# Semi-Bipartite Graph Visualization for Gene Ontology Networks

Kai Xu[1], Rohan Williams[2], Seok-Hee Hong[3], Qing Liu[1], and Ji Zhang[4]

[1] CSIRO, Australia
[2] Australian National University, Australia
[3] School of Information Technologies, University of Sydney, Australia
[4] The University of Southern Queensland, Australia

**Abstract.** In this paper we propose three layout algorithms for *semi-bipartite graphs*—bipartite graphs with edges in one partition—that emerge from microarray experiment analysis. We also introduce a method that effectively reduces visual complexity by removing less informative nodes. The drawing quality and running time are evaluated with five real-world datasets, and the results show significant reduction in crossing number and total edge length. All the proposed methods are available in visualization package GEOMI [1], and are well received by domain users.

## 1 Introduction

Expression microarrays [2] have been widely used to measure *gene expression level*—the activity level of genes—in biological experiments. A typical microarray experiment involves comparing the gene expression levels of diseased (e.g. cancerous) and healthy tissue. The genes that show opposite expression levels—active in cancer tissue but not in normal, or vice versa—are known as *differentially expressed genes*. They are commonly used as candidates to study the genetic cause of the disease. The Gene Ontology [3] is a directed acyclic graph containing known gene functions (*terms*). A *parent term* describes an abstract function shared by its *child terms*, which represent more specific functions. All the edges point from the parent to its children. A gene is *annotated* by a term if it has that function, or in other words, the term is an *annotation* of the gene.

It is a rare case that the functions in the Gene Ontology provide a direct answer to the genetic cause of a disease. More complex analyses are generally required, such as how genes regulate each other, and the global function of a gene group. Many existing tools, such as the family of over-representation methods [4], only list terms that are statistically important. More recent tools, such as BiNGO [5], start to show the hierarchical structures among terms. However, genes are not shown and their performance or visual quality are not properly evaluated. In this paper, we propose novel methods that address these problems. Our main contributions are:

- Introduction of a new graph type: semi-bipartite graph;
- Proposing three layout algorithms for the semi-bipartite graphs;

- A visual complexity reduction technique for semi-bipartite graphs;
- Implementation and evaluation of the proposed methods.

## 2 Related Work

### 2.1 Gene Ontology Visualization

To date most work on Gene Ontology analysis has focused on statistical models designed to identify terms that occur at a higher proportion than random expectation [4]; there are several methods that aim to complement such analysis with visualization. In the work by Baehrecke et al. [6], a TreeMap [7] is used to display the part of the Gene Ontology hierarchy identified by the over-represented terms. The terms with multiple parents have to be duplicated under each parent to convert the Gene Ontology hierarchy into a tree. The GObar [8] uses the Graphviz package [9] to produce a layered drawing of the Gene Ontology hierarchy, but only with graphs less than 20 nodes. In SpindleViz [10] a variation of the Sugiyama method [11] is proposed to display the Gene Ontology hierarchy in three dimensions. Other hierarchy visualization methods can also be applied to the Gene Ontology, such as those surveyed by Katifori et al. [12].

All the work on Gene Ontology visualization illustrates its importance. However, the genes and how they are annotated are missing from these methods. It has been shown that such information is important to gene functional analysis [13]. This issue is attempted by Robinson et al. [14] by showing the number of annotations each term has. More details are provided in GO PaD [15], which visualizes both genes and Gene Ontology terms in a network with genes linked to their annotations. However, only a small number of selected terms, which are not experiment specific, are shown in the visualization. The BiNGO plug-in [5] for CytoScape [16] uses the gray-scale of a term color to overlay the over-representation information onto the Gene Ontology hierarchy. The genes and how they are annotated are only shown implicitly.

### 2.2 Layered Drawings and Sugiyama Method

Layered layout algorithms are natural choices for visualizing the Gene Ontology because of its hierarchical structure. Many existing algorithms are based on the 4-step framework first proposed by Sugiyama et al. [11]. Various algorithms have been proposed for each step; the details can be found in the book by Di Battista et al. [17] and Kaufmann and Wagner [18]. An important part of the Sugiyama method is the bipartite graph cross minimization problem, which is also particularly relevant to the layout algorithms we propose in this paper. Given a bipartite graph $G = (V, W, E)$ with two parallel straight-lines $L_1$ and $L_2$, a *two-layered* drawing consists of placing vertices in the vertex set $V$ on $L_1$ and $W$ on $L_2$ respectively. Each edge is a line segment joining one vertex in $V$ and one in $W$. The embedding is fully determined by the vertex orderings of $V$ and $W$. The *one-sided crossing minimization* problem has a fixed ordering of vertices in $W$ on $L_2$, and the problem is shown to be NP-complete [19, 20]. A number of heuristics, approximation, and exact algorithms have been proposed [11, 19, 21]. Eades and Wormald [19] proposed a *Median method*, which produces a 3-approximate solution. The *Barycenter method* by Sugiyama *et al.* [11] is an $O(\sqrt{n})$-approximation algorithm [19]. Currently, the best known approximation algorithm is given by Nagamochi [21] that delivers a drawing with a 1.4664 factor approximation.

## 3 Semi-Bipartite Graph and Gene-Term Network

The data used in the Gene Ontology analysis can be treated as a bipartite graph, with the genes being one partition and the ontology being the other. However, the Gene Ontology partition is a directed acyclic graph itself. To accommodate this, we define a *semi-bipartite graph* as a graph $G = (V, W, E, F)$, where $V$ and $W$ are two sets of nodes, $E$ is the set of edges between $V$ and $W$, i.e., $E = \{(v_i, w_j) \mid v_i \in V, w_j \in W\}$, and $F$ is the set of edges between the nodes in $W$, i.e., $F = \{(w_i, w_j) \mid w_i, w_j \in W\}$.

With this definition, we introduce the *Gene-Term Network* [13]. It contains two types of nodes: genes and Gene Ontology terms. The genes are those in the most differentially expressed list, and the terms are those in the *induced Gene Ontology hierarchy* (*induced hierarchy* for short) that includes terms that are annotated to the genes of interest and all their ancestors in the Gene Ontology together with induced edges. There are two types of edges: *annotation edges* that connect genes to their annotation terms and *term edges* that link the terms in the induced Gene Ontology hierarchy. The gene-term network is a semi-bipartite graph with one partition being a directed acyclic graph. Formally, given a set of genes $V = \{v_1, v_2, \ldots, v_n\}$ and the Gene Ontology hierarchy $GO = \{GO_T, GO_E\}$ (where $GO_T$ and $GO_E$ are the set of Gene Ontology terms and edges respectively), a gene-term network is a semi-bipartite graph: $G = (V, W, E, F)$ where

- $W = \{w \mid w \in GO_T, \exists v \in V, w \in t(v) \textit{ or } w \in a(v)\}$, where $t(v)$ is the set of annotations of a gene $v$ and $a(v)$ is the set of ancestor terms of $t(v)$.
- $E = \{(v, w) \mid v \in V, w \in t(v)\}$ is a set of *annotation edges* linking genes and their annotations;
- $F = GO_E \cap (W \times W)$ is a set of *term edges* induced from $GO$.
- The induced hierarchy $P = (W, F)$ is a directed acyclic graph.

## 4 Layout Algorithms for Semi-Bipartite Graphs

### 4.1 Extended Bipartite Algorithms

This algorithm extends the barycenter method [11] and starts by drawing the induced hierarchy $P$ using the Sugiyama method, treating it as the partition with fixed order. The gene nodes $V$ are placed on a parallel layer and ordered according to the horizontal position of their annotation terms to minimize inter-partition edge crossings (we assume the drawing is top-down with horizontal layers and the same applies to other algorithms in this paper). The gene nodes are placed on the bottom layer because many of them annotate to the leaf terms of $P$; placing gene nodes above the hierarchy will introduce extra edge crossings. Please refer to [22] for algorithm details.

The running time of the algorithm is the sum of that of the Sugiyama method and barycenter computation. The former depends on the algorithms used for the various stages. In our implementation, we use the layering method by Gansner et al. [23] that minimizes the number of dummy nodes and requires polynomial time in the worst case. The crossing reduction algorithm is based on the median method [19] that runs in linear time ($O(|W| + |F|)$). Finally, the horizontal coordinate assignment uses a heuristic algorithm based on network simplex formulation [24] that requires polynomial time in the worst case. The barycenter can be computed in time linear to the size of gene node set and the number of annotation edges [19], i.e., $O(|V| + |E|)$. Therefore, the overall running time is $O(T(|W|, |F|)) + O(|V| + |E|)$, where $O(T(|W|, |F|))$ is the running time of the Sugiyama method for the induced hierarchy $P$.

### 4.2 Sub-Hierarchy Barycenter Algorithm

This algorithm aims to reduce crossings caused by annotation edges and achieves this by adjusting the child term order in the induced hierarchy according to the gene node order. It is based on the following observations:

- Assume a term $w$ has two child terms $w_1$ and $w_2$ with $w_1$ to the left of $w_2$ in the drawing. If most genes annotated to $w_1$ are to the right of $w_2$, changing the order of $w_1$ and $w_2$ is likely to reduce edge crossing. However, such a change will also affect all the descendants of $w_1$ and $w_2$. Therefore, the decision to change the order should be based on how genes are annotated to the sub-hierarchy rooted at $w_1$ and $w_2$.
- Following the previous example, assume $w_1$ and $w_2$ both have two child terms. The order change between $w_1$ and $w_2$ will affect the order among their child terms, but the reverse is not true. Therefore, the change of child order should be breadth-first.

Before describing the algorithm, we need to introduce *sub-hierarchy barycenter*, which is based on the position of all the gene nodes annotated to the terms in the sub-hierarchy rooted at a term (including the term itself). Formally, for a term $w$, its sub-hierarchy barycenter $b(w)$ is:

$$b(w) = average(x(v_i)), v_i \in V, \exists w_j \in sub(w), (v_i, w_j) \in F$$

where $average()$ computes the average value, $x(v_i)$ is the $x$-coordinate of $v_i$, and $sub(w)$ is the sub-hierarchy in the induced hierarchy rooted at $w$.

The *sub-hierarchy barycenter algorithm* starts with drawing the gene-term network with the extended bipartite algorithm. Then, it traverses the induced hierarchy bottom up to compute the sub-hierarchy barycenter for each term. After that, it traverses the induced hierarchy again breadth first to re-order the child terms accordingly. Finally, the gene nodes are re-ordered according to the new child term order. The whole process is repeated a fixed number of times. Because there is a total ordering of sub-hierarchy barycenter, there will be no ordering conflict when a term has multiple parents. Similarly, common child terms only need to be sorted once when they are shared by multiple parents. The details of the algorithm can be found in [22].

Besides the extended bipartite algorithm and Sugiyama method, the running time consists of three parts: that of the sub-hierarchy barycenter computation, the child term re-ordering, and the gene nodes ordering. During barycenter computation, each term $w$ is visited $1+|parent(w)|$ times ($|parent(w)|$ is the number of parents of $w$): once for computing its own barycenter; $|parent(w)|$ times for all its parents. Because each term edge is visited exactly once when computing the barycenter of the parent term, the total number of term visits is:

$$\sum_{w\in W}(|parent(w)|+1) = \sum_{w\in W}|parent(w)| + \sum_{w\in W} = |F| + |W|$$

During barycenter computation, each gene $v$ is visited $|t(v)|$ times: once for each term it annotates to. Therefore the total number of gene visits is $\sum_{v\in V} t(v) = |E|$, and the running time for sub-hierarchy barycenter computation is $O(|E| + |F| + |W|)$. For child term re-ordering, each term is only ordered once, so the running time is expected to be $O(|W|\log|W|)$. Similarly, the running time of gene nodes re-ordering is $O(|V|\log|V|)$. Therefore, the overall running time of the sub-hierarchy barycenter algorithm is $O(T(|W|,|F|)) + O(|E| + |F|) + O(|W|\log|W|) + O(|V|\log|V|)$.

### 4.3 Partition Merge Algorithm

In the previous algorithm, the crossing reduction is achieved by changing the induced hierarchy embedding and gene node ordering separately. The *partition merge algorithm* makes global crossing reduction possible by merging two partitions. There are two variations of the algorithm. The first one places all gene nodes on the layer beneath the induced hierarchy, which is achieved by assigning the direction of all annotation edges pointing to gene nodes. No change is required for the layered layout algorithm if the layering step starts with the sinks (gene nodes), such as the Longest Path Layering [17], otherwise a constraint is required that all gene nodes must be placed on the bottom layer.

Relaxing the bottom layer constraint can potentially reduce the total edge length, which is another factor important to graph readability [25]. To avoid comparing all possible edge direction assignment permutations, we propose a *level barycenter* heuristic that assigns a gene to a layer that is the average of the layer of its annotation terms. When term layer equals level barycenter value, the edge direction is from term to gene so that gene with one annotation is placed under the term. The algorithm details can be found in [22]. Moving gene nodes up the hierarchy may introduce extra edge crossings, but it is counter balanced by shorter annotation edges that are less likely to intersect with other edges. Overall, the level barycenter algorithm performs very well in terms of edge crossings. Please refer to the evaluation results in Section 6 for the details.

The running time of the first variation of the Partition Merge algorithm is similar to that of the Extended Bipartite algorithm with the addition of the edge direction assignment step, which requires $O(|E|)$ time. Therefore, its total running time is $O(|E|) + O(T(|V + W|, |E + F|))$. For the second variation, the level barycenter computation requires $O(|V| + |E|)$ time and then the direction assignment takes $O(|E|)$ time, which leads to a total running time of $O(|V| + |E|) + O(T(|V + W|, |E + F|))$.

## 5 Term Reduction

During this study we found that as the data size increases, the size of the induced hierarchy increases at a much higher rate than that of the gene nodes. The reason is that each new gene usually annotates to several new terms, and each new term in turn introduces several new ancestors. In an effort to reduce the size of the induced hierarchy, we observe that many terms in the top part of the hierarchy are abstract and not informative for functional analysis, so they can be removed to reduce visual complexity. To identify such terms, we introduce the concept of the *subordinate term*, which are Gene Ontology terms that have no gene annotated to them and do not show new relationships between terms with gene annotation. Formally, we define the *indirect gene set* $g'(w)$ of a term $w$ as the set of genes annotated to the term in its sub-hierarchy but not the term itself, i.e.,

$$g'(w) = \{v_i \mid v_i \in V, \exists\ w_j \in sub(w), (v, w_j) \in E, w_j \neq w\}$$

Similarly, the *direct gene set* of a term is the set of genes annotate to it, i.e.,

$$g(w) = \{v_i \mid v_i \in V, (v, w) \in E\}$$

Now, we can define a term $w$ is *subordinate* if

1. $g(w) = \emptyset$, and
2. $\exists\ w_i \in sub(w), g'(w_i) = g'(w), w_i \neq w$

This means that a subordinate term does not have any genes annotated to it, and its indirect gene set is the same as one of its descendants'. The latter means the term is not the first common ancestor of two terms with direct gene annotated to them, i.e., is not of structural importance in function analysis.

However, there are many cases in Gene Ontology where a term can be both a parent and grandparent of the same term: term $w_1$ is a parent of term $w_2$ and $w_3$, and $w_2$ is also a parent of $w_3$. In this example, $w_1$ always has the same indirect gene set as $w_2$ if it has no other child term. According to the previous definition, $w_1$ is a subordinate term if no genes are annotated to it. However, these cases are considered biologically important under some conditions. To accommodate such cases, we define that a term is *semi-subordinate* if

1. $g(w) = \emptyset$, and

2. $\forall\ w_i \in child(w), g'(w_i) = g'(w)$.

The revised Condition 2 means that a term is semi-subordinate only if its indirect gene set is the same as all of its childrens.

The algorithm of identifying subordinate terms traverses the induced hierarchy bottom up. It starts with a queue that contains only the leaf terms. Once a term is checked, all its parents are appended to the queue, and a term is always checked after all its child terms. The indirect gene set of a term is the union of the direct gene set of all its children. A term is subordinate if its indirect gene set coincides with one of its childrens, and its direct gene set is empty. The process is repeated until the queue is empty. The algorithm details are available in [22]. The only change required for semi-subordinate terms is the check condition.

Similar to the sub-hierarchy barycenter algorithm, each term $w$ is visited $1+|parent(w)|$ times: once for computing its indirect gene set, $|parent(w)|$ times for all its parents. Each gene $v$ is also visited $|t(v)|$ times: once for each term it annotates to. Note that the coincidence check can be done in constant time, because the indirect gene set of a child is always a subset of that of its parent and equal size is sufficient to show that they coincide. Therefore, the total running time is $O(|E|+|F|+|W|)$. The semi-subordinate algorithm has the same running time because the change it needs does not affect running time.

# 6 Evaluation

## 6.1 Dataset

The five sets of genes used in the evaluation were generated from the results of 91 microarray experiments with each measuring about 22000 mouse genes. Full details of the experiment and its findings can be found in [26]. Table 1 shows the size of the gene-ontology network for every dataset.

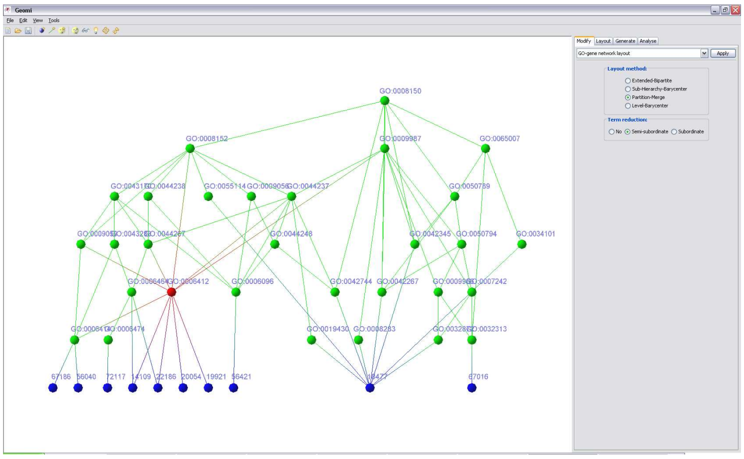
| Dataset | Total node $|V|+|W|$ | Total edge $|E|+|F|$ | Gene node $|V|$ | Term node $|W|$ | Annotation edge $|E|$ | Term edge $|F|$ |
|---|---|---|---|---|---|---|
| *cgg5* | 49 | 65 | 4 | 45 | 7 | 58 |
| *cgg4* | 150 | 243 | 10 | 140 | 19 | 224 |
| *cgg3* | 221 | 394 | 13 | 208 | 38 | 356 |
| *cgg2* | 374 | 627 | 43 | 331 | 89 | 538 |
| *cgg1* | 447 | 864 | 25 | 422 | 81 | 783 |

**Table 1.** Dataset size

## 6.2 Implementation

All algorithms and the term reduction method are implemented in GEOMI [1], which is a Java-based graph visualization system. Figure 1 is the user interface:

The layout is shown in the left panel with navigation functions including zoom in/out, pan, and rotate. In the right panel, users can choose among different layouts and two levels of term reduction. The blue nodes are genes, green nodes are terms, and the red node is the over-represented term. The same implementation of the Sugiyama method is used in all the algorithms, and the method used in each step is the same as described in Section 4.1. In the sub-hierarchy barycenter algorithm, the re-ordering of the child terms and gene nodes are repeated 20 times to find the embedding with minimal number of edge crossings.



GO-gene network layout

Layout method:

○ Extended-Bipartite
○ Sub-Hierarchy-Barycenter
● Partition-Merge
○ Level-Barycenter

Term reduction:

○ No ● Semi-subordinate ○ Subordinate

**Fig. 1.** The GEOMI plug-in with all four algorithms and term reduction (gene: blue, term: green, over-represented: red). User can choose the layout and term reduction from the panel on the right.

### 6.3 Layout Quality

We use edge crossings as one of the visual quality measurements, because it has been shown to be the most important layout aesthetics [25] for graph readability. Table 2 shows the number of edge crossings of the different layout methods. The two variations of the partition merge algorithm are shown as Partition Merge (all the gene nodes on the bottom layer) and Level Barycenter respectively. The extended bipartite algorithm—the baseline method—is outperformed by the other algorithms in all cases. The sub-hierarchy barycenter, partition merge, and level barycenter algorithm have similar performance, providing about 20% less edge crossings than the extended bipartite algorithm. The sub-hierarchy barycenter performs better for the larger datasets (*cgg2* and *cgg1*) than the smaller ones (*cgg4* and *cgg3*), while the opposite is true for the partition merge and level barycenter algorithm. The latter performed well with regards to edge crossings given its main goal is to reduce total edge length.

| Dataset | Extended Bipartite | Sub-Hierarchy Barycenter | Partition Merge | Level Barycenter |
|---|---|---|---|---|
| *cgg5* | 3 | 0 | 0 | 0 |
| *cgg4* | 162 | 135 | 119 | 126 |
| *cgg3* | 870 | 798 | 605 | 646 |
| *cgg2* | 2542 | 2028 | 1929 | 1918 |
| *cgg1* | 4164 | 3440 | 3883 | 3765 |

**Table 2.** Edge crossings of different layout methods

The total edge lengths are shown in Figure 2. Term edges are not included in the tests, because level barycenter algorithm has little impact on them. It is clear that the level barycenter provides a considerable reduction in total edge length and is more effective as the graph size increases.

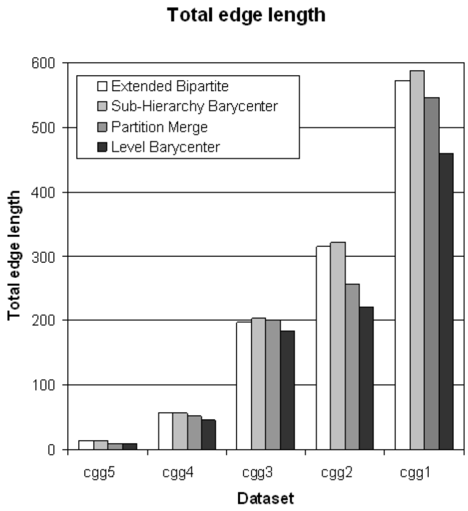


**Fig. 2.** The total annotation edge length

### 6.4 Term Reduction

The results of the term reduction are shown in Figure 3. Gene nodes are not included because they are not affected and the induced hierarchy accounts for the majority of the gene-term network. Term reduction is effective in reducing network size: the number of term nodes is reduced to at least half of what they were. Besides reduced graph size, other benefits of term reduction include less edge crossings and faster running time for the layout algorithms.

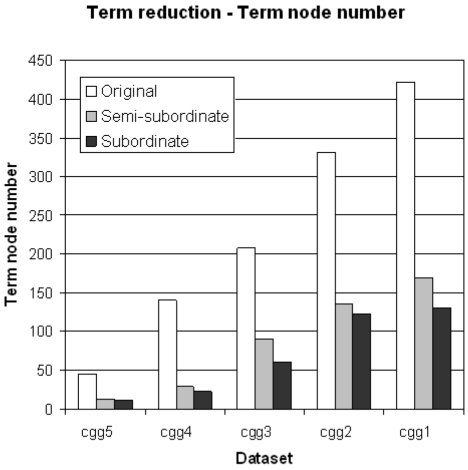


**Fig. 3.** The number of term nodes after term reduction

### 6.5 Running Time

Table 3 shows the running time of the four algorithms. All the tests were performed on a laptop equipped with an Intel Core 2 Duo 2.5 GHz CPU with 2 GB memory. The time is the average of 20 repetitions. In most cases all algorithms completed within one second, which means they scale well to graph with 447 nodes and 864 edges (*cgg1*). The relative speed among the algorithms is consistent with the running time analysis in Section 4.

| Dataset | Extended Bipartite | Sub-Hierarchy Barycenter | Partition Merge | Level Barycenter |
|---|---|---|---|---|
| *cgg5* | 0.128 | 0.406 | 0.11 | 0.25 |
| *cgg4* | 0.126 | 0.625 | 0.109 | 0.328 |
| *cgg3* | 0.338 | 0.719 | 0.297 | 0.453 |
| *cgg2* | 0.378 | 0.969 | 0.375 | 0.609 |
| *cgg1* | 0.607 | 1.375 | 0.594 | 1.016 |

**Table 3.** Running time of the algorithms in seconds.

### 6.6 User Feedback

A pilot user study has been conducted to gain feedback from domain experts. We asked two domain experts: one bioinformatician with statistics background and one molecular biologist, who are the most likely users. The users were asked to compare the proposed methods (using GEOMI) against the way they currently perform function analysis. They found the extra information gene-term network provides very useful. The layout produced by the sub-hierarchy barycenter and partition merge algorithm have similar visual quality; both are better than that of the extended bipartite algorithm. Term reduction is effective and necessary, especially when the dataset is large. One possible improvement is to keep the term levels as they are in the original Gene Ontology hierarchy, which is not the case currently.

## 7 Conclusions

In this paper we introduced the semi-bipartite graph for the visual analysis of microarray experiments using the Gene Ontology. Among the proposed layout algorithms, sub-hierarchy barycenter, partition merge, and level barycenter produce considerably less edge crossings than the baseline method—extended bipartite. The result of the level barycenter algorithm has the least total edge length. All the algorithms take less than or close to one second to complete for graphs with size up to 447 nodes and 864 edges. The term reduction technique effectively reduces graph size by more than half. All the methods have been implemented in GEOMI and well received by the domain experts.

## References

1. Ahmed, A., Dwyer, T., Forster, M., Fu, X., Ho, J., Hong, S.H., Koschützki, D., Murray, C., Nikolov, N.S., Taib, R., Tarassov, A., Xu, K.: GEOMI: Geometry for maximum insight. In Healy, P., Nikolov, N.S., eds.: Proceedings of the 13th International Symposium on Graph Drawing. Volume 3843 of Lecture Notes in Computer Science., Springer (2005) 468–479
2. Schena, M., Shalon, D., Davis, R.W., Brown, P.O.: Quantitative monitoring of gene expression patterns with a complementary dna microarray. Science **270**(5235) (October 1995) 467–470
3. Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T., Harris, M.A., Hill, D.P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J.C., Richardson, J.E., Ringwald, M., Rubin, G.M., Sherlock, G.: Gene ontology: tool for the unification of biology. Nature Genetics **25**(1) (May 2000) 25–29
4. Khatri, P., Draghici, S.: Ontological analysis of gene expression data: current tools, limitations, and open problems. Bioinformatics **21**(18) (Sep 2005) 3587–3595
5. Maere, S., Heymans, K., Kuiper, M.: Bingo: a cytoscape plugin to assess over-representation of gene ontology categories in biological networks. Bioinformatics **21**(16) (2005) 3448–3449
6. Baehrecke, E.H., Dang, N., Babaria, K., Shneiderman, B.: Visualization and analysis of microarray and gene ontology data with treemaps. BMC Bioinformatics **5** (2004) 84
7. Shneiderman, B.: Tree visualization with treemaps: A 2d space-filling approach. ACM Transactions on Graphics **11**(1) (1992) 92–99
8. Lee, J.S.M., Katari, G., Sachidanandam, R.: GObar: A gene ontology based analysis and visualization tool for gene sets. BMC Bioinformatics **6** (2005) 189

9. Gansner, E.R., North, S.C.: An open graph visualization system and its applications to software engineering. Software Practice and Experience **30**(11) (2000) 1203–1233
10. Joslyn, C.A., Mniszewski, S.M., Smith, S.A., Weber, P.M.: Spindleviz: A three dimensional, order theoretical visualization environment for the gene ontology. In: Proceedings of Joint BioLINK and 9th Bio-Ontologies Meeting. (2006)
11. Sugiyama, K., Tagawa, S., Toda, M.: Methods for visual understanding of hierarchical system structures. IEEE Transactions on Systems, Man, and Cybernetics **11**(2) (1981) 109–125
12. Katifori, A., Halatsis, C., Lepouras, G., Vassilakis, C., Giannopoulou, E.: Ontology visualization methods—a survey. ACM Comput. Surv. **39**(4) (2007) 10
13. Xu, K., Huang, X.X., Cotsapas, C., Hong, S.H., McCaughan, G., Gorrell, M., Little, P., Williams, R.: Combined visualisation and analysis of gene ontology annotations using multivariate representations of annotations and bipartite networks. Technical Report 09/166, CSIRO (2009)
14. Robinson, P.N., Wollstein, A., Bohme, U., Beattie, B.: Ontologizing gene-expression microarray data: characterizing clusters with gene ontology. Bioinformatics **20**(6) (2004) 979–981
15. Alterovitz, G., Xiang, M., Mohan, M., Ramoni, M.F.: GO PaD: the gene ontology partition database. Nucleic Acids Research **35** (2007) D322–D327
16. Shannon, P., Markiel, A., Ozier, O., Baliga, N.S., Wang, J.T., Ramage, D., Amin, N., Schwikowski, B., Ideker, T.: Cytoscape: A software environment for integrated models of biomolecular interaction networks. Genome Research **13** (2003) 2498–2504
17. Battista, G.D., Eades, P., Tamassia, R., Tollis, I.G.: Graph Drawing: Algorithms for the Visualization of Graphs. Prentice Hall (1999)
18. Kaufmann, M., Wagner, D., eds.: Drawing Graphs, Methods and Models. In Kaufmann, M., Wagner, D., eds.: Drawing Graphs. Volume 2025 of Lecture Notes in Computer Science., Springer (2001)
19. Eades, P., Wormald, N.C.: Edge crossings in drawings of bipartite graphs. Algorithmica **11**(4) (1994) 379–403
20. Garey, M.R., Johnson, D.S.: Crossing number is NP-complete. SIAM J. Algebraic and Discrete Methods **4**(3) (1983) 312–316
21. Nagamochi, H.: An improved bound on the one-sided minimum crossing number in two-layered drawings. Discrete & Computational Geometry **33**(4) (2005) 569–591
22. Xu, K., Williams, R., Hong, S.H., Liu, Q., Zhang, J.: Semi-bipartite graph visualization for gene ontology networks. Technical Report EP091883, CSIRO (2009) http://www.it.usyd.edu.au/ visual/kaixu/papers/gene-ontology-layout.pdf.
23. Gansner, E.R., Koutsofios, E., North, S.C., Vo, K.P.: A technique for drawing directed graphs. IEEE Transactions on Software Engineering **19**(3) (1993) 214–230
24. Chvatal, V.: Linear Programming. W. H. Freeman (1983)
25. Purchase, H.C.: Which aesthetic has the greatest effect on human understanding? In: Proceedings of the 5th International Symposium on Graph Drawing, London, UK, Springer-Verlag (1997) 248–261
26. Cowley, M.J., Cotsapas, C.J., Williams, R.B.H., Chan, E.K.F., Pulvers, J.N., Liu, M.Y., Luo, O.J., Nott, D.J., Little, P.F.R.: Intra- and inter-individual genetic differences in gene expression. Nature Precedings **http://hdl.handle.net/10101/npre.2008.1799.1** (2008)