ClusterInspector: a tool to visualize ontology-based relationships between biological entities

Sèverine Bérard, Laurent Tichit, Carl Herrmann

ClusterInspector: a tool to visualize ontology-based relationships between biological entities

Sèverine Bérard[1,2] Laurent Tichit[1] and Carl Herrmann[3]

[1] Institut de Mathématiques de Luminy, UMR CNRS 6206, Campus de Luminy, Case 907, 13288 Marseille Cedex 9, France
tichit@iml.univ-mrs.fr
[2] INRA Toulouse, Unité BIA Chemin de Borde Rouge, BP 27 Castanet-Tolosan 31326 Cedex, France
Severine.Berard@tlse.toulouse.inra.fr
[3] Laboratoire de Génétique et de Physiologie du Développement, UMR 6545, Campus de Luminy, Case 907, 13288 Marseille Cedex 9, France
herrmann@ibdm.univ-mrs.fr

Abstract: *In this paper, we define several quantitative measures that are useful when using ontologies to describe biological data. More precisely, we define the precision of a term inside an ontology, and use this concept to quantify the similarity between terms inside an ontology, or, by extension, between biological entities annotated to terms of the ontology. This similarity measure is used to build a hierarchical tree between biological objects annotated to a particular ontology. The graphical representation of this tree allows to easily interpret the relationships between objects of a set based on this ontology, and to identify relevant annotations. We have implemented this principle in ClusterInspector, a tool that allows to browse the resulting tree at various levels of the ontology.*

Keywords: Ontologies - Clustering

1 Introduction

While the pace of high-throughput experiments in biology is holding on, we need powerful tools to take advantage of the ever increasing amount of data, hence turning data into knowledge. Often, the result of high-throughput methods are clusters of biological objects (e.g. genes or proteins), which are likely to share particular features. This is for example the case when dealing with micro-array expression data, where hierarchical clustering techniques lead to groups of genes with similar expression behavior [1]. On the proteome side, numerous groups have proposed strategies to use interaction data to identify proteins that appear to tightly interact with each other [2,3,4,5]. Such integrative approaches can be used to better understand the interplay between biological processes, or to infer possible functions for yet unannotated gene products. In either case, the assumption is that these clusters are consistent, in the sense that they tend to associate genes which share particular features, such as functions or localization. Hence, the validation of these techniques often requires to check, first, whether this assumption is indeed satisfied. Features of genes or gene products are increasingly described via ontologies [6,8,9], which represent structured vocabularies mathematically designed as a directed acyclic graph (DAG). There are a vast number of such biological ontologies, and a standard, the Open Biomedical Ontologies (OBO) [7] has been worked out to unify their schemes.

In this paper, we propose a method and a tool, ClusterInspector, which allows to visualize the relationships between elements of a cluster, based on their annotation to a given ontology. It allows to identify possible

sub-groups which are more tightly related, or outliers, i.e., elements that have been mistakenly included. For example, using functional ontologies such as Gene Ontology (GO) one can spot genes that participate simultaneously in two complementary processes inside the same cluster. It also helps in characterizing the most relevant feature(s) of a set of elements, for example the most relevant functions. We believe that this tool will prove valuable whenever one is confronted with the biological interpretation of a set of genes or gene products.

The paper is organized as follows: in section 2, we will introduce the notion of precision and similarity inside an ontology, on which the clustering procedure is based. Section 3 will be devoted to the hierarchical clustering procedure, with details about the scoring scheme of the internal nodes and a presentation of the graphical interface. In Section 4, we give a simple example of a possible application of our method, and Section 5 will compare our work with other approaches.

2 Quantitative characteristics of an ontology

The core of ClusterInspector is a hierarchical clustering procedure, which is based on a measure of similarity between two elements with respect to the given ontology structure. The similarity measure itself relies on the notion of precision of a term inside an ontology. We will start by describing these rwo notions.

2.1 Precision

Since an ontology is a structured vocabulary, it implies a hierarchy between its terms. Hence, some terms are more general, while others are more specific. Intuitively, the deeper the term inside the DAG, the more precise it is. One might therefore use the depth, defined as the length of the shortest path from the root to a term, as a measure of its precision. The drawback of this definition is that it does not take into account the fact that some branches of the ontology are longer than others, and that the precision should therefore be weighted. For example, for the "biological process" ontology of GO, the terms GO:0046726 *"positive regulation of viral protein levels"* and GO:0000122 *"negative regulation of transcription from Pol II promoter"* both represent leaves of the DAG, but the first is at depth 3 while the latter is at depth 8. The definition of precision we propose here is an attempt to answer the following question: "Given the current annotation of a biological entity in an ontology, how much ambiguity is there in this annotation?". In other words, if we believe that an annotation reflects the incompleteness[4] of our current knowledge, and that it might evolve in the future by becoming annotated to more precise terms, how many more precise terms are possible? Therefore, the precision of a term t, denoted $p(t)$, will be related to the number of sub-terms which are below t, i.e., all terms (including the term itself) which can be reached starting from t, compared to the total number of terms N. Hence, the precision depends on the ratio n_t/N.

In order for the leaves of the DAG to have maximum precision 1, and the root minimum precision 0, we define:

$$p(t) = \frac{\ln(n_t/N)}{\ln(1/N)}. \tag{1}$$

The fact that all leaves have equal precision might seem a strong assumption, as some leaves in the ontology have not been equally explicited. But this maximum precision should rather be interpreted as the fact that a biological entity, annotated to a leaf of an ontology, cannot receive a more precise annotation in the current state of the ontology. Of course, not only do annotations evolve, but also the structure of the ontology changes with time, and therefore the precision of a given term might evolve. In Table 1, we give a list of the first 10 less

[4] We do not consider the possibility of mis-annotation, but only of incomplete annotation.

precise terms in the GO "biological process" ontology. This example shows that, while globally correlated to the depth of a term inside the ontology, the ranking by precision is not equivalent[5].

Term	precision	depth
biological process	0	0
physiological process	0.024	1
cellular process	0.028	1
cellular physiological process	0.041	2
metabolism	0.076	2
cellular metabolism	0.079	3
primary metabolism	0.107	3
development	0.171	1
macromolecule metabolism	0.186	3
regulation of biological process	0.190	1

Table 1. The first 10 term of the GO "biological process" ontology, ranked by increasing precision

2.2 Similarity

Once we have defined the precision of a term in the ontology, we can use this notion to quantify the similarity between terms. First, we define a term t to be an ancestor of a term t' if there exists a path upwards in the ontology DAG which leads from t' to t. Hence, by definition, the root of the ontology is a common ancestor to all other terms. With this definition of ancestors, we define the similarity between two terms t_1, t_2 as the precision of their most precise common ancestor.

$$s(t_1, t_2) \; = \; \max_{t_k \in \mathcal{S}_{common}} p(t_k) \; , \tag{2}$$

where $\mathcal{S}_{common}$ represents the set of common ancestors of t_1, t_2[6].

We can extend the notion of similarity to any pair of biological entities annotated to an ontology:

$$s(o, o') \; = \; \max_{t \in \mathcal{A}(o), t' \in \mathcal{A}(o')} s(t, t'), \tag{3}$$

where $\mathcal{A}(o)$ represents all the annotations t of object o to the ontology.

The similarity we have defined is not a distance in the strict sense; while it is symmetric, it does not respect the triangular inequality, nor does the similarity of a term to itself be maximal. Our similarity represents the worst case scenario: let us imagine that two genes are both annotated *directly* to a very general term in the *"biological process"* ontology of GO, like *"physiological process"*, and to nothing else. These two genes would have very low similarity ($s = 0.024$), which represents the lower bound of their future similarity based on more precise annotations. This worst case scenario is realized if, for example, the first gene gets annotated to a term in the "metabolism" branch, while the other gets annotated to a term in the distinct "localization" branch. If, on the contrary, both genes turn out to be involved in, say, catabolism, then their similarity will be

[5] Given the structure of the DAG, the depth of a term is not always unambiguously defined. We choose to define it as the length of the shortest path from the root to the term.

[6] These definition can easily be extended to a set of more than two terms.

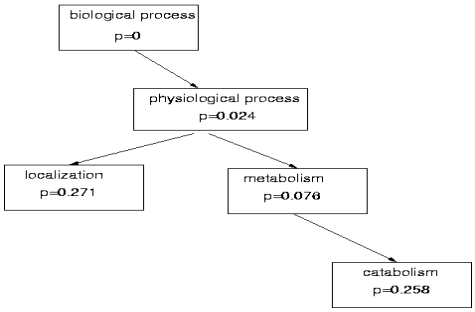

Figure 1. A small portion of the "biological process" ontology of GO illustrating the example of section 2.2.

greater than 0.024 (see Fig. 1). Hence, any value of a similarity at time θ represents the lower bound for the similarity at time $\theta' > \theta$ when the annotations will have become more precise[7].

3 ClusterInspector: a tool for cluster inspection

In the following section, we present the basic idea of ClusterInspector and the hierarchical clustering algorithm which is based on the previously defined similarity. Moreover, we explain how we score internal nodes of the tree obtained in order to select specific sub-clusters that appear particularly relevant. Finally, we explain how the graphical interface is implemented.

3.1 General principle

Based on the previously defined similarity, we developed a method and a tool, named ClusterInspector, which allows to graphically display the relationship between objects of a given set (e.g. genes or proteins) based on a given ontology. The method will iteratively group together similar objects, and display the whole set as a hierarchical tree. This visualization will enable to identify sub-groups which form a particularly homogeneous subset: proteins which share very similar functions in the case of a functional ontology, or genes expressed in similar body part in the case of an anatomical ontology. Our method is meant as a way to interpret clusters which come as an output of any kind of clustering procedure: hierarchical clustering of gene expression data, clustering based on protein-interaction data, experimentally purified protein complexes, etc...

Basically, the objects we consider are biological entities annotated to an ontology. However, since objects often carry multiple annotations inside a given ontology, we consider object|annotation pairs as elementary entities, rather that objects themselves. Therefore, the tree will generally have more leaves that the number of objects in the set. This is particularly interesting since it allows to identify for example pleiotropic genes which participate in several complementary processes inside a group of genes, or ubiquitously expressed genes.

The leaves of the tree will consist of all object|annotation pairs. The deepest nodes (i.e., those located immediately above the leaves) are annotated to the most precise common ancestor of the underlying leaves. Each node of the tree carries an annotation, which represents the most precise ancestor of the sub-nodes located immediately below. Finally, the root of the tree will represent the whole set of object|annotation pairs. The

[7] Again, this does not take into account the possibility of a mis-annotation.

length of a branch (T, T') is proportional to the increase in precision between the annotation t of node T and the annotation t' of its child node T'. More precisely:

$$L(T, T') \;=\; m \times |\, p(t) - p(t') \,| + 1 \;, \tag{4}$$

where m is an arbitrary multiplicative factor which can be tuned to optimally stretch the tree, and the $+1$ is an offset meant to avoid zero length branches. This allows in particular to detect possible outliers, i.e., sub-trees or leaves that get merged at a very late stage.

3.2 The algorithm

The hierarchical tree is constructed from a set of n object|annotation pairs and a given ontology. Each node of the tree represents a term of the ontology. Our algorithm is a greedy iterative clustering procedure. Schematically, at each step i of the algorithm, there are n_i sub-trees; let $T^i = \{T^i_j \mid j \in [1..n_i]\}$ be the set of sub-trees at a step i. The root node of a tree T^i_j carries an annotation t^i_j. We compute all pairwise similarities between the t^i_j's. Then, we merge together the closest sub-trees to form a new node. This procedure is repeated until a single tree is formed. More formally, we give below the pseudo-code of the algorithm. $A(t^i_{k_1}, t^i_{k_2})$ denotes the most precise common ancestor of $t^i_{k_1}$ and $t^i_{k_2}$.

Start with $i = 0$ and **repeat**:

1. Compute all pairwise similarities between the t^i_j for all $j \in [1..n_i]$;

2. Select the $T^i_k, k \in I \subset [1..n_i]$, such that $\forall k_1, k_2 \in I, k_1 \neq k_2$:

$$s(t^i_{k_1}, t^i_{k_2}) = \min_{l_1, l_2 \in [1..n_i]} s(t^i_{l_1}, t^i_{l_2}) \quad \text{and} \quad A(t^i_{k_1}, t^i_{k_2}) = t$$

and agglomerate them in a sub-tree $\mathcal{T}$ whose root is labelled by t;

3. Affectations $i \leftarrow i + 1$,
$T^{i+1} \leftarrow T^i - \{T^i_k \mid k \in I\} + \{\mathcal{T}\}$ and
$n_{i+1} \leftarrow n_i - |I| + 1$;

until $n_{i+1} = 1$.

Initially, $n_0 = n$ and T^0 represents the n leaves, i.e. the n object|annotation pairs. Note that, when computing the similarity between two leaves of the tree, we use eq. (2) and not eq. (3), since the annotation of a leaf is unique.

Assuming that the computation of the similarity between two terms can be done in constant time, the time complexity of the naive procedure presented above is $O(n^3)$. However, we can improve this complexity to $O(n^2 \log n)$ by using a Kruskal-like algorithm. The Kruskal algorithm [16] keeps in memory all the similarities computed at the first step, ordered increasingly, so at each next step it computes only the similarities between the new merged element and all others. This avoids to re-compute all values at each step.

Unfortunately, the assumption that the computation of the similarity between two terms could be done in constant time is not realistic. Fast computation of similarities requires precomputations on the ontology. Prior

to running ClusterInspector, we need to generate a file containing the genealogy of each term in the ontology (i.e., the full list of its ancestors up to the root), and a file containing the precision of every single term. Both files are generated by appropriate scripts (OntoGenealogy, OntoPrecision). All scripts are written in Perl and apply to any ontology in the OBO format. They are available by request from the authors, together with the graphical interface described below.

3.3 Scoring the internal nodes: the Sub-Tree Relevance Index

Each internal node of the tree represents a sub-cluster carrying an annotation. We would like to decide which of these possible sub-clusters are the most relevant ones from the point of view of the ontology considered. We propose a scoring method, which uses the precision presented previously: we define a Sub-Tree Relevance Index (SRI) in the following way:

$$\text{SRI}(T) \;=\; p(t) \times N(T) \tag{5}$$

where T represents a node of the tree, $p(t)$ the precision of its annotation, and $N(T)$ the number of leaves in the sub-tree. Note that $N(T)$ counts each object only once in the sub-tree, even if the same object appears several times associated to different annotations. When one moves from the root of the tree to the leaves, two competing effects take place: the precision of the node-annotations increases, but the size of the sub-trees decreases. Hence, a high value of the SRI means that we have a large sub-cluster annotated to a precise term, which makes this sub-tree relevant. One could also imagine other definitions for the SRI, for example a formula which emphasizes more the precision of the terms, in order to identify smaller, but more precisely annotated sub-clusters. We plan to implement other weighting schemes in the future.

Since the SRI is meant to help defining relevant subsets of objects, we need to define a threshold above which a sub-tree is considered as such. This threshold will depend on the ontology considered, but might also be influenced by the nature of the objects (e.g. proteins from different organisms). In order to evaluate the threshold, we generate a large number of random sets of objects, build the tree for each set using ClusterInspector and the ontology we are interested in, and collect the values of the SRIs at the nodes in order to build the distribution. Given the distribution, we can determine the value representing a particular percentile. As an example, we generated 500 random sets of 15 drosophila proteins[8], built the hierarchical tree for each set using ClusterInspector for the "biological process" ontology of GO and the fly_anatomy ontology. The cumulative distribution of the SRI is shown in Fig. 2, and shows that less than 0.4% of sub-nodes obtained from random sets have a SRI≥ 3 for any of these two ontologies. Hence, if we are interested in exploring sets of drosophila proteins from the point of view of these particular ontologies, any sub-tree with a SRI above 3 potentially represents an interesting set.

3.4 Graphical interface

This graphical software is a hierarchical clustering display tool. Its main objective is to lead to a fast and unambiguous visualization of the similarity between genes or proteins according to a given ontology. This tool allows loading a hierarchical clustered tree of genes (or gene products) stored in the Newick file format. The tree gets an automatic layout, following the rectangular clad tree layout algorithm (complexity in $O(n)$).

Each node of the tree is labeled by the ontology identifier. Depending on the type of the node, several other labels are added:

- The leaves are as well labeled by the name of the gene,

[8] The size of the initial set has only marginal impact on the random SRI distribution.

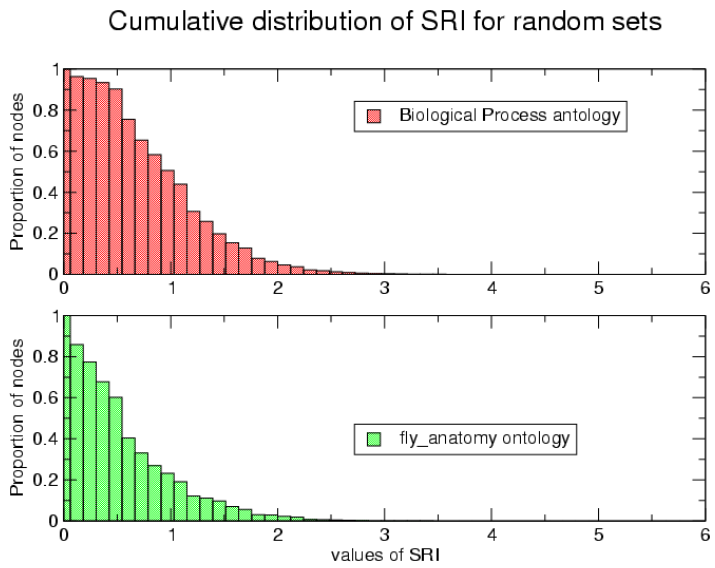

Figure 2. Cumulative distribution of the SRI for all internal nodes of hierarchical clustering tree built from random sets of 15 Drosophila proteins for the "biological process" GO ontology and the fly_anatomy ontology.

– The internal nodes are labeled by the SRI and the description of the ontological term.

The branch length follows eq. (4).

In order to display only interesting informations, the sub-trees are collapsible. It is hence possible to contract/expand a whole sub-tree by clicking on its root. Moreover, an automatic pruning mechanism is implemented, based on the SRI value: sub-Trees with SRI lower than a given value are hidden, except if they contain sub-trees with SRI greater than this value. The annotation of the nodes are displayed in pop-up boxes which appear when the mouse pointer is placed over the node. A zooming mechanism has as well been implemented.

Search functions have as well been included in this software, which enable to

– emphasize the sub-trees with the highest SRI,
– display all the genes (or gene products) of a sub-tree or the genes shared by several sub-trees.

The program is written in pure Java 1.5 and an ANTLR parser is used to parse the data from the Newick file format.

In future versions, several enhancements will be implemented in order to ease the navigation in the tree and the visualization of biological relevant informations.

4 Application

As an illustration of our method, we chose to analyze the genes which form the Toll signaling pathway in drosophila. This pathway is known to be involved both in innate response to microbial infection and to establishment of dorsal ventral polarity in Drosophila embryo through the establishment of a Dorsal gradient. The

eight genes involved are cactus (CG5848), capicua (cic), dorsal (dl), Myd88, pelle (pll), toll (Tl), tube (tub) and spätzle (spz). Building the hierarchical tree using ClusterInspector for the "biological process" ontology, we identify 5 sub-trees with a SRI ≥ 5 (see Fig. 3):

- GO:0006966 *"anti-fungal humoral response sensu Protostomia"* (SRI=6.0)
- GO:0008063 *"Toll signaling pathway"* (SRI=5.9)
- GO:0009620 *"response to fungi"* (SRI=5.5)
- GO:0009953 *"dorsal/ventral pattern formation"* (SRI=5.2)

Other sub-trees with SRI above the threshold determined earlier (i.e., 3) are either parents or children of the five mentioned sub-trees. The two main processes in which this pathway is involved clearly appear as the top ranking sub-trees. Moreover, it clearly appears that all genes but capicua and dorsal are involved in both processes, while dorsal only appears in the pattern formation sub-tree. Capicua does not appear in any of these sub-trees, but rather in a much smaller sub-tree *"embryonic development sensu Metazoan"* (SRI=0.9), which outlines the fact that this gene does not belong to the core components of the pathway.

5 Related work

Different projects have already addressed the question of integrating information from various sources to help interpret sets of biological entities. One such method, named THEA [10] (Tools for High-Throughput Experiment Analysis) was developed to help interpret clusters of genes from gene expression data. The software integrates various information sources, such as ontologies, in order to annotate groups of genes. This software takes as input a hierarchical clustering of genes based on expression profiles, and uses ontologies to annotate the nodes of this hierarchical tree. This is an important difference with our approach, where the hierarchical clustering results from the similarity measure between annotations of genes. Moreover, THEA annotates a cluster to a term if all members of the cluster are annotated (directly of through its descendents) to that term. Again, this is a difference with our approach, which is able to identify several annotations that might be relevant for a single set of genes.

Another tool, called GOToolBox [11], was developed to help analyzing Gene Ontology annotations for groups of genes. One module of GOToolBox, GO-Stats, allows to extract the most relevant GO terms associated with a set of genes. The relevance of a GO term for a given set of genes is computed using a statistical technique based on the hyper-geometrical distribution. The P-value can be corrected for multiple testing using various methods. This tool does not provide a way to display the relationship between the genes based on the ontology. In order to compare the output of GOToolBox with ClusterInspector, we investigated whether the GO terms which are ranked highest by GO-Stats correspond to the nodes with the highest SRI in ClusterInspector. Table 2 presents the list of the 30 first GO terms of the biological process ontology and lists the SRI/P-value computed by ClusterInspector/GO-Stats (see caption for details). The comparison highlights the differences between both approaches: (1) GO-Stats lists all GO terms that appear as ancestors of the terms to which the genes are annotated; hence, more than 200 terms are displayed in the GoToolBox output for this particular example. ClusterInspector selects only the terms which appear at a certain stage of the clustering procedure as the most precise ancestor of a set of sub-nodes, e.g. around 50 for this particular example. For example, the GO term rank third by GO-Stats (GO:0019732, *anti-fungal humoral response*) does not appear as a node annotation in our tree; (2) comparing the figures for the terms *"response to bacteria"* (GO:0042742, SRI=1.9, P-value=1.8E-4) and *"immune response"* (GO:0006955, SRI=1.9, P-value=8.9E-13) shows that both are rated similarly by ClusterInspector, but very differently by GO-Stats. This is because GO-Stats identifies statistically over/under-represented terms while ClusterInspector points at annotations which are represented in several instances, but not necessarily over-represented with respect to the genome background. If we have 5 out of 10

proteins which are involved in “transcription”, this does not represent a statistical bias but still represents an interesting information for characterizing the set.

Many other projects have been developed to help interpreting high-throughput data, mainly using GO: GOTreeMachine [12], GOSurfer [18], Onto-Express [13],... All these tools allow a statistical analysis of over/under-represented terms, and some display a sub-DAG of GO which covers the relevant terms. However, to our knowledge, our method is unique in that it uses a similarity measure based on the topology of the ontology. The tree we build is *not* a sub-DAG of the original ontology, since only a limited number of relevant terms appear.

As for the similarity measure of terms inside an ontology, other groups have addressed this question. The first one, [15], proposes a *semantic similarity measure* as a counterpart to sequence similarity measure. They first define a measure between GO terms, which takes into account the ”Information Content” of term by assuming a term is more informative than another if it occurs less often in annotation databases. To count the occurrences of a term, they use SWISS-PROT-Human proteins and the assumption that a term occurs if itself, or any of its children, occurs. Thus a probability of occurrence is affected to each term. This leads to the following formula: $sim(t_1, t_2) = -\ln P_{ma}(t_1, t_2)$, where $P_{ma}(t_1, t_2)$ is the probability of the common ancestor of t_1 and t_2 with minimum probability. The similarity between two genes is defined as the mean of the pairwise similarity of their annotations.

Analogously, Couto *et al.* [17] aim to define a *functional similarity* between gene-products. They first define a ”Semantic Similarity Measure” (SSM) on terms, using an idea similar to Lord *et al.*, i.e., the Information Content (IC) of a term depends on its probability of occurrence in the GOA database [14]. This formula is enhanced by the integration of depth and density of the term in the GO DAG, the whole being normalized to be in [0..1]. Hence, they define the functional similarity between two genes (FuSSiMeG) as the maximum similarity between their assigned terms, balanced by the IC of those terms:

$$\text{FuSSiMeG}(g_1, g_2) = \max_{t_1 \in \mathcal{A}(g_1), t_2 \in \mathcal{A}(g_2)} \{SSM(t_1, t_2) \times IC(t_1) \times IC(t_2)\} \tag{6}$$

where $\mathcal{A}(g)$ is the set of annotations of g. This very sophisticated formula suffers from the difficulty to adjust its parameters for depth and density.

These two approaches exclusively apply to GO, and rely more heavily on the quality of annotations, which enters the definition of similarity. On the contrary, our method can be used on any ontology and the quality of its result only depends on that of the chosen ontology. Moreover, our method, unlike these two approaches, allows to visualize the similarities between a set of genes.

6 Conclusion

In this paper, we have presented a new method to graphically interpret sets of data using ontologies. The main novelty of our approach is the fact that the relationship between objects with respect to a given ontology is determined using a similarity measure between terms of the ontology, and by extension between objects annotated to the ontology. It applies (1) to any ontology for which annotation data is available, and (2) to any kind of sets of biological entities. Its main advantages over existing methods is its simplicity, the fact that no parameters are to be specified by the user, and the quality of the graphical interface, which allows exploration of clusters at different precision levels of the ontology. We believe that it will prove useful for numerous applications in bioinformatics.

Acknowledgments: We thank B. Jacq, A. Guénoche and C. Brun for discussions and encouragements. SB acknowledges a CNRS postdoc grant in the framework of the ACI IMPBio "EIDIPP".

References

[1] Eisen MB, Spellman PT, Brown PO, Botstein D, *"Cluster analysis and display of genome-wide expression patterns"*, Proc. Natl. Acad. Sci. USA. 1998 Dec 8;95(25):14863-8

[2] Brun C, Chevenet F, Martin D, Wojcik J, Guénoche A, Jacq B, *"Functional classification of proteins for the prediction of cellular function from a protein-protein interaction network"*, Genome Biol. 2003;5(1):R6

[3] Brun C, Herrmann C, Guénoche A, *"Clustering proteins from interaction networks for the prediction of cellular functions"*, BMC Bioinformatics. 2004 Jul 13;5(1):95

[4] Vazquez A, Flammini A, Maritan A, Vespignani A, *"Global protein function prediction from protein-protein interaction networks"*, Nat Biotechnol 2003, 21(6):697-700

[5] Bader GD, Hogue CW, *"An automated method for finding molecular complexes in large protein interaction networks"*, BMC Bioinformatics 2003, 4:2

[6] Ashburner M, Mungall CJ, Lewis SE, *"Ontologies for biologists: a community model for the annotation of genomic data"*, Cold Spring Harb Symp Quant Biol. 2003;68:227-35

[7] see http://obo.sourceforge.net

[8] Bard J, Rhee SY, Ashburner M, *"An ontology for cell types"*, Genome Biol. 2005;6(2):R21. Epub 2005 Jan 14

[9] Ashburner M, et al. , *"Gene ontology: tool for the unification of biology."*, Nat Genet. 2000 May;25(1):25-9

[10] Pasquier C, Girardot F, Jevardat de Fombelle K, Christen R, *"THEA: ontology-driven analysis of microarray data"*, Bioinformatics. 2004 Nov 1;20(16):2636-43

[11] Martin D, Brun C, Remy E, Mouren P, Thieffry D, Jacq B, *"GOToolBox: functional analysis of gene datasets based on Gene Ontology"*, Genome Biol. 2004;5(12):R101

[12] Zhang B, Schmoyer D, Kirov S, Snoddy J, *"GOTree Machine (GOTM): a web-based platform for interpreting sets of interesting genes using Gene Ontology hierarchies"*, BMC Bioinformatics. 2004 Feb 18;5(1):16

[13] Draghici S, Khatri P, Bhavsar P, Shah A, Krawetz SA, Tainsky MA, *"Onto-Tools, the toolkit of the modern biologist: Onto-Express, Onto-Compare, Onto-Design and Onto-Translate"*, Nucleic Acids Res. 2003 Jul 1;31(13):3775-81

[14] Camon E, Barrell D, Lee V, Dimmer E, Apweiler R, *"The Gene Ontology Annotation (GOA) Database–an integrated resource of GO annotations to the UniProt Knowledgebase"*, In Silico Biol. 2004;4(1):5-6

[15] Lord PW, Stevens RD, Brass A, Goble CA, *"Investigating semantic similarity measures across the Gene Ontology: the relationship between sequence and annotation"*, Bioinformatics. 2003 Jul 1;19(10):1275-83

[16] Krsukal JB, *"On the shortest spanning sub-tree of a graph and the traveling salesman problem"*, Proceedings of the American Mathematical Society (1956) vol. 7, 48-50

[17] Couto FM, Silva AJ, Coutinho P, *"Implementation of a Functional Semantic Similarity Measure between Gene-Products"*, http://www.di.fc.ul.pt/tech-reports/03-29.pdf

[18] Zhong S, Tian L,Li C, Storch KF, Wong WH, *"Comparative Analysis of Gene Sets in the Gene Ontology Space under the Multiple Hypothesis Testing Framework"*, Proc. IEEE Comp Systems Bioinformatics (2004), 425-435

GO id	precision	SRI	P-value	Rank	description
GO:0006966	1.000	6.0	6.2E-18	1	antifungal humoral response (sensu Protostomia)
GO:0008063	0.848	5.9	2.2E-17	2	Toll signaling pathway
GO:0006960	0.848	5.9	5.9E-13	8	antimicrobial humoral response (sensu Protostomia)
GO:0009620	0.689	5.5	2.8E-15	5	response to fungi
GO:0009953	0.646	5.2	3.3E-14	6	dorsal/ventral pattern formation
GO:0007389	0.456	5.0	3.8E-10	15	pattern specification
GO:0009950	0.682	4.8	3.6E-12	12	dorsal/ventral axis specification
GO:0043207	0.411	4.1	1.9E-12	11	response to external biotic stimulus
GO:0000578	0.702	3.5	9.7E-10	16	embryonic axis specification
GO:0007166	0.445	3.1	1.1E-7	27	cell surface receptor linked signal transduction
GO:0009880	0.603	3.0	3.0E-8	21	embryonic pattern specification
GO:0009613	0.427	3.0	2.7E-13	7	response to pest, pathogen or parasite
GO:0009607	0.296	3.0	4.1E-8	23	response to biotic stimulus
GO:0007352	1.000	3.0	1.3E-8	19	zygotic determination of dorsal/ventral axis
GO:0006967	1.000	3.0	2.2E-8	20	antifungal polypeptide induction
GO:0007165	0.361	2.9	8.9E-6	32	signal transduction
GO:0007154	0.308	2.8	4.8E-5	36	cell communication
GO:0006952	0.310	2.8	3.3E-8	22	defense response
GO:0035172	0.848	2.5	7.7E-8	24	hemocyte proliferation (sensu Arthropoda)
GO:0050832	0.786	2.4	1.4E-15	4	defense response to fungi
GO:0030097	0.463	2.3	1.1E-7	26	hemopoiesis
GO:0007275	0.171	2.2	6.6E-6	31	development
GO:0042742	0.710	2.1	1.7E-3	57	defense response to bacteria
GO:0009887	0.270	1.9	8.6E-4	49	organogenesis
GO:0009617	0.634	1.9	2.1E-3	68	response to bacteria
GO:0006955	0.324	1.9	8.9E-13	9	immune response
GO:0006606	0.626	1.9	4.2E-4	41	protein-nucleus import
GO:0050830	0.924	1.8	5.6E-5	37	defense response to Gram-positive bacteria
GO:0006963	0.879	1.8	1.1E-4	38	antibacterial polypeptide induction
GO:0045449	0.497	1.5	1.9E-2	102	regulation of transcription

Table 2. *List of the 30 first GO "biological process" terms relevant for the 8 genes of the Toll pathway. Column 2 indicates the precision of the term, column 3 the SRI, column 4 the P-value as computed by GO-Stats, column 5 the rank according to this P-value, and column 6 the description of the term*

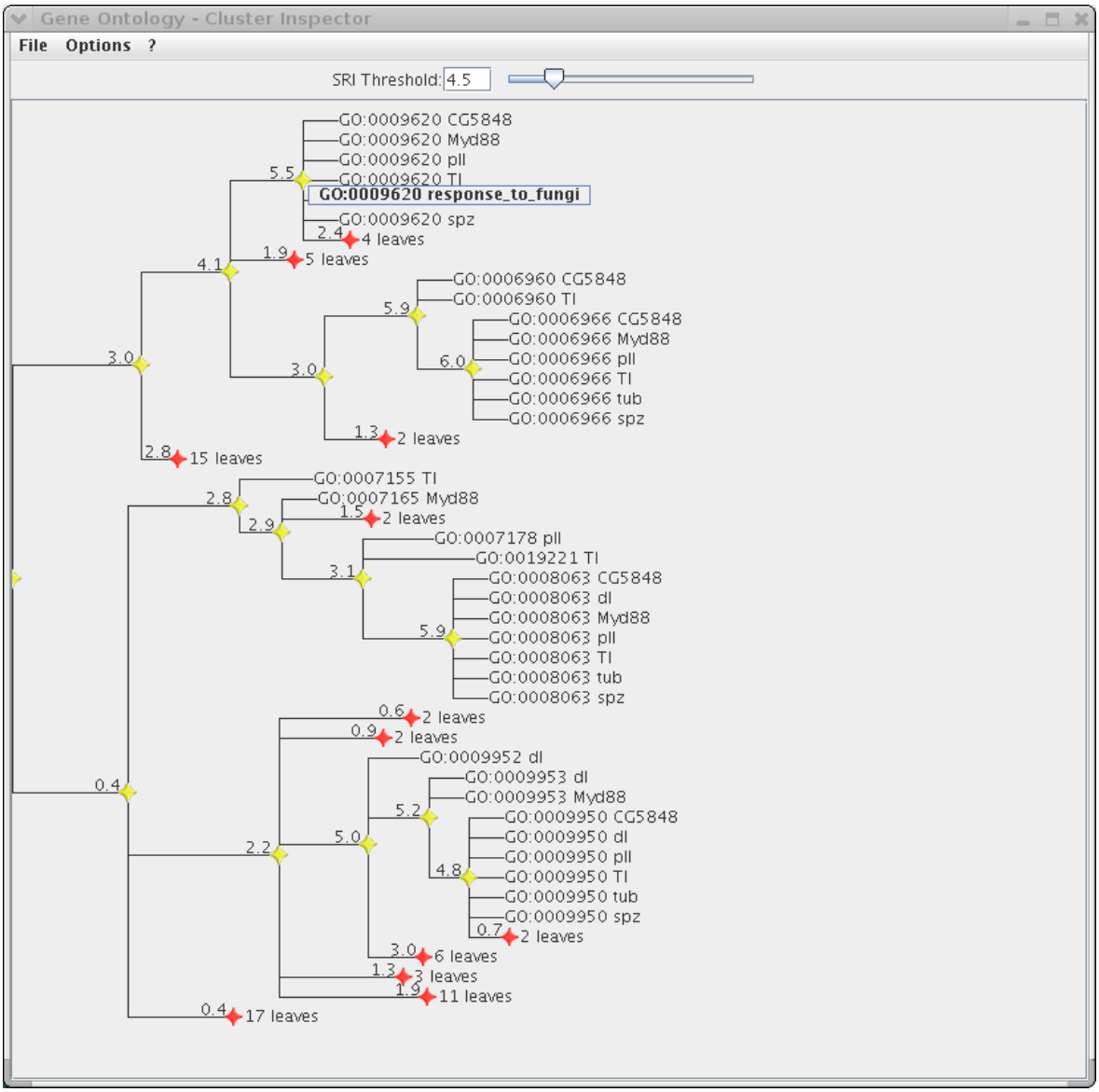

Figure 3. Screenshot of the ClusterInspector graphical interface showing the tree obtained for the Toll pathway genes with the biological process ontology. Sub-Trees with SRIs lower than 4.5 are hidden (except if they contain sub-trees with SRI greater than 4.5). Red stars indicates collapsed sub-trees and the number of underlying leaves is indicated. These sub-trees can be displayed by clicking on the node. Yellow stars indicate that a sub-tree is fully shown. Pop-up boxes display the annotation of a sub-tree, here “response to fungi”.