Introduction

In this vignette we describe some of the basic characteristics of the data available from the Gene Ontology (GO), [The Gene Ontology Consortium 2000] and how these data have been incorporated into Bioconductor. We assume that readers are familiar with the basic DAG structure of GO and with the mappings of genes to GO terms that are provide by GOA [Camon et al. 2004]. We consider these basic structures and properties quite briefly.

GO, itself, is a structured terminology. The ontology describes genes and gene products and is divided into three separate ontologies. One for cellular component (CC), one for molecular function (MF) and one for biological process (BP). We maintain those same distinctions where appropriate. The relationship between terms is a parent-child one, where the parents of any term are less specific than the child. The mapping in either direction can be one to many (so a child may have many parents and a parent may have many children). There is a single root node for all ontologies as well as separate root nodes for each of the three ontologies named above. These terms are structured as a directed acyclic graph (or a DAG).

GO itself is only the collection of terms; the descriptions of genes, gene products, what they do, where they do it and so on. But there is no direct association of genes to terms. The assignment of genes to terms is carried out by others, in particular the GOA project [Camon et al. 2004]. It is this assignment that makes GO useful for data analysis and hence it is the combined relationship between the structure of the terms and the assignment of genes to terms that is the concern of the GO.db package.

The basis for child-parent relationships in GO can be either an is-a relationship, where the child term is a more specific version of the parent. Or, it can be a has-a, or part-of relationship where the child is a part of the parent. For example a telomere is a part-of a chromosome.

Genes are assigned to terms on the basis of their LocusLink ID. For this reason we make most of our mappings and functions work for LocusLink identifiers. Users of specific chips, or data with other gene identifiers should first map their identifiers to LocusLink before using GOstats.

A gene is mapped only to the most specific terms that are applicable to it (in each ontology). Then, all less specific terms are also applicable and they are easily obtained by traversing the set of parent relationships down to the root node. In practice many of these mappings are
precomputed and easily obtained from the different hash tables provided by the GO.db package.

Mapping of a gene to a term can be based on many different things. GO and GOA provide an extensive set of evidence codes, some of which are given in Table 1, but readers are referred to the GO web site and the documentation for the GO.db package for a more comprehensive listing. Clearly for some investigations one will want to exclude genes that were mapped according to some of the evidence codes.

<table>
<thead>
<tr>
<th>Evidence Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMP</td>
<td>inferred from mutant phenotype</td>
</tr>
<tr>
<td>IGI</td>
<td>inferred from genetic interaction</td>
</tr>
<tr>
<td>IPI</td>
<td>inferred from physical interaction</td>
</tr>
<tr>
<td>ISS</td>
<td>inferred from sequence similarity</td>
</tr>
<tr>
<td>IDA</td>
<td>inferred from direct assay</td>
</tr>
<tr>
<td>IEP</td>
<td>inferred from expression pattern</td>
</tr>
<tr>
<td>IEA</td>
<td>inferred from electronic annotation</td>
</tr>
<tr>
<td>TAS</td>
<td>traceable author statement</td>
</tr>
<tr>
<td>NAS</td>
<td>non-traceable author statement</td>
</tr>
<tr>
<td>ND</td>
<td>no biological data available</td>
</tr>
<tr>
<td>IC</td>
<td>inferred by curator</td>
</tr>
</tbody>
</table>

Table 1: GO Evidence Codes

In some sense TAS is probably the most reliable of the mappings. IEA is a weak association and is based on electronic information, no human curator has examined or confirmed this association. As we shall see later, IEA is also the most common evidence code.

The sets of mappings of interest are roughly divided into three parts. First there is the basic description of the terms etc., these are provided in the GOTERMS hash table. Each element of this hash table is named using its GO identifier (these are all of the form GO: followed by seven digits). Each element is an instance of the GOTerms class. A complete description of this class can be obtained from the appropriate manual page (use class?GOTerms). From these data we can find the text string describing the term, which ontology it is in as well as some other basic information.

There are also a set of hash tables that contain the information about parents and children. They are provided as hash tables (the XX in the names below should be substituted for one of BP, MF, or CC).

- **GOXXPARENTS**: the parents of the term
- **GOXXANCESTOR**: the parents, and all their parents and so on.
- **GOXXCHILDREN**: the children of the term
- **GOXXOFFSPRING**: the children, their children and so on out to the leaves of the GO graph.
For the GOXPARENTS mappings (only) information about the nature of the relationship is included.

> GOTERMS"GO:0003700"

GOID: GO:0003700
Term: DNA-binding transcription factor activity
Ontology: MF
Definition: A transcription regulator activity that modulates transcription of gene sets via selective and non-covalent binding to a specific double-stranded genomic DNA sequence (sometimes referred to as a motif) within a cis-regulatory region. Regulatory regions include promoters (proximal and distal) and enhancers. Genes are transcriptional units, and include bacterial operons.

Synonym: GO:0000130
Synonym: GO:0001071
Synonym: GO:0001130
Synonym: GO:0001131
Synonym: GO:0001151
Synonym: GO:0001199
Synonym: GO:0001204
Synonym: DNA binding transcription factor activity
Synonym: gene-specific transcription factor activity
Synonym: sequence-specific DNA binding transcription factor activity
Synonym: nucleic acid binding transcription factor activity
Synonym: transcription factor activity
Synonym: bacterial-type DNA binding transcription factor activity
Synonym: bacterial-type RNA polymerase core promoter proximal region sequence-specific DNA binding transcription factor activity
Synonym: bacterial-type RNA polymerase transcription enhancer sequence-specific DNA binding transcription factor activity
Synonym: bacterial-type RNA polymerase transcription factor activity, metal ion regulated sequence-specific DNA binding
Synonym: bacterial-type RNA polymerase transcription factor activity, sequence-specific DNA binding
Synonym: metal ion regulated sequence-specific DNA binding bacterial-type RNA polymerase transcription factor activity
Synonym: metal ion regulated sequence-specific DNA binding transcription factor activity
Synonym: sequence-specific DNA binding bacterial-type RNA polymerase transcription factor activity
Synonym: transcription factor activity, bacterial-type RNA polymerase core promoter proximal region sequence-specific binding
Synonym: transcription factor activity, bacterial-type RNA polymerase
proximal promoter sequence-specific DNA binding
Synonym: transcription factor activity, bacterial-type RNA polymerase
transcription enhancer sequence-specific binding
Synonym: transcription factor activity, metal ion regulated
sequence-specific DNA binding
Secondary: GO:0000130
Secondary: GO:0001071
Secondary: GO:0001130
Secondary: GO:0001131
Secondary: GO:0001151
Secondary: GO:0001199
Secondary: GO:0001204

> GOMFPARENTS"GO:0003700"
   isa
"GO:0140110"

> GOMFCCHILDREN"GO:0003700"
   isa   isa   isa   isa   isa
"GO:0000981" "GO:0001216" "GO:0001217" "GO:0016987" "GO:0098531"

Here we see that the term GO:0003700 has two parents, that the relationships are is-a and that it has one child. One can then follow these chains of relationships or use the ANCESTOR and OFFSPRING hash tables to get more information.

The mappings of genes to GO terms is not contained in the GO package. Rather these mappings are held in each of the chip and organism specific data packages, such as hgu95av2GO and org.Hs.egGO are contained within packages hgu95av2.db and org.Hs.eg.db respectively. These mappings are from a Entrez Gene ID to the most specific applicable GO terms. Each such entry is a list of lists where the innermost list has these names:

- **GOID**: the GO identifier
- **Evidence**: the evidence code for the assignment
- **Ontology**: the ontology the GO identifier belongs to (one of BP, MF, or CC).

Some genes are mapped to a GO identifier based on two or more evidence codes. Currently these appear as separate entries. So you may want to remove duplicate entries if you are not interested in evidence codes. However, as more sophisticated use is made of these data it will be important to be able to separate out mappings according to specific evidence codes.

In this next example we consider the gene with Entrez Gene ID 4121, this corresponds to Affymetrix ID 39613_at.
> l1l = hgu95av2GO["39613_at"]
> length(l1l)

[1] 20

> sapply(l1l, function(x) x$Ontology)

GB:0005975 GO:0006486 GO:0036503 GO:0045047 GO:1904382 GO:1904382 GO:0000139
   "BP" "BP" "BP" "BP" "BP" "BP" "BP"
GO:0005783 GO:0005793 GO:0005794 GO:0005829 GO:0016020 GO:0031436
   "CC" "CC" "CC" "CC" "CC" "CC" "CC"
GO:0070062 GO:0004571 GO:0004571 GO:0005509 GO:0015923
   "CC" "MF" "MF" "MF" "MF"

> 

We see that there are 20 different mappings. We can get only those mappings for the BP ontology by using `getOntology`. We can get the evidence codes using `getEvidence` and we can drop those codes we do not wish to use by using `dropECode`.

> getOntology(l1l, "BP")

[1] "GO:0005975" "GO:0006486" "GO:0036503" "GO:0045047" "GO:1904382"

> getEvidence(l1l)

GO:0005975 GO:0006486 GO:0036503 GO:0045047 GO:1904382 GO:1904382 GO:0000139
   "IEA" "IEA" "IDA" "IMP" "IBA" "IDA" "IBA"
GO:0000139 GO:0005783 GO:0005793 GO:0005794 GO:0005829 GO:0016020 GO:0031436
   "TAS" "IBA" "IDA" "IDA" "HDA" "IDA" "IBA"
GO:0070062 GO:0004571 GO:0004571 GO:0005509 GO:0015923
   "HDA" "IBA" "IMP" "TAS" "IEA" "TAS"

> zz = dropECode(l1l)
> getEvidence(zz)

GO:0036503 GO:0045047 GO:1904382 GO:1904382 GO:0000139 GO:0000139 GO:0005783
   "IDA" "IMP" "IBA" "IDA" "IBA" "TAS" "IBA"
GO:0005793 GO:0005794 GO:0005829 GO:0016020 GO:0031410 GO:0070062 GO:0004571
   "IDA" "IDA" "HDA" "IDA" "HDA" "IBA" "IBA"
GO:0004571 GO:0004571 GO:0015923
   "IMP" "TAS" "TAS"
A Basic Description of GO

We now characterize GO and some of its properties. First we list some of the specific GO IDs that might be of interest (please feel free to propose even more).

- **GO:0003673** is the GO root.
- **GO:0003674** is the MF root.
- **GO:0005575** is the CC root.
- **GO:0008150** is the BP root.
- **GO:0000004** is biological process unknown
- **GO:0005554** is molecular function unknown
- **GO:0008372** is cellular component unknown

We can find out how many terms are in each of the different ontologies by:

```r
> zz = Ontology(GOTERM)
> table(unlist(zz))

<table>
<thead>
<tr>
<th>BP</th>
<th>CC</th>
<th>MF</th>
<th>universal</th>
</tr>
</thead>
<tbody>
<tr>
<td>27186</td>
<td>4058</td>
<td>11198</td>
<td>1</td>
</tr>
</tbody>
</table>
```

Or we can ask about the number of is-a and partof relationships in each of the three different ontologies.

```r
> BPisa = eapply(GOBPPARENTS, function(x) names(x))
> table(unlist(BPisa))

<table>
<thead>
<tr>
<th>isa negatively regulates</th>
<th>part of</th>
</tr>
</thead>
<tbody>
<tr>
<td>49258</td>
<td>2660</td>
</tr>
<tr>
<td>2656</td>
<td>3060</td>
</tr>
</tbody>
</table>

> MFisa = eapply(GOMFPARENTS, function(x) names(x))
> table(unlist(MFisa))

<table>
<thead>
<tr>
<th>isa part of</th>
</tr>
</thead>
<tbody>
<tr>
<td>13639</td>
</tr>
<tr>
<td>11</td>
</tr>
</tbody>
</table>

> CCisa = eapply(GOCCPARENTS, function(x) names(x))
> table(unlist(CCisa))

<table>
<thead>
<tr>
<th>isa part of</th>
</tr>
</thead>
<tbody>
<tr>
<td>4685</td>
</tr>
<tr>
<td>1833</td>
</tr>
</tbody>
</table>
```
Working with GO

Finding terms that have specific character strings in them is easily accomplished using `grep`. In the next example we first convert the data from `GOTERM` into a character vector to make it easier to do multiple searches.

```r
> goterms = unlist(Term(GOTERM))
> whmf = grep("fertilization", goterms)
```

So we see that there are 16 terms with the string “fertilization” in them in the ontology. They can be accessed by subsetting the `goterms` object.

```r
> goterms[whmf]
```

"fusion of sperm to egg plasma membrane involved in double fertilization forming two zygotes"

"double fertilization forming two zygotes"

"post-fertilization epigenetic regulation of gene expression"

"respiratory burst at fertilization"

"fertilization envelope"

"negative regulation of fertilization"

"fusion of sperm to egg plasma membrane involved in double fertilization forming a zygote and endosperm"

"fusion of sperm to egg plasma membrane involved in double fertilization forming two zygotes"

"regulation of double fertilization forming two zygotes"

"male-female gamete recognition during double fertilization forming two zygotes"
Working with chip specific meta-data

In some cases users will want to restrict their attention to the set of terms etc that map to genes that were assayed in the experiments that they are working with. To do this you should first get the appropriate chip specific meta-data file. Here we demonstrate some of the examples on the Affymetrix HGU95av2 chips and so use the package hgu95av2.db. Each of these packages has a data environment whose name is the basename of the package with a GO suffix, so in this case hgu95av2GO. Note that if there are many manufacturer ids that map to the same Entrez Gene identifier then these will be duplicate entries (with different keys).

We can get all the MF terms for our Affymetrix data.

```r
> affyGO = eapply(hgu95av2GO, getOntology)
> table(sapply(affyGO, length))
```

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1352</td>
<td>971</td>
<td>1432</td>
<td>1739</td>
<td>1630</td>
<td>1069</td>
<td>830</td>
<td>584</td>
<td>449</td>
<td>317</td>
<td>183</td>
<td>193</td>
<td>153</td>
<td>77</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>36</td>
<td>35</td>
<td>37</td>
<td>20</td>
<td>25</td>
<td>20</td>
<td>11</td>
<td>16</td>
<td>12</td>
<td>3</td>
<td>13</td>
<td>7</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>37</td>
<td>38</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

How many of these probes have multiple GO terms associated with them? What do we do if we want to compare two genes that have multiple GO terms associated with them?

What about evidence codes? To find these we apply a similar function to the affyGO terms.

```r
> affyEv = eapply(hgu95av2GO, getEvidence)
> table(unlist(affyEv, use.names=FALSE))
```

<table>
<thead>
<tr>
<th></th>
<th>EXP</th>
<th>HDA</th>
<th>HEP</th>
<th>HMP</th>
<th>IBA</th>
<th>IC</th>
<th>IDA</th>
<th>IEA</th>
<th>IEP</th>
<th>IGI</th>
<th>IMP</th>
<th>IPI</th>
<th>ISA</th>
<th>ISM</th>
<th>ISO</th>
<th>ISS</th>
<th>NAS</th>
<th>ND</th>
<th>RCA</th>
<th>TAS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>815</td>
<td>6973</td>
<td>53</td>
<td>116</td>
<td>485</td>
<td>37</td>
<td>137</td>
<td>716</td>
<td>665</td>
<td>632</td>
<td>18</td>
<td>109</td>
<td>5</td>
<td>218</td>
<td>422</td>
<td>883</td>
<td>175</td>
<td>59</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td></td>
<td>309</td>
<td>276</td>
<td>21</td>
<td>108</td>
<td>03</td>
<td>612</td>
<td>156</td>
<td>304</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

>
```r
> test1 = eapply(hgu95av2GO, dropECode, c("IEA", "NR"))
> table(unlist(sapply(test1, getEvidence),
+   use.names=FALSE))

EXP  HDA  HEP  HMP  IBA  IC  IDA  IEP  IGI  IMP  IPI  ISA  ISO  ISS  NAS  ND  RCA  TAS
 815  6973  53   116 48537 1371 71665 1095 2184 22883 17559 1343  309 27621 10803 612 156 30410
```

These functions make it somewhat straightforward to select subsets of the GO terms that are specific to different evidence codes.

### References
