The *GOSim* package

Holger Fröhlich

July 8, 2008

1 Introduction

The Gene Ontology (GO) has become one of the most widespread systems for systematically annotating gene products within the bioinformatics community and is developed by the Gene Ontology Consortium (17). It is specifically intended for describing gene products with a controlled and structured vocabulary. GO terms are part of a Directed Acyclic Graph (DAG), covering three orthogonal taxonomies or "aspects": *molecular function, biological process* and *cellular component*. Two different kinds of relationship between GO terms exist: the "is-a" relationship and the "part-of" relationship. Providing a standard vocabulary across any biological resources, the GO enables researchers to use this information for automated data analysis.

The *GOSim* package (5) provides the researcher with various information theoretic similarity concepts for GO terms (11; 12; 8; 7; 9; 3; 4). It additionally implements different methods for computing functional similarities between gene products based on the similarties between the associated GO terms. This can, for instances, be used for clustering genes according to their biological function (16; 6) and thus may help to get a better understanding of the biological aspects covered by a set of genes.

Since version 1.1 *GOSim* additionally offers the possibility of a GO enrichment analysis using the topGO package (1). Hence, *GOSim* acts now as an umbrella for different analysis methods employing the GO structure.

2 Usage of *GOSim*

To elucidate the usage of *GOSim* we show an example workflow and explain the employed similarity concepts. We create a character vector of Entrez gene IDs, which we assume to be from human:

```
> library(GOSim)
> genes = c("207", "208", "596", "901", "780", "3169", "9518",
+     "2852", "26353", "8614", "7494")
```

Next we investigate the GO annotation within the current ontology (which is *biological process* by default):

```
> getGOInfo(genes)
```

2.1 Term Similarities

Let us examine the similarity of the GO terms for genes "8614" and "2852" in greater detail:

```
> getTermSim(c("GO:0007166", "GO:0007267", "GO:0007584", "GO:0007165",
+     "GO:0007186"), method = "Resnik", verbose = FALSE)
```

```
           GO:0007166 GO:0007267 GO:0007584 GO:0007165 GO:0007186
GO:0007166  1.0000000  0.3693741   0.233135  0.3792863  0.4508958
GO:0007267  0.3693741  1.0000000   0.233135  0.3693741  0.3693741
GO:0007584  0.2331350  0.2331350   1.000000  0.2331350  0.2331350
GO:0007165  0.3792863  0.3693741   0.233135  1.0000000  0.3792863
GO:0007186  0.4508958  0.3693741   0.233135  0.3792863  1.0000000
```

This calculates Resnik's pairwise similarity between GO terms (11; 12):

$$sim(t, t') = IC_{ms}(t, t') := \max_{\hat{t} \in Pa(t,t')} IC(\hat{t}) \tag{1}$$

Here $Pa(t, t')$ denotes the set of all common ancestors of GO terms t and t', while $IC(t)$ denotes the information content of term t. It is defined as (e.g. (9))

$$IC(\hat{t}) = -\log P(\hat{t}) \tag{2}$$

i.e. as the negative logarithm of the probability of observing $\hat{t}$. The information content of each GO term is already precomputed for each ontology based on the empirical observation, how many times a specific GO term or any of its direct or indirect offsprings appear in the annotation of the GO with gene products.

```
> data("ICsBPhumanall")
> IC[c("GO:0007166", "GO:0007267", "GO:0007584", "GO:0007165",
+     "GO:0007186")]
```

```
GO:0007166 GO:0007267 GO:0007584 GO:0007165 GO:0007186
  6.270508   7.308268  10.169108   5.274649   6.771886
```

This loads the information contents of all GO terms within "biological process". Likewise, the data files `ICsMFhumanall` and `ICsCChumanall` contain the information contents of all GO terms within "molecular function" and "cellular component" for human. Since

GOSim version 1.1.4.0 the information content of GO terms relies on the mapping of Entrez gene IDs to GO terms provided by the libraries org.Dm.eg.db (fly), org.Hs.eg.db (human), org.Mm.eg.db (mouse), org.Pf.plasmo.db (malaria), org.Rn.eg.db (rat) and org.Sc.sgd.db (yeast). Additionally, it is possible to pass a user provided mapping via the function `setEvidenceLevel`. Please refer to the manual pages for details. If only GO terms having certain evidence codes should be considered, one must explicitely calculate the corresponding information contents in the function `calcICs`. Again, more information on this function can be found in the manual pages.

For the similarity computation in (Eq.: 1) normalized information contents are used, which are obtained by dividing the raw information contents by its maximal value:

```
> IC[c("GO:0007166", "GO:0007267", "GO:0007584", "GO:0007165",
+     "GO:0007186")]/max(IC)
```

To continue our example from above, let us also calculate Jiang and Conrath's pairwise similarity between GO terms, which is the default, for compairson reasons (7):

```
> getTermSim(c("GO:0007166", "GO:0007267", "GO:0007584", "GO:0007165",
+     "GO:0007186"), verbose = FALSE)
```

```
           GO:0007166 GO:0007267 GO:0007584 GO:0007165 GO:0007186
GO:0007166  1.0000000  0.7623340  0.2841403  0.9283904  0.9639472
GO:0007267  0.7623340  1.0000000  0.2095176  0.8339436  0.7262812
GO:0007584  0.2841403  0.2095176  1.0000000  0.3557498  0.2480874
GO:0007165  0.9283904  0.8339436  0.3557498  1.0000000  0.8923376
GO:0007186  0.9639472  0.7262812  0.2480874  0.8923376  1.0000000
```

Jiang and Conrath's similarity measure is defined as

$$sim(t, t') = 1 - \min(1, IC(t) - 2IC_{ms}(t, t') + IC(t')) \tag{3}$$

i.e. the similarity between t and t' is 0, if their normalized distance is at least 1.

Likewise, we can also compute Lin's pairwise similarity between GO terms (8):

```
> getTermSim(c("GO:0007166", "GO:0007267", "GO:0007584", "GO:0007165",
+     "GO:0007186"), method = "Lin", verbose = FALSE)
```

```
           GO:0007166 GO:0007267 GO:0007584 GO:0007165 GO:0007186
GO:0007166  1.0000000  0.7565931  0.3944322  0.9137423  0.9615578
GO:0007267  0.7565931  1.0000000  0.3710118  0.8164726  0.7296516
GO:0007584  0.3944322  0.3710118  1.0000000  0.4198663  0.3827587
GO:0007165  0.9137423  0.8164726  0.4198663  1.0000000  0.8757122
GO:0007186  0.9615578  0.7296516  0.3827587  0.8757122  1.0000000
```

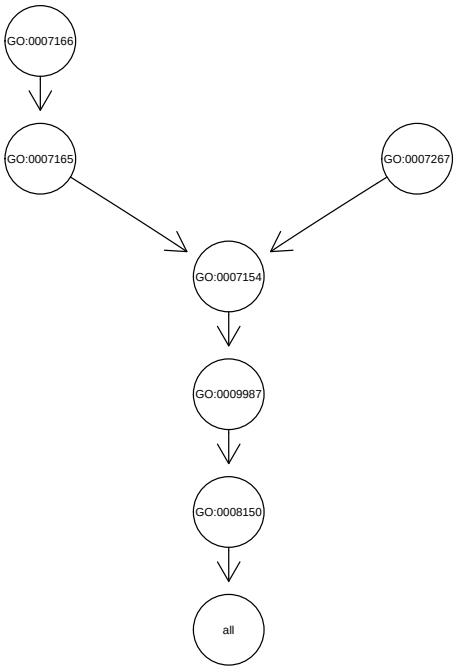

Figure 1: Example of a GO graph starting with leaves GO:0007166 and GO:0007267.

It is defined as:

$$sim(t, t') = \frac{2IC_{ms}(t, t')}{IC(t) + IC(t')} \tag{4}$$

Resnik's, Jiang-Conraths's and Lin's term similarities all refer to $IC_{ms}(t, t')$, the information content of the minimum subsumer of t and t', i.e. of the lowest common ancestor in the hierarchy. For illustration let us plot the GO graph with leaves GO:0007166 and GO:0007267 and let us compute their minimum subsumer (see Fig. 1):

```
> library(Rgraphviz)
> G = getGOGraph(c("GO:0007166", "GO:0007267"))
> plot(G)
```

```
> getMinimumSubsumer("GO:0007166", "GO:0007267")
```

```
[1] "GO:0007154"
```

In contrast to the above defined similarity measures Couto et al. (4) introduced a concept, which is not based on the minimum subsumer, but on the set of all disjunctive common ancestors. Roughly speaking, the idea is not to consider the common ancestor having the highest information content only, but also others, if they are somehow "separate" from each other, i.e. there is a path to t and t' not passing any other of the disjunctive common ancestors.

```
> getDisjCommAnc("GO:0007166", "GO:0007267")

[1] "GO:0007154"
```

In this case the set of disjunctive common ancestors only consists of the minimum subsumer, because any path from the other ancestors to GO:0007166 and GO:0007267 would have to pass the minimum subsumer (see Fig. 1).

Based on the notion of disjunctive common ancestors Resnik's similarity concept can be extended by defining:

$$sim(t,t') = IC_{share}(t,t') = \frac{1}{|DisjCommAnc|} \sum_{t \in DisjCommAnc} IC(t) \tag{5}$$

Likewise, Jiang-Conraths's and Lin's measures can be extended as well by replacing $IC_{ms}(t,t')$ by $IC_{share}(t,t')$.

```
> getTermSim(c("GO:0007166", "GO:0007267", "GO:0007584", "GO:0007165",
+     "GO:0007186"), method = "CoutoResnik", verbose = FALSE)

           GO:0007166 GO:0007267 GO:0007584 GO:0007165 GO:0007186
GO:0007166  1.0000000  0.3693741   0.233135  0.3693741  0.3792863
GO:0007267  0.3693741  1.0000000   0.233135  0.3693741  0.3693741
GO:0007584  0.2331350  0.2331350   1.000000  0.2331350  0.2331350
GO:0007165  0.3693741  0.3693741   0.233135  1.0000000  0.3693741
GO:0007186  0.3792863  0.3693741   0.233135  0.3693741  1.0000000
```

Finally, it should be mentioned that also the depth and density enriched term similarity by Couto et al. (3) has been integrated into *GOSim*:

```
> setEnrichmentFactors(alpha = 0.5, beta = 0.3)
> getTermSim(c("GO:0007166", "GO:0007267", "GO:0007584", "GO:0007165",
+     "GO:0007186"), method = "CoutoEnriched", verbose = FALSE)

           GO:0007166 GO:0007267 GO:0007584 GO:0007165 GO:0007186
GO:0007166  1.0000000  0.2030138  0.2021858  0.1636249  0.2151779
GO:0007267  0.2030138  1.0000000  0.2184200  0.1793893  0.2141354
GO:0007584  0.2021858  0.2184200  1.0000000  0.1820661  0.2112405
GO:0007165  0.1636249  0.1793893  0.1820661  1.0000000  0.1730192
GO:0007186  0.2151779  0.2141354  0.2112405  0.1730192  1.0000000
```

2.2 Functional Gene Similarities

The special strength of *GOSim* lies in the possibility not only to calculate similarities for individual GO terms, but also for genes based on their complete GO anntation. Since *GOSim* version 1.1 for this purpose four basic ideas have been implemented:

1. Maximum and average pairwise GO term similarity

2. Average of best matching GO term similarities (14).

3. Computation of a so-called *optimal assignment* of terms from one gene to those of another one (6).

4. Embedding of each gene into a feature space defined by the gene's similarity to certain prototype genes (16; 6). Within this feature space similarities naturally arise as dot products between the feature vectors. These dot products can be understood as so-called *kernel functions* (15), as used in e.g. Support Vector Machines (2).

2.3 Maximum and Average Pairwise GO Term Similarity

The idea of the maximum pairwise GO term similarity is straight forward. Given two genes g and g' annotated with GO terms $t_1, ..., t_n$ and $t'_1, ..., t'_m$ we define the functional similarity between between g and g' as

$$sim_{gene}(g, g') = \max_{\substack{i = 1, , ..., n \\ j = 1, ..., m}} sim(t_i, t'_j) \tag{6}$$

where sim is some similarity measure to compare GO terms t_i and t'_j. Instead of computing the maximum pairwise GO term similarity one may also take the average here.

To normalize the similarities later on, we can peform the transformation

$$sim_{gene}(g, g') \leftarrow \frac{sim_{gene}(g, g')}{\sqrt{sim_{gene}(g, g) sim_{gene}(g', g')}} \tag{7}$$

The consequence will be a similarity of 1 for g with itself and between 0 and 1 for g with any other gene. Another possibility is to use the Lin's normalization (see Eq. 4)

$$sim_{gene}(g, g') \leftarrow \frac{2 sim_{gene}(g, g')}{sim_{gene}(g, g) + sim_{gene}(g', g')} \tag{8}$$

2.4 Average of Best Matching GO Term Similarities

The idea of this approach (14) is to assign each GO term t_i occuring in gene g to its best matching partner $t'_{\pi i}$ in gene g'. Hence multiple GO terms from gene g can be assigned to one GO term from gene g'. A similarity score is computed by taking the average similarity of assigned GO terms. Since, however, genes can have an unequal number of GO terms the result depends on whether GO terms of gene g are assigned to those of gene g' or vice versa. Hence, in (14) it was proposed to either take the maximum or the average of both similarity scores. Both strategies are implemented in *GOSim*.

2.4.1 Optimal Assignment Gene Similarities

To elucidate the idea of the optimal assignment, consider the GO terms associated with gene "8614" on one hand and gene "2852" on the other hand:

```
> getGOInfo(c("8614", "2852"))
```

```
           GO:0007166
go_id      "GO:0007166"
Term       "cell surface receptor linked signal transduction"
Definition "Any series of molecular signals initiated by the binding of an extracell
           GO:0007267
go_id      "GO:0007267"
Term       "cell-cell signaling"
Definition "Any process that mediates the transfer of information from one cell to a
           GO:0007584
go_id      "GO:0007584"
Term       "response to nutrient"
Definition "A change in state or activity of a cell or an organism (in terms of move
           GO:0007165
go_id      "GO:0007165"
Term       "signal transduction"
Definition "The cascade of processes by which a signal interacts with a receptor, ca
           GO:0007186
go_id      "GO:0007186"
Term       "G-protein coupled receptor protein signaling pathway"
Definition "The series of molecular signals generated as a consequence of a G-protei
           GO:0007186
go_id      "GO:0007186"
Term       "G-protein coupled receptor protein signaling pathway"
Definition "The series of molecular signals generated as a consequence of a G-protei
```

Given a similarity concept sim to compare individual GO terms, the idea is now to assign each term of the gene having fewer annotation to exactly one term of the other gene such that the overall similarity is maximized. Hence, in the optimal assignment More formally optimal assignment problem can be stated as follows: Let π be some permutation of either an n-subset of natural numbers $\{1, ..., m\}$ or an m-subset of natural numbers $\{1, ..., n\}$ (this will be clear from context). Then we are looking for the quantity

$$sim_{gene}(g, g') = \begin{cases} \max_\pi \sum_{i=1}^{n} sim(t_i, t'_{\pi(i)}) & \text{if } m > n \\ \max_\pi \sum_{j=1}^{m} sim(t_{\pi(j)}, t'_j) & \text{otherwise} \end{cases} \tag{9}$$

The computation of (9) corresponds to the solution of the classical maximum weighted bipartite matching (optimal assignment) problem in graph theory and can be carried out

in $O(\max(n,m)^3)$ time (10). To prevent that larger lists of terms automatically achieve a higher similarity we may further sim_{gene} divide 9 by $\max(m,n)$.

In our example, using Lin's GO term similarity measure the following assignments are found:

$$\texttt{GO:0007165} \rightarrow \texttt{GO:0007267} \quad (10)$$

$$\texttt{GO:0007186} \rightarrow \texttt{GO:0007166} \quad (11)$$

The resulting similarity matrix is:

```
> getGeneSim(c("8614", "2852"), similarity = "OA", similarityTerm = "Lin",
+     verbose = FALSE)

          8614      2852
8614 1.0000000 0.5926768
2852 0.5926768 1.0000000
```

Note the difference to a gene similarity that is just based on the maximum GO term similarity and to a gene similarity that is based on the average of best matching GO terms:

```
> getGeneSim(c("8614", "2852"), similarity = "max", similarityTerm = "Lin",
+     verbose = FALSE)

          8614      2852
8614 1.0000000 0.9615578
2852 0.9615578 1.0000000

> getGeneSim(c("8614", "2852"), similarity = "funSimMax", similarityTerm = "Lin",
+     verbose = FALSE)

        8614    2852
8614 1.00000 0.93765
2852 0.93765 1.00000
```

2.4.2 Feature Space Embedding of Gene Products

To calculate the feature vectors for each gene we can either define certain prototype genes a priori or we use one of the heuristics implemented in the function `selectPrototypes`. The default behavior is to select the 250 best annotated genes, i.e. which have been annotated with GO terms most often:

```
> proto = selectPrototypes(verbose = FALSE)
```

We now calculate for each gene g feature vectors $\phi(g)$ by using their similarity to all prototypes $p_1, ..., p_n$:

$$\phi(g) = (sim'(g, p_1), ..., sim'(g, p_n))^T \tag{12}$$

Here sim' by default is the maximum pairwise GO term similarity. Alternatively, one can use the optimal assignment similarity for sim' as well. Both similarity measures can by itself again be combined with arbitrary GO term similarity concepts. The default is the Jiang-Conrath term similarity.

Because the feature vectors are very high-dimensional we usually perform a principal component analysis (PCA) to project the data into a lower dimensional subspace:

```
> PHI = getGeneFeaturesPrototypes(genes, prototypes = proto, verbose = FALSE)
```

This uses the above define prototypes to calculate feature vectors and performs a PCA afterwards. The number of principal components is chosen such that at least 95% of the total variance in feature space can be explained (this is a relatively conservatve criterion).

We can now plot our genes in the space spanned by the first 2 principal components to get an impression of the relative "position" of the genes to each other in the feature space (see Fig. 2). The feature vectors are normalized to Euclidian norm 1 by default:

```
> x = seq(min(PHI$features[, 1]), max(PHI$features[, 1]), length.out = 100)
> y = seq(min(PHI$features[, 2]), max(PHI$features[, 2]), length.out = 100)
> plot(x, y, xlab = "principal component 1", ylab = "principal component 2",
+     type = "n")
> text(PHI$features[, 1], PHI$features[, 2], labels = genes)
```

Finally, we can directly calculate the similarities of the genes to each other, this time using the Resnik's GO term similarity concept. These similarities may then be used to cluster genes with respect to their function:

```
> sim = getGeneSimPrototypes(genes, prototypes = proto, similarityTerm = "Resnik",
+     verbose = FALSE)
> h = hclust(as.dist(1 - sim$similarity), "ward")
> plot(h, xlab = "")
```

This produces a hierarchical clustering of all genes using Ward's method (see Fig. 3).

It should be mentioned that up to now all similarity computations were performed within the ontology "biological process". One could imagine to combine functional similarities between gene products with regard to different taxonomies. An obvious way for doing so would be to consider the sum of the respective similarities:

$$sim_{total}(g, g') = sim_{Ontology1}(g, g') + sim_{Ontology2}(g, g') \tag{13}$$

Of course, one could also use a weighted averaging scheme here, if desired.

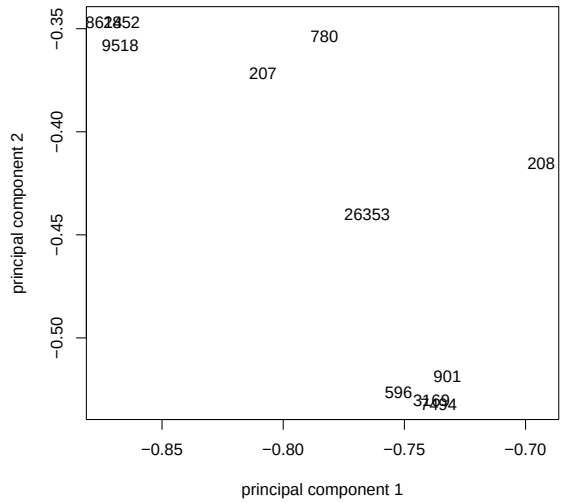

Figure 2: Embedding of the genes into the feature space spanned by the first 2 principal components

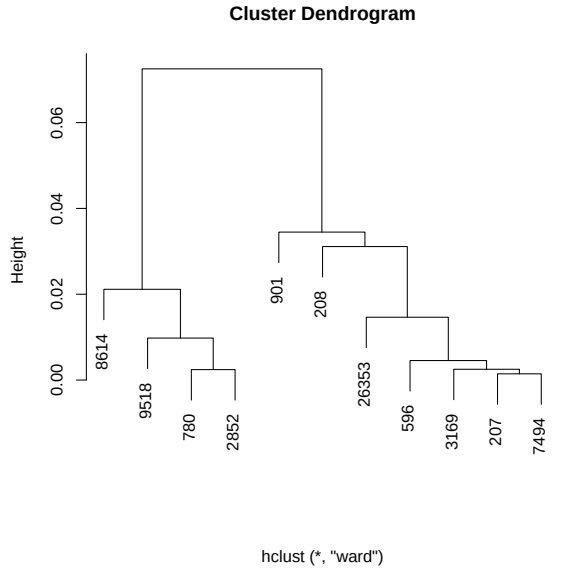

Figure 3: Possible functional clustering of the genes using Ward's method.

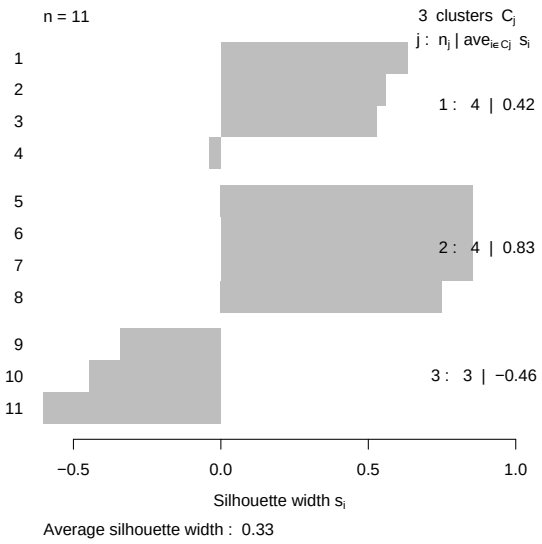

Figure 4: Silhouette plot of a possible given grouping of genes.

2.5 Cluster Evaluations

GOSim has the possibility to evaluate a given clustering of genes or terms by means of their GO similarities. Supposed, based on other experiments (e.g. microarry), we have decided to put genes "8614", "9518", "780", "2852" in one group, genes "3169", "207", "7494", "596" in a second and the rest in a third group. Then we can ask ourselves, how similar these groups are with respect to their GO annotations:

```
> ev = evaluateClustering(c(2, 3, 2, 3, 1, 2, 1, 1, 3, 1, 2), sim$similarity)
> plot(ev$clustersil, main = "")
```

A good indiciation of the clustering qualitiy can be obtained by looking at the cluster silhouettes (13) (see Fig. 4). This shows that clusters 1 and 2 are relatively homogenous with respect to the functional similarity of the genes contained in it, while the genes in cluster 3 are more dissimilar.

2.6 GO Enrichment Analysis

Since version 1.1 *GOSim* also offers the possibility of a GO enrichemnt analysis. Suppose, we may now want to get a clearer picture of the genes involved in cluster 1. For this purpose we use the topGO tool (1).

```
> gomap <- get("gomap", env = GOSimEnv)
> allgenes = unique(c(sample(names(gomap), 1000), genes))
> analyzeCluster(c("8614", "9518", "780", "2852"), allgenes)
```

```
Building most specific GOs .....        ( 805 GO terms found. )

Build GO DAG topology ..........        ( 1900 GO terms and 3256 relations. )

Annotating nodes ...............        ( 799 genes annotated to the GO terms. )

                        -- Elim Algorithm --

                the algorithm is scoring 34 nontrivial nodes
                parameters:
                        test statistic:  Fisher test
                        cutOff:  0.01

         Level 8:       2 nodes to be scored        (0 eliminated genes)

         Level 7:       3 nodes to be scored        (0 eliminated genes)

         Level 6:       5 nodes to be scored        (0 eliminated genes)

         Level 5:       5 nodes to be scored        (23 eliminated genes)

         Level 4:       7 nodes to be scored        (23 eliminated genes)

         Level 3:       7 nodes to be scored        (23 eliminated genes)

         Level 2:       4 nodes to be scored        (23 eliminated genes)

         Level 1:       1 nodes to be scored        (23 eliminated genes)
$GOTerms
           go_id                                                Term
15517 GO:0007167 enzyme linked receptor protein signaling pathway

15517 Any series of molecular signals initiated by the binding of an extracellular l

$p.values
 GO:0007167
0.004595884

$genes
$genes$`GO:0007167`
 [1] "10256"  "10817"  "163126" "1907"   "2048"   "207"    "2264"   "2662"
 [9] "2788"   "2984"   "3937"   "4915"   "5295"   "55715"  "5795"   "5894"
```

```
[17] "780"    "819"    "8527"   "8660"   "8822"   "9451"   "9518"
```

The result shows that the four genes are mainly involved in cell-cell signaling, in an enzyme linked receptor protein signaling pathway and in the response to nutrient.

References

[1] Adrian Alexa, Jörg Rahnenführer, and Thomas Lengauer. Improved scoring of functional groups from gene expression data by decorrelating GO graph structure. *Bioinformatics*, 22(13):1600 – 1607, 2006.

[2] C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20:273 – 297, 1995.

[3] F. Couto, M. Silva, and P. Coutinho. Implementation of a Functional Semantic Similarity Measure between Gene-Products. Technical Report DI/FCUL TR 03–29, Department of Informatics, University of Lisbon, 2003.

[4] F. Couto, M. Silva, and P. Coutinho. Semantic Similarity over the Gene Ontology: Family Correlation and Selecting Disjunctive Ancestors. In *Conference in Information and Knowledge Management*, 2005.

[5] H. Fröhlich, N. Speer, A. Poustka, and T. Beissbarth. GOSim - An R-Package for Computation of Information Theoretic GO Similarities Between Terms and Gene Products. *BMC Bioinformatics*, 8:166, 2007.

[6] H. Fröhlich, N. Speer, and A. Zell. Kernel based functional gene grouping. In *Proc. Int. Joint Conf. Neural Networks*, pages 6886 – 6891, 2006.

[7] J. Jiang and D. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of the International Conference on Research in Computational Linguistics*, Taiwan, 1998.

[8] D. Lin. An information-theoretic definition of similarity. In Morgan Kaufmann, editor, *Proceedings of the 15th International Conference on Machine Learning*, volume 1, pages 296–304, San Francisco, CA, 1998.

[9] P.W. Lord, R.D. Stevens, A. Brass, and C.A. Goble. Semantic similarity measures as tools for exploring the gene ontology. In *Proceedings of the Pacific Symposium on Biocomputing*, pages 601–612, 2003.

[10] K. Mehlhorn and S. Näher. *The LEDA Platform of Combinatorial and Geometric Computing*. Cambridge University Press, 1999.

[11] P. Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, volume 1, pages 448–453, Montreal, 1995.

[12] P. Resnik. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambigiguity in natural language. *Journal of Artificial Intelligence Research*, 11:95–130, 1999.

[13] P.J. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J. Comp. and Applied Mathematics*, 20:53–65, 1987.

[14] Andreas Schlicker, Francisco S Domingues, Jörg Rahnenführer, and Thomas Lengauer. A new measure for functional similarity of gene products based on Gene Ontology. *BMC Bioinformatics*, 7:302, 2006.

[15] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

[16] N. Speer, H. Fröhlich, C. Spieth, and A. Zell. Functional grouping of genes using spectral clustering and gene ontology. In *Proc. Int. Joint Conf. Neural Networks*, pages 298 – 303, 2005.

[17] The Gene Ontology Consortium. The gene ontology (GO) database and informatics resource. *Nucleic Acids Research*, 32:D258–D261, 2004.