Package ‘KEGGgraph’

November 6, 2025

Type Package

Title KEGGgraph: A graph approach to KEGG PATHWAY in R and Bioconductor

Version 1.71.0

Date 2025-04-14

Author Jitao David Zhang, with inputs from Paul Shannon and Hervé Pagès

Maintainer Jitao David Zhang <`jitao_david.zhang@roche.com`>

Description KEGGGraph is an interface between KEGG pathway and graph object as well as a collection of tools to analyze, dissect and visualize these graphs. It parses the regularly updated KGML (KEGG XML) files into graph models maintaining all essential pathway attributes. The package offers functionalities including parsing, graph operation, visualization and etc.

License GPL (>= 2)

LazyLoad yes

LazyData yes

Depends R (>= 3.5.0)

Imports methods, XML (>= 2.3-0), graph, utils, RCurl, Rgraphviz

Suggests RBGL, testthat, RColorBrewer, org.Hs.eg.db, hgu133plus2.db, SPIA

Collate kegg2graph-functions.R parse.R annotation.R graph.R kgmlfile.R misc.R vis.R

URL `https://accio.github.io/research/#software`

biocViews Pathways, GraphAndNetwork, Visualization, KEGG

Encoding UTF-8

git_url https://git.bioconductor.org/packages/KEGGgraph

git_branch devel

git_last_commit 09a8170

git_last_commit_date 2025-10-29

Repository Bioconductor 3.23

Date/Publication 2025-11-05

Contents

`colorectalcancerSPIA` *Colorectal cancer dataset*

Description

Colorectal cancer dataset provided by `SPIA` package. It is just a copy during the development of `SPIA` package in case the package is not available. It will be removed when the `SPIA` package is stable.

see the description of `SPIA` package.

Format

see the format of `SPIA` package.

Source

Yi Hong and Kok Sun Ho and Kong Weng Eu and Peh Yean Cheah, A susceptibility gene set for early onset colorectal cancer that integrates diverse signaling pathways: implication for tumorigenesis, Clin Cancer Res, 2007, 13(4),1107-14.

`expandKEGGNode` *Expand KEGG node of paralogues*

Description

The function expands KEGG node of paralogues, and is mainly used internally. The end-users are not expected to call it unless they know exactly what they are doing.

Usage

```
expandKEGGNode(node)
```

Arguments

`node`	An object of `KEGGNode-class`

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

`expandKEGGPathway` *Expand KEGG Pathway*

Description

The function expands paralogue nodes in KEGG pathway and returns expanded KEGG pathway, KEGG node and edge data is maintained.

Usage

```
expandKEGGPathway(pathway)
```

Arguments

`pathway`	An object of `KEGGPathway-class`

Details

The function expands nodes with paralogues in KEGG pathway and copy neccessary edges.

Value

An object of `KEGGPathway-class`

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

See Also

`expandKEGGNode`

Examples

```
sfile <- system.file("extdata/hsa04010.xml",package="KEGGgraph")
kegg.pathway <- parseKGML(sfile)
kegg.expandpathway <- expandKEGGPathway(kegg.pathway)
```

`getDisplayName-methods` *Get a character string as label for display*

Description

In KGML files, 'graph' element has a 'name' attribute to store the displaying name of a node, which is straighforward for end users. For example, biologists have no idea about a node 'hsa:1432' but its display name 'MAPK14' helps them to link this node to their knowledge. This method extract DisplayName from graph objects for KEGGNode and graph, where the method for graph returns the display names of its nodes.

Methods

object = "KEGGNode" An object of `KEGGNode-class`

object = "graph" A KEGG graph object

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

References

KGML Document Manual `https://www.genome.jp/kegg/docs/xml/`

Examples

```
sfile <- system.file("extdata/hsa04010.xml",package="KEGGgraph")
pathway <- parseKGML(sfile)

nodes <- nodes(pathway)
subnodes <- nodes[10:15]

sapply(subnodes, getDisplayName)
## compare them with getName, one 'displayName' may correspond to many paralogues
sapply(subnodes, getName)
```

`getEntryID-methods` *Get entry ID for single or list of KEGGNode or KEGGedge object(s)*

Description

The method extracts EntryIDs from `KEGGNode-class` or `KEGGEdge-class` object(s).

In case of `KEGGEdge-class` objects, the entryID of the nodes involved in the binary are returned as a vector *in the order specified by the direction of the relation*, that is, if the edge is defined as A->B, then the entryID returned from the edge equals to c(getEntryID(A), getEntryID(B)).

Methods

- **obj = "KEGGEdge"** Object of `KEGGEdge-class`
- **obj = "list"** A wrapper for list of `KEGGNode-class` or `KEGGEdge-class` objects

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

References

KGML Document Manual `https://www.genome.jp/kegg/docs/xml/`

Examples

```
sfile <- system.file("extdata/hsa04010.xml",package="KEGGgraph")
pathway <- parseKGML(sfile)

nodes <- nodes(pathway)
node <- nodes[[7]]
getEntryID(node)

edges <- edges(pathway)
edge <- edges[[7]]
getEntryID(edge)

getEntryID(nodes[1:4])
getEntryID(edges[1:4])
```

`getKEGGgeneLink-methods` *Get KEGG gene link*

Description

Tranlsate a object into a link point to the gene on KEGG website.

This method complies with the Gene link rule of the KEGG website.

Methods

- **object = "character"** A KEGGID, for example 'hsa:1423'

Examples

```
getKEGGgeneLink("hsa:1423")
```

`getKEGGID-methods` *Get KEGG ID*

Description

Get KEGGID from a `KEGGNode-class` object.

The `KEGGNode-class` can be either another pathway (KEGGID in the form like 'hsa\d*'), KEGG Gene ('hsa:\d*') or compound ('cpd:C\d*'). In case of the KEGG Gene ID, the organism prefix is removed when the value is returned.

Methods

object = "KEGGNode" An object of `KEGGNode-class`

Examples

```
wntfile <- system.file("extdata/hsa04310.xml",package="KEGGgraph")
wnt <- parseKGML(wntfile)
nodes <- nodes(wnt)
getKEGGID(nodes[[1]])
getKEGGID(nodes[[26]])
```

`getKEGGnodeData` *Get or set list of KEGG node or edge data*

Description

The 'get' methods extracts KEGG node (edge) attributes from a graph produced by calling `parseKGML2Graph` or `KEGGpathway2Graph`. The 'set' methods writes a list into the edge or node data.

Usage

```
getKEGGnodeData(graph, n)
getKEGGedgeData(graph, n)
```

Arguments

`graph`	a graph object by parsing KGML file, where KEGG node and edge attributes are maintained
`n`	optional character string, name of the desired node or edge. If is missing all node Data is returned

Details

Node and edge data is stored as list within environments in graphs to save memory and speed up graph manipulations. When using `getKEGGnodeData` or `getKEGGedgeData` is called, the list is extracted out of the environment and returned.

Value

Either a list or single item of `KEGGNode-class` or `KEGGEdge-class` object(s).

Note

These functions will be unified into ’KEGGnodeData’ and ’KEGGnodeData<-’ forms.

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

Examples

```
sfile <- system.file("extdata/hsa04010.xml",package="KEGGgraph")
gR <- parseKGML2Graph(sfile,expandGenes=TRUE)
getKEGGnodeData(gR,"hsa:4214")
getKEGGedgeData(gR,"hsa:4214~hsa:5605")
```

`getKGMLurl` *Get KGML file (url) with KEGG PATHWAY ID and (optional) organism*

Description

The function simply returns the KGML file url given KEGG PATHWAY ID. If the KEGG PATHWAY ID contains no organism prefix, user can specify the ’organism’ parameter. Otherwise the ’organism’ option is ignored.

retrieveKGML is a simple wrapper to getKGMLurl, which downloads the KGML file with `download.file` in utils package.

Usage

```
getKGMLurl(pathwayid, organism = "hsa")
retrieveKGML(pathwayid, organism, destfile,method="auto", ...)
kgmlNonmetabolicName2MetabolicName(destfile)
getCategoryIndepKGMLurl(pathwayid, organism="hsa", method="auto", ...)
```

Arguments

`pathwayid`	KEGG PATHWAY ID, e.g. ’hsa00020’
`organism`	three-alphabet organism code, if pathwayid contains the ocde this option is ignored
`destfile`	Destination file, to which the remote KGML file should be saved
`method`	Method to be used for downloading files, passed to `download.file` function. Currently supports "internal", "auto" and "lynx"
`...`	Parameters passed to download.file

Details

The function `getKGMLurl` takes the pathway identifier (can be in the form of 'hsa00020' or with 'pathway' prefix, for example 'path:hsa00020'), and returns the url to download KGML file.

The mapping between pathway identifier and pathway name can be found by KEGGPATHNAME2ID (or reversed mappings) in KEGG.db package. See vignette for example.

`retrieveKGML` calls `download.file` to download the KGML file from KEGG REST API remotely.

Before July 2017, KEGG FTP server was used to download the KGML files. Since then the REST API service of KEGG is used instead.

Value

KGML File URL of the given pathway.

Note

So far the function does not check the correctness of the 'organism' prefix, it is the responsibility of the user to garantee the right spelling.

There were a period when the metabolic and non-metabolic pathways were saved separately in different directories, and `KEGGgraph` was able to handle them. `kgmlNonmetabolicName2MetabolicName` is used to translate non-metabolic pathway KGML URL to that of metabolic pathway. `getCategoryIndepKGMLurl` determines the correct URL to download by attempting both possibilities. They were mainly called internally. Now since the KGML file is to be downloaded in each pathway's main page instead from the FTp server, these functions are no more needed and will be removed in the next release.

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

References

Plea from KEGG (available as of Aug 2011) `https://www.genome.jp/kegg/docs/plea.html`

Examples

```
getKGMLurl("hsa00020")
getKGMLurl("path:hsa00020")
getKGMLurl("00020",organism="hsa")
getKGMLurl(c("00460", "hsa:00461", "path:hsa00453", "path:00453"))

hasConnection <- RCurl::url.exists(getKGMLurl("cel00010"))
if(hasConnection) {
  tmp <- tempfile()
  retrieveKGML(pathwayid='00010', organism='cel', destfile=tmp, method="auto")
} else {
  warnings("No connection to KEGG webservice")
}
```

`getName-methods` *Get 'name' attribute*

Description

Get 'name' attribute for given object, this method can be used for almost all objects implemented in KEGGgraph package to extract their name slot. See manual pages of individual objects for examples.

Methods

object = "KEGGEdgeSubType" An object of `KEGGEdgeSubType-class`

object = "KEGGNode" An object of `KEGGNode-class`

object = "KEGGPathway" An object of `KEGGPathway-class`

object = "KEGGPathwayInfo" An object of `KEGGPathwayInfo-class`

object = "KEGGReaction" An object of `KEGGReaction-class`

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

References

KGML Document Manual `https://www.genome.jp/kegg/docs/xml/`

Examples

```
sfile <- system.file("extdata/hsa04010.xml",package="KEGGgraph")
pathway <- parseKGML(sfile)

## get pathway name
getName(pathway)

## get node name
nodes <- nodes(pathway)
getName(nodes[[2]])

## get edge name: it is not informative since the nodes are identified
## with file-dependent indices
edges <- edges(pathway)
getName(edges[[7]])

## get subtype name
subtype <- getSubtype(edges[[2]])[[1]]
getName(subtype)
```

`getNamedElement` *Extract the value in a vector by name*

Description

The function extracts the value(s) in a named vector by given name(s), in case no element is found with the given name, NA will be returned

Usage

```
getNamedElement(vector, name)
```

Arguments

`vector`	A named vector of any data type
`name`	Wanted name

Value

The elements with the given name, 'NA' in case no one was found

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

Examples

```
vec <- c(first="Hamburg", second="Hoffenheim",third="Bremen")
getNamedElement(vec, "third")
getNamedElement(vec, "last")
```

`getPathwayInfo-methods` *Get KEGG pathway info*

Description

KEGG stores additional information of the pathways in their KGML files, which can be extracted by this function.

The method returns the attributes of the pathway including its full title, short name, organism, image file link (which can be downloaded from KEGG website) and web link.

Methods

object = "KEGGPathway" An object of `KEGGPathway-class`

Examples

```
sfile <- system.file("extdata/hsa04010.xml",package="KEGGgraph")
pathway <- parseKGML(sfile)
getPathwayInfo(pathway)
```

`getReactions-methods` *Get KEGG reactions*

Description

In KGML, the pathway element specifies one graph object with the *entry* elements as its nodes and the *relation* and *reaction* elements as its edges. The *relation* elements are saved as *edges* in objects of `KEGGPathway-class`, and the *reactions* elements are saved as a slot of the object, which can be retrieved with the function `getReactions`.

Regulatory pathways are always viewed as protein networks, so there is no 'reaction' information saved in their KGML files. Metabolic pathways are viewed both as both protein networks and chemical networks, hence the `KEGGPathway-class` object may have reactions information.

Methods

object = "KEGGPathway" An object of `KEGGPathway-class`

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

References

KGML Document manual `https://www.genome.jp/kegg/docs/xml/`

See Also

`KEGGPathway-class`

Examples

```
mapfile <- system.file("extdata/map00260.xml", package="KEGGgraph")
maptest <- parseKGML(mapfile)
maptest

mapReactions <- getReactions(maptest)
mapReactions[1:3]
```

`getRgraphvizEdgeNames` *Get Rgraphviz compatitable edge names*

Description

Get Rgraphviz compatitable edge names, where the out- and in-nodes sharing a edge are concatenated by "~".

Usage

```
getRgraphvizEdgeNames(graph)
```

Arguments

`graph`	A graph object

Value

A list of names, the order is determined by the edge order.

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

References

Rgraphviz package

Examples

```
tnodes <- c("Hamburg","Dortmund","Bremen", "Paris")
tedges <- list("Hamburg"=c("Dortmund", "Bremen"),
"Dortmund"=c("Hamburg"), "Bremen"=c("Hamburg"), "Paris"=c())
tgraph <- new("graphNEL", nodes = tnodes, edgeL = tedges)
getRgraphvizEdgeNames(tgraph)
```

`getSubtype-methods` *Get subtype*

Description

KEGG stores sub-type of interactions between entities in the KGML files, which can be extracted with this method. The descriptions for the subtypes can be explored at the KGML document manual in the references.

See `KEGGEdge-class` for examples. The method for graphs is a wrapper to extract all subtype information from one graph.

Methods

object = "graph" A graph object of KEGGgraph. The method returns a list of subtypes in the same order of edges

object = "KEGGEdge" An object of KEGGEdge, which stores the subtype information

Author(s)

Jitao David Zhang mailto:jitao_david.zhang@roche.com

References

KGML Document manual https://www.genome.jp/kegg/docs/xml/

Examples

```
sfile <- system.file("extdata/hsa04010.xml",package="KEGGgraph")
pathway <- parseKGML(sfile)

edges <- edges(pathway)
subtype <- getSubtype(edges[[1]])
subtype
```

`getTitle-methods` *Get title for given element*

Description

The methods get title attribute for given KGML element, for example for objects of KEGGPathway-class or KEGGPathwayInfo-class

Methods

object = "KEGGPathway" An object of KEGGPathway-class

object = "KEGGPathwayInfo" An object of KEGGPathwayInfo-class

Author(s)

Jitao David Zhang mailto:jitao_david.zhang@roche.com

References

KGML Document manual https://www.genome.jp/kegg/docs/xml/

Examples

```
sfile <- system.file("extdata/hsa04010.xml",package="KEGGgraph")
pathway <- parseKGML(sfile)

getTitle(pathway)

pi <- getPathwayInfo(pathway)
getTitle(pi)
```

`getType-methods` *Get type attribute*

Description

This method can be used to extract generic type attribute from several objects implemented in KEGGgraph package.

The meanings and descriptions of the types can be found at KGML manual listed in the reference.

Methods

object = "KEGGEdge" An object of `KEGGEdge-class`

object = "KEGGNode" An object of `KEGGNode-class`

object = "KEGGReaction" An object of `KEGGReaction-class`

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

References

KGML Manual `https://www.genome.jp/kegg/docs/xml/`

Examples

```
mapfile <- system.file("extdata/map00260.xml", package="KEGGgraph")
maptest <- parseKGML(mapfile)

## node type
node <- nodes(maptest)[[3]]
getType(node)

## edge type
edge <- edges(maptest)[[5]]
getType(edge)

## reaction type
reaction <- getReactions(maptest)[[5]]
getType(reaction)
```

`getValue-methods` *Get 'value' attribute*

Description

Get 'value' attribute, mainly used internally and is not expected to be called by users.

Methods

object = "KEGGEdgeSubType" An object of `KEGGEdgeSubType-class`

`graphDensity` *Graph density*

Description

The graph density is defined as d = E/(V*(V-1)/2) where E is the number of edges and V of nodes.

Usage

```
graphDensity(graph)
```

Arguments

`graph`	A graph object

Details

The density of a graph lies between [0,1]

Value

A value between [0,1]

Author(s)

Jitao David Zhang `jitao_david.zhang@roche.com`

References

Aittokallio and Schwikowski (2006), Graph-based methods for analysing networks in cell biology, Briefings in Bioinformatics, 7, 243-255.

Examples

```
tnodes <- c("Hamburg","Dortmund","Bremen", "Paris")
tedges <- list("Hamburg"=c("Dortmund", "Bremen"),
"Dortmund"=c("Hamburg"), "Bremen"=c("Hamburg"), "Paris"=c())
tgraph <- new("graphNEL", nodes = tnodes, edgeL = tedges)
graphDensity(tgraph)
```

`isHomoList` *Determines whether a list is homogenous*

Description

If a list contains objects of the same class with the given class name, we call it a homogenous list and the function returns `TRUE`, otherwise it returns `FALSE`.

Usage

```
isHomoList(list, class)
```

Arguments

`list`	A list
`class`	The class name to be validated

Value

logical

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

Examples

```
testlist <- list("home1"="Hamburg","home2"="Heidelberg",
"home3"="Tianjin")
isHomoList(testlist,"character")
testlist$lucky <- 16
isHomoList(testlist,"character")
```

`KEGGEdge-class` *Class 'KEGGEdge'*

Description

A class to represent 'relation' elements in KGML files and edge objects in a KEGG graph

Objects from the Class

Objects are normally created by `parseRelation` function, which is not intended to be called by user directly

Slots

`entry1ID`: The entryID of the first KEGGNode

`entry2ID`: The entryID of the second KEGGNode

`type`: The type of the relation, see `getType-methods`

`subtype`: The subtype(s) of the edge, a list of `KEGGEdgeSubType`

Methods

getEntryID `signature(obj = "KEGGEdge")`: Get entryIDs of the edge in the order specified by the direction of the edge

getType `signature(object = "KEGGEdge")`: Get the relation type

getName `signature(object = "KEGGEdge")`: Get the names of edges in the convention of Rgraphviz, 'node1~node2'

show `signature(object = "KEGGEdge")`: Show method

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

References

KGML Manual `https://www.genome.jp/kegg/docs/xml/`

See Also

`KEGGNode-class`

Examples

```
mapfile<- system.file("extdata/map00260.xml", package="KEGGgraph")
maptest <- parseKGML(mapfile)

x <- edges(maptest)[[1]]
class(x)

## examples to extract information from KEGGEdge
getName(x)
getEntryID(x)

getType(x)
getSubtype(x)

subtype <- getSubtype(x)[[1]]
getName(subtype)
```

`KEGGEdgeSubtype` *KEGG edge subtypes*

Description

Edge subtypes defined by the KEGG database.

Format

A `data.frame` with 17 rows and 11 columns

`KEGGEdgeSubType-class` *Class "KEGGEdgeSubType"*

Description

A class to represent subtype in KEGG

Objects from the Class

Objects can be created by calls of the form `new("KEGGEdgeSubType", ...)`.

Slots

`name`: Object of class `"character"`, name of the subtype

`value`: Object of class `"character"`, value of the subtype

Methods

- **getName** `signature(object = "KEGGEdgeSubType")`: getting subtype name
- **getValue** `signature(object = "KEGGEdgeSubType")`: getting subtype value
- **show** `signature(object = "KEGGEdgeSubType")`: show method

Note

Please note that 'KEGGEdgeSubtype' is a data frame storing subtype predefinitions, the 'type' with lowercases. 'KEGGEdgeSubType' is however a class representing these subtypes.

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

See Also

`KEGGEdge-class`

Examples

```
showClass("KEGGEdgeSubType")
## use example(KEGGEdge-class) for more examples
```

`KEGGEdgeType` *KEGG edge types*

Description

Edge types defined by the KEGG database.

Format

A `data.frame` with values and explanations of edge types.

`KEGGGraphics-class` *class 'KEGGGraphics'*

Description

A class to represent 'graphics' element in KGML files

Objects from the Class

This method is mainly used to extract visualization information from KGML files.

Objects can be created by calling `parseGraphics`

Slots

`name`**:** Object of class `"character"` graphics name

`x`**:** Object of class `"integer"` x coordinate in KEGG figure

`y`**:** Object of class `"integer"` y coordinate in KEGG figure

`type`**:** Object of class `"character"` graphics type (shape)

`width`**:** Object of class `"integer"` witdh of the symbol

`height`**:** Object of class `"integer"` height of the symbol

`fgcolor`**:** Object of class `"character"` foreground color

`bgcolor`**:** Object of class `"character"` background color

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

References

KGML Manual `https://www.genome.jp/kegg/docs/xml/`

See Also

`parseGraphics`

Examples

```
showClass("KEGGGraphics")
```

`KEGGGroup-class` *Class "KEGGGroup"*

Description

Class to represent 'group' nodes in KEGG pathways

Objects from the Class

The objects are usually created by `parseEntry` function and is not intended to be called directly by users.

Slots

`component`**:** Component of the group

`entryID`**:** see the slot of `KEGGNode-class`

`graphics`**:** see the slot of `KEGGNode-class`

`link`**:** see the slot of `KEGGNode-class`

`map`**:** see the slot of `KEGGNode-class`

`name`**:** see the slot of `KEGGNode-class`

`reaction`**:** see the slot of `KEGGNode-class`

`type`**:** see the slot of `KEGGNode-class`

Extends

Class `"KEGGNode"`, directly.

Methods

getComponent `signature(object = "KEGGNode")`: returns components of the group, in a vector of strings

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

See Also

`KEGGNode-class`

Examples

```
showClass("KEGGGroup")
```

`KEGGNode-class` *Class "KEGGNode"*

Description

The class to present 'entry' element in KGML files and nodes in KEGG graphs

Objects from the Class

Objects can be created by calls of the function `parseEntry` and is not intended to be directly created by users.

Slots

- `entryID`**:** entryID, the 'id' attribute of 'entry' elements in KGML files. In each KGML file the entryID is specified by auto-increment integers, therefore entryIDs from two individual KGML files are not unique. However, if 'expandGenes' option is specified in `KEGGpathway2Graph` function, the unique KEGGID will replace the default integer as the new entryID, which is unique in biological context
- `name`**:** Name of the node
- `type`**:** Type of the node, use `data(KEGGNodeType)` to see available values
- `link`**:** URL link of the node
- `reaction`**:** Reaction of the node
- `map`**:** Map of the node
- `graphics`**:** Graphic details (including display name) of the node, an object of `KEGGGraphics`

Methods

- **getDisplayName** `signature(object = "KEGGNode")`: get display name
- **getEntryID** `signature(obj = "KEGGNode")`: get entryID, in case of gene-expanded graphs this is the same as getKEGGID
- **getKEGGID** `signature(object = "KEGGNode")`: get KEGGID
- **getType** `signature(object = "KEGGNode")`: get the type of the node
- **<-name** `signature(object = "KEGGNode")`: replace name
- **getComponent** `signature(obj = "KEGGNode")`: returns entryID (the same as `getEntryID`), for compatibility with `KEGGGroup-class`
- **show** `signature(object = "KEGGNode")`: show method

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

References

KGML Document manual `https://www.genome.jp/kegg/docs/xml/`

See Also

`KEGGEdge-class`, `parseEntry`

Examples

```
## We show how to extract information from KEGGNode object
sfile <- system.file("extdata/hsa04010.xml",package="KEGGgraph")
pathway <- parseKGML(sfile)

ns <- nodes(pathway)
node <- ns[[1]]

show(node)
getName(node)
getDisplayName(node)
getEntryID(node)
getKEGGID(node)
```

`KEGGNodeType` *KEGG node types*

Description

Node types defined by the KEGG database.

Format

A `data.frame` with values and explanations of KEGG node types.

`KEGGPATHID2NAME` *KEGG pathway identifier to name*

Description

The data provides a translation mechanism between KEGG pathway identifiers, for instance `hsa04010`, and pathway names, for instance `MAPK signaling pathway`.

Format

An `AnnDbBiMap`

`KEGGPathway-class` *Class "KEGGPathway"*

Description

A class to represent KEGG pathway

Objects from the Class

Objects can be created by calls of the form `new("KEGGPathway", ...)`. Normally they are created by `parseKGML`.

Slots

`pathwayInfo`**:** An object of `KEGGPathwayInfo-class`

`nodes`**:** List of objects of `KEGGNode-class`

`edges`**:** List of objects of `KEGGEdge-class`

`reactions`**:** List of objects of `KEGGReaction-class`

Methods

edges `signature(object = "KEGGPathway", which = "ANY")`: KEGGEdges of the pathway

edges<- `signature(object = "KEGGPathway")`: setting edges

getName `signature(object = "KEGGPathway")`: getting pathway name

getTitle `signature(object = "KEGGPathway")`: getting pathway title

nodes<- `signature(object = "KEGGPathway", value = "ANY")`: setting nodes

nodes `signature(object = "KEGGPathway")`: KEGGNodes of the pathway

getPathwayInfo `signature(object = "KEGGPathway")`: getting KEGGPathwayInfo

getTitle `signature(object = "KEGGPathway")`: getting title of the pathway

show `signature(object = "KEGGPathway")`: display method

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

References

KGML Document manual `https://www.genome.jp/kegg/docs/xml/`

See Also

`parseKGML`, `KEGGEdge-class`, `KEGGNode-class`, `KEGGReaction-class`

Examples

```
## We show how to extract information from KEGGPathway objects
## Parse KGML file into a 'KEGGPathway' object
mapfile <- system.file("extdata/map00260.xml", package="KEGGgraph")
maptest <- parseKGML(mapfile)

## short summary of the pathway
maptest

## get information of the pathway
getPathwayInfo(maptest)

## nodes of the pathway
nodes <- nodes(maptest)
node <- nodes[[3]]
getName(node)
getType(node)
getDisplayName(node)

## edges of the pathway
edges <- edges(maptest)
edge <- edges[[3]]
getEntryID(edge)
getSubtype(edge)
```

`KEGGpathway2Graph` *Parses KEGGpathway to graph*

Description

The function parses an object of `KEGGPathway-class` into graph.

Usage

```
KEGGpathway2Graph(pathway, genesOnly = TRUE, expandGenes = TRUE)
```

Arguments

`pathway`	An instance of `KEGGPathway-class`
`genesOnly`	logical, should only the genes are maintained and other types of nodes (compounds, etc) neglected? TRUE by default
`expandGenes`	logical, should homologue proteins expanded? TRUE by default

Details

When 'expandGenes=TRUE', the nodes have unique names of KEGGID (in the form of 'org:xxxx', for example 'hsa:1432'), otherwise an auto-increment index given by KEGG is used as node names.

In the latter case, the node names are duplicated and graphs cannot be simply merged before the nodes are unique.

KEGG node and edge data is stored in 'nodeData' and 'edgeData' slots respectively, which can be extracted by `getKEGGnodeData` and `getKEGGedgeData`.

Value

A directed graph.

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

See Also

`parseKGML2Graph`

Examples

```
sfile <- system.file("extdata/hsa04010.xml",package="KEGGgraph")
kegg.pathway <- parseKGML(sfile)
gR.compact<- KEGGpathway2Graph(kegg.pathway,expandGenes=FALSE)
```

`KEGGpathway2reactionGraph` *Convert chemical reaction network of KEGG pathway into graph*

Description

Regulatory pathways are always viewed as protein networks, so there is no 'reaction' information saved in their KGML files. Metabolic pathways are viewed both as both protein networks and chemical networks, hence the `KEGGPathway-class` object may have reactions information among chemical compounds.

This functions extracts reaction information from KEGG pathway, and convert the chemical compound reaction network into directed graph.

Usage

```
KEGGpathway2reactionGraph(pathway)
```

Arguments

`pathway`	A `KEGGPathway-class` object, usually as the result of the function `parseKGML`

Details

The direction of the graph is specified by the role of the compound in the reaction, the edges goes always out of ’substrate’ and points to ’product’.

For now there is no wrapper to parse the KGML file directly into a reaction graph. In future there maybe one, but we don’t want to confuse users with two similar functions to parse the file into a graph (since we assume that most users will need the protein graph, which can be conveniently parsed by `parseKGML2Graph`).

From version 1.18.0, reaction graphs returned by `KEGGpathway2reactionGraph` can be merged with other reaction graphs or pathway graphs.Thus users can combine pathway and reaction graph in one KGML file into a single graph.

Value

A directed graph with compounds as nodes and reactions as edges.

If the pathway does not contain any chemical reactions, a warning message will be printed and `NULL` is returned.

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

References

KGML Document manual `https://www.genome.jp/kegg/docs/xml/`

See Also

`KEGGPathway-class`

Examples

```
mapfile <-  system.file("extdata/map00260.xml",package="KEGGgraph")
map <- parseKGML(mapfile)
cg <- KEGGpathway2reactionGraph(map)
cg
nodes(cg)[1:3]
edges(cg)[1:3]
```

`KEGGPathwayInfo-class` *Class "KEGGPathwayInfo"*

Description

A class to represent information of a KEGG pathway

Objects from the Class

Objects can be created by calls of the function `parsePathwayInfo`.

Slots

`name`: Object of class `"character"` Pathway name

`org`: Object of class `"character"` Organism

`number`: Object of class `"character"` Number

`title`: Object of class `"character"` Title of the pathway

`image`: Object of class `"character"` Image URL

`link`: Object of class `"character"` URL Link

Methods

getTitle `signature(object = "KEGGPathwayInfo")`: get title of the pathway

show `signature(object = "KEGGPathwayInfo")`: show method

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

References

KGML Document Manual `https://www.genome.jp/kegg/docs/xml/`

Examples

```
sfile <- system.file("extdata/hsa04010.xml",package="KEGGgraph")
pathway <- parseKGML(sfile)
pi <- getPathwayInfo(pathway)

class(pi)

getTitle(pi)
```

`KEGGReaction-class` *Class "KEGGReaction"*

Description

A class to present 'reaction' elements in KGML files

Objects from the Class

Objects can be created by calls of the function `parseReaction`.

Slots

`name`**:** Object of class `"character"` the KEGGID of this reaction, e.g. "rn:R02749"

`type`**:** Object of class `"character"` the type of this reaction, either 'reversible' or 'irreversible'

`substrateName`**:** Object of class `"character"`, KEGG identifier of the COMPOUND database or the GLYCAN database e.g. "cpd:C05378"

`substrateAltName`**:** Object of class `"character"` alternative name of its parent substrate element

`productName`**:** Object of class `"character"` specifies the KEGGID of the product

`productAltName`**:** Object of class `"character"` alternative name of its parent product element

Methods

show `signature(object = "KEGGReaction")`: show method

getName `signature(object = "KEGGReaction")`: get the KEGGID of the reaction

getType `signature(object = "KEGGReaction")`: get the type of the reaction

getSubstrate `signature(object = "KEGGReaction")`: get the name of substrate

getProduct `signature(object = "KEGGReaction")`: get the name of product

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

References

KGML Document Manual `https://www.genome.jp/kegg/docs/xml/`

Examples

```
## We show how to extract reactions from a 'KEGGPathway' object
mapfile <- system.file("extdata/map00260.xml", package="KEGGgraph")

maptest <- parseKGML(mapfile)
mapReactions <- getReactions(maptest)

## More details about reaction
reaction <- mapReactions[[1]]
getName(reaction)
getType(reaction)
getSubstrate(reaction)
getProduct(reaction)
```

`kgmlFileName2PathwayName` *Convert KGML file name to pathway name*

Description

The function uses KEGG package and converts KGML file name into human readable pathway name.

Usage

```
kgmlFileName2PathwayName(filename)
```

Arguments

`filename`	A KGML file name

Details

So far it only supports KGML files organized by species.

NOTE: there is issue of package loading sequence to use this function: the 'KEGG.db' must be loaded before 'KEGGgraph' to use it properly. Otherwise the `mget` returns error of 'KEGGPATHID2NAME' is not a environment. So far I don't where does this bug come from, so I commented out the examples.

Value

A character string of pathway name

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

`mergeGraphs` *A function to merge KEGG graphs*

Description

The function merges a list of KEGG graphs into one graph object. The merged graph have unique nodes, and edges are merged into non-duplicate sets.

For the reason of speed, `mergeGraphs` discards KEGG node and edge informations. To maintain them while merging graphs, please use `mergeKEGGgraphs`.

Usage

```
mergeGraphs(list, edgemode = "directed")
```

Arguments

`list`	A list of graph objects, which can be created by `parseKGML2Graph`
`edgemode`	Edge mode of the graph product, by default 'directed'

Details

The function takes a list of graphs and merges them into a new graph. The nodes of individual graphs must be unique. The function takes care of the removal of duplicated edges.

Value

A directed graph

Note

It is known that graphs from C.elegance pathways have problem when merging, because the nodes name are not consistent betweeen edge records and entry IDs.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

See Also

`parseKGML2Graph`

`mergeKEGGgraphs` *Merge KEGG graphs, also merging KEGGNode and KEGGEdge attributes*

Description

`mergeKEGGgraphs` extends function `mergeGraphs` and merges a list of KEGG graphs. Both `mergeGraphs` and `mergeKEGGgraphs` can be used to merge graphs, while the latter form is able to merge the nodes and edges attributes from KEGG, so that the nodes and edges have a one-to-one mapping to the results from `getKEGGnodeData` and `getKEGGEdgeData`.

See details below.

Usage

```
mergeKEGGgraphs(list, edgemode = "directed")
```

Arguments

`list`	A list of named KEGG graphs
`edgemode`	character, 'directed' by default

Details

`mergeGraphs` discards the node or edge attributes, hence `getKEGGnodeData` or `getKEGGedgeData` will return `NULL` on the resulting graph.

`mergeKEGGgraphs` calls `mergeGraphs` first to merge the graphs, then it also merges the KEGGnodeData and KEGGedgeData.so that they are one-to-one mapped to the nodes and edges in the merged graph.

Value

A graph with nodeData and edgeData

Note

From version 1.21.1, lists containing NULL should also work.

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

See Also

`mergeGraphs`

Examples

```
sfile <- system.file("extdata/hsa04010.xml",package="KEGGgraph")
gR <- parseKGML2Graph(sfile,expandGenes=TRUE)

wntfile <- system.file("extdata/hsa04310.xml",package="KEGGgraph")
wntR <- parseKGML2Graph(wntfile, expandGenes=TRUE)

graphlist <- list(mapkG=gR, wntG=wntR)
mergedKEGG <- mergeKEGGgraphs(graphlist)

mergedKEGG

## list containing NULL works also
nlist <- list(gR, wntR, NULL)
nmergedKEGG <- mergeKEGGgraphs(nlist)
```

`neighborhood` *Return the neighborhood set of given vertices*

Description

The function returns the neighborhood set of given vertices in the form of list. Optionally user can choose to include the given vertices in the list, too.

Usage

```
neighborhood(graph, index, return.self = FALSE)
```

Arguments

`graph`	An object of `graphNEL`
`index`	Names of nodes, whose neighborhood set should be returned
`return.self`	logical, should the vertex itself also be returned?

Details

Let v be a vertex in a (di)graph, the out-neighborhood or successor set (N+(v), x belongs to V(G) and v->x) and the in-neighborhood or predecessor set (N-(v), x belongs to V(G) and x->v) are jointly returned.

The returned list is indexed by the given node indices, `NULL` is returned in case of non-existing node.

The nodes are unique, that is, duplicated nodes are removed in results.

Value

A list indexed by the given node indices, each entry containing the neighborhood set of that node (or furthermore including that node).

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

References

D.B. West. Introduction to Graph Theory, Second Edition. Prentice Hall, 2001

Examples

```
V <- c("Hamburg","Stuttgart","Berlin","Paris","Bremen")
E <- list("Hamburg"=c("Berlin","Bremen"),
          "Stuttgart"=c("Berlin","Paris"),
          "Berlin"=c("Stuttgart","Bremen"),
          "Paris"=c("Stuttgart"),
          "Bremen"=c("Hamburg","Berlin"))
g <- new("graphNEL", nodes=V, edgeL=E, edgemode="directed")
if(require(Rgraphviz) & interactive()) {
  plot(g, "neato")
}

## simple uses
neighborhood(g, "Hamburg")
neighborhood(g, c("Hamburg", "Berlin","Paris"))

## in case of non-existing nodes
neighborhood(g, c("Stuttgart","Ulm"))
```

```
## also applicable to non-directed graphs
neighborhood(ugraph(g), c("Stuttgart","Berlin"))
```

`parseEntry` *Parse ENTRY elements of KGML document*

Description

ENTRY elements contain information of nodes (proteins, enzymes, compounds, maps, etc) in KEGG pathways. 'parseEntry' function parses the elements into `link{KEGGNode-class}` or `KEGGGroup-class` objects. It is not expected to be called directly by the user.

Usage

```
parseEntry(entry)
```

Arguments

`entry`	XML node of KGML file

Details

See `https://www.genome.jp/kegg/docs/xml/` for more details about 'entry' as well as other elements in KGML files.

Value

An object of `link{KEGGNode}` or (in case of a group node) `link{KEGGGroup}`

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

References

`https://www.genome.jp/kegg/docs/xml/`

See Also

`parseGraphics`, `parseKGML`, `KEGGNode-class`, `KEGGGroup-class`

`parseGraphics` *Parse 'graphics' elements in KGML files*

Description

The function parses 'graphics' elements in KGML files, and it is mainly used internally.

Usage

```
parseGraphics(graphics)
```

Arguments

`graphics`	XML node

Details

The function is called by other parsing functions and not intended to be called directly by user.

Value

An object of `KEGGGraphics-class`.

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

References

KGML Document manual `https://www.genome.jp/kegg/docs/xml/`

See Also

`KEGGGraphics-class`

`parseKGML` *KGML file parser*

Description

The function parses KGML files according to the KGML XML documentation.

Usage

```
parseKGML(file)
```

Arguments

`file`	Name of KGML file

Details

The function parses KGML file (depending on XML package).

Value

An object of `KEGGPathway-class`.

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

References

KGML Manual `https://www.genome.jp/kegg/docs/xml/`

See Also

`parseEntry`, `parseRelation`, `parseReaction`, `KEGGPathway-class`,

Examples

```
sfile <- system.file("extdata/hsa04010.xml",package="KEGGgraph")
kegg.pathway <- parseKGML(sfile)
kegg.pathway
```

`parseKGML2DataFrame` *Parse KGML file into a data frame*

Description

This function extends the `parseKGML2Graph` function, by converting the resulting graph into a three-column data frame representing out-nodes (the `from` column in the data frame), in-nodes (`to`), types and subtypes of edges that connect them (`type` and `subtype`, respectively). It can be used, for example, for exporting KEGG pathway networks in plain text files.

Usage

```
parseKGML2DataFrame(file, reactions=FALSE,...)
```

Arguments

`file`	A KGML file
`reactions`	Logical, whether metabolic reactions should be parsed and returned as part of the data frame. Default:`FALSE`
`...`	Other parameters passed to `KEGGpathway2Graph`

Details

The out- and in-nodes are represented in the form of KEGG identifiers. For human EntrezIDs the function `translateKEGGID2GeneID` can be used.

Multile edges are supported: in case more than one subtypes of edges exist between two nodes, they are all listed in the resulting data frame.

Value

A four-column data frame, representing the graph structure: out-nodes (the `from` column), in-nodes (`to`), edge type (`type`) and subtype (`subtype`).

Author(s)

Jitao David Zhang

See Also

`parseKGML2Graph`, `KEGGpathway2Graph` and `translateKEGGID2GeneID`.

Examples

```
sfile <- system.file("extdata/hsa04010.xml",package="KEGGgraph")
gdf <- parseKGML2DataFrame(sfile)
head(gdf)
dim(gdf)

rfile <- system.file("extdata/hsa00020.xml",package="KEGGgraph")
dim(dfWr <- parseKGML2DataFrame(rfile, reactions=TRUE))
dim(dfWOr <- parseKGML2DataFrame(rfile, reactions=FALSE))
stopifnot(nrow(dfWr)>nrow(dfWOr))

## not expanding genes: only the KGML-specific identifiers are used then
## only for expert use
## NOT RUN
gdf.ne <- parseKGML2DataFrame(sfile, expandGenes=FALSE)
dim(gdf.ne)
head(gdf.ne)
## NOT RUN
```

`parseKGML2Graph` *Parse KGML files into KEGG graph*

Description

This function is a wrapper for parseKGML and KEGGpathway2Graph. It takes two actions: first it reads in the KGML file and parses it into an object of `KEGGPathway-class`, the second step it calls `KEGGpathway2Graph` function to return the graph model.

Usage

```
parseKGML2Graph(file, ...)
```

Arguments

`file`	Name of KGML file
`...`	other parameters passed to KEGGpathway2Graph, see `KEGGpathway2Graph`

Details

Note that groups of genes will be split into single genes by calling the `KEGGpathway2Graph` function. Edges that connected to groups will be duplicated to connect each member of the group.

Value

A graph object.

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

Examples

```
sfile <- system.file("extdata/hsa04010.xml",package="KEGGgraph")
gR <- parseKGML2Graph(sfile,expandGenes=TRUE)
gR
```

`parseKGMLexpandMaps` *A convenient function to parse KGML and expand its containing maps into one graph object*

Description

The function does several tasks implemented in the KEGGgraph package in sequence to make expanding maps easier.

Usage

```
parseKGMLexpandMaps(file, downloadmethod = "auto", genesOnly = TRUE, localdir,...)
```

Arguments

`file`	A KGML file
`downloadmethod`	passed to `download.file` function as 'method', see its documentation for more details
`genesOnly`	logical, should only the genes nodes remain in the returned graph object?
`localdir`	character string, if specified, the function tries to read files with the same base name from a local directory, useful when there are file copies on the client.
`...`	Other parameters passed to `download.file`

Details

In KEGG pathways there're usually pathways contained('cross-linked') in other pathways, for example see https://www.genome.jp/kegg/pathway/hsa/hsa04115.html, where p53 signalling pathway contains other two pathways 'apoptosis' and 'cell cycle'. This function parses these pathways (refered as 'maps' in KGML manual), download their KGML files from KEGG REST API, parse them individually, and merge all the children pathway graphs with the parental pathway into one graph object. The graph is returned as the function value.

Since different graphs does not have unique node identifiers unless the genes are expanded, so by using this function user has to expand the genes. Another disadvantage is that so far due to the implementation, the KEGGnodeData and KEGGedgeData is lost during the merging. This however will probably be changed in the future version.

Value

A directed graph object

Author(s)

Jitao David Zhang `jitao_david.zhang@roche.com`

References

KGML Document manual `https://www.genome.jp/kegg/docs/xml/`

See Also

for most users it is enough to use `parseKGML2Graph`

`parsePathwayInfo` *Parse information of the pathway from KGML files*

Description

The function parses the information of the given pathway from KGML files into an object of `KEGGPathwayInfo-class`. It is used internally and is not expected to be called by users directly.

Usage

```
parsePathwayInfo(root)
```

Arguments

`root`	Root element of the KGML file

Value

An object of `KEGGPathwayInfo-class`

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

References

KGML Document Manual `https://www.genome.jp/kegg/docs/xml/`

`parseReaction` *Parse reaction from KGML files*

Description

The function parses 'reaction' element in KGML files. It is used interally and not expected to be called by users.

Usage

```
parseReaction(reaction)
```

Arguments

`reaction`	A node of the type 'reaction' in KGML files

Details

See the reference manual for more information about 'reaction' type

Value

An object of `KEGGReaction-class`

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

References

KGML Document Manual `https://www.genome.jp/kegg/docs/xml/`

`parseRelation` *Parse RELATION elements from KGML files*

Description

RELATION elements in KGML files record the binary relationships between ENTRY elements, corresponding to (directed) edges in a graph. 'parseRelation' function parses RELATION elements into `KEGGEdge-class` objects from KGML files. It is not expected to be called directly by the user.

Usage

```
parseRelation(relation)
```

Arguments

`relation`	XML node of KGML file

Details

See `https://www.genome.jp/kegg/docs/xml/` for more details about 'relation' as well as other elements in KGML files.

Value

An object of `link{KEGGEdge}`.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

References

`https://www.genome.jp/kegg/docs/xml/`

See Also

`KEGGEdge-class`, `parseEntry`

`parseSubType` *Parse KGML relation subtype*

Description

The function parses KGML relation subtype, called internally and not intended to be used by end users.

Usage

```
parseSubType(subtype)
```

Arguments

`subtype`	KGML subtype node

Value

An object of `KEGGEdgeSubType-class`

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

`plotKEGGgraph` *Plot KEGG graph with Rgraphviz*

Description

The function provides a simple interface to Rgraphviz to render KEGG graph with custom styles.

`KEGGgraphLegend` gives the legend of KEGG graphs

Usage

```
plotKEGGgraph(graph, y = "neato", shortLabel = TRUE,
useDisplayName=TRUE, nodeRenderInfos,  ...)
KEGGgraphLegend()
```

Arguments

`graph`	A KEGG graph, by calling `parseKGML2Graph`
`y`	the layout method, `neato` by default
`shortLabel`	logical, should be short label used instead of full node name?
`useDisplayName`	logical, should the labels of nodes rendered as the 'display name' specified in the KGML file or render them simply with the node names?
`nodeRenderInfos`	List of node rendering info
`...`	Other functions passed to renderGraph, not implemented for now

Details

Users are not restricted to this function, alternatively you can choose other rendering functions.

Value

The graph after layout and rendering is returned.

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

Examples

```
opar <- par(ask=TRUE)
sfile <- system.file("extdata/hsa04010.xml",package="KEGGgraph")
gR <- parseKGML2Graph(sfile,expandGenes=TRUE)
subs <-  c("hsa:1432",edges(gR)$`hsa:1432`,
    "hsa:5778","hsa:5801","hsa:84867",
    "hsa:11072","hsa:5606","hsa:5608",
    "hsa:5494","hsa:5609")
gR.sub <- subGraph(subs, gR)
if(require(Rgraphviz))
  plotKEGGgraph(gR.sub)

KEGGgraphLegend()
par(opar)
```

`pvalue2asterisk` *Return common significance sign (asterisk) associated with given p value*

Description

A p-value of 0.05, 0.01, 0.001 correspond to one, two or three asterisks. If 'sig.1' is set to `TRUE`, then the p-value of 0.1 returns '.'.

Usage

```
pvalue2asterisk(pvalues, sig.1 = FALSE)
```

Arguments

`pvalues`	A numeric value
`sig.1`	logical, whether the significance sign of 0.1 should be returned

Value

A character string containing the signs

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

Examples

```
pvalue2asterisk(0.03)
pvalue2asterisk(0.007)
pvalue2asterisk(3e-5)
pvalue2asterisk(0.55)
```

`queryKEGGsubgraph` *Query the subgraph of a given KEGG graph with Entrez GeneID (s)*

Description

Given a list of genes (identified by Entrez GeneID), the function subsets the given KEGG gragh of the genes as nodes (and maintaining all the edges between).

Usage

```
queryKEGGsubgraph(geneids, graph, organism = "hsa", addmissing = FALSE)
```

Arguments

`geneids`	A vector of Entrez GeneIDs
`graph`	A KEGG graph
`organism`	a three-alphabet code of organism
`addmissing`	logical, in case the given gene is not found in the graph, should it be added as single node to the subgraph?

Details

This function solves the questions like 'How is the list of gene interact with each other in the context of pathways?'

Limited by the `translateKEGGID2GeneID`, this function supports only human for now. We are working to include other organisms.

If 'addmissing' is set to `TRUE`, the missing gene in the given list will be added to the returned subgraph as single nodes.

Value

A subgraph with nodes representing genes and edges representing interactions.

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

See Also

`translateGeneID2KEGGID`

Examples

```
sfile <- system.file("extdata/hsa04010.xml",package="KEGGgraph")
gR <- parseKGML2Graph(sfile,expandGenes=TRUE)
geneids <- c(5594, 5595, 6197, 5603, 1843,5530, 5603)
sub <- queryKEGGsubgraph(geneids, gR)
if(require(Rgraphviz) && interactive()) {
  plot(sub, "neato")
}

## add missing nodes
list2 <- c(geneids, 81029)
sub2 <- queryKEGGsubgraph(list2, gR,addmissing=TRUE)
if(require(Rgraphviz) && interactive()) {
  plot(sub2, "neato")
}
```

`randomSubGraph` *Randomly subset the given graph*

Description

The function is intended to be a test tool. It subset the given graph repeatedly.

Usage

```
randomSubGraph(graph, per = 0.25, N = 10)
```

Arguments

`graph`	A graph object
`per`	numeric, the percentage of the nodes to be sampled, value between (0,1)
`N`	Repeat times

Value

The function is called for its side effect, `NULL` is returned

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

Examples

```
tnodes <- c("Hamburg","Dortmund","Bremen", "Paris")
tedges <- list("Hamburg"=c("Dortmund", "Bremen"),
"Dortmund"=c("Hamburg"), "Bremen"=c("Hamburg"), "Paris"=c())
tgraph <- new("graphNEL", nodes = tnodes, edgeL = tedges)
randomSubGraph(tgraph, 0.5, 10)
```

`splitKEGGgroup` *Split KEGG group*

Description

The function split 'group' entries in KGML files. Most of the cases they are complexes. During the splitting the function copies the edges between groups and nodes (or between groups and groups) correspondingly, so that the existing edges also exist after the groups are split.

Usage

```
splitKEGGgroup(pathway)
```

Arguments

`pathway`	An object of `KEGGPathway-class`

Details

By default the groups (complexes) in KEGG pathways are split.

Value

An object of `KEGGPathway-class`

Author(s)

Jitao David Zhang mailto:jitao_david.zhang@roche.com

References

KGML Manual https://www.genome.jp/kegg/docs/xml/

See Also

KEGGpathway2Graph

Examples

```
sfile <- system.file("extdata/hsa04010.xml",package="KEGGgraph")
kegg.pathway <- parseKGML(sfile)
kegg.pathway.split <- splitKEGGgroup(kegg.pathway)

## compare the different number of edges
length(edges(kegg.pathway))
length(edges(kegg.pathway.split))
```

`subGraphByNodeType` *Subset KEGG graph by node types*

Description

The function subsets KEGG graph by node types, mostly used in extracting gene networks.

Usage

```
subGraphByNodeType(graph, type = "gene", kegg=TRUE)
```

Arguments

`graph`	A KEGG graph object produced by calling parseKGML2Graph
`type`	node type, see KEGGNodeType for details
`kegg`	logical, should the KEGG Node and Edge attributes be maintained during the subsetting? By default set to 'TRUE'

Value

A subgraph of the original graph

Author(s)

Jitao David Zhang mailto:jitao_david.zhang@roche.com

Examples

```
sfile <- system.file("extdata/hsa04010.xml",package="KEGGgraph")
sGraph <- parseKGML2Graph(sfile,expandGenes=TRUE, genesOnly=FALSE)
sGraphGene <- subGraphByNodeType(sGraph, type="gene")
```

`subKEGGgraph` *Subset KEGG graph, including subsetting node and edge attributes*

Description

`subKEGGgraph` extends generic method `subGraph` and subsets the KEGG graph. Both 'subKEGGgraph' and 'subGraph' can be used to subset the graph, the difference lies in whether the node and edge attributes from KEGG are also subset (`subKEGGgraph`) or not (`subGraph`).

See details below.

Usage

```
subKEGGgraph(nodes, graph)
```

Arguments

`nodes`	Node names to subset
`graph`	A graph parsed from KGML files, produced by `parseKGML2Graph`, `KEGGpathway2Graph` or `parseKGMLexpandMaps`

Details

`subGraph` does not subset the node or edge attributes, hence the results of `getKEGGnodeData` and `getKEGGedgeData` does not map to the nodes and edges in the subgraph in a one-to-one manner, with attributes of removed nodes and edges still remaining in the subGraph.

`subKEGGgraph` calls `subGraph` first to subset the graph, and then it also subsets the KEGGnodeData and KEGGedgeData so that they are one-to-one mapped to the nodes and edges in the subgraph.

Value

A graph with nodeData and edgeData.

Author(s)

Jitao David Zhang `mailto:jitao_david.zhang@roche.com`

Examples

```
sfile <- system.file("extdata/hsa04010.xml",package="KEGGgraph")
gR <- parseKGML2Graph(sfile,expandGenes=TRUE)
subs <-  c("hsa:1432",edges(gR)$`hsa:1432`,"hsa:5778","hsa:5801",
"hsa:84867","hsa:11072","hsa:5606","hsa:5608","hsa:5494","hsa:5609")
gR.keggsub <- subKEGGgraph(subs, gR)
gR
gR.keggsub
```

`subtypeDisplay-methods` *Get display information for relation subtypes*

Description

To render KEGG pathway graphs, we have created a custom style of edges to represent their subtypes. 'subtypeDisplay' extracts this information

Methods

object = "graph" An KEGG graph

object = "KEGGEdge" An object of `KEGGEdge-class`

object = "KEGGEdgeSubType" An object of `KEGGEdgeSubType-class`

`translateKEGGgraph` *Tranlate the KEGG graph from being indexed by KEGGID to another identifer*

Description

The function translates the KEGG graph into a graph of equivalant topology while index with unique identifiers given by user. The new identifiers could be, for example, GeneSymbol or other identifiers mapped to KEGGID.

Usage

```
translateKEGGgraph(graph, newNodes)
```

Arguments

`graph`	A KEGG graph
`newNodes`	A character vector giving the new nodes, must be of the same length and same order of the nodes of the given graph

Details

The function is still experimental and users are welcomed to report any difficulties

Value

Another graph indexed by the given identifier

Author(s)

Jitao David Zhang <jitao_david.zhang@roche.com>

Examples

```
sfile <- system.file("extdata/hsa04010.xml",package="KEGGgraph")
gR <- parseKGML2Graph(sfile,expandGenes=TRUE)

subG <- subKEGGgraph(c("hsa:1848","hsa:1432","hsa:2002","hsa:8986"),gR)
symbols <- c("DUSP6","MAPK14","ELK1","RPS6KA4")
sub2G <- translateKEGGgraph(subG, symbols)
sub2G
nodes(sub2G)

if(require(Rgraphviz) & interactive()) {
plot(sub2G, "neato")
}
```

`translateKEGGID2GeneID` *Translate between KEGGID and Entrez Gene ID*

Description

`translateKEGGID2GeneID` translates KEGGID to NCBI Entrez Gene ID, and `translateGeneID2KEGGID` translates Entrez Gene ID back to KEGGID.

Usage

```
translateKEGGID2GeneID(x, organism="hsa")
translateGeneID2KEGGID(x, organism="hsa")
```

Arguments

`x`	KEGGID, e.g. 'hsa:1432', or Entrez Gene ID, e.g. '1432'
`organism`	Three alphabet code for organisms. The mapping between the orgniams and codes can be found at https://www.genome.jp/kegg/kegg3.html

Details

The KEGGID are unique identifiers used by KEGG PATHWAY to identify gene products. After parsing the KEGG pathway into graph, the graph use KEGGID as its nodes' names.

`translateKEGGID2GeneID` converts KEGGIDs into entrez GeneID, which can be translated to other types of identifiers, for example with biomaRt package or organism-specific annotation packages. See vignette for examples.

`translateKEGG2GeneID` is maintained for back-compatibility and wrapps `translateKEGGID2GeneID`.

Value

Entrez GeneID of the given KEGG ID(s)

Note

This function works so far only with human KEGGIDs, since for them the Entrez GeneID can be derived easily with removing the organism prefix.

The complete functional function will be implemented in the later release of the package.

Author(s)

Jitao David Zhang

Examples

```
egNodes <- c("hsa:1432", "hsa:11072")
translateKEGGID2GeneID(egNodes)
translateGeneID2KEGGID("1432")
```

Index