Microarray annotation

Benedikt Brors

Computational Oncology Group

Dept. Theoretical Bioinformatics

German Cancer Research Center

b.brors@dkfz.de

Why do we need microarray clone annotation?

- Often, the result of microarray data analysis is a list of genes.
- The list has to be summarized with respect to its biological meaning. For this, information about the genes and the related proteins has to be gathered.
- If the list is small (let's say, 1–30), this is easily done by reading database information and/or the available literature.
- Sometimes, lists are longer (100s or even 1000s of genes). Automatic parsing and extracting of information is needed.
- To get complete information, you will need the help of an experienced computational biologist (aka 'bioinformatician'). However, there is a lot that you can do on your own.

Databases

- Sequences are contained in *primary sequence databases* like EMBL/Genbank or SwissProt. Primary nucleic acid databases have a high degree of redundancy.

- Some databases are *curated*, i.e. curators watch over the entries and ensure quality, remove redundancy, and annotate domain structure, function etc. This is a slow process, thus curated databases are limited in size and not really up-to-date.

- Meta databases collect further information and relate them to primary databases. Examples are **OMIM** (online mendelian inheritance in man) for disease-related genes, **LocusLink** for genomic location, **PFAM** for protein domain structure, and **GeneCards** for comprehensive information from other databases on human genes.

The relation of clone information to genes and proteins

- Microarrays are produced using information on *expressed sequences* as EST clones, cDNAs, partial cDNAs etc.

The relation of clone information to genes and proteins

- Microarrays are produced using information on *expressed sequences* as EST clones, cDNAs, partial cDNAs etc.

- At the other end, functional information is generated (and available) for *proteins*. Hence, there is a need to map a clone sequence ID to a protein ID. This is non-trivial.

The relation of clone information to genes and proteins

- Microarrays are produced using information on *expressed sequences* as EST clones, cDNAs, partial cDNAs etc.

- At the other end, functional information is generated (and available) for *proteins*. Hence, there is a need to map a clone sequence ID to a protein ID. This is non-trivial.

- First, there are usually hundreds of ESTs (and several cDNA sequences) that map to the same gene. The Database *Unigene* tries to resolve this clustering by sequence clustering.

The relation of clone information to genes and proteins

- Microarrays are produced using information on *expressed sequences* as EST clones, cDNAs, partial cDNAs etc.

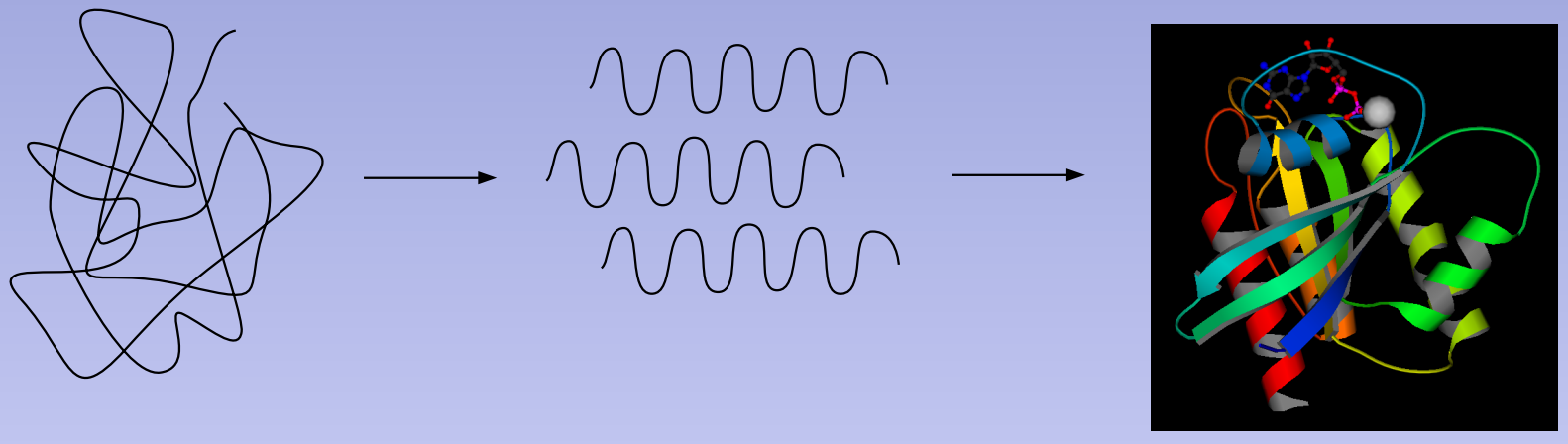
- At the other end, functional information is generated (and available) for *proteins*. Hence, there is a need to map a clone sequence ID to a protein ID. This is non-trivial.

- First, there are usually hundreds of ESTs (and several cDNA sequences) that map to the same gene. The Database *Unigene* tries to resolve this clustering by sequence clustering.

- An alternative approach is taken by *Locus Link*. This is a quite stable repository of genomic loci, supposed to be a single gene. Since the emphasis is on well-characterised loci, Locus Link is not complete.

- There are other projects like RefSeq (NCBI) or TIGR Gene Indices. According to the cross-references available for a certain microarray, one or the other may be advantageous.

Genome	Transcripts	Protein
genes hidden and interrupted by introns	often truncated highly redundant used for designing microarrays	function annotated



ENSEMBL (Sanger) Golden Path (UCSC) LocusLink	Unigene / GeneNEST RefSeq TIGR Gene Indices	SwissProt GOA function annotation

The Human Genome Sequence

- With the completion of the human genome sequence, you'd think that such ambiguities can be resolved. In fact, that is not the case.

The Human Genome Sequence

- With the completion of the human genome sequence, you'd think that such ambiguities can be resolved. In fact, that is not the case.

- Part of the problem is due to the fact that it is hard to predict gene structure (intron/exon) without knowing the entire mRNA sequence, which happens for about two-thirds of all genes.

The Human Genome Sequence

- With the completion of the human genome sequence, you'd think that such ambiguities can be resolved. In fact, that is not the case.

- Part of the problem is due to the fact that it is hard to predict gene structure (intron/exon) without knowing the entire mRNA sequence, which happens for about two-thirds of all genes.

- Then, there are errors in the assembly (putting together the sequence snippets). A typical symptom is that a gene appears to map to multiple loci on the same chromosome, with very high sequence similarity.

The Human Genome Sequence

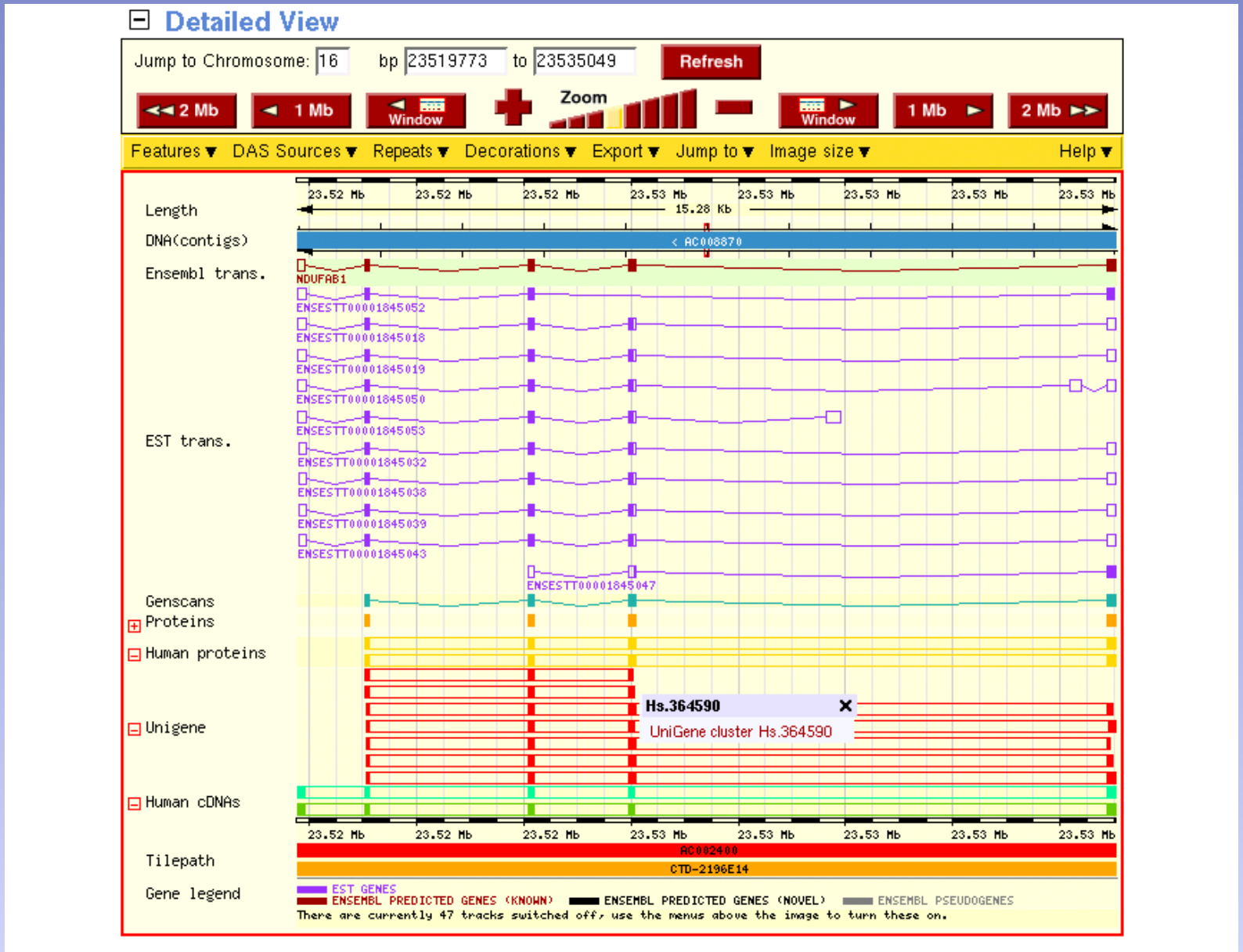
- With the completion of the human genome sequence, you'd think that such ambiguities can be resolved. In fact, that is not the case.

- Part of the problem is due to the fact that it is hard to predict gene structure (intron/exon) without knowing the entire mRNA sequence, which happens for about two-thirds of all genes.

- Then, there are errors in the assembly (putting together the sequence snippets). A typical symptom is that a gene appears to map to multiple loci on the same chromosome, with very high sequence similarity.

- But there are also sequences that are nearly indentical, but duplicated. This has happened not long ago in evolution by means of transposable elements.

Genomic mapping: ENSEMBL Browser



Some figures

- Currently, it's estimated that the human genome contains about 25,000 – 30,000 genes that code for 50,000 – 100,000 different transcripts (and thus, proteins).

- Unigene (human section) contains 105,680 clusters, but 45,999 of them are of size 2 or less.

- RefSeq DNA contains 28,097 human sequences.

- ENSEMBL contains 21,787 predicted genes, 31,609 predicted transcripts. Fully computational methods like Genscan produce more than 65,000 predictions.

- Locus Link contains 15,248 genes with known function, and further 6038 genes without function annotation.

Function annotation

- Probably, the most important thing you want to know is what the genes or their products are concerned with, i.e. their **function**.

- Function annotation is difficult: Different people use different words for the same function, or may mean different things by the same word. The context in which a gene was found (e.g. “TGFβ-induced gene”) may not be particularly associated with its function.

- Inference of function from sequence alone is error-prone and sometimes unreliable. The best function annotation systems (GO, SwissProt) use human beings who read the literature before assigning a function to a gene.

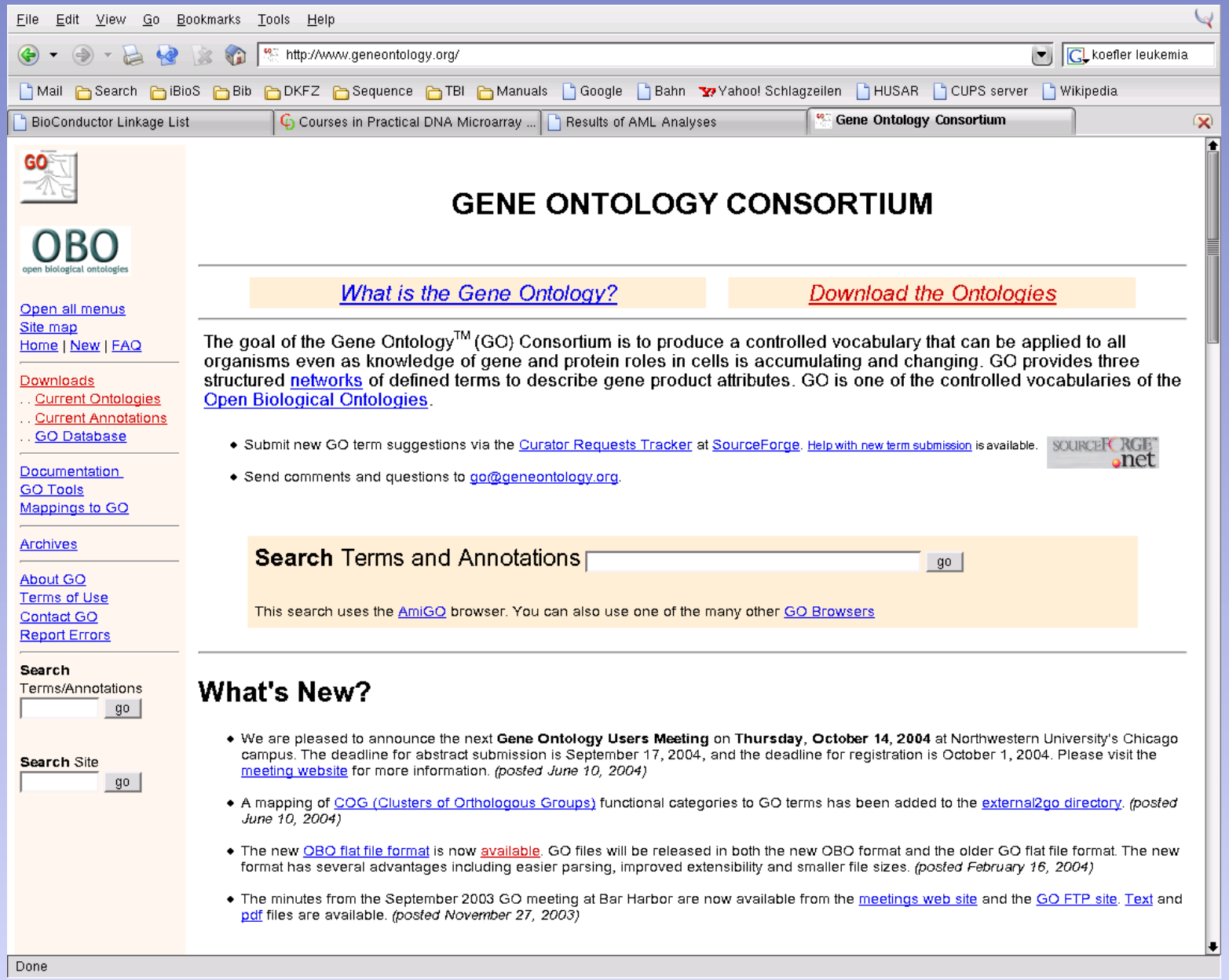
The Gene Ontology system

- To overcome some of the problems, an annotation system has been created: Gene Ontology (http://www.geneontology.org). Ontology means here the art (or science) of giving everything its correct name.

- It represents a unified, consistent system, i.e. terms occur only once, and there is a dictionary of allowed words.

- Furthermore, terms are related to each other: the hierarchy goes from very general terms to very detailed ones.

The Gene Ontology site



The Gene Ontology hierarchy

Actual annotation

- Gene Ontology by itself is only a system for annotating genes and proteins. It does not relate database entries to a special annotation value.

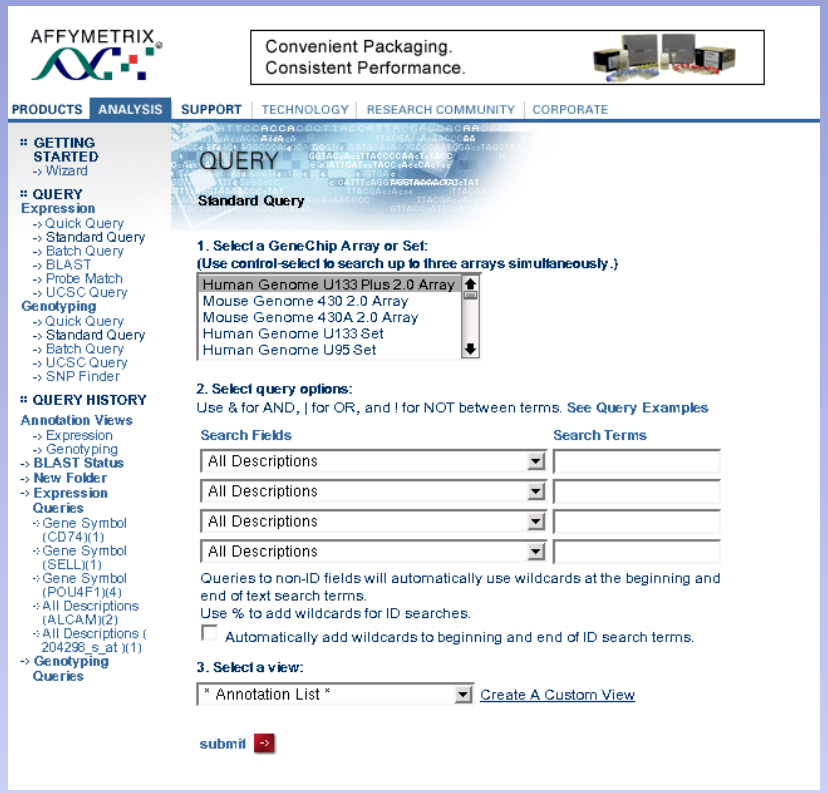
- Luckily, research communities for several model organisms have agreed on entering Gene Ontology information into the databases. As this is done ‘by hand’, GO annotation for most organisms is far from complete.

Available Gene Ontology information

TIGR *Bacillus anthracis* Ames	4414	4414	4416	4416	200	200	4417	6	Download Mar 12, 2004
TIGR *Arabidopsis thaliana* README	9638	9638	24945	24945	5835	5835	25701	13653	Download Feb 10, 2004
TIGR *Coxiella burnetii* RSA 493	1359	1359	1349	1349	176	176	1365	4	Download Mar 12, 2004
TIGR Gene Index README	80031	0	100151	0	78400	0	126557	1	Download Apr 16, 2004
TIGR *Shewanella oneidensis* MR-1	3696	3696	3696	3696	241	241	3696	5	Download Mar 12, 2004
TIGR *Vibrio cholerae*	2923	2923	2728	2728	191	191	2924	9	Download Mar 12, 2004
GO Annotations @ EBI Human README	18094	8143	20606	7604	15787	7077	22720	13816	Download Jun 4, 2004
GO Annotations @ EBI Mouse README	17488	2751	20683	2336	14932	2690	23060	3522	Download Jun 4, 2004
GO Annotations @ EBI PDB README	20731	0	21969	0	11289	0	22989	5	Download Jun 4, 2004
GO Annotations @ EBI Rat README	14810	302	17731	296	11665	281	19453	490	Download Jun 4, 2004
GO Annotations @ EBI UniProt README	602033	22971	719095	19706	440522	21270	818792	26991	Download Jun 4, 2004

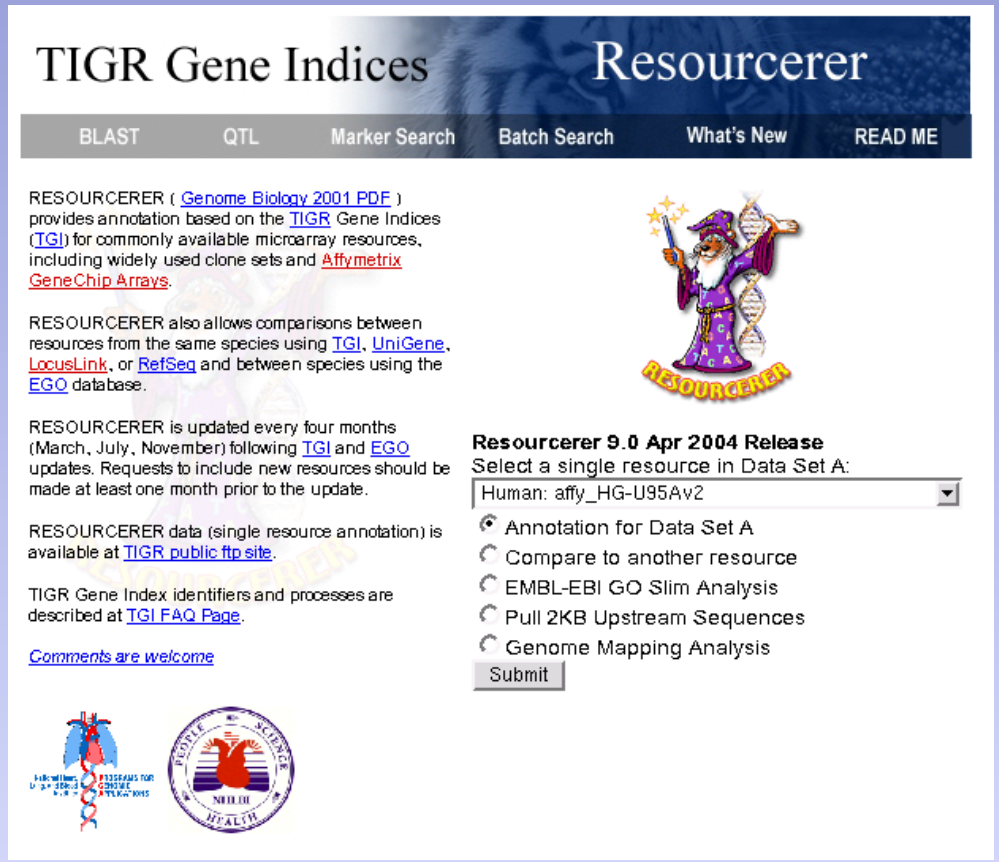
The NetAffx System

- For Affymetrix arrays, annotation is provided by the supplier via the NetAffx system (http://www.affymetrix.com/analysis/netaffx/)

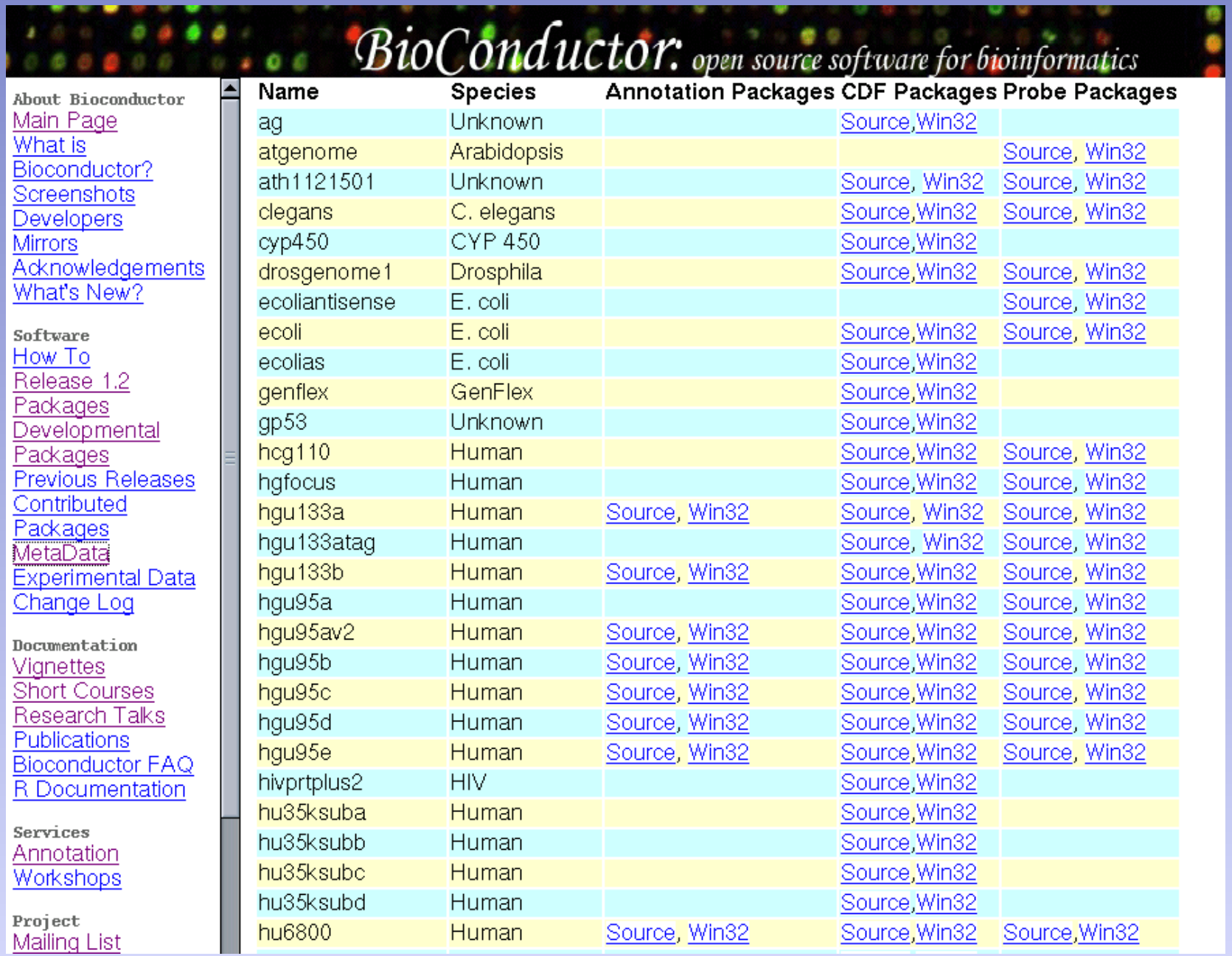


Alternative pre-compiled annotation

- The Institute of Genomic Research (TIGR) has its own pre-compiled annotation for most commercial arrays (Affymetrix, Agilent, Incyte etc.): `http://www.tigr.org/tigr-scripts/magic/r1.pl`



Data packages in Bioconductor



BioConductor: open source software for bioinformatics

About Bioconductor
Main Page
What is Bioconductor?
Screenshots
Developers
Mirrors
Acknowledgements
What's New?

Software
How To
Release 1.2 Packages
Developmental Packages
Previous Releases
Contributed Packages
MetaData
Experimental Data
Change Log

Documentation
Vignettes
Short Courses
Research Talks
Publications
Bioconductor FAQ
R Documentation

Services
Annotation
Workshops

Project
Mailing List

Name	Species	Annotation Packages	CDF Packages	Probe Packages
ag	Unknown		Source,Win32	
atgenome	Arabidopsis			Source, Win32
ath1121501	Unknown		Source, Win32	Source, Win32
clegans	C. elegans		Source,Win32	Source, Win32
cyp450	CYP 450		Source,Win32	
drosgenome1	Drosphila		Source,Win32	Source, Win32
ecoliantisense	E. coli			Source, Win32
ecoli	E. coli		Source,Win32	Source, Win32
ecolias	E. coli		Source,Win32	
genflex	GenFlex		Source,Win32	
gp53	Unknown		Source,Win32	
hcg110	Human		Source,Win32	Source, Win32
hgfocus	Human		Source,Win32	Source, Win32
hgu133a	Human	Source, Win32	Source, Win32	Source, Win32
hgu133atag	Human		Source, Win32	Source, Win32
hgu133b	Human	Source, Win32	Source,Win32	Source, Win32
hgu95a	Human		Source,Win32	Source, Win32
hgu95av2	Human	Source, Win32	Source,Win32	Source, Win32
hgu95b	Human	Source, Win32	Source,Win32	Source, Win32
hgu95c	Human	Source, Win32	Source,Win32	Source, Win32
hgu95d	Human	Source, Win32	Source,Win32	Source, Win32
hgu95e	Human	Source, Win32	Source,Win32	Source, Win32
hivprtplus2	HIV		Source,Win32	
hu35ksuba	Human		Source,Win32	
hu35ksubb	Human		Source,Win32	
hu35ksubc	Human		Source,Win32	
hu35ksubd	Human		Source,Win32	
hu6800	Human	Source, Win32	Source,Win32	Source,Win32

Bioconductor metadata packages

- These packages contain one-to-one and one-to-many mappings for frequently used chips, especially Affymetrix arrays.

- Information available includes gene names, gene symbol, database accession numbers, Gene Ontology function description, enzmye classification number (EC), relations to PubMed abstracts, and others.

- The data use the framework of the `annotate` package, so I will briefly explain how it works.

Environments in R

- To quickly find information on one subject in a long list, a data structure called *hash table* is frequently used in computer science.

- A hash table is a list of key/value pairs, where the key is used to find the corresponding value. To go the other way round, you have to use pattern matching, which is much slower.

- In R, hash tables are implemented as *environments*. For the moment, we do not care about the philosophy behind it and simply treat it as another word for hash table.

Setting up environments

To set up a new environment:

```
symbol.hash = new.env(hash=TRUE)
```

To create a key/value pair:

```
assign("1234_at", "EphA3", env=symbol.hash)
```

To list all keys of an environment:

```
ls(env=symbol.hash)
```

To get the value for a certain key:

```
get("1234_at", env=symbol.hash)
```

The annotate package

- That's all standard R. The annotate package gives one further function, `multiget`, which retrieves more than one entry at a time, and definitions for special data, e.g. PubMed abstracts, or chromosomal location objects.

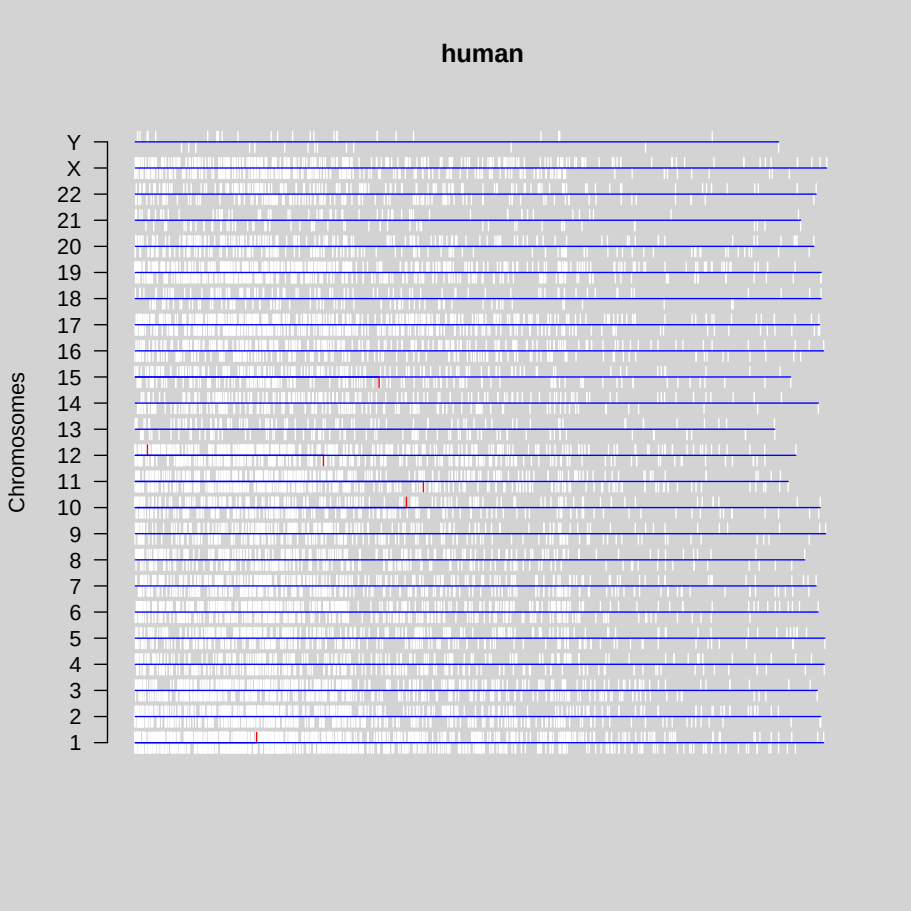
- ChromLoc objects are quite useful if you want to associate gene expression with certain positions on a chromosome, e.g. if aberration occurs in your samples.

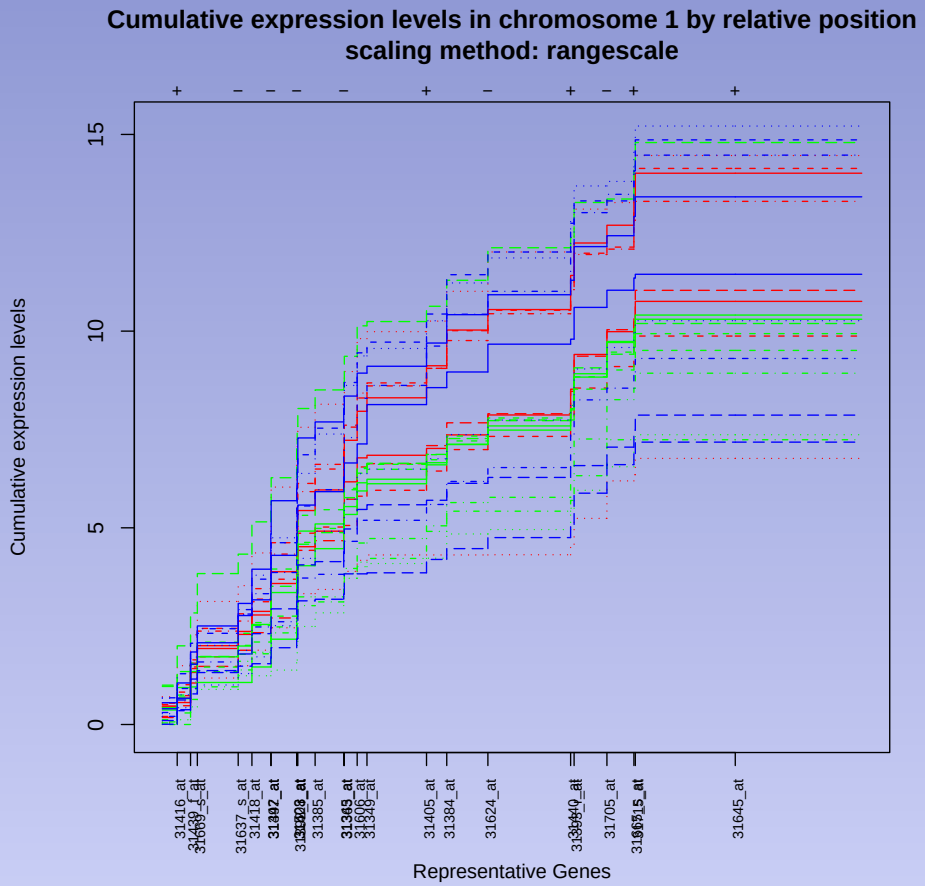
- You can construct a ChromLoc object on your own (→ Vignette), or use the function `buildChromLocation`. For chip HGU95a_v2:

```
library(hgu95av2)
cl.95a = buildChromLocation("hgu95av2")
```

Plots for ChromLocation objects

- Plotting methods are available via library `geneplotter`





How to get annotation for a set of genes

- Suppose you have found some interesting genes. The index in the matrix is in `index.int`. To get the gene names:

```
gnam.int = geneNames(exprset)[index.int]
```

- To find the description:

```
multiget(gnam.int, env=hgu95av2GENENAME)
```

- To get EC Numbers (relating to KEGG pathways):

```
multiget(gnam.int, env=hgu95av2ENZYME)
```

Some caveats

- Because of the non-unique matching of sequences to the genome, array features are sometimes annotated with more than one position:

```
a = ls(env=hgu95av2CHRLOC)
table(sapply(multiget(a, env=hgu95av2CHRLOC),
    length))
```

```
    1     2     3     4     5     6     7     8
11793   647   127    29    13    10     2     1
```

- For the 800 or so sequences with more than one location, only the first one is used, although there is no warning. It should be desirable to resolve the ambiguities by hand, but nobody has done yet.

- There are even 14 probe sets on HGU95A_v2 that map to 2 chromosomes; however, these are located on some special extrachromosomal segment and annotated with “X” and “Y”.

- N.B. There is a special annotation package for Affymetrix arrays, `annaffy`. It does not provide much other functionality than `annotate`, but allows to do the same things differently (and maybe more conveniently).

Pattern matching

- To find something in character vectors or character lists, some pattern matching is required.
- If you have real full names, use `match`, e.g.

```
match("1234_at", rownames(exprs(exprset)) )
```

- This will give you the index of ‘‘`1234_at`’’. It works also with more than one gene:

```
match(gnam.int, rownames(exprs(exprset)) )
```

will give all indeces for genes in `gnam.int`.

- If you want to use regular expression matching, use `grep`.

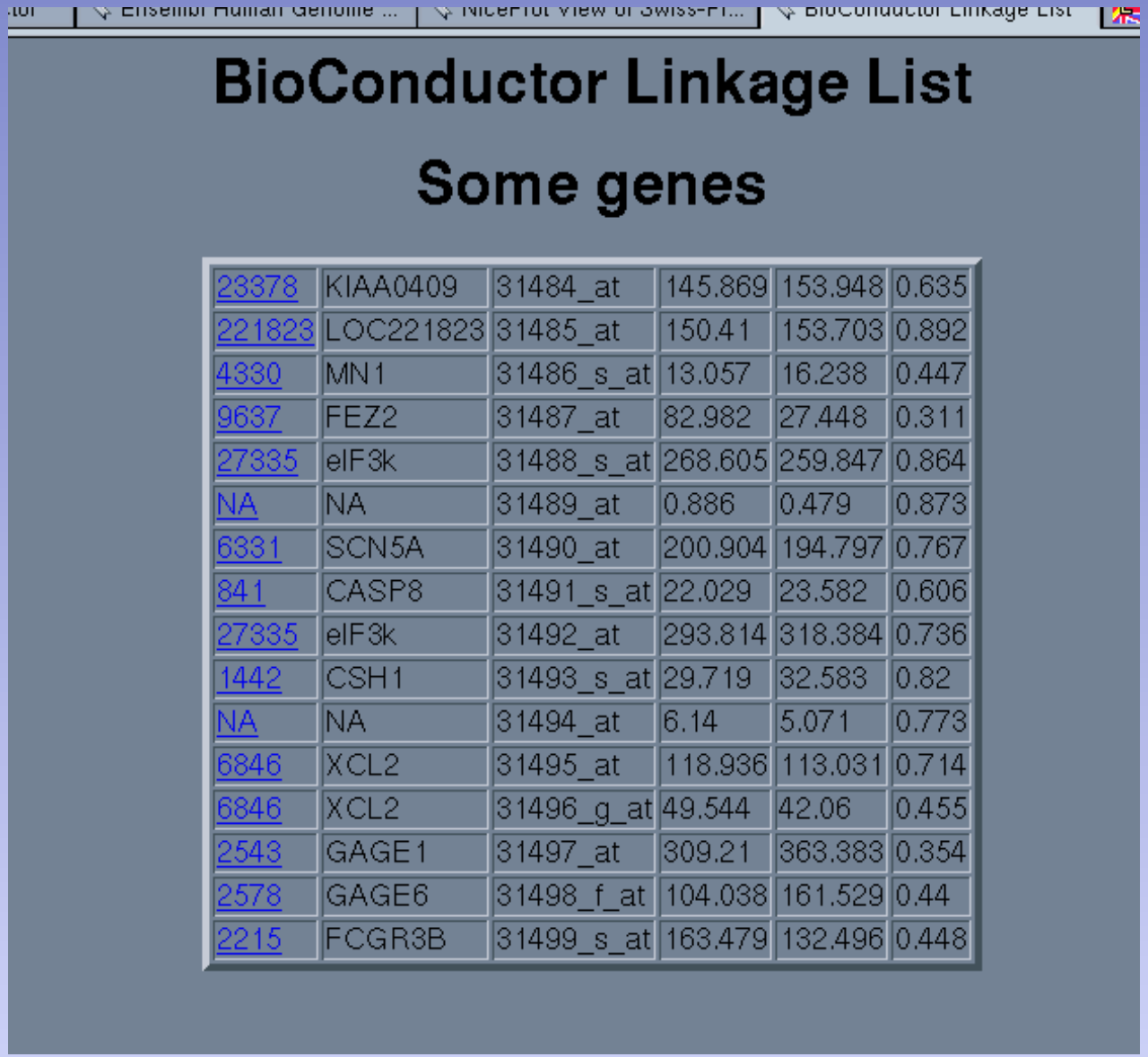
Export of annotation to HTML

- `annotate` is able to export tables of gene annotations to HTML, which is much nicer to browse than text tables

- Suppose, from a t-test you have for some genes `igenes`: mean of genes in class 1, `igenes.gp1`, mean in class 2, `igenes.gp2`, and P-value `igenes.pval`. To construct pretty HTML output:

```
igenes.ll = multiget(igenes, env=hgu95av2LOCUSID)
igenes.sym = multiget(igenes, env=hgu95av2SYMBOL)
ll.htmlpage(igenes.ll, "HOWTO.igenes", "Some genes",
  list(igenes,sym, igenes, round(igenes.gp1,3),
  round(igenes.gp2,3),round(igenes.pval,3)))
```

The result



BioConductor Linkage List

Some genes

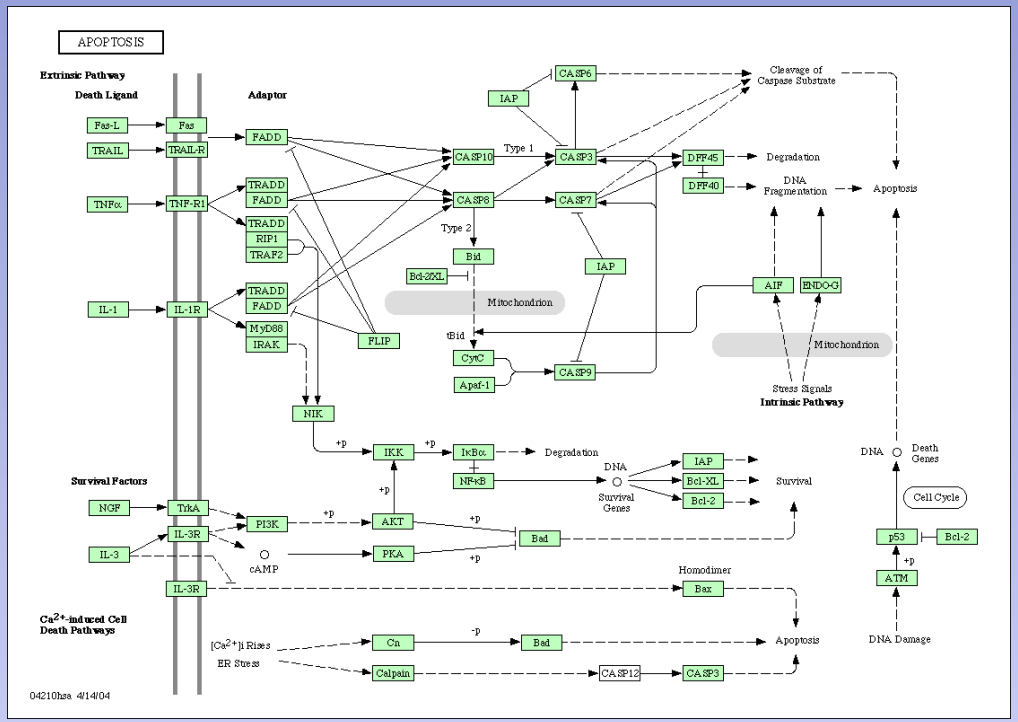
23378	KIAA0409	31484_at	145.869	153.948	0.635
221823	LOC221823	31485_at	150.41	153.703	0.892
4330	MN1	31486_s_at	13.057	16.238	0.447
9637	FEZ2	31487_at	82.982	27.448	0.311
27335	eIF3k	31488_s_at	268.605	259.847	0.864
NA	NA	31489_at	0.886	0.479	0.873
6331	SCN5A	31490_at	200.904	194.797	0.767
841	CASP8	31491_s_at	22.029	23.582	0.606
27335	eIF3k	31492_at	293.814	318.384	0.736
1442	CSH1	31493_s_at	29.719	32.583	0.82
NA	NA	31494_at	6.14	5.071	0.773
6846	XCL2	31495_at	118.936	113.031	0.714
6846	XCL2	31496_g_at	49.544	42.06	0.455
2543	GAGE1	31497_at	309.21	363.383	0.354
2578	GAGE6	31498_f_at	104.038	161.529	0.44
2215	FCGR3B	31499_s_at	163.479	132.496	0.448

Pathways

- For biological interpretation of function, most people want to use *pathways*
- A pathway is something like a bunch of interacting proteins and/or nucleic acids that allow for mass flux (metabolism) or information flux (signal transduction)
- The problem is that interaction information for proteins is quite rare (except for yeast)
- Some textbook pathways exist, but only few in computer-readable format

Pathway databases

- For metabolic pathways, some databases exist: KEGG (http://www.genome.ad.jp/kegg/), and EcoCyc (http://ecocyc.org), HumanCyc (http://humancyc.org) from SRI



Signal transduction information

- KEGG has some very limited information on signal transduction
- The database TRANSPATH wants to cover signal transduction. But information is incomplete, and you have to pay for part of the information (available via HNB)
- Other sources are `www.biocarta.com` and `www.stke.org` (requires registration)

Some software packages for function analysis

- There are some packages that allow to map gene expression profiles to biological information, like pathways.
- One example is GeneMAPP (`www.genemapp.org`) which also has a collection of user-contributed pathways.
- GoMiner (`http://discover.nci.nih.gov/gominer`) tries to find statistically significantly enriched terms in a gene list. This is, however, very crude and tends to favor annotations with very few total number of associated genes.
- Ingenuity (`http://www.ingenuity.com`) has its own database with interaction information, and software to infer pathways from microarray experiments. It seems to be quite capable, but is also expensive

Dealing with GO annotations

- Since the annotation system is hierarchical, i.e. for each term there is a hierarchical list of more general terms, we can compare functions of genes on every level we wish.

Dealing with GO annotations

- Since the annotation system is hierarchical, i.e. for each term there is a hierarchical list of more general terms, we can compare functions of genes on every level we wish.

- Technically, this amounts to the problem of finding the least common parent node between to genes of interest.

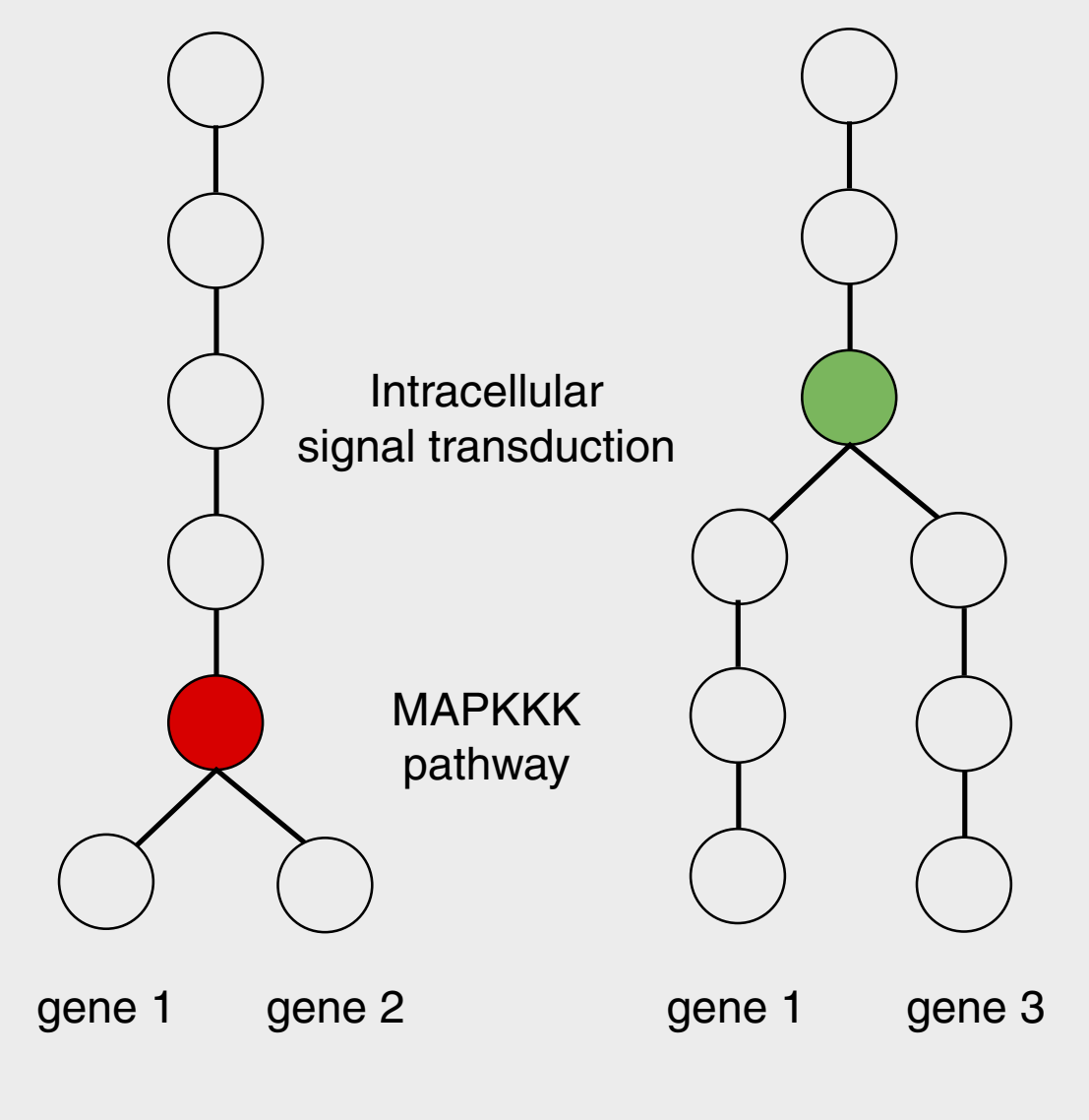
Dealing with GO annotations

- Since the annotation system is hierarchical, i.e. for each term there is a hierarchical list of more general terms, we can compare functions of genes on every level we wish.

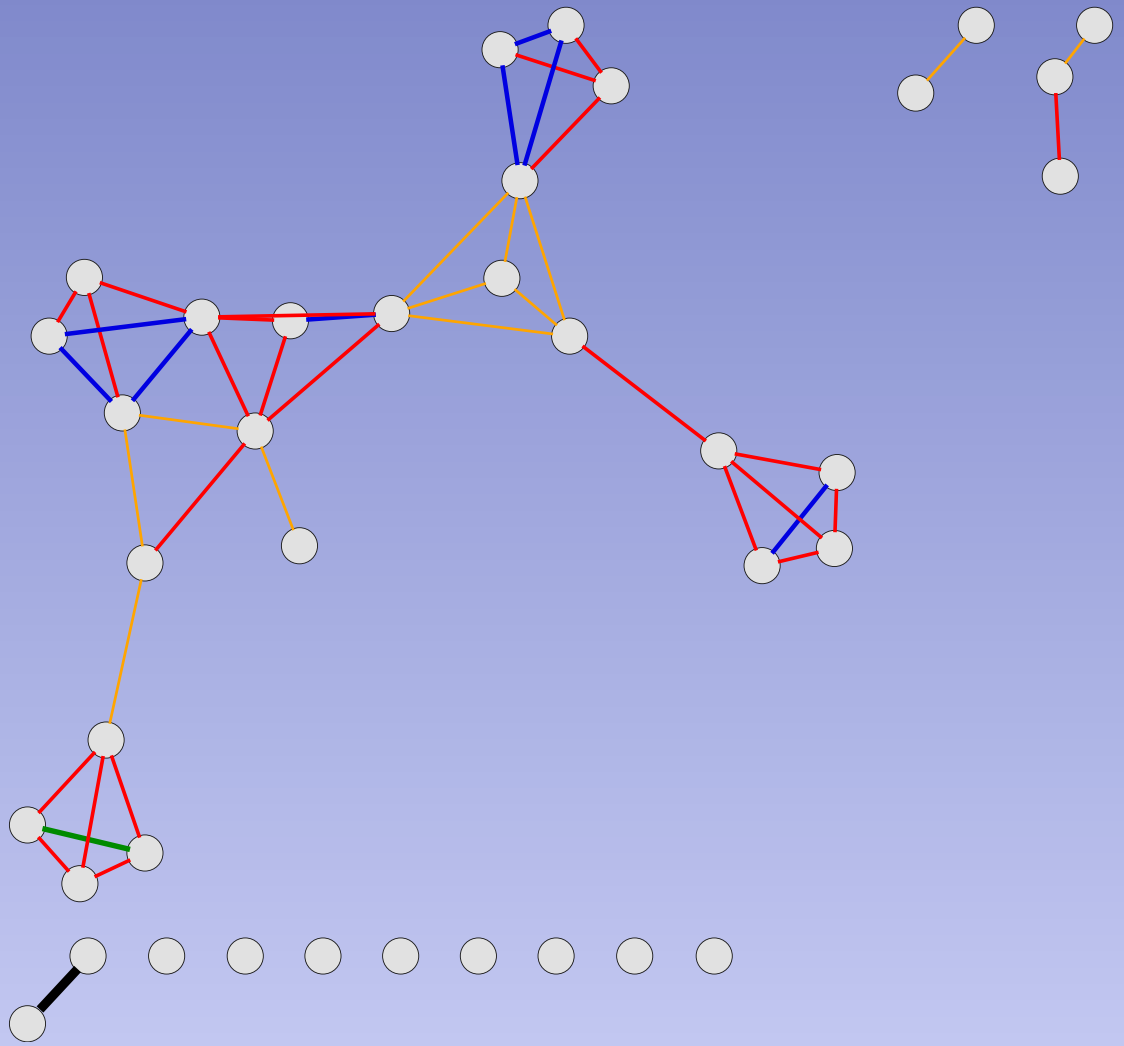
- Technically, this amounts to the problem of finding the least common parent node between to genes of interest.

- This can be used to find clusters of functionally related genes in a list that comes out of some other analysis.

Comparing GO-annotated genes



GO functional clusters as a graph



Graphs as analysis tools

- Graphs are quite useful for bioinformatic analysis, and have a long-standing history in sequence analysis.

- Recently, some functionality has been built into R to deal with graphs (`graph, Rgraphviz, RBGL`). Certainly, the most useful capability is to visualize graphs via `Rgraphviz`. The R package is an interface to the external program `graphviz` (from AT&T). Big graphs should be visualized by means of `ggobi`, however.

- Some other immediate use is to construct PubMed co-citation graphs for genes of interest. Functions for this exist. However, for many other applications the meaning of graphs or graph-theoretic algorithms is not clear, so a lot of work remains to be done.