec y$, $T_i(x) < H_i(y)$ is TRUE. It is closed if there is no other gene z such that: (1) $\forall x, y \in \mathcal{S}$ s.t. $x \prec y$, $T(x) < H(z)$ and $H(y) > T(z)$ are both TRUE in every $\mathcal{S}_i$ containing $\mathcal{S}$, or (2) if x and y are the first and the last gene in $\mathcal{S}$, either $T(z) < H(x)$ or $H(z) > T(y)$ holds for every $\mathcal{S}_i$ containing $\mathcal{S}$, where '$x \prec y$' means x appears before y.

Proof: According to Def. 2, the theorem can be directly proved by showing $T_i(x) < H_i(y) \Leftrightarrow \mathcal{R}(x, y) = false$. Suppose $T_i(x) < H_i(y)$ but $\mathcal{R}(x, y)$=TRUE in $\mathcal{S}_i$. Then, there must be an EDG containing both x and y in $\mathcal{S}_i$. Thus, $H_i(y) \leq T_i(x)$. This contradicts the supposition. In turn, if $\mathcal{R}(x, y)$=FALSE in $\mathcal{S}_i$, we have $T_i(x) < H_i(y)$ since $x \prec y$. Hence the proof. □

3.1.2 *Template-driven Pattern Growth*

A sequence that is not a significant chain of $\mathcal{S}_i$ can be still a significant chain of $\mathcal{S}_j$ ($i \neq j$). To further reduce the search space, we introduce a template-driven pattern growth method specifically designed for microarray data, where the number of samples is usually much less than the number of genes.

The template-driven enumeration method examines every sample s_i in turn. That is, every sample is considered as a template exactly once. When sample s_i is used as the template, we only consider the subsequences $\mathcal{S}$ such that $\mathcal{S} \sqsubseteq \mathcal{S}_i$. The sequences that are not contained in any sample will not be enumerated. Moreover, different from the traditional sequence mining methods, such as PrefixSpan [19] and BIDE [20], during the growth of sequence $\mathcal{S}$, we do not pre-compute all possible one-item extensions to $\mathcal{S}$, and thus no projected database needs to be generated and maintained. Instead, when $\mathcal{S}_i$ is taken as the template, we only consider the genes y in $\mathcal{S}_i$ s.t. $H(y) > T(x)$, where x represents the last gene in $\mathcal{S}$. After all samples are considered, all possible significant chains will be identified. During the enumeration, once a significant chain is identified not to be closed, the following search can be pruned. The correctness of this method is ensured by Theorems 2 and 3.

3.2 Phenotype structure discovery

A block (or submatrix) is the basic element of a phenotype structure, which consists of a subset of samples and the corresponding p-signature. Thus, phenotype structure discovery can be naturally divided into the following three components: candidate p-signatures generation, block derivation from candidate p-signatures, and quality test of block combinations.

3.2.1 *Candidate p-signature generation*

According to Definition 3, a p-signature must be a significant chain. Thus, a naive candidate p-signature generating method is to check all significant chains, which, however, is infeasible in practice. The following theorem states that the candidate p-signatures can only result from the closed significant chains.

Theorem 4: Suppose that $(\mathbb{S}, \mathbb{G})$ and $(\mathbb{S}', \mathbb{G}')$ are two candidate phenotype structures, where $\mathbb{S}=\{\mathbb{S}_1, \mathbb{S}_2, \ldots, \mathbb{S}_k\}$, $\mathbb{S}'=\{\mathbb{S}'_1, \mathbb{S}'_2, \ldots, \mathbb{S}'_k\}$, $\mathbb{G}=\{p_1, p_2, \ldots, p_k\}$, and $\mathbb{G}'=\{p'_1, p'_2, \ldots, p'_k\}$. If $\forall i$, $1\leq i\leq k$, $p_i \sqsubseteq p'_i$ and p'_i is closed, then $Q(\mathbb{S}', \mathbb{G}')\geq Q(\mathbb{S}, \mathbb{G})$.

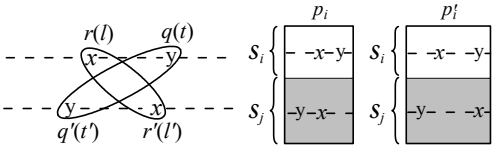


Fig. 5. $PD(p'_i, p'_i|_s)\geq PD(p_i, p_i|_s)$

Proof: We use Fig.5 as an illustration example, where the shadowed blocks are the projections of p_i and p'_i on all samples in $\mathbb{S}_j$. For a sample s in $\mathbb{S}_j$, the two dashed lines denote p_i and $p_i|_s$ (or p'_i and $p'_i|_s$), where (x, y) is a reverse pair. The position of x in p_i (resp. $p_i|_s$) is indicated by r (resp. r'), and that of y in p_i (resp. $p_i|_s$) is indicated by q (resp. q'). Similarly, the position of x in p'_i (resp. $p'_i|_s$) is indicated by l (resp. l'), and that of y in p'_i (resp. $p'_i|_s$) is indicated by t (resp. t'). Then, $[dist_x(p'_i, s) + dist_y(p'_i, s)] - [dist_x(p_i, s) + dist_y(p_i, s)] = (l' - l + t - t') - (r' - r + q - q') = [(l' - t') - (r' - q')] + [(t - l) - (q - r)]$. Since $p_i \sqsubseteq p'_i$, $[(t - l) - (q - r)] \geq 0$. Likewise, since $p_i|_s \sqsubseteq p'_i|_s$, $[(l' - t') - (r' - q')] \geq 0$. Therefore, the preceding formula is no less than 0. Extending the conclusion to any reverse pair in p_i, we conclude that $PD(p'_i, p'_i|_s) \geq PD(p_i, p_i|_s)$. Moreover, since s is any sample in $\mathbb{S}_j$, we have that $\sum_{\forall s\in\mathbb{S}_j} PD(p'_i, p'_i|_s) \geq \sum_{\forall s\in\mathbb{S}_j} PD(p_i, p_i|_s)$. Similarly, we have that $\sum_{\forall s\in\mathbb{S}_i} PD(p'_j, p'_j|_s) \geq \sum_{\forall s\in\mathbb{S}_i} PD(p_j, p_j|_s)$. Thus, $Q(\mathbb{S}', \mathbb{G}') \geq Q(\mathbb{S}, \mathbb{G})$. Hence the proof. □

Theorem 4 implies that the template-driven pattern growth method discussed in Section 3.1 for closed significant chains can be also applied for candidate p-signature generation.

3.2.2 *Block derivation from candidate p-signatures*

For every candidate p-signature, we find the set of samples containing it as a candidate phenotype block. Then the combinations of candidate blocks will be tested for their quality. Next, we show that a vertical bitmap representation of g^*-sequences can be utilized for efficient block derivation and closure checking/pruning.

First, we create a bitmap for each gene after the step of trivial g^*-sequence identifying. The bitmap consists of m sections each corresponding to a sample. If a gene x appears in EDG_j of $\mathcal{S}_i$, the j-th bit of section i in the bitmap for gene x is set to 1; otherwise, it is set to 0.

Second, for a candidate p-signature $p'_k = p_k x$, we find all samples characterized by p'_k via several bit operations. Suppose that the bitmap $B(p_k)$ for p_k is given. If bit j of section i is 1, sample $\mathcal{S}_i$ must contain p_k and the last item of p_k appears in EDG_j. According to Theorem 3, $\mathcal{S}_i$ will contain p'_k if only the index of the fist bit of value '1' in section i of $B(x)$, say l, is larger than that of the last bit of value '1' in section i of $B(p_k)$, say t.

Fig. 6. Bitmaps

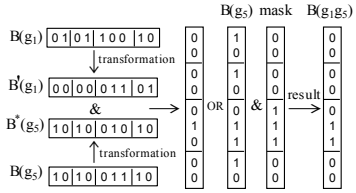


Fig. 7. Deriving bitmap for $g_1 g_5$

Finally, we transform $B(p_k)$ such that, in every section, all bits after t are set to 1, and the rest bits are set to 0. We also transform $B(x)$ such that, for every section, only bit l is set to 1 and the rest bits are all set to 0. We denote the two transformed bitmaps as $B'(p_k)$ and $B^*(x)$, respectively. If section i in $B(T)$, i.e., the result of $B'(p_k)$ *AND* $B^*(x)$, is not 0, sample $\mathcal{S}_i$ must contain p'_k; otherwise, not. Moreover, to update $B(p'_k)$, $B(T)$ is *OR*ed with $B(x)$, and the result is *AND*ed with a mask, which is derived by setting all bits in the sections of value '0' in $B(T)$ to 0, and setting the bits in the other sections to 1. As such, we obtain $B(p'_k)$, the exact bitmap for p'_k.

For example, Fig. 6 is the bitmap representation of the data in Fig. 3 after trivial g^*-sequence identifying. $(g_4 g_8 \langle g_2) g_3 g_9 \rangle$ is a closed trivial pattern. Fig. 7 is an illustration of the above process on $g_1 g_5$. For g_1 and g_5, the transformed bitmaps are $B'(g_1)$ and $B^*(g_5)$, respectively. The final bitmap for $g_1 g_5$ is $B(g_1 g_5)$, which indicates $g_1 g_5$ is only contained by $\mathcal{S}_3$. Further, since $H_3(g_5)=2$ and $T_3(g_5)=3$, we know that $g_1 g_5$ ends at EDG_2 and EDG_3 of $\mathcal{S}_3$. And, since $T_3(g_1)=1$, we can immediately conclude that $g_1 g_5$ is closed according to Theorem 3. Note that the process can be also used to perform closure checking/pruning. That is, once a candidate p-signature is found not closed, any p-signature containing it can be removed according to Theorem 4. Based on the bitmap representation, closure checking/pruning can be performed efficiently without recording any previously discovered patterns.

3.2.3 *Quality test of block combinations*

A phenotype structure is a combination of blocks. Given the valid block derivation as mentioned above, the next step is to develop an efficient block-combination test strategy to find the one with maximal quality score as in Eq.(5). Clearly, it is intractable to enumerate all block combinations. Thus, we adopt a heuristic framework to tackle the problem.

Our method can be treated as a box-filling process: Given m label-unknown samples together with some initially empty boxes, we want to fill the boxes with the samples, so that every sample is put into a box and the samples in a box are homogenous in the sense that they could be characterized by a common p-signature. Since the class label of every sample is unknown, we adopt a progressive exploring methodology. Specifically, we first try whether s_1, $s_1 s_2$, ..., $s_1 s_2 \cdots s_m$

should be put into the first box in order. Then, in a similar manner, recursively fill the other boxes using the remaining samples. During the process, the projection divergence PD based cross-projection can be used to evaluate the quality of the current box-filling state. Thus, if we try all possible box-filling states without any pruning, the process can be treated as searching through a sample enumeration tree as shown in Fig. 8, which corresponds to the reduced dataset in Fig. 6.

In the enumeration tree, each node corresponds to a unique sample combination, under which the corresponding closed p-signatures are listed. Any possible box-filling state could find a counterpart in the tree, that is, a combination of some nodes together with one of the listed closed p-signatures. Theorem 5 ensures that all closed p-signatures can be discovered from the enumeration tree.

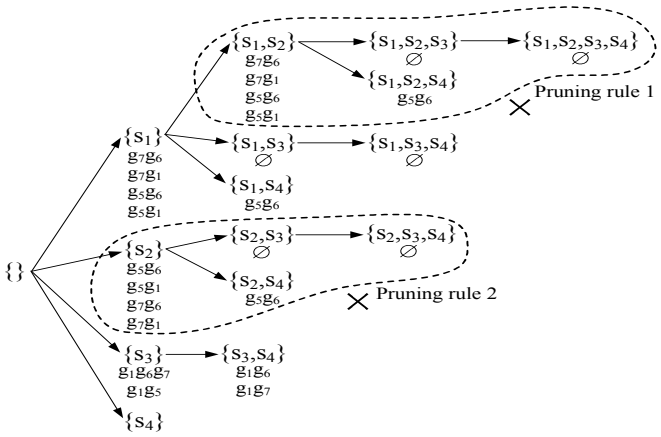


Fig. 8. The enumeration tree for the dataset in Table 1

Algorithm 1: Discover The Phenotype Structure

Input: a preprocessed expression matrix of n samples, SDB

Output: the phenotype structure, Result $=(\mathbb{S}, \mathbb{G})$

Q_{best}=0;
for i=1 **to** n **do**
  X=$\{s_i\}$; $candiX$=$\{s_{i+1}, s_{i+2}, \ldots, s_n\}$;
  `Recursive_Search` (X, $candiX$, Q_{best});
`Output` (Result);

Function: `Recursive_Search`(X, $candiX$, Q_{best})
$cResult$=$\varnothing$;
compute p-signatures contained by X;
derive the blocks at X;
if *(pruning rule 2 or 3 is activated)* **then return**;
prune $candiX$ by rule 1;
while ($|S|$>Min_s) **do**
  select a representative block $(\mathbb{S}, p)$ as described in section 3.2.3;
  $cResult$=$cResult$+$(\mathbb{S}, p)$;
  S=S−$\mathbb{S}$;
if (Q_c>Q_{best}) **then**
  Q_{best}=Q_c;
  handle outliers;
  Result =$cResult$;
if ($candiX$≠$\varnothing$) **then**
  $candiX$=$candiX$−$\{s_k\}$ s.t. k=$min\{j|j$∈$candiX\}$;
  `Recursive_Search` (X∪$\{s_k\}$, $candiX$, Q_{best});

Theorem 5: Let S be the set of all samples, and $\mathbb{S}_i$ be a subset of S. For a p-signature, p_i, of $\mathbb{S}_i$, if it is closed in $\mathbb{S}_i$, it must also be closed in S.

Proof: Suppose p_i is closed in $\mathbb{S}_i$ but not closed in S. Then, there must exist a pattern $p_i' \supset p_i$ such that any sample containing p_i in S also contains p_i'. However, this contradicts the assumption. Hence the proof. □

Two greedy combination strategies, *aggressive greed* and *progressive greed*, can be integrated into the searching process. In what follows, we first introduce the basic ideas of the two strategies. Pruning strategies that can be used to further reduce the search space will also be discussed.

Aggressive Greed: This approach is based on the intuition that the best individual blocks constitute the best combination. The PD based cross-projection is used to guide the block selection. The general idea is to make the block selected in each step as distinctive as possible. Specifically, the first block is selected based on the value of $\frac{\sum_{\forall s \in \mathbb{S}-\mathbb{S}_i} PD(p_i, p_i|_s)}{|\mathbb{S}-\mathbb{S}_i|}$. That is, the block whose p-signature has the maximum average PD to its projections on the remaining samples is selected; The remaining blocks are selected based on the value of $\sum \mathfrak{B}(i,j)$ (Eq. (6)), where j is the index of the block to be selected and i is the index of any block having been selected. The block with the maximum average difference on $\mathfrak{B}(i,j)$ will be selected.

In this approach, each block will be examined only once. That is, once block i is determined in step i, it will remain unchanged. This strategy heavily depends on the quality of the first selected block. If a bad block is selected in the first, the remaining selections will be based on this block.

Progressive Greed: To address the limitation of the previous approach, the progressive greed approach allows to update a previously selected block by a new block if such an update can improve the quality of the block combination.

During the tree traversal, for the current node X, we derive the most distinctive block $(\mathbb{S}_i, p_i)$ from it. Then, we remove $\mathbb{S}_i$ from the complete sample set S and search the remaining sample set $S-\mathbb{S}_i$ to seek the next block $(\mathbb{S}_j, p_j)$ such that $\mathfrak{B}(i,j)$ is maximum while $\mathbb{S}_i \cap \mathbb{S}_j$ is minimum. This ensures to select the block with the maximum average difference and the minimum overlap with the selected blocks. The process proceeds recursively until every sample is assigned to a block. When such a block combination is obtained, it is considered as a candidate. Instead of immediately returning this candidate as the result, we track back to node X and continue searching the remaining tree branches to generate new candidates in a similar way. During the traversal, we always keep track of the current best result and its quality score Q_{best}. Once a new candidate is generated, we compare its quality score, Q_c, with Q_{best}, and update Q_{best} to Q_c if $Q_c > Q_{best}$. Experimental results show that this method greatly improves the quality of the results due to the quality-guaranteed block updating way. Algorithm 1 formalizes the progressive heuristic.

Next, we introduce several effective pruning techniques to further reduce the search space.

Pruning Rule 1: Let X be a node in the enumeration tree, where a set of blocks, $\{(\mathbb{S}_1, p_1), (\mathbb{S}_2, p_2), \ldots, (\mathbb{S}_k, p_k)\}$ are derived. Let $Y=\{s_i | s_l \prec_{ORD} s_i$ and $s_i \in \bigcap_{i=1}^{k} \mathbb{S}_i\}$, where s_l is the sample of the lowest order in X and $s_l \prec_{ORD} s_i$ means the order of s_i is lower than that of s_l in enumerating. The subtree rooted at node $X \cup s_i$ can be completely pruned if $s_i \in Y$.

Proof: Let $X'=X \cup Z$ be any descendant node of X, and $Y'=Y \cap Z$ be a non-empty sample subset. Then, if we select a block at node X', it must be a duplication of the choice made on node $X'-Y'$, which is also a descendant node of X. This is because $Y' \subseteq Y$ and any sample in Y is common to all blocks derived at X. Thus, we can remove the samples in Y from consideration. Hence the proof. □

For example, let X be node $\{s_1\}$ in Fig. 8. According to pruning rule 1, $Y=\{s_2\}$. Thus, the subtree rooted at node $\{s_1 s_2\}$ can be completely pruned. This is because any block derived along this branch can be found in some later branch.

After applying the above rule, $s_i \in Y$ will be removed to a set $P(X)$, which is useful in some other pruning rule.

Pruning Rule 2: Let X be a node in the enumeration tree, where a set of blocks, $\{(\mathbb{S}_1, p_1), (\mathbb{S}_2, p_2), \ldots, (\mathbb{S}_k, p_k)\}$ are derived, and let $Y=\bigcap_{i=1}^{k} \mathbb{S}_i$. We can prune the entire subtree rooted at node X if there exists a sample s_i that satisfies: (1) $s_i \in Y$; and (2) $s_i \notin X$; and (3) $s_i \prec_{ORD} s_l$, where s_l is a sample of the lowest order in X; and (4) $s_i \notin P(X')$, where X' is any ancestor of X;

Proof: The existence of s_i, which satisfies (1), indicates that the set of blocks derived at X is the same as that derived at another node $X \cup s_i$. Further, condition (2) and (3) implies node $X \cup s_i$ occurs before X. Condition (4) ensures that the same result has ever been explored. Hence the proof. □

For example, let X be node $\{s_2\}$ in Fig. 8. According to pruning rule 2, s_1 is the sample satisfying all four conditions. Therefore, the subtree rooted at node $\{s_2\}$ can be completely pruned. This is because any block derived along this branch must have ever been discovered in some previous branch.

Pruning Rule 3: Let $\mathbb{S}$ be the set of all samples. Suppose a block $(\mathbb{S}_1, p_1)$ is currently selected as the first block candidate and Q_{best} is the quality of the best candidate result obtained so far. Then, any combination taking $(\mathbb{S}_1, p_1)$ as the first block can be pruned if $\max(PD_1, PD_2) \leq Q_{best}$, where PD_1 and PD_2 are expressed as Eq.(7) and Eq.(8), respectively.

$$PD_1 = \max_{\forall s \in \mathbb{S} - \mathbb{S}_1} PD(p_1, p_1|_s), \tag{7}$$

$$PD_2 = \max_{\substack{\forall i, \mathbb{S}_i \cap \mathbb{S}_1 = \varnothing \\ \forall s \in \mathbb{S} - \mathbb{S}_i}} PD(p_i, p_i|_s). \tag{8}$$

Proof: Given $(\mathbb{S}_1, p_1)$, the first block selected by a block combination test. Let $(\mathbb{S}_i, p_i)$ be any other block, which may occur in the same combination together with $(\mathbb{S}_1, p_1)$ and m_i be the number of samples in $\mathbb{S}_i$. Then, according to Eq.(6), we can derive the upper bound, Q_{u_1}, of $\mathfrak{B}(1, i)$. That is, $Q_{u_1} \leq \frac{m_2 PD_1 + m_1 PD_2}{m_1 + m_2} \leq \max(PD_1, PD_2)$. Similarly, $\forall i \neq j$ $(i, j \neq 1)$, the upper bound, Q_{u_2}, of $\mathfrak{B}(i, j)$ is also $\max(PD_1, PD_2)$. The conclusion can be drawn from Definition 5. □

If the first block is not distinctive, the quality of a phenotype structure may not be high . Thus, once the first block, $(\mathbb{S}_1, p_1)$, is selected, we can prune all combinations that contain $(\mathbb{S}_1, p_1)$ with the low upper bound of quality ($\leq Q_{best}$).

Fig. 8 illustrates the search which can be pruned by the corresponding rules.

3.3 Refinement

To avoid explicitly setting the number of blocks, FINDER uses Min_s as a terminal condition to stop the block combination test. A small number of samples may not be assigned to any block. Also, due to the heuristic block selection strategy, some samples may be assigned to multiple blocks.

Such cases can be dealt with by reassigning those samples to the current result according to certain criterion. In our solution, we break every current p-signature into the smaller fragments. Then, a sample is reassigned by combining the decisions from all fragments. The process is treated as a voting based on PD and the cross-projection. That is, for a sample s to be reassigned, we project the fragments of every block onto it and compute the average projection distance PD_{avg}. s is assigned to the block with minimum PD_{avg}.

Since the originally discovered p-signatures are closed sequences, the generated fragments should be sequence generators [21]. The intuition behind our solution is the Occam's razor principle [22] and the fact that a sequence generator and a closed sequence containing it characterize the same sample set [21]. Next, a top-down recursive process is given to break a p-signature into the smaller fragments (sequence generators).

Suppose that p_i is a closed p-signature. We first generate all its immediate sub-patterns, $p_{i_1}, p_{i_2}, \ldots, p_{i_n}$, by removing a single item from p_i, respectively. We then compare the supports of p_i and p_{i_x} for all $x \in [1, n]$. If the support of p_{i_x}, i.e., the number of samples containing p_{i_x}, is larger than that of p_i, i.e., $supp(p_{i_x}) > supp(p_x)$, we remove p_{i_x} and all its immediate sub-patterns from considering. Otherwise, we recursively continue the process for p_{i_x}. The patterns that can not be further reduced are left as the final fragments.

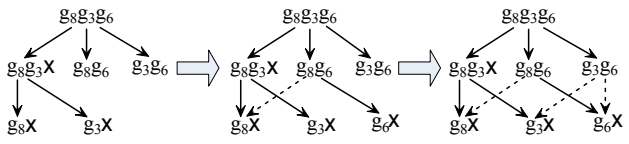


Fig. 9. Refine $g_8 g_3 g_6$

Example 3: Suppose that $g_8 g_3 g_6$ is a given original p-signature. The process discussed above can be illustrated by Fig. 9. The dataset in Fig. 3 is used as the sample dataset. First, $g_8 g_3$, $g_8 g_6$ and $g_3 g_6$ are generated by removing g_6, g_3 and g_8, respectively. Then, $g_8 g_3$ together with its two immediate sub-patterns, i.e., g_8 and g_3, is removed since $supp(g_8 g_3)=4 > supp(g_8 g_3 g_6)=3$. For $g_8 g_6$, since g_8 has been removed, we only need to compare the supports of g_6 and $g_8 g_6$, and thus g_6 is removed. Finally, for $g_3 g_6$, since both g_3 and g_6 have been removed and there is no further way to break the sequences, $g_8 g_3 g_6$ is broken into $g_8 g_6$ and $g_3 g_6$. During

the support comparison process, the bitmap based method (see Fig. 7) can be also used to improve the efficiency.

4 Performance Evaluation

We study the performance of FINDER by evaluating its efficiency and effectiveness. The algorithms are coded in C++. All experiments are conducted on a 2.0-GHz HP PC with 1G memory running Window XP. Both real and synthetic datasets are used in the experiments.

TABLE 3
The information of three real microarray datasets

dataset	# sample	# gene	class$_1$: # class$_1$	class$_2$: # class$_2$	class$_3$: # class$_3$
Colon	62	2000	negative:40	positive:22	N/A
Leukemia	38	5000	B-ALL:19	T-ALL:8	AML:11
HBC	22	3326	BRCA1:7	BRAC2:8	Sporadic:7

Real datasets: We use the clinical data on colon tumor [9], ALL-AML leukemia [7] and Hereditary Breast Cancer (HBC) [23]. Table 3 shows the statistics of these three datasets: the number of samples (# sample), the number of genes (# gene), the label of class i (class$_i$) and the number of samples in class i (# class$_i$).

Synthetic datasets: The synthetic data generator takes the following parameters: (1) k, the number of "blocks" in a phenotype structure; (2) MAX_s and MIN_s, the maximum and minimum numbers of samples in a "block"; (3) MAX_g and MIN_g, the maximum and minimum numbers of genes in a "block"; and (4) N_s and N_g, the number of samples and genes. The synthetic datasets are first initialized with random values. A number of submatrices are then embedded by setting MIN_s=5, MAX_s=0.6×N_s, MIN_g=10, and MAX_g=0.02×N_g, where N_s varies from 25 to 45, and N_g varies from 1000 to 3000.

Unless otherwise specified, the default parameters setting for FINDER are δ=0.3, Min_s=0.3, Max_s=0.5.

4.1 Efficiency

We evaluate the efficiency of FINDER by studying how response time varies with respect to $\#sample$ and $\#gene$, and how response time varies with respect to Min_s, Max_s and δ. Since no previous work can be directly applied to the problem setting in this paper, we implemented a naive two-step method as the baseline method. First, all candidate p-signatures are mined using BIDE [20], one of the state-of-the-art closed sequence mining algorithm; Second, do an exhaustive combination test over all derived blocks. Two greedy strategies proposed in this paper are also implemented, which are called A-FINDER (aggressive approach) and P-FINDER (progressive approach), respectively.

4.1.1 Scalability

Before the phenotype structure discovery step, all trivial g^*-sequences will be identified and genes with no discriminative power will be removed. We first evaluate the efficiency of the trivial pattern pruning (TPP in short) step. From Figures 10(a) and 10(b), we see that TPP's running time increases as $\#gene$ increases and decreases as $\#sample$ increases. This is because increasing $\#gene$ may result more and longer p-signatures. As $\#sample$ increases, less and shorter p-signatures will be generated. In Figure 10(c), TPP's running time decreases as δ increases. This is intuitive since increasing δ often produces the larger EDGs. Thus less and shorter p-signatures are generated and tested in TPP.

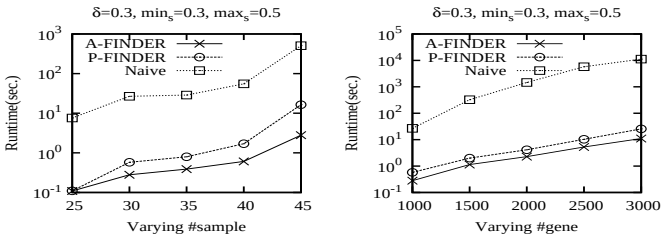


Fig. 14. Scalability of PSD

Next, we evaluate the scalability of phenotype structure discovery (PSD in short) step using the synthetic datasets. In Figure 14, PSD's running time becomes longer as $\#sample$ and $\#gene$ increases. This is because larger $\#sample$ may lead to more sample combinations to be tested and the increasing of $\#gene$ makes the number of EDGs in every g^*-sequence larger. Note that FINDER is *two or three orders of magnitude* faster than the naive method. The reason is two-fold. First, due to the template-driven pattern growth method, we do not need to create and scan any projected database. Closure checking can also be efficiently done based on the head-tail matrix and the bitmap based block derivation. Second, during block combination test, effective pruning strategies are applied.

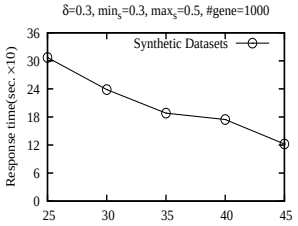
4.1.2 Effects of the parameters

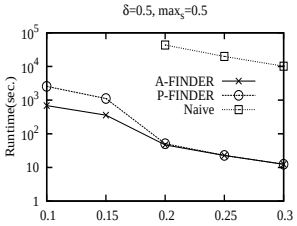
In Fig. 11, the response time decreases as Min_s increases. This is intuitive since smaller "blocks" are removed with larger Min_s. In Fig. 12, the response time is hardly affected by increasing Max_s from 0.5. This indicates that "blocks" with too many samples rarely occur in the real datasets. The result also indicates that the phenotype structure defined by the sequence model does exist in the real data and their sizes are relatively stable. In Fig. 13, the response time increases as δ increases. This is because larger δ leads to more genes left after TPP, thus more EDGs remain. Note that FINDER is always 2∼3 orders of magnitude faster than the naive method.

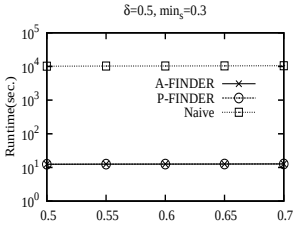
In Fig. 11∼13, we also observe that A-FINDER and P-FINDER have similar efficiency performance. Later we will show that P-FINDER is more accurate than A-FINDER.

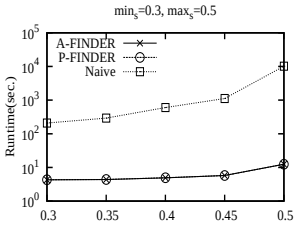
4.2 Effectiveness

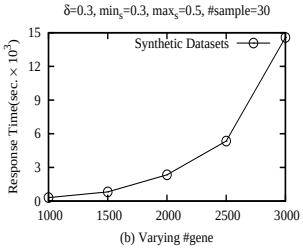
The effectiveness of FINDER is evaluated from both theoretical and practical aspects. From the theoretical aspect, we compare FINDER with two representative unsupervised phenotype structure discovery methods, i.e., ESPD [4] and HARP [6]. Moreover, we conducted a specific set of experiments to show the superiority of PD w.r.t other simple dissimilarity measures (e.g edit distance). From the practical aspect, we

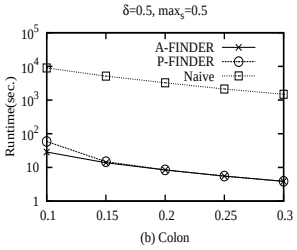


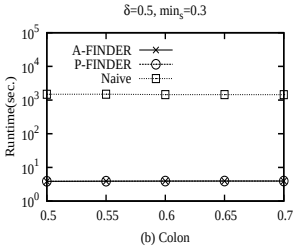


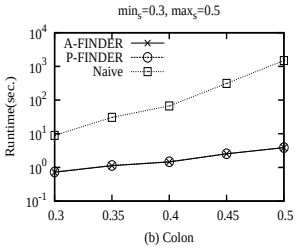


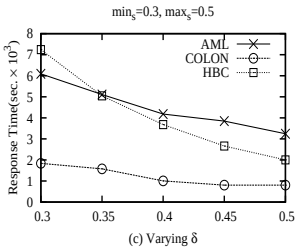


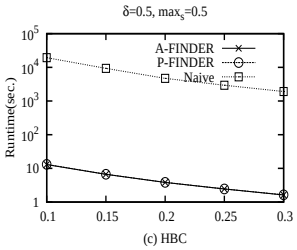


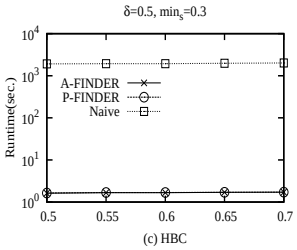












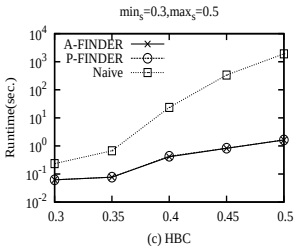


Fig. 10. Scalability of TPP Fig. 11. Varying Min_s Fig. 12. Varying Max_s Fig. 13. Varying δ

show that the genes selected by FINDER are indeed relevant to the considered phenotypes. Note that microarray are highly noisy. The identified pattern may be caused by random noises. Therefore, we further check the statistical significance (p-values) of the discovered phenotype structures.

4.2.1 Accuracy and Statistical Significance Evaluation

Phenotype structure discovery involves two key elements, i.e., the partition of samples and the selection of genes. Correspondingly, we conducted two sets of experiments to compare FINDER with ESPD and HARP. In the first set of experiments, precision, recall and accuracy are used to evaluate the correctness of the partition of samples, the computations of which follow a common evaluation framework proposed in [24]. In the second set of experiments, we use gene selection rate (GSR, the ratio between the number of the selected genes and the total number of genes) to evaluate the succinctness of the selected genes. Moreover, we calculate the p-value (determined by hypergeometric test) for each block within a discovered phenotype structure given the selected genes. The results are shown in Tables 4∼6.

Let $H=\{H_1, H_2, \ldots, H_k\}$ be the "true" clusters of a given data set, and $C=\{C_1, C_2, \ldots, C_l\}$ be the found clusters by a specific algorithm on the same data set. For each H_i, $i\in[1,k]$, we determine C_j, $j\in[1,l]$, with which H_i shares the largest number of samples. The precision and recall of C_j is defined as the number of samples common to C_j and H_i divided by the total number of samples in C_j and H_i, respectively.

As for accuracy, [24] indicates that the accuracy of classification specified by $\frac{correctlypredictedobjects}{allobjects}$ can be used to judge the clustering quality. Concretely, each sample s is denoted as a bitvector of length l if l clusters $C_1, C_2, \ldots, C_l$ is found. The j-th bit in the bitvector equals 1 if s contains p_j, the p-signature of C_j; otherwise 0. Then, we use SVM to induce models on the binary feature representation, and estimate the classification accuracy via ten-fold cross-validation.

As can be seen from the tables, FINDER significantly improves the precision, recall and accuracy with much smaller GSR. For example, in Colon dataset, P-FINDER improves the accuracy from 54.8% (HARP) and 53.2% (ESPD) to 88.7%, while reducing GSR from 91.9% (HARP) and $15‰$ (ESPD) to $1‰$. Note that ESPD and HARP are both combination discriminability-based methods. The result confirms the intuition that permutation provides more information than combination, since the former disclose not only the co-occurrence of the genes but also the ordering relationship among them. The decrease of GSR can help to reduce the cost of subsequent biological validation of selected genes [17].

Given GSR, we use the hypergeometric distribution to calculate the p-value for each block of a phenotype structure. A p-value indicates the probability that a phenotype structure is formed by chance. Specifically, it is computed as follows:

$$p = 1 - \sum_{i=0}^{k-1} \frac{\binom{M}{i}\binom{m-M}{t-i}}{\binom{m}{t}} \tag{9}$$

In the above equation, m is the total number of samples in a given dataset, and M is the number of samples annotated to a particular phenotype. Eq.(9) calculates the probability that seeing at least k samples annotated to that particular

TABLE 4
The correctness of the partition of samples

Dataset		ESPD			HARP			A-FINDER			P-FINDER		
		precision	recall	accuracy	precision	recall	accuracy	precision	recall	accuracy	precision	recall	accuracy
Colon	positive	37.9%	50%	53.2%	37.5%	40.9%	54.8%	76%	86.4%	85.5%	80%	90.9%	88.7%
	negative	66.7%	55%		65.8%	62.5%		91.2%	85%		94.6%	87.5%	
Leukemia	B-ALL	69.2%	47.4%	60.5%	69.2%	47.4%	55.3%	100%	63.2%	76.3%	100%	94.7%	97.4%
	T-ALL	30.8%	50%		66.7%	25%		87.5%	87.5%		88.9%	100%	
	AML	83.3%	90.9%		45.5%	90.9%		55.6%	90.9%		100%	100%	
HBC	BRAC1	57.1%	57.1%	50%	40%	28.6%	40.9%	71.4%	71.4%	68.2%	85.7%	85.7%	81.8%
	BRAC2	57.1%	50%		42.6%	37.5%		62.5%	62.5%		75%	75%	
	Sporadic	37.5%	42.9%		40%	57.1%		71.4%	71.4%		85.7%	85.7%	
average		55%	55.4%	54.6%	50.9%	48.7%	50.3%	77%	77.3%	76.7%	88.7%	89.9%	89.3%

TABLE 5
GSR and p-value comparison

Dataset		ESPD			HARP			A-FINDER			P-FINDER		
		GSR		p-value	GSR		p-value	GSR		p-value	GSR		p-value
Colon	positive	15%	15%	0.455	78.6%	91.9%	0.501	1.5‰	2.5‰	4.115e-08	1‰	1‰	1.045e-09
	negative	15%		0.455	22.5%		0.501	1.5‰		4.115e-08	1‰		1.045e-09
Leukemia	B-ALL	6‰	6‰	0.085	46.8%	86.7%	0.085	0.6‰	1.8‰	1.861e-05	0.6‰	1.8‰	5.658e-10
	T-ALL	6‰		0.257	45.2%		0.106	0.8‰		4.928e-06	0.8‰		1.84e-07
	AML	6‰		1.436e-06	35.4%		9.184e-03	0.6‰		7.537e-04	0.8‰		8.31e-10
HBC	BRAC1	15.5‰	15.5‰	0.107	44.6%	89.4%	0.523	0.9‰	2.5‰	1.355e-02	0.9‰	2.2‰	6.215e-04
	BRAC2	15.5‰		0.182	50.1%		0.523	0.9‰		7.207e-02	0.9‰		8.321e-03
	Sporadic	15.5‰		0.51	38.9%		0.51	0.9‰		1.355e-02	0.9‰		6.215e-04
average		11.8‰	12.2‰	0.2564	45.3%	89.3%	0.384	0.9‰	2.3‰	1.25e-02	0.9‰	1.7‰	1.200e-03

TABLE 6
The performance of edit distance

Dataset		A-FINDER with ED						P-FINDER with ED					
		precision	recall	p-value	GSR		accuracy	precision	recall	p-value	GSR		accuracy
Colon	positive	68%	77.3%	1.471e-05	4.5‰	5.5‰	71.0%	73.1%	86.4%	1.412e-07	1.5‰	2.5‰	79.0%
	negative	73%	67.5%	7.779e-02	3.5‰			83.3%	75%	3.387e-04	2‰		
Leukemia	B-ALL	78.6%	57.9%	8.520e-03	1.2‰	3‰	63.2%	86.4%	57.9%	2.569e-03	1‰	2.4‰	68.4%
	T-ALL	80%	50%	4.295e-03	1.6‰			83.3%	62.5%	6.187e-04	1.4‰		
	AML	47.4%	81.8%	1.465e-02	1.4‰			52.6%	90.9%	1.521e-03	1.2‰		
HBC	BRAC1	50%	57.1%	0.182	2.1‰	3.9‰	54.5%	57.1%	57.1%	0.107	1.8‰	3.3‰	59.1%
	BRAC2	66.7%	50%	9.626e-02	1.8‰			71.4%	62.5%	3.223e-02	1.5‰		
	Sporadic	50%	57.1%	0.182	2.1‰			80%	57.1%	2.073e-02	1.8‰		
average		64.2%	62.3%	7.069e-02	2.3‰	4.1‰	62.9%	73.4%	68.7%	2.063e-02	1.5‰	2.7‰	68.8%

TABLE 7
The genes discovered from Leukemia dataset

Gene	Rank							
	t-test	Information gain	Sum of variances	Twoing rule	Gini index	Sum minority	Max minority	1D SVM
MB-1*	4	18	26	26	26	41	34	21
CST3*	49	4	3	3	3	2	2	4
MacMarcks*	19	38	29	29	29	21	13	27
TCL1*	42	30	61	61	61	>100	>100	>100
IGHM*	69	>100	>100	>100	>100	>100	83	>100
TCRB	>100	>100	>100	>100	>100	>100	>100	>100
GUK1	>100	>100	>100	>100	>100	>100	>100	>100
GLUL	>100	>100	>100	>100	>100	>100	>100	>100
ER-60	>100	>100	>100	>100	>100	>100	>100	>100

phenotype in randomly chosen t samples. This approach is widely used to evaluate the statistical significance of the result in many existing tools, such as Gene Ontology[1] and GO TermFinder[2]. A smaller p-value indicates a stronger statistical significance. If most of the blocks of a phenotype structure are of small p-values, the phenotype structure is unlikely formed by chance. As can be seen from Table 5, the phenotype structures discovered by FINDER are of very small p-values.

1. http://www.geneontology.org
2. http://search.cpan.org/dist/GO-TermFinder/

To show the power of the ordered gene expression values in the phenotype structure discovery more clearly, we visualize the phenotype structures discovered from the three real datasets in Fig. 15(a)∼15(c), where the strength of gene expression is mapped into the darkness of color. The stronger the gene expresses, the darker the color is. The gene orders (p-signatures) and the sample labels are given at the top and the left of every block, respectively. '$*$' marks the samples not properly grouped. Clearly, in each block of a phenotype structure, the mapped expression values are always from lightness to darkness. The order among genes can be used

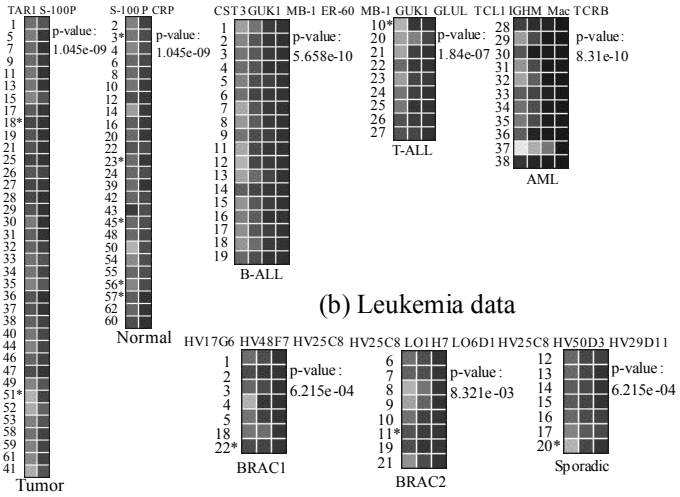


Fig. 15. The result visualization

to discover the phenotype structures of statistical significance.

As ever mentioned, PD is more suitable than ED in our task. In Table 6, a set of experiments is conducted to show this. For simplicity, we call A-FINDER (resp. P-FINDER) with edit distance as A-FINDER* (resp. P-FINDER*). By comparing the results with that in Table 4 and 5, it is not difficult to see that the performances of A-FINDER* and P-FINDER* are better than that of ESPD and HARP but worse than that of A-FINDER and P-FINDER. The former could be intuitively explained in such a way that the sequence model discloses not only the co-occurrence of the genes but also the ordering information among them. Thus, more information could be exploited by this model. As for the latter, it could be explained by three differences between edit distance and projection divergence. *First*, ED and PD have different application scenarios. In ED, the three basic operations, i.e. insertion, deletion and substitution, correspond to three possible mutational events during evolution. That is, they have specific biological significance. Thus, ED is more suitable for the biological data such as DNA or Protein sequences. In our case, since no any evolutional mutation is involved, ED is not suitable; *Second*, ED measures the dissimilarity between sequences by only combining the difference on individual items while PD concerns the problem from the interrelation among items. Thus, PD may uncover difference even when ED fails; *Third*, unless user specifying, ED assigns each operation with equal differentiability weight, which is unrealistic. However, PD can naturally utilize $[dist_x(p_i, s)+dist_y(p_i, s)]$ as an intuitive coefficient to weight the differentiability of x and y.

4.2.2 Biological Significance of the Discovered Patterns

Different from the existing methods, FINDER characterizes the phenotype structure from a sequence point of view. It incorporates the interrelationship among genes. Some genes ignored by the previous methods may play an important role in the disease phenotype. In this part, we present some interesting results discovered by FINDER from the Leukemia dataset [7] and show that FINDER is able to find not only the genes identified by the existing methods, but also some important genes ignored by the existing methods.

Table 7 lists all genes involved in the phenotype structure discovered from the Leukemia dataset. The results from eight statistics based gene ranking methods [25] are used as the benchmark. If a gene is ranked within top-100 by two or more traditional methods, it is marked with '$*$'. For the traditional methods, the higher ranked genes are often considered more interesting. As shown in Table 7, genes MB-1, CST3 and MacMarcks are top-ranked genes by all eight methods. They are also discovered by FINDER. MB-1 gene encodes the Ig-alpha protein of the B-cell antigen component. It is a sensitive and specific reagent for B-lineage blasts that will aid in the classification of B-cell precursor ALL and in the identification of biphenotypic leukemia presenting as AML [26]; Indicated by GENE[3], a searchable database of genes in NCBI, providing various detailed information for the studied genes, CST3 encodes the most abundant extracellular inhibitor of cysteine proteases, which is found in high concentrations in biological fluids and is expressed in virtually all organs of the body. A mutation in this gene is associated with amyloid angiopathy (e.g. AML); MacMarcks gene is proven to be immune-related [27]. Tumor is often immune-related, thus it is biologically plausible to find MacMarcks in the phenotype structure of Leukemia. Genes IGHM and TCL1 are identified by two and five methods in Table 7, respectively. As GENE states, IGHM is the antigen recognition molecule of B cells, which is involved in immune responses and Ag binding, so it is not surprising to relate IGHM to Leukemia; TCL1 is activated in T-cell leukemias by translocations and inversions that juxtapose it to regulatory elements of T-cell receptor genes, and activation of TCL1 in mature T-cells causes T-cell leukemia in humans [28].

For the genes without '$*$', although they are not top-ranked by the eight traditional methods, extensive biological evidences indicate that these genes are also related to leukemia. For example, TCRB is ranked outside top-100 in Table 7. However, TCRA is reported by five methods in Table 7 [25]. From the gene description in the Leukemia dataset [7], we know that the two are both T-cell receptors. They have very similar function. Moreover, GENE confirms that chromosomal abnormalities involving TCRB are closely associated with T-cell lymphomas. Also, we find two other interesting cases involved with the order among genes GUK1, GLUL and ER-60. That is, the gene sequence <MB-1 GUK1 GLUL> identifies T-ALL phenotype with precision=88.9% and recall=100%, and the gene sequence <CST3 GUK1 MB-1 ER-60> identifies B-ALL phenotype with precision=100% and recall=94.7%. From Table 5, we can see that the corresponding GSRs in the two cases are 0.8‰ and 0.6‰, respectively, while the p-values are $1.84e$-07 and $5.658e$-10, respectively. That is, although we select only a small set of genes, the number of which is far less than that selected by ESPD (6‰ and 6‰) and HARP (45.2% and 46.8%), the result discovered by FINDER is of much higher statistical significance than that discovered by ESPD (0.257

3. http://www.ncbi.nlm.nih.gov/gene

and 0.085) and HARP (0.106 and 0.085). The similar cases are also discovered in the other two real datasets, as shown in Table 5. It is the order among genes, which is ignored by singleton or combination discriminability based methods, that enables FINDER to discover the statistical significant phenotype structures with higher accuracy and fewer genes. Moreover, such order may provide a possible explanation to some diseases from a new point of view. For example, due to the small p-value, it is statistically reasonable to infer that the cause of T-ALL may be that gene GLUL expresses more than gene GUK1 and gene GUK1 expresses more than gene MB-1 in an individual. The similar case can be also found in the other results, as shown in Figure 15. Thus, the relationship between the ordered expressions among genes and the phenotype structure is worthy of further study.

5 Related work

In addition to the biclustering algorithms [11]–[14] discussed in Section 1, our work is also related to some previous work on sequential pattern mining and contrast data mining.

Sequential pattern was first introduced in [29]. Since then, many efficient algorithms to extract the full set of sequential patterns have been proposed, such as SPADE [30] , PrefixSpan [19] and SPAM [31], etc. Subsequently, to reduce the generation of an explosive number of subsequences, some methods mining only concise representations are proposed [20], [21], [32]. Moreover, to incorporate some user-specific interests, a number of constraint-based sequential pattern mining algorithms were also presented [33], [34]. However, the extremely high dimensionality of microarray data often makes it difficult to directly apply these methods on microarray data analysis.

Contrast data mining aims to mine patterns and models distinguishing different classes/conditions [35]. Emerging pattern [36] and interesting rule groups [37] are two interesting contrast patterns applicable to microarray data analysis, where data are modeled as the set of co-occurrent items. However, the ordering information among items may be utilized to find patterns with more discriminant power. Correspondingly, sequential classification rules were introduced [34], which exploit the order among items for contrast analysis. In [38], Petra et al. unified the common contrast patterns into a framework named supervised descriptive rule discovery. However, this framework is available only when the class labels are given and often returns too much results.

As one of our recent work, we proposed a novel concept, non-redundant contrast sequence rule (NR-rule for short), and a related mining algorithm, NRMINER, in [39]. However, the work in [39] is quite different from that in this paper. This is because: (1) NR-rule makes sense only when the class labels are available, and thus NRMINER is an supervised learning algorithm. However, in this paper, we are interested in the phenotype structure discovery problem from an unsupervised perspective, where the class labels are assumed previously unknown. As mentioned in Section 1, the problem addressed in this paper is more challenging than that addressed in the previous work with known class labels; (2) the work in [39] only focuses on a set of sequence rules, where each rule has higher discriminative power than any of its subrules. However, in this paper, we aims to find a partition of samples such that each group of samples is characterized by a representative sequence pattern distinguishing this group from others. Since the cases considered in the two tasks are quite different, they are naturally different in spirits and details.

6 Conclusions

We model the phenotype structure discovery problem from a sequence perspective. Different from the existing methods, the proposed g^*-sequences model uses the ordered gene expression values as the discriminative signatures. It enables to find highly accurate phenotype structure with a small number of genes. We show the problem of phenotype structure discovery is NP-complete and develop a progressive exploring strategy to tackle the computational challenge. In the FINDER algorithm, a novel sequence dissimilarity measurement and a cross projection approach enable to try exploring candidate phenotype structures in a quality-guaranteed way. Various effective techniques are developed to further improve the efficiency. Extensive experimental results on real and synthetic datasets show that our method dramatically improves the accuracy of the discovered phenotype structure (in terms of statistical and biological significance) while using much less genes compared to the existing methods. Moreover, FINDER is $2\sim3$ orders of magnitude faster than the alternative methods.

Acknowledgments

Supported by 863 program (2012AA011004), National Science Fund for Distinguished Young Scholars (61025007), National Science Fund of China Key Program (60933001), National Natural Science Foundation of China (61272182, 61100028, 61173029, 61173030), New Century Excellent Talents (NCET-11-0085), China Postdoctoral Science Foundation (2012T50263, 2011M500568) and Fundamental Research Funds for the Central Universities (N110404005).

References

[1] S. Tavazoie, J. Hughes, M. Campbell, R. Cho, and G. Church, "Systematic determination of genetic network architecture," *Nat. Genetics*, vol. 22, pp. 281–85, 1999.

[2] M. Eisen, P. Spellman, P. Brown, and D. Botstein, "Cluster analysis and display of genome-wide expression patterns," *Proc. Natl. Acad. Sci. USA*, vol. 95, pp. 14 863–68, 1998.

[3] A. Alizadeh, "Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling," *Nature*, vol. 403, pp. 503–11, 2000.

[4] C. Tang, A. Zhang, and M. Ramanathan, "Espd: a pattern detection model underlying gene expression profiles," *Bioinformatics*, vol. 20, no. 6, pp. 829–838, 2004.

[5] J. R. Nevins and A. Potti, "Mining gene expression profiles: expression signatures as cancer phenotypes," *Nature Reviews Genetics*, vol. 8, no. 8, pp. 601–609, 2007.

[6] K. Y. Yip, D. W. Cheung, and M. K. Ng, "Harp: A practical projected clustering algorithm," *TKDE*, vol. 16, no. 11, pp. 1387–1397, 2004.

[7] T. R. Golub, D. K. Slonim, P. Tamayo, and et al., "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, pp. 531–537, 1999.

[8] J. Luo, D. J. Duggan, Y. Chen, and et al, "Human prostate cancer and benign prostatic hyperplasia: molecular dissection by gene expression profiling." *Cancer Res*, vol. 61, no. 12, pp. 4683–8, 2001.

[9] U. Alon, N. Barkai, D. A. Notterman, and et al., "Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays," *PNAS*, vol. 96, no. 12, pp. 6745–6750, 1999.

[10] M. Xiong, X. Fang, and J. Zhao, "Biomarker identification by feature wrappers," *Genome Research*, vol. 11, no. 11, pp. 1878–1887, 2001.

[11] J. Liu and W. Wang, "Op-cluster: Clustering by tendency in high dimensional space," in *ICDM*, 2003, pp. 187–194.

[12] Y. Cheng and G. M. Church, "Biclustering of expression data," in *ISMB*, 2000, pp. 93–103.

[13] X. Xu, Y. Lu, and A. Tung, "Mining shifting-and-scaling co-regulation patterns on gene expression profiles," in *ICDE*, 2006, pp. 89–100.

[14] A. Ben-Dor, B. Chor, R. M. Karp, and Z. Yakhini, "Discovering local structure in gene expression data: the order-preserving submatrix problem," in *RECOMB*, 2002, pp. 49–57.

[15] Q. Fang, W. Ng, and J. Feng, "Discovering significant relaxed order-preserving submatrices," in *KDD*, 2010, pp. 433–442.

[16] S. Dong, C. L. Nutt, R. A. Betensky, and et. al., "Histology-based expression profiling yields novel prognostic markers in human glioblastoma," *J Neuropathol Exp Neurol.*, vol. 64, no. 11, pp. 948–955, 2005.

[17] H. Liu and H. Motoda, *Computational Methods of Feature Selection*. Danvers, MA: Chapman & Hall/CRC, 2007.

[18] D. Zuckerman, "On unapproximable versions of np-complete problems," *SIAM Journal on Computing*, vol. 25, no. 6, pp. 1293–1304, 1996.

[19] J. Pei, J. Han, and et al., "Prefixspan: Mining sequential patterns by prefix-projected growth," in *ICDE*, 2001, pp. 215–224.

[20] J. Wang and J. Han, "Bide: Efficient mining of frequent closed sequences," in *ICDE*, 2004, pp. 79–90.

[21] D. Lo, S.-C. Khoo, and J. Li, "Mining and ranking generators of sequential patterns," in *SDM*, 2008, pp. 553–564.

[22] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*. New Jersey, USA: Addison Wesley, 2005.

[23] I. Hedenfalk, D. Duggan, Y. Chen, and et al., "Gene-expression profiles in hereditary breast cancer," *New England Journal of Medicine*, vol. 344, no. 8, pp. 539–548, 2001.

[24] E. Müller, S. Günnemann, I. Assent, and T. Seidl, "Evaluating clustering in subspace projections of high dimensional data," *PVLDB*, vol. 2, no. 1, pp. 1270–1281, 2009.

[25] Y. Su, T. M. Murali, V. Pavlovic, M. Schaffer, and S. Kasif, "Rankgene: identification of diagnostic genes based on expression data," *Bioinformatics*, vol. 19, no. 12, pp. 1578–1579, 2003.

[26] V. Buccheri and B. Mihaljevic, "mb-1: a new marker for b-lineage lymphoblastic leukemia," *Blood*, vol. 82, no. 3, pp. 853–857, 1993.

[27] S. Chang, K. Stacey, J. Chen, and et al, "Mechanisms of regulation of the macmarcks gene in macrophages by bacterial lipopolysaccharide," *J Leukoc Biol.*, vol. 66, no. 3, pp. 528–534, 1999.

[28] Y. Pekarsky, C. Hallas, and C. M. Croce, "The role of tcl1 in human t-cell leukemia," *Oncogene*, vol. 20, no. 40, pp. 5638–5643, 2001.

[29] R. Agrawal and R. Srikant, "Mining sequential patterns," in *ICDE*, 1995, pp. 3–14.

[30] M. J. Zaki, "Spade: An efficient algorithm for mining frequent sequences," *Machine Learning*, vol. 42, no. 1/2, pp. 31–60, 2001.

[31] J. Ayres, J. Flannick, and et al., "Sequential pattern mining using a bitmap representation," in *KDD*, 2002, pp. 429–435.

[32] X. Yan, J. Han, and R. Afshar, "Clospan: Mining closed sequential patterns in large databases," in *SDM*, 2003.

[33] B. Ding, D. Lo, J. Han, and S.-C. Khoo, "Efficient mining of closed repetitive gapped subsequences from a sequence database," in *ICDE*, 2009, pp. 1024–1035.

[34] M. J. Zaki, "Sequence mining in categorical domains: Incorporating constraints," in *CIKM*, 2000, pp. 422–429.

[35] J. Bailey and G. Dong, *Contrast Data Mining: Concepts, Algorithms, and Applications*. Danvers, MA: Chapman & Hall/CRC, 2012.

[36] G. Dong and J. Li, "Efficient mining of emerging patterns: Discovering trends and differences." in *KDD*, 1999, pp. 43–52.

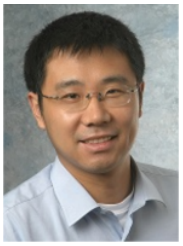
[37] G. Cong, A. K. H. Tung, X. Xu, and et al., "Farmer: Finding interesting rule groups in microarray datasets," in *SIGMOD*, 2004, pp. 143–154.

[38] P. K. Novak, N. Lavrac, and G. I. Webb, "Supervised descriptive rule discovery: A unifying survey of contrast set, emerging pattern and subgroup mining," *Journal of Machine Learning Research*, vol. 10, pp. 377–403, 2009.

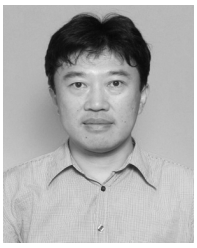
[39] Y. Zhao, G. Wang, Y. Li, and Z. Wang, "Finding novel diagnostic gene patterns based on interesting non-redundant contrast sequence rules," in *ICDM*, 2011, pp. 972–981.

Yuhai Zhao received his B.E., M.E. and Ph.D. in computer science, from Northeastern University, China, in 1999, 2004 and 2007, respectively. Currently he is an associate professor in the School of Information Science and Engineering, Northeastern University, China. He is a member of IEEE ACM, and a member of CCF. His major research interests include data mining and bioinformatics.

Guoren Wang received his BSc, MSc and PhD degrees, in computer science, from Northeastern University, China, in 1988, 1991 and 1996, respectively. Currently he is a professor in the School of Information Science and Engineering, Northeastern University, China. His major research interests are XML data management, query processing and optimization, high-dimensional indexing, parallel database systems, P2P data management and uncertain data management.



Xiang Zhang received his Ph.D. from the Department of Computer Science at the University of North Carolina at Chapel Hill in 2011. Currently he is an assistant professor in the Electric Engineering and Computer Science Department at Case Western Reserve University. His major research interests include graph mining, network analysis, high-dimensional data analysis, bioinformatics and database.



Jeffrey Yu Xu Jeffrey Xu Yu received his B.E., M.E. and Ph.D. in computer science, from the University of Tsukuba, Japan, in 1985, 1987 and 1990, respectively. He was a research fellow (Apr. 1990 – Mar. 1991) and a faculty member (Apr. 1991 – July 1992) in the Institute of Information Sciences and Electronics, University of Tsukuba, a Lecturer in the Department of Computer Science, Australian National University (July 1992 – June 2000). Currently he is a Professor in the Department of Systems Engineering and Engineering Management, the Chinese University of Hong Kong. His major research interests include data mining, data stream, XML query processing and optimization and graph database.

Zhanghui Wang Zhanghui Wang received his B.E.in computer science from Shengyang Institute of Aeronautical Engineering, China, in 2007, and received his M.E. in computer science from Northeastern University, China, in 2010. Currently he is a PH.D candidate in computer science, Northeastern University, China. His major research interests include data mining and bioinformatics.