Package ‘graphite’

December 14, 2018

Version 1.28.1
Date 2018-11-24
Title GRAPH Interaction from pathway Topological Environment
Author Gabriele Sales <gabriele.sales@unipd.it>, Enrica Calura
<enrica.calura@gmail.com>, Chiara Romualdi
<chiara.romualdi@unipd.it>
Maintainer Gabriele Sales <gabriele.sales@unipd.it>
Description Graph objects from pathway topology derived from Biocarta,
HumanCyc, KEGG, NCI, Panther, PathBank, PharmGKB, Reactome and SMPDB
databases.
License AGPL-3
Depends R (>= 2.10), methods
Imports AnnotationDbi, checkmate, graph, httr, rappdirs, stats, utils
Suggests a4Preproc, ALL, BiocStyle, clipper, codetools,
lgu133plus2.db, lgu95av2.db, impute, knitr, org.Hs.eg.db,
parallel, R.rsp, RCy3, rmarkdown, SPIA (>= 2.2), testthat,
topologyGSA (>= 1.4.0)
LazyData yes
VignetteBuilder R.rsp
biocViews Pathways, ThirdPartyClient, GraphAndNetwork, Network,
Reactome, KEGG, BioCarta, Metabolomics
git_url https://git.bioconductor.org/packages/graphite
git_branch RELEASE_3_8
git_last_commit 5039302
git_last_commit_date 2018-11-18
Date/Publication 2018-12-13

R topics documented:

as.list.PathwayList ......................................................... 2
buildPathway ............................................................. 3
as.list.PathwayList  

Conversion of PathwayLists into lists.

Description
Converting a PathwayList into a list of Pathways.

Usage

## S3 method for class 'PathwayList'
as.list(x, ...)

Arguments

x  
a PathwayList object

...  
extra arguments to as.list

Value
A list of pathways.

Author(s)
Gabriele Sales

See Also
PathwayList

Examples

as.list(pathways("hsapiens", "kegg"))
**buildPathway**

*Build a Pathway object.*

**Description**

This function creates a new object of type `Pathway` given a data frame describing its edges.

**Usage**

```r
buildPathway(id, title, species, database, proteinEdges, 
metaboliteEdges = NULL, mixedEdges = NULL, 
timestamp = NULL)
```

**Arguments**

- `id`  
  the pathway identifier.
- `title`  
  the title of the pathway.
- `species`  
  the species the pathway belongs to.
- `database`  
  the name of the database the pathway derives from.
- `proteinEdges`  
  a data.frame of edges between proteins (or genes). 
  Must have the following columns: `src_type`, `src`, `dest_type`, `dest`, `direction` and `type`. 
  Direction must be one of the two strings: "directed" or "undirected".
- `metaboliteEdges`  
  interactions between metabolites. 
  Can be NULL. Otherwise, it must have the same structure as `proteinEdges`.
- `mixedEdges`  
  interactions between metabolites and proteins. 
  Can be NULL. Otherwise, it must have the same structure as `proteinEdges`.
- `timestamp`  
  when the pathway was annotated, by default the time `buildPathway` is called.

**See Also**

`Pathway-class`

**Examples**

```r
data <- data.frame(src_type = "ENTREZID", src="672", 
dest_type = "ENTREZID", dest="7157", 
direction="undirected", type="binding")
pathway <- buildPathway("#1", "example", "hsapiens", "database", data)
```

# Example with metabolites:
```r
data <- data.frame(src_type = "ENTREZID", src="672", 
dest_type = "ENTREZID", dest="7157", 
direction="undirected", type="binding")
metabolite_data <- data.frame(src_type = "CHEBI", src="77750", 
dest_type = "ENTREZID", dest="7157", 
direction="undirected", type="binding")
pathway <- buildPathway("#1", "example", "hsapiens", "database", 
data, metabolite_data)
```
convertIdentifiers  

Convert the node identifiers of a pathway.

**Description**

Converts the node identifiers of pathways.

If the option Ncpus is set to a value larger than 1 and the package parallel is installed, the conversion procedure will automatically use multiple cores.

**Usage**

`convertIdentifiers(x, to)`

**Arguments**

- `x` can be a list of pathways or a single pathway
- `to` a string describing the type of the identifier. Can assume the values "entrez", "symbol" or the name of one of the columns provided by an Annotation package (for example, "UNIPROT").

**Value**

A Pathway object.

**See Also**

Pathway

**Examples**

```r
r <- pathways("hsapiens", "reactome")
convertIdentifiers(r$'mTOR signalling', "symbol")
```

---

cytoscapePlot  

Plot a pathway graph in Cytoscape

**Description**

Renders the topology of a pathway as a Cytoscape graph.

**Usage**

`cytoscapePlot(pathway, ..., cy.ver = 3)`

**Arguments**

- `pathway` a Pathway object.
- `...` optional arguments forwarded to `pathwayGraph`.
- `cy.ver` select a Cytoscape version. Only version 3 is supported in this release.
Details

Requires the RCy3 package.

Value

An invisible list with two items:

- graph: the graphNEL object sent to Cytoscape.
- suid: the RCy3 network SUID.

See Also

Pathway
pathwayGraph

Examples

```r
## Not run:
r <- pathways()
cytoscapePlot(convertIdentifiers(reactome$`Unwinding of DNA`, "symbol"))
## End(Not run)
```

Pathway-class  

Class “Pathway”

Description

A biological pathway.

Variants

A Pathway instance actually stores multiple variants of the same biological data.

This is the list of included variants:

- proteins: includes only interactions among proteins;
- metabolites: includes only interactions among metabolites;
- mixed: includes all available interactions.

Methods

- pathwayId(p): Returns the native ID of the pathway.
- pathwayTitle(p): Returns the title of the pathway.
- pathwayDatabase(p): Returns the name of the database the pathway was derived from.
- pathwaySpecies(p): Returns the name of the species in which the pathway was annotated.
- pathwayTimestamp(p): Returns the date of pathway data retrieval.
- pathwayURL(p): Returns the URL of the pathway in its original database, if available.
- convertIdentifiers(p, to): Returns a new pathway using a different type of node identifiers.
pathwayDatabases

edges(p, which = c("proteins", "metabolites", "mixed"), stringsAsFactors = TRUE):
Returns a data.frame describing the edges of this pathway.
The option which selects the desired pathway variant (see section "Variants" above).
If stringsAsFactors is TRUE, strings are converted to factors.

nodes(p, which = c("proteins", "metabolites", "mixed")): Returns the names of the nodes
belonging to this pathway.
The option which selects the desired pathway variant (see section "Variants" above).

plot(p): Shows the pathway topology in Cytoscape.

runClipper(p, expr, classes, method, ...): Runs a clipper analysis over the pathway.
runTopologyGSA(p, test, exp1, exp2, alpha, ...): Runs a topologyGSA analysis over the
pathway.

Author(s)
Gabriele Sales

See Also
pathways

---

pathwayDatabases  List the available pathway databases.

Description
Obtains the list of pathway databases available through graphite.

Usage
pathwayDatabases()

Value
Returns a data.frame with two columns: species and database.

Author(s)
Gabriele Sales

See Also
pathways

Examples
pathwayDatabases()
pathwayGraph

Graph representing the topology of a pathway

Description

Builds a graphNEL object representing the topology of a pathway.

Usage

pathwayGraph(pathway, which = "proteins", edge.types = NULL)

Arguments

- **pathway**: a Pathway object.
- **which**: the pathway variant you want.
  
  See Pathway documentation for a list of the supported variants.
- **edge.types**: keep only the edges matching the type names in this vector.

Value

A graphNEL object.

See Also

Pathway

graphNEL

Examples

```r
r <- pathways("hsapiens", "reactome")
pathwayGraph(r$"mTOR signalling", edge.types="Binding")
```

PathwayList-class

Class "PathwayList"

Description

A collection of pathways from a single database.

Extends

Class "Pathways", directly.
Methods

l[[i]]: Returns a selection of the pathways contained in the pathway list.
l[[[i]]] Access one of the pathways contained in the pathway list.
l$'title' Access one of the pathways by its title.
convertIdentifiers(l, to) Returns a new list of pathways using a different type of node identifiers.
length(l) Returns the number of pathways contained in the list.
names(l) Returns the titles of the pathways contained in the list.
prepareSPIA(l, pathwaySetName, print.names=FALSE) Prepares the pathways for a SPIA analysis.
runClipper(l, expr, classes, method, maxNodes=150, ...) Runs a clipper analysis over all the pathways in the list.
runTopologyGSA(l, test, exp1, exp2, alpha, maxNodes=150, ...) Runs a topologyGSA analysis over all the pathways in the list.

Author(s)

Gabriele Sales

See Also

pathways

pathways

Retrieve a list of pathways.

Description

Retrieve a list of pathways from a database for a given species.

graphite currently supports the following databases:

- BioCarta
- HumanCyc
- KEGG
- NCI-Nature Pathway Interaction Database
- PANTHER
- PharmGKB
- Reactome
- SMPDB

Call the pathwayDatabase function for more details.

Usage

pathways(species, database)
Arguments

species one of the supported species

database the name of the pathway database

Value

A PathwayList object.

See Also

PathwayList, pathwayDatabases

Examples

pathways("hsapiens", "reactome")

Pathways-class Class "Pathways"

Description

A virtual class acting as a common parent to all other classes representing pathway databases.

Objects from the Class

A virtual Class: No objects may be created from it.

Methods

No methods defined with class "Pathways" in the signature.

Author(s)

Gabriele Sales

See Also

PathwayList
prepareSPIA

Prepare pathway dataset needed by runSPIA.

Description

Prepare pathway dataset needed by runSPIA. See runSPIA and spia for more details.

Usage

prepareSPIA(db, pathwaySetName, print.names = FALSE)

Arguments

db a PathwayList object or a list of Pathways.
pathwaySetName name of the output pathway set.
print.names print pathway names as the conversion advances.

References


See Also

runSPIA
spia
PathwayList

runClipper

Run a topological analysis on an expression dataset using clipper.

Description

clipper is a package for topological gene set analysis. It implements a two-step empirical approach based on the exploitation of graph decomposition into a junction tree to reconstruct the most relevant signal path. In the first step clipper selects significant pathways according to statistical tests on the means and the concentration matrices of the graphs derived from pathway topologies. Then, it “clips” the whole pathway identifying the signal paths having the greatest association with a specific phenotype.

If the option Ncpus is set to a value larger than 1 and the package parallel is installed, the conversion procedure will automatically use multiple cores.
runClipper

Usage

runClipper(x, expr, classes, method, which = "proteins", seed = NULL, ...)

Arguments

x a PathwayList, a list of Pathways or a single Pathway object.
expr a matrix (size: number p of genes x number n of samples) of gene expression.
classes a vector (length: n) of class assignments.
method the kind of test to perform on the cliques. It could be either "mean" or "variance".
which the pathway variant you want.
See Pathway documentation for a list of the supported variants.
seed if not NULL, set the seed for the random number generator used by clipper.
... additional options: see for details easyClip.

When invoked on a PathwayList, you can use the named option maxNodes to limit the analysis to those pathways with at most a given number of nodes.

Details

The expression data and the pathway have to be annotated in the same set of identifiers.

References


See Also

collider

easyClip

Examples

if (require(clipper) & require(ALL) & require(a4Preproc)) {
  data(ALL)
  pheno <- as(phenoData(ALL), "data.frame")
  samples <- unlist(lapply(c("NEG", "BCR/ABL"), function(t) {
    which(grepl("^B\d+$", pheno$BT) & (pheno$mol.biol == t))[1:10]
  }))
  classes <- c(rep(1,10), rep(2,10))
  expr <- exprs(ALL)[,samples]
  rownames(expr) <- paste("ENTREZID", featureData(addGeneInfo(ALL))$ENTREZID, sep = ":")
  k <- as.list(pathways("hsapiens", "kegg"))
  selected <- k[c("Bladder cancer", "Hippo signaling pathway - multiple species")]
  runClipper(selected, expr, classes, "mean", pathThr = 0.1)
}
runSPIA

**Run SPIA analysis**

**Description**

Run a topological analysis on an expression dataset using SPIA.

**Usage**

```r
runSPIA(de, all, pathwaySetName, ...)
```

**Arguments**

- `de` A named vector containing log2 fold-changes of the differentially expressed genes. The names of this numeric vector are Entrez gene IDs.
- `all` A vector with the Entrez IDs in the reference set. If the data was obtained from a microarray experiment, this set will contain all genes present on the specific array used for the experiment. This vector should contain all names of the 'de' argument.
- `pathwaySetName` The name of a pathway set created with `prepareSPIA`.
- `...` Additional options to pass to `spia`.

**Details**

The spia option "organism" is internally used. It is an error use it in the additional options.

**Value**

The same of spia, without KEGG links. A data frame containing the ranked pathways and various statistics: pSize is the number of genes on the pathway; NDE is the number of DE genes per pathway; tA is the observed total perturbation accumulation in the pathway; pNDE is the probability to observe at least NDE genes on the pathway using a hypergeometric model; pPERT is the probability to observe a total accumulation more extreme than tA only by chance; pG is the p-value obtained by combining pNDE and pPERT; pGfdr and pGFWER are the False Discovery Rate and respectively Bonferroni adjusted global p-values; and the Status gives the direction in which the pathway is perturbed (activated or inhibited).

**References**


**See Also**

`spia`
Examples

```r
if (require(SPIA) && require(hgu133plus2.db)) {
  data(colorectalCancer)
  top$ENTREZ <- mapIds(hgu133plus2.db, top$ID, "ENTREZID", "PROBEID", multiVals = "first")
  top <- top[!is.na(top$ENTREZ) & !duplicated(top$ENTREZ),]
  top$ENTREZ <- paste("ENTREZID", top$ENTREZ, sep = ":")
  tg1 <- top[top$adj.P.Val < 0.05,]
  DE_Colorectal = tg1$logFC
  names(DE_Colorectal) <- tg1$ENTREZ
  ALL_Colorectal <- top$ENTREZ
  biocarta <- pathways("hsapiens", "biocarta")[1:20]
  biocarta <- convertIdentifiers(biocarta, "ENTREZID")
  prepareSPIA(biocarta, "biocartaEx")
  runSPIA(de = DE_Colorectal, all = ALL_Colorectal, "biocartaEx")
}
```

---

**runTopologyGSA**

Run a topological analysis on an expression dataset using topologyGSA.

**Description**

Use graphical models to test the pathway components highlighting those involved in its deregulation.

If the option `Ncpus` is set to a value larger than 1 and the package `parallel` is installed, the conversion procedure will automatically use multiple cores.

**Usage**

```r
runTopologyGSA(x, test, exp1, exp2, alpha, ...)
```

**Arguments**

- `x` a `PathwayList`, a list of `Pathways` or a single `Pathway` object.
- `test` Either "var" and "mean". Determine the type of test used by `topologyGSA`.
- `exp1` Experiment matrix of the first class, genes in columns.
- `exp2` Experiment matrix of the second class, genes in columns.
- `alpha` Significance level of the test.
- `...` Additional parameters forwarded to `topologyGSA`.

When invoked on a `PathwayList`, can use the named option "maxNodes" to limit the analysis to those pathways having up to this given number of nodes.

**Details**

This function produces a warning and returns NULL when the number of genes in common between the expression matrices and the pathway is less than 3.
References

Massa MS, Chiogna M, Romualdi C. Gene set analysis exploiting the topology of a pathway. BMC System Biol. 2010 Sep 1;4:121.

See Also

pathway.var.test pathway.mean.test

Examples

```r
if (require(topologyGSA)) {
  data(examples)
  colnames(y1) <- paste("SYMBOL", colnames(y1), sep = ":")
  colnames(y2) <- paste("SYMBOL", colnames(y2), sep = ":")

  k <- pathways("hsapiens", "kegg")
  p <- convertIdentifiers(k["Fc epsilon RI signaling pathway"], "SYMBOL")
  runTopologyGSA(p, "var", y1, y2, 0.05)
}
```
## Index

### *Topic analysis*
- runClipper, 10
- runSPIA, 12
- runTopologyGSA, 13

### *Topic classes*
- Pathway-class, 5
- PathwayList-class, 7
- Pathways-class, 9

### *Topic clipper*
- runClipper, 10

### *Topic spia*
- runSPIA, 12

### *Topic topologyGSEA*
- runTopologyGSA, 13

### *Topic topology*
- runClipper, 10
- runSPIA, 12
- runTopologyGSA, 13

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>[],PathwayList-method (PathwayList-class), 7</td>
<td></td>
</tr>
<tr>
<td>[[,PathwayList-method (PathwayList-class), 7</td>
<td></td>
</tr>
<tr>
<td>$,PathwayList-method (PathwayList-class), 7</td>
<td></td>
</tr>
<tr>
<td>as.list.PathwayList, 2</td>
<td></td>
</tr>
<tr>
<td>buildPathway, 3</td>
<td></td>
</tr>
<tr>
<td>clipper, 11</td>
<td></td>
</tr>
<tr>
<td>convertIdentifiers, 4</td>
<td></td>
</tr>
<tr>
<td>convertIdentifiers,Pathway-method (Pathway-class), 5</td>
<td></td>
</tr>
<tr>
<td>convertIdentifiers,PathwayList-method (PathwayList-class), 7</td>
<td></td>
</tr>
<tr>
<td>cytoscapePlot, 4</td>
<td></td>
</tr>
<tr>
<td>easyClip, 11</td>
<td></td>
</tr>
<tr>
<td>edges,Pathway,character-method (Pathway-class), 5</td>
<td></td>
</tr>
<tr>
<td>edges,Pathway,missing-method (Pathway-class), 5</td>
<td></td>
</tr>
<tr>
<td>graphNEL, 5, 7</td>
<td></td>
</tr>
<tr>
<td>length,PathwayList-method (PathwayList-class), 7</td>
<td></td>
</tr>
<tr>
<td>names,PathwayList-method (PathwayList-class), 7</td>
<td></td>
</tr>
<tr>
<td>nodes,Pathway-method (Pathway-class), 5</td>
<td></td>
</tr>
<tr>
<td>Pathway, 2, 4, 5, 7, 10, 11, 13</td>
<td></td>
</tr>
<tr>
<td>Pathway-class, 5</td>
<td></td>
</tr>
<tr>
<td>pathway.mean.test, 14</td>
<td></td>
</tr>
<tr>
<td>pathway.var.test, 14</td>
<td></td>
</tr>
<tr>
<td>pathwayDatabase, 8</td>
<td></td>
</tr>
<tr>
<td>pathwayDatabase (Pathway-class), 5</td>
<td></td>
</tr>
<tr>
<td>pathwayDatabases, 6, 9</td>
<td></td>
</tr>
<tr>
<td>pathwayGraph, 4, 5, 7</td>
<td></td>
</tr>
<tr>
<td>pathwayId (Pathway-class), 5</td>
<td></td>
</tr>
<tr>
<td>PathwayList, 2, 9–11, 13</td>
<td></td>
</tr>
<tr>
<td>PathwayList-class, 7</td>
<td></td>
</tr>
<tr>
<td>Pathways, 7</td>
<td></td>
</tr>
<tr>
<td>pathways, 6, 8, 8</td>
<td></td>
</tr>
<tr>
<td>Pathways-class, 9</td>
<td></td>
</tr>
<tr>
<td>pathwaySpecies (Pathway-class), 5</td>
<td></td>
</tr>
<tr>
<td>pathwayTimestamp (Pathway-class), 5</td>
<td></td>
</tr>
<tr>
<td>pathwayTitle (Pathway-class), 5</td>
<td></td>
</tr>
<tr>
<td>pathwayURL (Pathway-class), 5</td>
<td></td>
</tr>
<tr>
<td>plot,Pathway,ANY-method (Pathway-class), 5</td>
<td></td>
</tr>
<tr>
<td>prepareSPIA, 10, 12</td>
<td></td>
</tr>
<tr>
<td>prepareSPIA,list-method (prepareSPIA), 10</td>
<td></td>
</tr>
<tr>
<td>prepareSPIA,PathwayList-method (PathwayList-class), 7</td>
<td></td>
</tr>
<tr>
<td>runClipper, 10</td>
<td></td>
</tr>
<tr>
<td>runClipper,list-method (runClipper), 10</td>
<td></td>
</tr>
<tr>
<td>runClipper,Pathway-method (Pathway-class), 5</td>
<td></td>
</tr>
<tr>
<td>runClipper,PathwayList-method (PathwayList-class), 7</td>
<td></td>
</tr>
<tr>
<td>runClipperMulti (runClipper), 10</td>
<td></td>
</tr>
<tr>
<td>runSPIA, 10, 12</td>
<td></td>
</tr>
<tr>
<td>runTopologyGSA, 13</td>
<td></td>
</tr>
<tr>
<td>runTopologyGSA,PathwayList-method (runTopologyGSA), 13</td>
<td></td>
</tr>
</tbody>
</table>
runTopologyGSA, Pathway-method
(Pathway-class), 5
runTopologyGSA, PathwayList-method
(PathwayList-class), 7
runTopologyGSAMulti (runTopologyGSA), 13
spia, 10, 12