BIOINFORMATICS APPLICATIONS NOTE
Vol. 21 no. 2 2005, pages 275–276
doi:10.1093/bioinformatics/bth495

OntologyTraverser: an R package for GO analysis

A. Young, N. Whitehouse, J. Cho and C. Shaw*

Baylor College of Medicine, Department of Molecular and Human Genetics, Houston, TX, USA

Received on April 2, 2004; revised on August 18, 2004; accepted on August 19, 2004
Advance Access publication August 27, 2004

ABSTRACT

Summary: Gene Ontology (GO) annotations have become a major tool for analysis of genome-scale experiments. We have created OntologyTraverser—an R package for GO analysis of gene lists. Our system is a major advance over previous work because (1) the system can be installed as an R package, (2) the system uses Java to instantiate the GO structure and the SJava system to integrate R and Java and (3) the system is also deployed as a publicly available web tool.

Availability: Our software is academically available through http://franklin.imgen.bcm.tmc.edu/OntologyTraverser/. Both the R package and the web tool are accessible.

Contact: cashaw@bcm.tmc.edu

Genome-scale experiments, such as microarray studies and large-scale library sequencing generate complex and difficult to interpret lists of genes. Biological annotations group individual genes into coherent categories, and annotation-based evaluation of results can be more interpretable than single gene based analysis. Many annotation systems are available. Perhaps the most ambitious scheme is the Gene Ontology project (GO, http://www.geneontology.org). We have developed OntologyTraverser—an R-based and web-deployed analysis system that uses GO annotations.

Many other groups have also developed analysis tools for GO-based consideration of gene lists (Dennis *et al*., 2003; Al-Shahrour *et al*., 2004; Beissbarth and Speed, 2004). The OntologyTraverser has many benefits over these existing methods. First, the OntologyTraverser is an R based system—an R package that can function with the existing open source R software (http://R-cran.org). Second, the system provides statistical testing and reporting of results at each GO node—an advance over other software that require the preselection of levels within the GO for analysis. Third, the system is deployed as a web tool to provide open community access through a web-browser interface.

GO-BASED ANALYSIS

The GO aims to locate gene-products (genes) to a 3-fold nested vocabulary of biological terms. Each gene may have a collection of terminal annotations to nodes within the vocabulary. Since the vocabulary is nested, information is also contained in the paths through the vocabulary to these terminal annotations. Genes with annotations at or below a term in the vocabulary share evidence for the term's biological property.

Experimental results are often distilled to lists of genes with unusual behavior. The experiment-derived lists can be mapped to the GO data structure. The GO structure can be instantiated, and the paths through the GO vocabulary to the terminal annotations for the genes in the list can be recovered. The path information can be tabulated, and counts can be determined for the number of paths determined by the list that pass through each node in the GO.

ARCHITECTURE AND IMPLEMENTATION

Our system, OntologyTraverser, is an R package composed of two loosely coupled and reusable components: a pure R component and a Java component. The R element is the wrapper for the software. R handles the gene list identifiers, static data needed to associate genes with terminal GO annotations and the statistical analysis of the GO traversal results. The Java component handles the actual instantiation of the GO data structure. Java also handles the tree-traversal to locate terminal GO nodes and to determine the paths through the ontology to reach those nodes.

The design choices in OntologyTraverser offer a real advantage over existing methods. The R element provides rapid and flexible implementation and integrates with existing tools like Bioconductor. However, pure R is not particularly suited to instantiation of the GO vocabulary and rapid traversal of the nested structure. Java is an ideal choice given the extensive toolkit of XML parsing tools and Java's ability to handle the large GO data structure. Computation is rapid, and the calculation of results for a gene list comprising of an entire 20 000 gene Affymetrix microarray requires <5 min on our server.

STATISTICS FOR COUNTS

Statistical consideration of GO results requires analysis of the counts at each GO node. Our system considers the paths from the root to each terminal annotation for genes

*To whom correspondence should be addressed.

in the list. The gene list determines the terminal annotations, and the path counts are obtained by traversing the GO structure to obtain counts at or below each GO node. A graphic depicting our approach appears on our website, http://franklin.imgen.bcm.tmc.edu/OntologyTraverser/

The counts for the gene list are compared to reference counts derived from a reference gene list. In the default case, the reference list is the entire probe set printed on the microarray, but our software will accommodate any reference list. Enrichment analysis at each GO node is performed under a statistical sampling model for the counts. The most common model in use is the hyper-geometric or sampling without replacement model for the counts. This approach is also called Fisher's Exact test because of its historical roots in the testing of counts from 2×2 tables. This null model supposes that counts at each node are sampled at random from the available possible counts determined by the reference list. As counts become large, the binomial distribution can be used to approximate the hyper-geometric model.

Two types of counts may be considered in the analysis: GO term counts and probe counts. The GO term approach treats each GO annotation as an independent object to be sampled. Because a given gene may be annotated to many GO identifiers, a single gene in the list can generate a multiplicity of individual GO counts. The multiple GO counts spawned by a single gene in the gene list has proved problematic for the sampling without replacement model; others have also noted this dependency (Al-Shahrour *et al*., 2004; Beissbarth and Speed, 2004). In our software, we count the unique set of probe identifiers mapped at or below each GO node to determine the parameters in the hyper-geometric model. We find that this approach eliminates some of the dependency in the counts and makes the hyper-geometric model more appropriate.

As in other scenarios with genome-scale data, the GO results present a large multiple testing problem. The individual P-values obtained at each GO node must be adjusted to account for the multiple comparison problem. We provide two P-values at each GO node: a raw, unadjusted marginal P-value and a linear step up Benjamini–Hochberg P-value to control FDR (Benjamini, 1995). Since our system is implemented in R, end-users can add extra P-value adjustments. We also provide a fold-change statistic. The fold-change statistic considers the fold-enrichment of the node under study, normalizing the probe counts into frequencies by calculating the total number of probes annotated through the GO level.

REPORTS

The display of GO results is a challenge. We have implemented several report formats. First, we provide TSV and HTML summaries in a tabular format where each row represents a GO node. We have also implemented an interactive, nested report for end-users using XML and XSL style sheets. The reports generated from XML and XSL style sheets permit interactivity; these reports implement Javascript mouse-over pop-up boxes with text describing the statistical results at each node. All formats are available through the web tool at http://franklin.imgen.bcm.tmc.edu/OntologyTraverser/

AVAILABILITY

Our software is accessible as a web tool and as a downloadable R package under the GPL2 license. The web tool functions through a browser interface. The R package depends on the Sjava system (Lang, 2000) and also the R.oo package (Bengtsson, 2003). Access to our software and additional information is available at http://franklin.imgen.bcm.tmc.edu/OntologyTraverser/

ACKNOWLEDGEMENT

This work was supported by NIH-U01 DK63588-01.

REFERENCES

Al-Shahrour,F., Diaz-Uriarte,R. and Dopazo,J. (2004) FatiGO: a web tool for finding significant associations of Gene Ontology terms with groups of genes. *Bioinformatics*, **20**, 578–580.

Beissbarth,T. and Speed,T. (2004) GOstat: find statistically overrepresented Gene Ontologies within a group of genes. *Bioinformatics*, **20**, 1464–1465.

Bengtsson,H. (2003) The R.oo package—Object-Oriented Programming with References Using Standard R Code. In *Proceedings of the 3rd International Workshop on Distributed Statistical Computing*. http://www.ci.tuwien.ac.at/Conferences/DSC-2003/Proceedings/

Benjamini,Y.a.H.Y. (1995) Controlling the false discovery rate: a practical and powerful approach to multiple testing. *J. R. Stat. Soc. B*, **57**, 289–300.

Dennis,G.,Jr, Sherman,B.T., Hosack,D.A., Yang,J., Gao,W., Lane,H.C. and Lempicki,R.A. (2003) DAVID: Database for Annotation, Visualization, and Integrated Discovery. *Genome Biol.*, **4**, 3.

Lang,D.T. (2000) The Omegahat environment: new possibilities for statistical computing. *J. Comput. Stat. Graph.*, **9**, 423–451.