We will now perform some analyses that integrate knowledge about biological network and databases. In this session, we will be using a dataset from gene expression (mRNA) study of patients with acute lymphoblastic leukemia (ALL). The dataset is available as part of the `ALL` Bioconductor package.

1. Read in the datasets all the necessary packages required (Note: using 'library' is equivalent to 'require')

```
library("ALL")
library("hgu95av2.db")
library("GO.db")
library("annotate")
library("genefilter")
library("GOstats")
library("RColorBrewer")
library("xtable")
library("Rgraphviz")

# read the data in
data(ALL, package="ALL")
```

2. First, we will perform GO test to investigate whether among the statistically significant genes, some GO terms (think of it like a pathway) are over-represented. We will not be using ALL patients here, but rather will only use the controls (Negative) and patients with BRC/ABL type of ALL

```
## For this data we can have ALL1/AF4 or BCR/ABL, select only the latter
subsetType <- "ALL1/AF4"
## Subset of interest: 37BRC/ABL + 42NEG = 79 samples
Bcell <- grep("^B", as.character(ALL$BT))
bcrAblOrNegIdx <- which(as.character(ALL$mol) %in% c("NEG", subsetType))
bcrAblOrNeg <- ALL[, intersect(Bcell, bcrAblOrNegIdx)]
bcrAblOrNeg$mol.biol = factor(bcrAblOrNeg$mol.biol)
```

3. Another important step when performing GO test is to define the 'universe' (of marbles) that can be drawn (see lecture slides for the marble analogy). The ALL study was conducted using Affymetrix human genome u95 version 2 array (hgu95av2), so to start with only genes covered by this array can be drawn. But we also need to exclude genes without EntrezID as these genes cannot be mapped to GO terms that utilize EntrezID as mapping key, as well as genes without GO terms.

```
## Remove genes that have no entrezGene id
entrezIds <- mget(featureNames(bcrAblOrNeg), envir=hgu95av2ENTREZID)
haveEntrezId <- names(entrezIds)[sapply(entrezIds, function(x) !is.na(x))]
numNoEntrezId <- length(featureNames(bcrAblOrNeg)) - length(haveEntrezId)
bcrAblOrNeg <- bcrAblOrNeg[haveEntrezId, ]

## Remove genes with no GO mapping
haveGo <- sapply(mget(featureNames(bcrAblOrNeg), hgu95av2GO),
function(x) {
if (length(x) == 1 && is.na(x))
FALSE
else TRUE
})
numNoGO <- sum(!haveGo)
bcrAblOrNeg <- bcrAblOrNeg[haveGo, ]
```

4. Since there is imbalance in gender ratio between cases and controls, we will not be investigating the $Y$ chromosome (why?) and for genes covered by more than one probe we will choose the probes with the largest variations (measured by IQR here).

```
# calculate IQR for each probe and select those with IQR>0.5 only
iqrCutoff <- 0.5
bcrAblOrNegIqr <- apply(exprs(bcrAblOrNeg), 1, IQR)
selected <- bcrAblOrNegIqr > iqrCutoff

## Drop those that are on the Y chromosome
## because there is an imbalance of men and women by group
chrN <- mget(featureNames(bcrAblOrNeg), envir=hgu95av2CHR)
onY <- sapply(chrN, function(x) any(x=="Y"))
onY[is.na(onY)] <- FALSE
selected <- selected & !onY
nsFiltered <- bcrAblOrNeg[selected,]

# for genes with multiple probes, select probe with largest IQR
numNsWithDups <- length(featureNames(nsFiltered))
nsFilteredIqr <- bcrAblOrNegIqr[selected]
uniqGenes <- findLargest(featureNames(nsFiltered), nsFilteredIqr,
"hgu95av2")
nsFiltered <- nsFiltered[uniqGenes, ]
numSelected <- length(featureNames(nsFiltered))
##set up some colors
BCRcols = ifelse(nsFiltered$mol == subsetType, "goldenrod", "skyblue")
cols = brewer.pal(10, "RdBu")

## Define gene universe based on results of non-specific filtering
affyUniverse <- featureNames(nsFiltered)
entrezUniverse <- unlist(mget(affyUniverse, hgu95av2ENTREZID))
if (any(duplicated(entrezUniverse)))
 stop("error in gene universe: cant have duplicate Entrez Gene Ids")
```

5. Now, we have define the 'universe' of marbles that can be drawn. We will perform t-test on each probe comparing the expression of control and cases and select the probes with p-values less than 0.05 as statistically-significant.

```
ttestCutoff <- 0.05
ttests = rowttests(nsFiltered, "mol.biol")
smPV = ttests$p.value < ttestCutoff
pvalFiltered <- nsFiltered[smPV, ]
selectedEntrezIds <- unlist(mget(featureNames(pvalFiltered), hgu95av2ENTREZID))
```

6. We can now test whether some GO terms that define Biological Process (BP) are over-represented among the statistically-significant genes. Once the GO test is done, we save the list of significantly over-represented terms into an HTML file.

```
# specify the test
 hgCutoff <- 0.001
 params <- new("GOHyperGParams",
 geneIds=selectedEntrezIds,
 universeGeneIds=entrezUniverse,
 annotation="hgu95av2.db",
 ontology="BP",
```

```
 pvalueCutoff=hgCutoff,
 conditional=FALSE,
 testDirection="over")

# test and save
hgOver <- hyperGTest(params)
html.name <- paste('YOURNAME','BP_over.html',sep="")
htmlReport(hgOver, file=html.name)
```