A graph-theoretic modeling on GO space for biological interpretation of gene clusters

Sung Geun Lee, Jung Uk Hur and Yang Seok Kim

1 Bioinformatics Unit, ISTECH Inc., No 704, Hyundai Town Vill 848-1, Janghang-dong, Ilsan-gu, Goyang city, Gyunggido, 411-380, Republic of Korea and 2 Cancer Metastasis Research Center, Yonsei University College of Medicine, 134 Shinchon-dong, Seodaemun-gu, Seoul, 120-752, Republic of Korea

Received on February 22, 2003; revised on June 4, 2003; accepted on August 9, 2003
Advance Access Publication January 22, 2004

ABSTRACT

Motivation: With the advent of DNA microarray technologies, the parallel quantification of genome-wide transcriptions has been a great opportunity to systematically understand the complicated biological phenomena. Amidst the enthusiastic investigations into the intricate gene expression data, clustering methods have been the useful tools to uncover the meaningful patterns hidden in those data. The mathematical techniques, however, entirely based on the numerical expression data, do not show biologically relevant information on the clustering results. Results: We present a novel methodology for biological interpretation of gene clusters. Our graph theoretic algorithm extracts common biological attributes of the genes within a cluster or a group of interest through the modified structure of gene ontology (GO) called GO tree. After genes are annotated with GO terms, the hierarchical nature of GO terms is used to find the representative biological meanings of the gene clusters. In addition, the biological significance of gene clusters can be assessed quantitatively by defining a distance function on the GO tree. Our approach has a complementary meaning to many statistical clustering techniques; we can see clustering problems from a different viewpoint by use of biological ontology. We applied this algorithm to the well-known data set and successfully obtained the biological features of the gene clusters with the quantitative biological assessment of clustering quality through GO Biological Process.

Availability: The software is available on request from the authors.

Contact: sglee@istech21.com

INTRODUCTION

Over the past decade, DNA microarray technologies have been highlighted for their notable ability of parallel monitoring of the genome-wide transcriptional profiling. The gene expression data present both great challenges and successes. They serve as valuable clues to understand systematically the complicated genetic behaviors of life. Meanwhile, the underlying structures of genes reveal demanding complexity. With the fast progress of microarray technologies, their data analysis techniques have been intensively explored as well. Clustering has been a useful data-mining tool since early days, for discovering similar expression patterns without prior knowledge (Ben-Dor et al., 1999; Eisen et al., 1998; Tamayo et al., 1999; Tavazoie et al., 1999). Each clustering method has a chosen (dis)similarity measure and its own optimized algorithm to partition given numerical expression data into groups. Generally, different clustering algorithms yield different clustering results for the same data: the number of clusters and their constituents. It may be safely stated that the workability of each clustering method depends on the characteristics of given data and that diverse clustering techniques unveil various aspects of given data. Nonetheless, the overflowing clustering techniques can further confuse biologists, due to the lack of adequate standards for cluster validity.

There are many mathematical methods in the literature that can be employed for assessing the quality of clustering results (Azuaje et al., 2002; Halkidi et al., 2001; Tibshirani et al., 2000). For example, such numerical validation methods have been used to check the compactness of a cluster or to examine the clear separation between clusters. The performance of a clustering algorithm would be improved, if the algorithm could either minimize intracluster distance or maximize intercluster distance. Yeung et al. (2001) utilized the leave-one-out approach to assess the predictive power of clustering algorithms. However, these methods for numerical optimization do not include any biological considerations. The biological meanings of the results are therefore interpreted manually and this work can be time-consuming for large-scale data.

Several alternative approaches have been attempted: incorporating the biological knowledge of genes for supervised clustering (Dettling and Buhlmann, 2002), utilizing the MEDLINE database by use of MeSH keyword hierarchies (Masys et al., 2002). These methods have the added advantage of automatically integrating biological knowledge and this can provide a unique opportunity for the biologist to annotate our results. We present a novel methodology that benefits from both statistical and biological views by using biological ontologies to interpret biological results.
An example of GO codes from a part of GO text format. In the GO text file during our recent experiment—some part of the file is shown above in the left side—death (GO ID: 0016265) was the fifth children of biological process (GO ID: 0008150) whose GO code is 200000000000000; hence the GO code of death is 250000000000000. In the same manner, other GO terms can be easily coded as represented above.

et al., 2001), statistically evaluating gene/protein groups for particular attributes by existing annotations (Robinson et al., 2002), and proposing a figure of merit based on the functional annotation and cluster membership of each gene (Gibbons and Roth, 2002). They used the biological information of each gene, obtained either from text mining of the scientific literature or from the public database, for automatic assessment or interpretation of gene clusters. Although they provide good reference methodologies, mostly they emphasize either assessment or interpretation of gene clusters separately, in some cases without regard to the multi-functions of genes.

Here, we provide a novel algorithm to find the significant biological features of a gene cluster/group of interest through the modified structure of gene ontology (GO) called GO tree. Using the natural transformation of GO directed acyclic graph (DAG) structure with a distance function on it, our graph theoretic algorithm extracts common or representative GO terms for a gene cluster by taking multi-functionality of genes into account. Furthermore, a new quantitative measure is integrated for the biological assessment of gene groups through GO Biological Process.

**GRAPH MODELING ON GO SPACE**

**Gene ontology**

Every academic work starts from precise definitions of technical terms and develops from the coherent use of these terms. Nonetheless, in biology dealing with diverse organisms that have their own complicated mechanisms of life, the vocabulary has been used rather divergently from species to species. The GO Consortium was formed to converge the efforts to make the controlled vocabulary of various genomic databases about diverse species in such a way that it can show the essential features shared by all the organisms, especially the eukaryotes (Ashburner et al., 2000; The Gene Ontology Consortium, 2001).

**GO tree and GO code**

GO hierarchy is naturally described as a DAG (Ashburner et al., 2000; Fig. 1). GO has three ontology files corresponding to its three categories, namely molecular function, biological process and cellular component. From this hierarchy, an acyclic digraph can be easily obtained for each category with GO.
terms as nodes. The recognition of the GO hierarchical system as a digraph with top-down directions makes us easily catch the structure of the ontology. Nonetheless, to facilitate calculation, we will transform the original digraph of GO into our desired form, an ordered tree that is a directed tree with an order defined for the children of every node of the tree. GO DAG is not a directed tree since a GO term may have more than one parent. In other words, a GO term may have multiple paths from the root. Our aim is to construct an ordered tree from this hierarchy of GO by defining one or more GO code(s) to each GO term so that GO terms can be computationally manipulated in a tree structure (Fig. 1).

Note that the same GO term may occur in different lines of the ontology file. To build an ordered tree, GO terms should be distinguished from one another if they are placed in different lines of the ontology file. This may be justified from a biological viewpoint that in the gene ontology, what counts is not a GO term itself but which path the GO term takes from the root. Each appearance of a GO term is considered distinct if a distinct path leads to it from the root.

A GO code is assigned to a GO term in each line of the ontology files. A GO term is transformed into a GO code atid as using the unique path Γ from the top category (root) to the GO term, where \( H = H_0 + 1 \) and \( H_0 \) is the length of a longest path from the root to a GO term in the ontology file. The resulting graph is an ordered tree having GO codes as nodes and one of the three GO category names as the root. We will call this ordered tree as GO tree and we can obtain three GO trees from the three GO categories, respectively. In the following sections, we will say that a node is on the level \( N \) of GO tree for \( N = 1, 2, \ldots, H \), if the depth of the node is \( N - 1 \). Moreover, given two GO codes \( A \) and \( B \) such that \( \text{level of } A = m \) and \( \text{level of } B = n \) with \( m < n \), then we will say that \( B \) is on a lower level than \( A \), or the level of \( B \) is greater than that of \( A \).

**METRIC STRUCTURE OF GO TREE**

The goal is to measure to what extent a gene cluster/group is associated with known GO functional categories. For example, in Figure 2, in terms of biological hierarchy, how could you say that cluster Clr1 = \{B_1, C_1, C_3, D_1, E_1\} is better clustered than cluster Clr2 = \{C_2, C_3, D_2, D_3, D_4\} or vice versa? We need an adequate measurement for this. The concept of usual distance \( d(x, y) \) between two nodes \( x \) and \( y \), i.e., the length of the unique path between the two nodes in GO tree, is not appropriate to use. In Figure 2, for example, \( d(B_1, B_2) = d(B_1, D_1) = 2 \) and \( d(B_1, B_2) = d(C_2, C_3) = d(D_2, D_3) = 2 \). Even if every pair of the two nodes above has the same path length, their relationships are quite different from each other. It is likely that \( B_1 \) and \( D_1 \) are more closely related than \( B_1 \) and \( B_2 \); similarly, \( C_2 \) and \( C_3 \) than \( B_1 \) and \( B_2 \); \( D_2 \) and \( D_3 \) than \( C_2 \) and \( C_3 \).

A weight function may be defined on \( E \), the edge set of GO tree \( T_G = (V_C, E) \). In defining a weight function, we make two fundamental assumptions on GO tree primarily for simplicity of modeling. First, the information of GO terms is more specific and more detailed on a lower level than on a higher level. Second, GO terms located at the same level contain equivalent level of information. With these assumptions, we will construct the metric structure of GO tree.

**Lowest common ancestor**

Lowest common ancestor (LCA) is an essential concept of our cluster analysis. Given a non-empty subset \( U \subseteq V_C \), where \( V_C \) is the set of nodes of GO tree \( T_G = (V_C, E) \), \( v \) is a common ancestor of \( U \) if every node in \( U \) is on a subtree of \( T_G \) having \( v \) as the root and \( v_0 \) is an LCA of \( U \) if \( v_0 \) is a common ancestor of \( U \) and the level of \( v_0 \) is greater than or equal to the level of \( w \) for any common ancestor \( w \) of \( U \). As seen intuitively, the existence and uniqueness of the LCA of any subset of \( V_C \) can be easily proved. For example, in Figure 2, if \( U_1 = \{C_1, C_3, D_2\}, U_2 = \{C_3, E_4, E_6\} \) and \( U_3 = \{C_3, D_2, D_3, E_5, E_7\} \), then the LCAs of \( U_1 \), \( U_2 \) and \( U_3 \) are \( A_1, C_3 \) and \( B_2 \), respectively.

**Principal distance**

In this section, we will define a metric on GO tree to measure ‘the closeness’ between two GO terms. First, a unique positive real number is assigned to each level of \( T_G \). Let \( H_0 \) be the height of \( T_G \) and let \( H = H_0 + 1 \). Suppose that \( W: I_H \to R^+ \) is a function such that \( W(i) > W(i + 1) \) where \( I_H = \{1, 2, 3, \ldots, H\} \). The weight of level \( i \) is then defined as \( W(i) \).

In our present modeling of GO tree, \( H = 15 \) and \( W(k) = 150 - 10(k - 1) \) for \( k \in I_H \). Hereafter, given a GO code \( v_i \) in

---

**Fig. 2. Metric relationship of GO.** The levels of \( A_i, B_i, C_i, D_i \) and \( E_i \) are 1, 2, 3, 4 and 5, respectively.
we will use $W(v_i)$ in place of $W$ of level of $v_i$ for notational convenience.

Suppose that $v_1$ and $v_2$ are two nodes in $T_G$. Then we define principal distance $Pd$ as follows:

$$Pd(v_1, v_2) = \begin{cases} 0, & \text{if } v_1 = v_2, \\ W(w_0), & \text{otherwise}, \end{cases}$$

where $w_0$ is the lowest common ancestor of $v_1$ and $v_2$. For example, in Figure 2, $Pd(C_1, V_1) = W(C_1), Pd(C_3, D_2) = W(B_2)$ and $Pd(C_3, E_2) = W(A_2)$. This definition of $Pd$ is somewhat geometrical. Alternatively, we can define $Pd$ in an algebraic way by using GO codes. Let $N_0$ be the set of natural numbers including zero. Then, given two GO codes $v_1 = a_1 a_2 \cdots a_H$ and $v_2 = b_1 b_2 \cdots b_H$ with $a_i, b_i \in N_0$,

$$Pd(v_1, v_2) = \begin{cases} 0, & \text{if } a_i = b_i \text{ for all } i, \\ W(L), & \text{otherwise}, \end{cases}$$

where $L = \max_{1 \leq i \leq H} \{|a_i| - |b_i|\}$. Now, we will show that $Pd$ is a metric on the set $V_C$ of all GO codes.

**Proposition 1.** $Pd \!: V_C \to R$ is a distance function, i.e. a metric.

**Proof.** It is trivial that $Pd$ is reflexive and symmetric from the definition of $Pd$. To show that $Pd$ is transitive, suppose that $x = a_1 a_2 \cdots a_H, y = b_1 b_2 \cdots b_H \in V_C$ and $Pd(x, y) = t$. Then, for any $z = c_1 c_2 \cdots c_H \in V_C$, if $Pd(y, z) = s$ and $s \leq t$, $Pd(x, z) \leq Pd(x, y) + Pd(y, z)$ since $Pd(x, z) \geq 0$. Similarly, if $Pd(y, z) = s$ and $s < t$, $Pd(x, y) \leq Pd(x, z) + Pd(y, z)$. By the above proposition, we can think of $T_G$ as a metric space and hence we get a useful ruler $Pd$ to measure the distance between any two nodes of $T_G$.

**MaxPd and AverPd**

Mathematically, the following three sets $\{1\}, \{1, 1\}, \{1, 1, 1\}$ are equal in the set notation. Yet, we want to take the number of occurrences of elements into account. In that case, such set is called as a multiset. Now, given a multiset $G = \{v_1, v_2, \ldots, v_n\}$ of GO codes in GO tree, $MaxPd$ is defined as the maximum value of principal distances between two elements in $G$ and $AverPd$ as the arithmetic average of principal distances from every pair of GO codes in $G$. In mathematical notations,

$$MaxPd(G) = \max_{1 \leq i \neq j \leq n} \{Pd(v_j, v_j)\} \quad \text{and} \quad AverPd(G) = \sum_{1 \leq i \neq j \leq n} \frac{Pd(v_i, v_j)}{n C_2}$$

where,

$$n C_2 = \frac{n(n - 1)}{2}$$

$MaxPd$ is used to give the comprehensive biological meanings of a gene cluster by finding a LCA of the cluster. If the LCA of a cluster is located at higher levels (level 1 or 2) of GO tree $T_G$, the cluster is not well organized or has some false positives that have inconsistent biological meanings with the other genes in that cluster. If the LCA is positioned at relatively lower levels (level 4 or lower than 4) of $T_G$, the clustering can be considered nicely done. Evidently, such conclusion follows from the current GO hierarchy and the weight function of our GO algorithm. $MaxPd$ equivalently weighs every gene in a cluster in its computation. The resultant GO code from $MaxPd$ may therefore be placed at relatively higher levels on account of just one false positive. While this might be bad in that it is not flexible, it can be also considered good in that it informs us of the existence of some functional outliers.

$AverPd$ signifies the most frequent GO codes among the genes of the GO codes at which most genes are concentrated in GO space. $AverPd$ tries to infer the strongest meanings of a gene cluster from its most concentrated subcluster and hence it does not concern a few functional outliers in that cluster. Moreover, $AverPd$ can produce several candidates according to their score (i.e. arithmetic average of principal distances), whereas the resultant GO codes from $MaxPd$ can be more than one, only given the multi-functions of genes since the LCA of a cluster is unique.

**Algorithmic Approach**

Given a cluster $C$ of genes that are annotated with GO terms, our main goal is to find the common biological meanings shared by the genes of the cluster. Using various resources (e.g. literature data mining or publicly available GO annotations), several GO terms can be extracted for each gene since a single gene may have multiple functions or be involved in multiple biological processes. How many GO terms a single gene will have mainly depends on the current accumulation of experimental results and on their refined processing into proper GO annotations. After GO term extraction, each GO term is transformed into corresponding GO codes. The representative GO codes for a cluster are then computed by $MaxPd$ or $AverPd$ using principal distance.

For an algorithmic approach, our procedure is formalized as follows. Suppose that a cluster $C$ consists of the genes $C_1, C_2, \ldots, C_n$. If each gene $C_i$ has $t_i$ GO terms and hence their corresponding $k_i$ GO codes, denoted by $c[i, j]$ with $1 \leq j \leq k_i$, then the maximum number of combinations $\{c[i, j], c[j, j], \ldots, c[n, n]\}$ of GO codes is $k_1 \times k_2 \times \cdots \times k_n$. Assuming that every $k_i$ is approximately 3, the number of combinations is about $3^n$. If so, given $n$ genes, we have to compute $3^n$ cases. Without appropriate modifications to reduce the operations, this requires an exponential time algorithm that is computationally expensive as $n$ becomes large. To cope with this problem, we consider the ordered GO codes $g[m]$ where $1 \leq m \leq \alpha, \alpha$ is a constant related to the input data $c[i, j]$ and the total number of GO codes $\Omega$.

384
in GO tree with $\alpha \leq \Omega$. The key is that the resulting biological terms are also GO terms. Each $g[m]$ is compared with $c[i, j]$ for $1 \leq i \leq n, 1 \leq j \leq k_i$ and the optimal combinations $(g[m], c[i, j])$ producing high proximity between them are chosen. That is, among the possible choices, the combinations that yield the most specific information, i.e. the lowest-leveled GO terms, will be selected. In this way, operations can be reduced down to $\alpha \times n \times \max_{1 \leq i \leq n} k_i$. If $k_i = 3$ as above, the required operations are about $3\alpha n$.

MaxPd is used to find a LCA of $C$. Let $Nr(g[m], t) = \{w \in V_C|Pd(g[m], w) \leq t\}$ for $t \in R$. If $C \subseteq Nr(g[m_0], t_0)$ with $t_0 = W(g[m_0])$ for some $m_0$, then $g[m_0]$ is a common ancestor of $C$. Furthermore, if $W(g[m]) \leq W(g[m])$ for any common ancestor $g[m]$ of $C$, $g[m_0]$ is a LCA of $C$. The pseudo-code of MaxPd can be concisely written as follows:

Step 1. Choose $g[m]$ such that $\max\{Pd(g[m], c[i, j])|1 \leq j \leq k_i \leq W(g[m])\}$ for all $i$.

Step 2. Among $g[m]$ chosen from step 1, select $g[m]$ with the minimum weight of level.

AverPd is used to find an optimal GO code $g[m_0]$ such that the average distance between $g[m_0]$ and each gene in $C$ is smaller than that of any $g[m]$, when measured in $Pd$. The following is the pseudo-code of AverPd.

Step 1. For each $m$ and $i$, Compute $S(m, i) = \min\{Pd(g[m], c[i, j])|1 \leq j \leq k_i\}$.

Step 2. For each $m$, calculate $f(m) = \sum_{1 \leq i \leq s} S(m, i)/n$.

Step 3. Choose $g[m_0]$ such that $f(m_0) \leq f(m)$ for all $m$.

**SAMPLE DATA**

The budding yeast *Saccharomyces cerevisiae*

Our algorithm was applied to the well-known Eisen *et al.* (1998) data set. Using the hierarchical clustering methods producing graphical dendrograms, Eisen *et al.* successfully clustered the gene expression profiles of the budding yeast *S. cerevisiae*. We thoroughly investigated the data in terms of our modeling. For GO term extraction, the Saccharomyces Genome Database (SGD) was used from http://www.geneontology.org/ (Dwight *et al.*, 2002). The SGD and GO versions tested are Revision 1.605 and 2.691 (Biological Process), respectively. The numbers of GO terms (nodes) in GO DAG are 5345 (Molecular Function), 6977 (Biological Process) and 1201 (Cellular Component), and the numbers of corresponding GO codes in GO tree are 8792, 36327 and 2039, respectively. It took at most 3 s to run the MaxPd and AverPd processes for each cluster using a 2.4 GHz PC under Windows environment with 512 MB RAM.

**Biological interpretation of gene clusters through GO codes**

We interpreted the top 10 clusters of Figure 2 in Eisen *et al.* through GO Biological Process. In Table 1, AverPd successfully computed the representative biological meanings that are almost the same as those given by Eisen *et al.* Our GO code representation is more descriptive and specific than just a keyword. The results of MaxPd also show that the gene clusters other than 2, 7 and 9 have inconsistent functional contexts. These functional discrepancies in a cluster may be caused by either the innate functional diversities of the sub-clusters or the lack of proper GO annotations of some genes in that cluster. Although only the first-ranked candidate term is shown in Table 1, multiple candidate terms can be selected by their scores.

**Biological significance of gene clusters by AverPd score**

AverPd is proposed as a new quantitative measure for estimating how well gene clusters of expression profiles are gathered together along with known functional categories. To examine the effectiveness of AverPd, we compared three kinds of gene groups constructed from Eisen *et al.* raw data of about 2470 ORFs: the original 10 clusters of Eisen *et al.*, another 10 clusters by average linkage hierarchical clustering, and 20 randomly chosen gene groups with no prior knowledge of microarray data. The 20 randomly chosen groups are again divided into two separate classes. One type of random groups has equal number of 50 genes and the other has increasing number of genes by 10 from 60 to 150. As shown in Table 2, the AverPd scores of the randomly selected gene groups are around 120 irrespective of the gene numbers. On the other hand, those of the Eisen top 10 clusters are fairly low, mostly not over 70. Another 10 clusters by hierarchical clustering are moderate. Figure 3 shows the distinct patterns of the three kinds of groups according to the functional tightness of the clusters through GO Biological Process. We can therefore assess the biological consistency of a gene cluster by AverPd score and ensure from Figure 3 that statistical clustering techniques show guilt-by-association rule much better than random partitions.

**DISCUSSION**

The GO hierarchy is nicely organized to enable the quantitative formulations between GO terms. Currently there are several algorithms and softwares using GO for identifying the most over-represented or characteristic GO terms of a gene group (Doniger *et al.*, 2003; Khatri *et al.*, 2002; Zeeberg *et al.*, 2003). They use various statistical tests such as Fisher’s exact test and graph visualizations, tree or DAG. They consider GO term frequencies among genes or compare specific GO term-related gene groups. Their methods are effective especially in visualizations but do not fully quantify the GO hierarchy represented by a graph structure: a tree or DAG is used mainly for visualization, not for essential computation. In our modeling, the topological property of GO hierarchy is entirely used to calculate the functional closeness of genes that are represented by GO codes in a metric GO space.
Table 1. Biological interpretation of top 10 clusters in Eisen et al. (1998) by GO codes

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Eisen keywords</th>
<th>AverPd (score value)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (11/11)</td>
<td>Cytoskeleton, Cell cycle: BP—cellular process—cell growth and/or maintenance</td>
<td>BP—cellular process—cell growth and/or maintenance—cell organization and biogenesis—cytoplasm organization and biogenesis—cytoskeleton organization and biogenesis—microtubule-based process—microtubule cytoskeleton organization and biogenesis (72.0)</td>
</tr>
</tbody>
</table>
| 3 (14/14) | mRNA splicing: BP—physiological processes | BP—physiological processes—metabolism—nucleobase
nucleoside
nucleotide and nucleic acid metabolism—RNA metabolism—RNA processing—mRNA processing—mRNA splicing (88.0) |
| 5 (22/22) | Protein synthesis: BP—physiological processes | BP—physiological processes—metabolism—biosynthesis—macromolecule biosynthesis—protein biosynthesis (43.0) |
| 6 (15/15) | ATP synthesis: BP—physiological processes | BP—physiological processes—cell growth and/or maintenance—transport—hydrogen transport—proton transport—energy coupled proton transport—down the electrochemical gradient—ATP synthesis coupled proton transport (54.0) |
| 7 (8/8) | Chromatin structure: BP—cellular process—cell growth and/or maintenance—cell organization and biogenesis—nuclear organization and biogenesis—chromosome organization and biogenesis (sensu Eukarya)—establishment and/or maintenance of chromatin architecture—chromatin assembly/disassembly | BP—cellular process—cell growth and/or maintenance—cell organization and biogenesis—nuclear organization and biogenesis—chromosome organization and biogenesis (sensu Eukarya)—establishment and/or maintenance of chromatin architecture—chromatin assembly/disassembly (0.0) |
| 8 (137/138) | Protein synthesis: BP—physiological processes—metabolism | BP—physiological processes—metabolism—biosynthesis—macromolecule biosynthesis—protein biosynthesis (12.0) |
| 9 (5/5) | DNA replication: BP—cellular process—cell growth and/or maintenance—cell proliferation—cell cycle | BP—cellular process—cell growth and/or maintenance—cell proliferation—cell cycle—DNA replication and chromosome cycle—DNA replication—DNA dependent DNA replication—DNA replication initiation (22.0) |
| 10 (15/15) | TCA cycle, Oxidative phosphorylation: BP—physiological processes—metabolism | BP—physiological processes—metabolism—energy pathways—energy derivation by oxidation of organic compounds—cellular respiration—aerobic respiration (72.0) |

GO Biological Process (BP) was used for analysis. The figures in parentheses beside cluster numbers mean ‘the number of genes having GO annotations/total gene number in the cluster’.

Our methodology can be used in several ways. It can assist in the biological assessment of the clustering results of DNA microarray data. Suppose that \( C \) is a gene cluster obtained by some clustering algorithm and that \( C_0 \) is a maximal subcluster of \( C \) whose genes are annotated with GO terms. It is not always the case that all the genes of \( C \) have appropriate GO annotations due to the current limited knowledge of gene functions and hence \( C_0 \subseteq C \). We can then assess the clustering quality of \( C \) by \( \text{AverPd}(C_0) \). If the value of \( \text{AverPd}(C_0) \) is sufficiently small, \( C \) can be regarded as biologically well-clustered in GO space. It can be also coupled with any clustering technique to predict efficiently the functional category of the unknown genes in a cluster. Thus, it may be helpful for automated functional annotation. If the \( \text{AverPd} \) score of a given gene cluster is below a pre-defined threshold, the genes in \( C - C_0 \) may be strongly related with the GO terms of \( \text{AverPd}(C_0) \).

Our algorithm can be easily generalized. It is applicable not only to cluster analysis of DNA microarray data, but also to any kinds of group analysis on other biological entities such as proteins that can be annotated with GO terms. This methodology can be further applied to any systematic ontology having an identical structure with GO, i.e., an acyclic digraph.

Despite these advantages, certain limitations also remain. We cannot obtain much biomedical information such as disease-related genes usually not contained in GO annotations. The disease-related terms are richer in Medical Subject...
A graph-theoretic modeling on GO space

Table 2. AverPd scores of three kinds of gene groups

<table>
<thead>
<tr>
<th>Gene groups</th>
<th>AverPd score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top 10 clusters in Figure 2 of Eisen et al. paper</td>
<td>72 0 88 47 43 54 0 12 22 72</td>
</tr>
<tr>
<td>Another 10 clusters by hierarchical clustering (average linkage)</td>
<td>14 84 85 85 106 90 67 110 102 105</td>
</tr>
<tr>
<td>Randomly chosen 10 groups with equal gene numbers (50)</td>
<td>121 119 114 120 116 119 120 118 123 120</td>
</tr>
<tr>
<td>Randomly chosen 10 groups with gene numbers increasing by 10 from 60 to 150</td>
<td>120 123 123 117 119 119 117 114 118 118</td>
</tr>
</tbody>
</table>

Fig. 3. AverPd scores of three kinds of gene groups. The group patterns are distinct according to the functional similarity of gene clusters. Random clusters are near 120, whereas Eisen top 10 clusters are mostly not over 70.

Headings (MeSH; http://www.nlm.nih.gov/mesh/meshhome.html) than in GO. In modeling GO tree, we assumed that the information specificity of GO terms on the same level is equivalent. But, it does not hold in general biological conditions. In that case, we may give an advantage or penalty of weight to some special relationships we focus on so that a more sophisticated weight system will be needed on GO tree.

One of the important aspects of GO hierarchy is that it is so dynamic and flexible. It can be considered positive in that it leaves room for further improvements. In making a robust algorithm, however, such properties may have an undesirable effect on the reliability of the algorithm, since results can be variable according to the changing GO hierarchy. However, several recent tests have assured that the more recently they are updated, the better results they produce. The results will evidently get better as more quality GO annotations are accumulating and GO hierarchy gets improving.

ACKNOWLEDGEMENTS

We are grateful to Hyong Song and Hwan Hyo Park for providing useful suggestions on implementation. We thank the anonymous referees for the valuable comments on our manuscript. Special thanks go to our colleagues Wan Seon Lee, Ji Young Oh, Tae-Hoon Chung and Jinho Yoo for helpful discussion.

REFERENCES


