Package ‘ROntoTools’

May 30, 2024

Type Package
Title R Onto-Tools suite
Version 2.32.0
Author Calin Voichita <calin@wayne.edu> and Sahar Ansari
         <saharansari@wayne.edu> and Sorin Draghici <sorin@advaitabio.com>
Maintainer Sorin Draghici <sorin@advaitabio.com>
Description Suite of tools for functional analysis.
biocViews NetworkAnalysis, Microarray, GraphsAndNetworks
License CC BY-NC-ND 4.0 + file LICENSE
Depends methods, graph, boot, KEGGREST, KEGGgraph, Rgraphviz
Suggests RUnit, BiocGenerics
RoxygenNote 5.0.1
git_url https://git.bioconductor.org/packages/ROntoTools
git_branch RELEASE_3_19
git_last_commit b64b430
git_last_commit_date 2024-04-30
Repository Bioconductor 3.19
Date/Publication 2024-05-29

Contents

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>alpha1MR</td>
<td>2</td>
</tr>
<tr>
<td>alphaMLG</td>
<td>3</td>
</tr>
<tr>
<td>compute.fisher</td>
<td>4</td>
</tr>
<tr>
<td>compute.normalInv</td>
<td>5</td>
</tr>
<tr>
<td>KEGGpathway2Graph</td>
<td>6</td>
</tr>
<tr>
<td>keggPathwayGraphs</td>
<td>6</td>
</tr>
<tr>
<td>keggPathwayNames</td>
<td>7</td>
</tr>
<tr>
<td>nodeWeights</td>
<td>8</td>
</tr>
<tr>
<td>pDis</td>
<td>9</td>
</tr>
</tbody>
</table>
Compute alpha weights

Transform a vector of p-values into weights.

Usage

alpha1MR(pv, threshold = max(pv))

Arguments

pv vector of p-values
threshold the threshold value that was used to select DE genes

Details

Computes a set of weights from p-values using the formula $1 - \frac{pv}{threshold}$.

Author(s)

Calin Voichita and Sorin Draghici

See Also

pe
Examples

```
load(system.file("extdata/E-GEOD-21942.topTable.RData", package = "ROntoTools"))

head(alphaMLG(top$adj.P.Val))
```

---

**alphaMLG**  
*Compute alpha weights*

**Description**

Transform a vector of p-values into weights.

**Usage**

```
alphaMLG(pv, threshold = max(pv))
```

**Arguments**

- `pv`: vector of p-values
- `threshold`: the threshold value that was used to select DE genes

**Details**

Computes a set of weights from p-values using the formula \(-\log_{10}(pv/threshold)\).

**Author(s)**

Calin Voichita and Sorin Draghici

**See Also**

`pe`

**Examples**

```
load(system.file("extdata/E-GEOD-21942.topTable.RData", package = "ROntoTools"))

head(alphaMLG(top$adj.P.Val))
```
compute.fisher

Combine independent p-values using the Fisher method

Usage

compute.fisher(p, eps = 1e-06)

Arguments

- p: a vector of independent p-values
- eps: the minimal p-value considered (all p-values smaller will be set to this value)

Value

the combined p-value

Author(s)

Calin Voichita and Sorin Draghici

References


See Also

pe, compute.normalInv

Examples

p <- c(.1, .01)
compute.fisher(p)
**compute.normalInv**

*Combine independent p-values using the normal inversion method*

**Description**

Combine independent p-values using the normal inversion method

**Usage**

```r
compute.normalInv(p, eps = 1e-06)
```

**Arguments**

- `p`: a vector of independent p-values
- `eps`: the minimal p-value considered (all p-values smaller will be set to this value)

**Value**

the combined p-value

**Author(s)**

Calin Voichita and Sorin Draghici

**References**


**See Also**

`pe`, `compute.fisher`

**Examples**

```r
p <- c(.1, .01)
compute.normalInv(p)
```
KEGGpathwayGraphs

Modified version of the same function from KEGGgraph

Description
Modified version of the same function from KEGGgraph

Usage
KEGGPathway2Graph(pathway, genesOnly = TRUE, expandGenes = TRUE)

keggPathwayGraphs
Download and parse KEGG pathway data

Description
Download and parse KEGG pathway data

Usage
keggPathwayGraphs(organism = "hsa", targRelTypes = c("GErel", "PCrel", "PPrel"), relPercThresh = 0.9, nodeOnlyGraphs = FALSE, updateCache = FALSE, verbose = TRUE)

Arguments
organism          organism code as defined by KEGG
orgRelTypes       target relation types
relPercThresh     percentage of the number of relation types over all possible realtions in the pathway
nodeOnlyGraphs    allow graphs with no edges
updateCache       re-download KEGG data
verbose           show progress of downloading and parsing

Value
A list of graphNEL objects encoding the pathway information.

Author(s)
Calin Voichita and Sorin Draghici

See Also
keggPathwayNames
keggPathwayNames

Examples

# The pathway cache provided as part of the pathway contains only the
# pathways that passed the default filtering. We recommend, re-downloading
# the pathways using the updateCache parameter
kpg <- keggPathwayGraphs("hsa")

# to update the pathway cache for human run:
# kpg <- keggPathwayGraphs("hsa", updateCache = TRUE)
# this is time consuming and depends on the available bandwith.

head(names(kpg))

kpg[['path:hsa04110']]
head(nodes(kpg[['path:hsa04110']]))
head(edges(kpg[['path:hsa04110']]))

keggPathwayNames Obtain KEGG pathway titles

Description

Obtain KEGG pathway titles

Usage

keggPathwayNames(organism = "hsa", updateCache = FALSE, verbose = TRUE)

Arguments

organism organism code as defined by KEGG
updateCache re-download KEGG data
verbose show progress of downloading and parsing

Value

A named vector of pathway titles. The names of the vector are the pathway KEGG IDs.

Author(s)

Calin Voichita and Sorin Draghici

See Also

keggPathwayGraphs
nodeWeights

Examples

```r
kpn <- keggPathwayNames("hsa")

# to update the pathway cache for human run:
# kpn <- keggPathwayNames("hsa", updateCache = TRUE)
# this is time consuming and depends on the available bandwidth.

head(kpn)
```

---

nodeWeights | Retrieve the node weights of a graph

Description

A generic function that returns the node weights of a graph. If `index` is specified, only the weights of the specified nodes are returned. The user can control which node attribute is interpreted as the weight.

Usage

```r
nodeWeights(object, index, ..., attr = "weight", default = 1)
```

## S4 method for signature 'graph,character'
```r
nodeWeights(object, index, attr, default)
```

## S4 method for signature 'graph,numeric'
```r
nodeWeights(object, index, attr, default)
```

## S4 method for signature 'graph,missing'
```r
nodeWeights(object, index, attr, default)
```

Arguments

- **object** A graph, any object that inherits the graph class.
- **index** If supplied, a character or numeric vector of node names or indices.
- **...** Unused.
- **attr** The name of the node attribute to use as a weight. You can view the list of defined node attributes and their default values using nodeDataDefaults.
- **default** The value to use if `object` has no node attribute named by the value of `attr`. The default is the value 1.
Details

The weights of all nodes identified by the index are returned. If index is not supplied, the weights of all nodes are returned.

By default, nodeWeights looks for an node attribute with name "weight" and, if found, uses these values to construct the node weight vector. You can make use of attributes stored under a different name by providing a value for the attr argument. For example, if object is a graph instance with an node attribute named "WTS", then the call nodeWeights(object, attr="WTS") will attempt to use those values.

If the graph instance does not have an node attribute with name given by the value of the attr argument, default will be used as the weight for all nodes. Note that if there is an attribute named by attr, then its default value will be used for nodes not specifically customized. See nodeData and nodeDataDefaults for more information.

Value

A named vector with the node weights. The names of the vector are the names of the specified index, or all nodes if index was not provided.

Author(s)

Calin Voichita and Sorin Draghici

See Also

nodes, nodeData

Examples

library(graph)
V <- LETTERS[1:4]
g <- graphNEL(nodes = V, edgemode = "directed")
nodeWeights(g)
nodeWeights(g, "B")
nodeWeights(g, attr = "WT", default = 3)
Usage

$pDis(x, \text{graphs}, \text{ref} = \text{NULL}, \text{nboot} = 2000, \text{verbose} = \text{TRUE}, \text{cluster} = \text{NULL}, \text{seed} = \text{NULL})$

Arguments

- \(x\) named vector of log fold changes for the differentially expressed genes; \(\text{names}(x)\) must use the same id’s as \(\text{ref}\) and the nodes of the graphs
- \(\text{graphs}\) list of pathway graphs as objects of type graph (e.g., \text{graphNEL}); the graphs must be weighted graphs (i.e., have an attribute \text{weight} for both nodes and edges)
- \(\text{ref}\) the reference vector for all genes in the analysis; if the reference is not provided or it is identical to \(\text{names}(x)\) a cut-off free analysis is performed
- \(\text{nboot}\) number of bootstrap iterations
- \(\text{verbose}\) print progress output
- \(\text{cluster}\) a cluster object created by \text{makeCluster} for parallel computations
- \(\text{seed}\) an integer value passed to \text{set.seed()} during the bootstrap permutations

Details

See details in the cited articles.

Value

An object of class \(\text{pDisRes-class}\).

Author(s)

Calin Voichita, Sahar Ansari and Sorin Draghici

References


See Also

Summary, \text{keggPathwayGraphs}, \text{setNodeWeights}, \text{setEdgeWeights}

Examples

```
# load a multiple sclerosis study (public data available in Array Express
# ID: E-GEOD-21942)
# This file contains the top table, produced by the limma package with
# added gene information. All the probe sets with no gene associate to them,
# have been removed. Only the most significant probe set for each gene has been
# kept (the table is already ordered by p-value)
```
# The table contains the expression fold change and significance of each probe set in peripheral blood mononuclear cells (PBMC) from 12 MS patients and 15 controls. 

load(system.file("extdata/E-GEOD-21942.topTable.RData", package = "ROntoTools"))
head(top)

# select differentially expressed genes at 1% and save their fold change in a vector fc and their p-values in a vector pv
fc <- top$logFC[top$adj.P.Val <= .01]
names(fc) <- top$entrez[top$adj.P.Val <= .01]

pv <- top$P.Value[top$adj.P.Val <= .01]
names(pv) <- top$entrez[top$adj.P.Val <= .01]

# alternativly use all the genes for the analysis
# NOT RUN:
# fc <- top$logFC
# names(fc) <- top$entrez

# pv <- top$P.Value
# names(pv) <- top$entrez

# get the reference
ref <- top$entrez

# load the set of pathways
kpg <- keggPathwayGraphs("hsa")

# set the beta information (see the cited documents for meaning of beta)
kpg <- setEdgeWeights(kpg)

# inclute the significance information in the analysis (see Voichita:2012 for more information)
# set the alpha information based on the pv with one of the predefined methods
kpg <- setNodeWeights(kpg, weights = alphaMLG(pv), defaultWeight = 1)

# perform the pathway analysis
# in order to obtain accurate results the number of bootstraps, nboot, should be increase to a number like 2000
pDisRes <- pDis(fc, graphs = kpg, ref = ref, nboot = 100, verbose = TRUE)

# obtain summary of results
head(Summary(pDisRes))

## pDisPathway-class

Class that encodes the result of pDis analysis for a single pathway

### Description

Class that encodes the result of pDis analysis for a single pathway
Slots

map: an object of type graph (e.g., graphNEL).
input: named vector of fold changes for genes on this pathway. The names of the genes are the original IDs used in the analysis
ref: vector of reference IDs on this pathway
boot: an object of class boot encoding the bootstrap information.
pDis: the gene primary dis-regulation for all genes on the pathway, as computed by primary dis-regulation.
asGS: pathway was considered as gene set

Author(s)
Calin Voichita, Sahar Ansari and Sorin Draghici

See Also
pDis, pDisRes-class

pDisRes-class

Primary dis-regulation (pDis) result class

Description
This class is used to encode the results of the pathway analysis performed by the function pDis.

Details
The slots input and ref record global information related to the whole analysis, while the pathways slot records the specific results as pDisPathway-class for each one of the pathways used in the analysis.

Slots

pathways: A list of pDisPathway-class objects.
input: named vector of fold changes used for the analysis. The names of the vector are the IDs originaly used.
ref: character vector containing the IDs used as reference in the analysis.
cutOffFree: boolean value indicating if a cut-of-free analysis has been performed.

Author(s)
Calin Voichita, Sahar Ansari and Sorin Draghici

See Also
pDis, pDisPathway-class
Description

Pathway-Express: Pathway analysis of signaling pathways

Usage

pe(x, graphs, ref = NULL, nboot = 2000, verbose = TRUE, cluster = NULL, seed = NULL)

Arguments

x
named vector of log fold changes for the differentially expressed genes; names(x) must use the same id’s as ref and the nodes of the graphs

graphs
list of pathway graphs as objects of type graph (e.g., graphNEL); the graphs must be weighted graphs (i.e., have an attribute weight for both nodes and edges)

ref
the reference vector for all genes in the analysis; if the reference is not provided or it is identical to names(x) a cut-off free analysis is performed

nboot
number of bootstrap iterations

verbose
print progress output

cluster
a cluster object created by makeCluster for parallel computations

seed
an integer value passed to set.seed() during the bootstrap permutations

Details

See details in the cited articles.

Value

An object of class peRes-class.

Author(s)

Calin Voichita and Sorin Draghici

References


See Also

Summary, plot, peRes, missing-method, keggPathwayGraphs, setNodeWeights, setEdgeWeights

Examples

# load a multiple sclerosis study (public data available in Array Express
# ID: E-GEOD-21942)
# This file contains the top table, produced by the limma package with
# added gene information. All the probe sets with no gene associate to them,
# have been removed. Only the most significant probe set for each gene has been
# kept (the table is already ordered by p-value)
# The table contains the expression fold change and significance of each
# probe set in peripheral blood mononuclear cells (PBMC) from 12 MS patients
# and 15 controls.
load(system.file("extdata/E-GEOD-21942.topTable.RData", package = "ROntoTools"))
head(top)

# select differentially expressed genes at 1% and save their fold change in a
# vector fc and their p-values in a vector pv
fc <- top$logFC[top$adj.P.Val <= .01]
names(fc) <- top$entrez[top$adj.P.Val <= .01]

pv <- top$P.Value[top$adj.P.Val <= .01]
names(pv) <- top$entrez[top$adj.P.Val <= .01]

# alternativly use all the genes for the analysis
# NOT RUN:
# fc <- top$logFC
# names(fc) <- top$entrez

# pv <- top$P.Value
# names(pv) <- top$entrez

# get the reference
ref <- top$entrez

# load the set of pathways
kpg <- keggPathwayGraphs("hsa")

# set the beta information (see the citated documents for meaning of beta)
kpg <- setEdgeWeights(kpg)

# inlcude the significance information in the analysis (see Voichita:2012
# for more information)
# set the alpha information based on the pv with one of the predefined methods
kpg <- setNodeWeights(kpg, weights = alphaMLG(pv), defaultWeight = 1)

# perform the pathway analysis
# in order to obtain accurate results the number of boostraps, nboot, should
# be increase to a number like 2000
peRes <- pe(fc, graphs = kpg, ref = ref, nboot = 100, verbose = TRUE)

# obtain summary of results
head(Summary(peRes))

---

**peEdgeRenderInfo**

Extract edge render information from a pePathway-class object

---

**Description**

Extract edge render information from a pePathway-class object

**Usage**

```r
peEdgeRenderInfo(x, pos.col = "black", pos.lty = "solid", pos.ah = "vee",
                 neg.col = "black", neg.lty = "dashed", neg.ah = "tee",
                 zero.col = "lightgray", zero.lty = "dotted", zero.ah = "none")
```

**Arguments**

- `x` an object of class `pePathway-class`
- `pos.col` color of the edges with positive weight
- `pos.lty` line type of the edges with positive weight
- `pos.ah` arrow head of the edges with positive weight
- `neg.col` color of the edges with negative weight
- `neg.lty` line type of the edges with negative weight
- `neg.ah` arrow head of the edges with negative weight
- `zero.col` color of the edges with zero weight
- `zero.lty` color of the edges with zero weight
- `zero.ah` color of the edges with zero weight

**Value**

a named list as expected by `edgeRenderInfo`

**Author(s)**

Calin Voichita and Sorin Draghici
peNodeRenderInfo

Extract node render information from a pePathway-class object

Description

Extract node render information from a pePathway-class object

Usage

```
peNodeRenderInfo(x, y = "Pert", input.shape = "box",
                default.shape = "ellipse", pos.col = "red", neg.col = "blue",
                zero.col = "white")
```

Arguments

- **x**: an object of class `pePathway-class`
- **y**: a string representing the factor to be represented (Pert, Acc or input; see `pePathway-class`)
- **input.shape**: shape of nodes that have measured expression change
- **default.shape**: shape of all other nodes

Examples

```r
# load experiment
load(system.file("extdata/E-GEOD-21942.topTable.RData", package = "ROntoTools"))
fc <- top$logFC[top$adj.P.Val <= .01]
names(fc) <- top$entrez[top$adj.P.Val <= .01]
ref <- top$entrez

# load the set of pathways
kpg <- keggPathwayGraphs("hsa")
kpg <- setEdgeWeights(kpg)
kpg <- setNodeWeights(kpg, defaultWeight = 1)

# perform the pathway analysis
peRes <- pe(fc, graphs = kpg, ref = ref, nboot = 100, verbose = TRUE)

p <- peRes@pathways[[50]]
g <- layoutGraph(p@map, layoutType = "dot")
graphRenderInfo(g) <- list(fixedsize = FALSE)
edgeRenderInfo(g) <- peEdgeRenderInfo(p)
nodeRenderInfo(g) <- peNodeRenderInfo(p)

# notice the different type of edges in the graph (solid/dashed/dotted)
renderGraph(g)
```
Value

a named list as expected by `nodeRenderInfo`

Author(s)

Calin Voichita and Sorin Draghici

See Also

`nodeRenderInfo`, `par`

Examples

```r
# load experiment
load(system.file("extdata/E-GEOD-21942.topTable.RData", package = "ROntoTools"))
fc <- top$logFC[top$adj.P.Val <= .01]
names(fc) <- top$entrez[top$adj.P.Val <= .01]
ref <- top$entrez

# load the set of pathways
kpg <- keggPathwayGraphs("hsa")
kpg <- setEdgeWeights(kpg)
kpg <- setNodeWeights(kpg, defaultWeight = 1)

# perform the pathway analysis
peRes <- pe(fc, graphs = kpg, ref = ref, nboot = 100, verbose = TRUE)

p <- peRes@pathways[[50]]
g <- layoutGraph(p@map, layoutType = "dot")
graphRenderInfo(g) <- list(fixedsize = FALSE)
edgeRenderInfo(g) <- peEdgeRenderInfo(p)
nodeRenderInfo(g) <- peNodeRenderInfo(p)

# notice the different type of nodes in the graph (box/circle)
# the color of each node represents the perturbation (red = positive, blue = negative)
# the shade represents the strength of the perturbation
renderGraph(g)
	nodeRenderInfo(g) <- peNodeRenderInfo(p, "Acc")

# now, the color of each node represents the accumulation (red = positive, blue = negative)
# notice that square nodes with no parents have no accumulation
renderGraph(g)
```
pePathway-class

Class that encodes the result of Pathway-Express for a single pathway

Description

Class that encodes the result of Pathway-Express for a single pathway

Slots

map: an object of type graph (e.g., graphNEL).
input: named vector of fold changes for genes on this pathway. The names of the genes are the original IDs used in the analysis
ref: vector of reference IDs on this pathway
boot: an object of class boot encoding the bootstrap information.
Pert: the gene perturbation factors for all genes on the pathway, as computed by Pathway-Express.
Acc: the gene accumulations for all genes on the pathway, as computed by Pathway-Express.
asGS: pathway was considered as gene set

Author(s)

Calin Voichita and Sorin Draghici

See Also

pe, peRes-class

peRes-class

Pathway-Express result class

Description

This class is used to encode the results of the pathway analysis performed by the function pe.

Details

The slots input and ref record global information related to the whole analysis, while the pathways slot records the specific results as pePathway-class for each one of the pathways used in the analysis.

Slots

pathways: A list of pePathway-class objects.
input: named vector of fold changes used for the analysis. The names of the vector are the IDs originally used.
ref: character vector containing the IDs used as reference in the analysis.
cutOffFree: boolean value indicating if a cut-of-free analysis has been performed.
Author(s)
Calin Voichita and Sorin Draghici

See Also
pe, pePathway-class

Description
Display graphical representation of pathway level statistic like: i) two way comparison between the measured expression change and one of the factors computed by Pathway-Express (pe) or ii) the bootstrap statistics of the same factors.

Usage
## S4 method for signature 'pePathway,missing'
plot(x, y, ..., type = "two.way", eps = 1e-06)

## S4 method for signature 'pePathway,character'
plot(x, y, main = "", ..., type = "two.way",
eps = 1e-06)

Arguments
x an object of type pePathway-class
y if provided, the factor to be plotted (either Acc (default) or Pert; see pePathway-class)
... Arguments to be passed to methods, such as par
type type of plot (either two.way (default) or boot)
eps any value smaller than this will be plotted as 0
main title

Author(s)
Calin Voichita and Sorin Draghici

See Also
pe, plot, peRes, missing-method, peNodeRenderInfo, peEdgeRenderInfo
Examples

```r
# load experiment
load(system.file("extdata/E-GEOD-21942.topTable.RData", package = "ROntoTools"))
fc <- top$logFC[top$adj.P.Val <= .01]
names(fc) <- top$entrez[top$adj.P.Val <= .01]
ref <- top$entrez

# load the set of pathways
kpg <- keggPathwayGraphs("hsa")
kpg <- setEdgeWeights(kpg)
kpg <- setNodeWeights(kpg, defaultWeight = 1)

# perform the pathway analysis (for more accurate results use nboot = 2000)
peRes <- pe(fc, graphs = kpg, ref = ref, nboot = 100, verbose = TRUE)
plot(peRes@pathways[[50]])
plot(peRes@pathways[[50]], "Pert", main = "Perturbation factor")
plot(peRes@pathways[[50]], type = "boot")
plot(peRes@pathways[[50]], "Pert", type = "boot", main = "Perturbation factor")
```

---

**plot,peRes,missing-method**

*Plot Pathway-Express result*

**Description**

Display a two-way plot using two of the p-values from the Pathway-Express analysis.

**Usage**

```r
## S4 method for signature 'peRes,missing'
plot(x, y, ..., comb.pv.func = compute.fisher,
     adjust.method = "fdr", threshold = 0.05, eps = 1e-06)

## S4 method for signature 'peRes,character'
plot(x, y, ..., comb.pv.func = compute.fisher,
     adjust.method = "fdr", threshold = 0.05, eps = 1e-06)
```

**Arguments**

- `x` an object of type `peRes-class`
- `y` vector of two p-values names to be combined using `comb.pv.func` (default: `c("pAcc", "pORA")`).
- `...` Arguments to be passed to methods, such as `par`. 
setEdgeWeights

comb.pv.func  the function to combine the p-values - takes as input a vector of p-values and returns the combined p-value (default: compute.fisher).

adjust.method  the name of the method to adjust the p-value (see p.adjust)

threshold  corrected p-value threshold

eps  any value smaller than this will be considered as eps (default: 1e-6).

Author(s)
Calin Voichita and Sorin Draghici

See Also
pe, summary.peRes, plot, pePathway, missing-method

Examples
# load experiment
load(system.file("extdata/E-GEOD-21942.topTable.RData", package = "ROntoTools"))
fc <- top$logFC[top$adj.P.Val <= .01]
names(fc) <- top$entrez[top$adj.P.Val <= .01]
ref <- top$entrez

# load the set of pathways
kpg <- keggPathwayGraphs("hsa")
kpg <- setEdgeWeights(kpg)
kpg <- setNodeWeights(kpg, defaultWeight = 1)

# perform the pathway analysis (for more accurate results use nboot = 2000)
peRes <- pe(fc, graphs = kpg, ref = ref, nboot = 100, verbose = TRUE)
plot(peRes)
plot(peRes, c("pPert","pORA"), comb.pv.func = compute.normalInv, threshold = .01)

setEdgeWeights  Set gene weights based on edge type

Description
setEdgeWeights

Usage
setEdgeWeights(graphList, edgeTypeAttr = "subtype",
edgeWeightByType = list(activation = 1, inhibition = -1, expression = 1,
repression = -1), defaultWeight = 0, combineWeights = sum,
nodeOnlyGraphs = FALSE)
Arguments

- **graphList**: a list of graphNEL objects
- **edgeTypeAttr**: edge attribute to be considered as the edge type. If the edge has multiple types, the edge type attribute is considered as a comma separated list of types
- **edgeWeightByType**: named list of weights, where the names of the list are the edge type (values of the attribute defined by `edgeTypeAttr`)
- **defaultWeight**: default value for an edge with a type not defined in `edgeWeightByType`
- **combineWeights**: for the edges with multiple types, the function to be applied on the vector of weights
- **nodeOnlyGraphs**: boolean value marking if graphs with no edges should be returned or not; note that graphs with all edge weights equal to 0 are considered node only graphs

Value

The `graphList` with the edge weights set.

Author(s)

Calin Voichita and Sorin Draghici

Examples

```r
# load the set of pathways
kpg <- keggPathwayGraphs("hsa")
kpg <- setEdgeWeights(kpg)

edgeWeights(kpg[['path:hsa04110']])
```

**setNodeWeights**

**Set node weights**

Description

Set node weights

Usage

```r
setNodeWeights(graphList, weights = NULL, defaultWeight = 1)
```
subGraphByNodeType

Arguments

- `graphList` a list of graph (e.g., `graphNEL`) objects
- `weights` named vector or matrix; if vector, the node is going to have the same weight in all graphs it appears; if matrix, the rows represent nodes and columns represent graphs and the node will have different weights in each pathway
- `defaultWeight` the default weight for all nodes not set by the parameter `weights`

Value

The `graphList` with the node weights set.

Author(s)

Calin Voichita and Sorin Draghici

Examples

# load the set of pathways
kpg <- keggPathwayGraphs("hsa")

kpg <- setNodeWeights(kpg)

nodeWeights(kpg[["path:hsa04110")])

Description

Modified version of the same function from KEGGgraph

Usage

`subGraphByNodeType(graph, type = "gene")`
Summary,pDisRes-method

Summarize the results of a Pathway-Express analysis

Description

Summarize the results of a Pathway-Express analysis

Usage

## S4 method for signature 'pDisRes'
Summary(x, ..., na.rm = FALSE)

Arguments

- **x**: Primary dis-regulation analysis result object obtained using `pDis`
- **...**: see `summary.pDisRes`
- **na.rm**: ignored

Summary,peRes-method

Summarize the results of a Pathway-Express analysis

Description

Summarize the results of a Pathway-Express analysis

Usage

## S4 method for signature 'peRes'
Summary(x, ..., na.rm = FALSE)

Arguments

- **x**: Pathway-Express analysis result object obtained using `pe`
- **...**: see `summary.peRes`
- **na.rm**: ignored
### Description

Summarize the results of a primary dis-regulation (pDis) analysis

### Usage

```r
tsummary.pDisRes(object, ..., pathNames = NULL, totalpDis = TRUE, normalize = TRUE, ppDis = TRUE, pORA = TRUE, comb.pv = c("ppDis", "pORA"), comb.pv.func = compute.fisher, order.by = "pComb", adjust.method = "fdr")
```

### Arguments

- **object**: pDis analysis result object obtained using `pDis`
- **...**: ignored
- **pathNames**: named vector of pathway names; the names of the vector are the IDs of the pathways
- **totalpDis**: boolean value indicating if the total primary dis-regulation should be computed
- **normalize**: boolean value indicating if normalization with regards to the bootstrap simulations should be performed on totalpDis
- **ppDis**: boolean value indicating if the significance of the total primary dis-regulation in regards to the bootstrap permutations should be computed
- **pORA**: boolean value indicating if the over-representation p-value should be computed
- **comb.pv**: vector of the p-value names to be combine (any of the above p-values)
- **comb.pv.func**: the function to combine the p-values; takes as input a vector of p-values and returns the combined p-value
- **order.by**: the name of the p-value that is used to order the results
- **adjust.method**: the name of the method to adjust the p-value (see `p.adjust`)

### See Also

- `pDis`

### Examples

```r
# load experiment
load(system.file("extdata/E-GEOD-21942.topTable.RData", package = "ROntoTools"))
fc <- top$logFC[top$adj.P.Val <= .01]
names(fc) <- top$entrez[top$adj.P.Val <= .01]
ref <- top$entrez

# load the set of pathways
```
kpg <- keggPathwayGraphs("hsa")
kpg <- setEdgeWeights(kpg)
kpg <- setNodeWeights(kpg, defaultWeight = 1)

# perform the pathway analysis
pDisRes <- pDis(fc, graphs = kpg, ref = ref, nboot = 100, verbose = TRUE)

# obtain summary of results
head(summary(pDisRes))

kpn <- keggPathwayNames("hsa")
head(summary(pDisRes))

head(summary(pDisRes, pathNames = kpn, totalpDis = FALSE,
      pORA = FALSE, comb.pv = NULL, order.by = "pDis"))

---

summary.peRes

**Summarize the results of a Pathway-Express analysis**

**Description**

Summarize the results of a Pathway-Express analysis

**Usage**

summary.peRes(object, ..., pathNames = NULL, totalAcc = TRUE, totalPert = TRUE, normalize = TRUE,
               pPert = TRUE, pAcc = TRUE, pORA = TRUE,
               comb.pv = c("pPert", "pORA"), comb.pv.func = compute.fisher,
               order.by = "pComb", adjust.method = "fdr")

**Arguments**

- **object** Pathways-Express result object obtained using `pe`
- **pathNames** named vector of pathway names; the names of the vector are the IDs of the pathways
- **totalAcc** boolean value indicating if the total accumulation should be computed
- **totalPert** boolean value indicating if the total perturbation should be computed
- **normalize** boolean value indicating if normalization with regards to the bootstrap simulations should be performed on totalAcc and totalPert
- **pPert** boolean value indicating if the significance of the total perturbation in regards to the bootstrap permutations should be computed
- **pAcc** boolean value indicating if the significance of the total accumulation in regards to the bootstrap permutations should be computed
summary.peRes

pORA  boolean value indicating if the over-representation p-value should be computed
comb.pv  vector of the p-value names to be combine (any of the above p-values)
comb.pv.func  the function to combine the p-values; takes as input a vector of p-values and returns the combined p-value
order.by  the name of the p-value that is used to order the results
adjust.method  the name of the method to adjust the p-value (see p.adjust)

See Also

pe

Examples

# load experiment
load(system.file("extdata/E-GEOD-21942.topTable.RData", package = "ROntoTools"))
fc <- top$logFC[top$adj.P.Val <= .01]
names(fc) <- top$entrez[top$adj.P.Val <= .01]
ref <- top$entrez

# load the set of pathways
kpg <- keggPathwayGraphs("hsa")
kpg <- setEdgeWeights(kpg)
kpg <- setNodeWeights(kpg, defaultWeight = 1)

# perform the pathway analysis
peRes <- pe(fc, graphs = kpg, ref = ref, nboot = 100, verbose = TRUE)

# obtain summary of results
head(summary(peRes))

kpn <- keggPathwayNames("hsa")

head(summary(peRes))

head(summary(peRes, pathNames = kpn, totalAcc = FALSE, totalPert = FALSE, pAcc = FALSE, pORA = FALSE, comb.pv = NULL, order.by = "pPert"))
Index

* internal
  KEGGpathway2Graph, 6
  subGraphByNodeType, 23
  alpha1MR, 2
  alphaMLG, 3
  compute.fisher, 4, 5, 21
  compute.normalInv, 4, 5
  edgeRenderInfo, 15, 16
  graphNEL, 6, 10, 12, 13, 18, 22, 23
  KEGGpathway2Graph, 6
  keggPathwayGraphs, 6, 7, 10, 14
  keggPathwayNames, 6, 7
  nodeData, 9
  nodeRenderInfo, 17
  nodes, 9
  nodeWeights, 8
  nodeWeights, graph, character-method
    (nodeWeights), 8
  nodeWeights, graph, missing-method
    (nodeWeights), 8
  nodeWeights, graph, numeric-method
    (nodeWeights), 8
  p.adjust, 21, 25, 27
  par, 16, 17, 19, 20
  pDis, 9, 12, 24, 25
  pDisPathway-class, 11
  pDisRes-class, 12
  pe, 2–5, 13, 18, 19, 21, 24, 26, 27
  peEdgeRenderInfo, 15, 19
  peNodeRenderInfo, 16, 19
  pePathway-class, 18
  peRes-class, 18
  plot, pePathway, character-method
    (plot, pePathway, missing-method), 19
  plot, peRes, character-method
    (plot, peRes, missing-method), 20
  plot, peRes, missing-method, 20
  setEdgeWeights, 10, 14, 21
  setNodeWeights, 10, 14, 22
  subGraphByNodeType, 23
  Summary, 10, 14
  Summary, pDisRes-method, 24
  Summary, peRes-method, 24
  summary, pDisRes, 24, 25
  summary, peRes, 21, 24, 26