Package: mully (via r-universe)

August 20, 2024

Type Package

Title Create, Modify and Visualize Multi-Layered Networks

Version 2.1.38

Maintainer Zaynab Hammoud <zaynabhassanhammoud@gmail.com>

Description Allows the user to create graphs with multiple layers. The user can also modify the layers, the nodes, and the edges. The graph can also be visualized. Zaynab Hammoud and Frank Kramer (2018) <doi:10.3390/genes9110519>. More about multilayered graphs and their usage can be found in our review paper: Zaynab Hammoud and Frank Kramer (2020) <doi:10.1186/s41044-020-00046-0>.

URL https://github.com/frankkramer-lab/mully

BugReports https://github.com/frankkramer-lab/mully/issues

License GPL (>= 2)

Encoding UTF-8

RoxygenNote 7.2.3

Imports igraph (>= 1.3.5.9097), rgl (>= 1.0.0), randomcoloR, graphics, stats

Suggests knitr, rmarkdown

VignetteBuilder knitr

Repository https://frankkramer-lab.r-universe.dev

RemoteUrl https://github.com/frankkramer-lab/mully

RemoteRef HEAD

RemoteSha 5c14a3ffa789dfa38b554b12f5608546416def0c

Contents

`mully-package` *The mully package*

Description

R package to create, modify and visualize graphs with multiple layers.

Introduction

Network theory has been used for many years in the modeling and analysis of complex systems, as epidemiology, biology and biomedicine . As the data evolves and becomes more heterogeneous and complex, monoplex networks become an oversimplification of the corresponding systems. This imposes a need to go beyond traditional networks into a richer framework capable of hosting objects and relations of different scales, called Multilayered Network **Mully**, **mul**ti**l**a**y**er networks, is an R package that provides a multilayer network framework. Using this package, the user can create, modify and visualize graphs with multiple layers. This package is an extension to the [igraph package](https://github.com/igraph/rigraph) that provides a monolayer graph framework. The package is implemented as a part of [the Multipath Project](https://www.sys-med.de/en/junior-research-groups/multipath/) directed by [Dr. Frank Kramer](https://www.uni-augsburg.de/de/fakultaet/fai/i

.

Creating graphs

mully, addLayer, addNode, addEdge, removeLayer, removeNode, removeEdge, getNodeAttributes, getEdgeAttributes

Visualization

plot.mully, plot3d

Further information

More information and references can be found in the mully paper:

https://www.mdpi.com/2073-4425/9/11/519

---

addEdge *Add an edge*

Description

Add an edge

Usage

```
addEdge(g, nodeStart, nodeDest, attributes)
```

Arguments

g	The input graph
nodeStart	The first endpoint of the edge
nodeDest	The second endpoint of the edge
attributes	The attributes to assign to the edge

Value

The mully graph, with the added edge

Examples

```
g=mully::demo()
addEdge(g,"dr3","g2",attributes=list(name="newEdge"))
```

addLayer *Add a layer or a set of layers to a graph*

Description

Add a layer or a set of layers to a graph

Usage

```
addLayer(g, nameLayer)
```

Arguments

g	The input graph.
nameLayer	The name or the list of the names of the layers to be added. The layer names must be unique.

Value

The graph, with the layers added.

Examples

```
g = mully("MyFirstMully",direct = FALSE)
g = addLayer(g, c("Gene", "Drug", "Disease"))
```

addNode *Add a node with assigned layer and attributes to a graph*

Description

Add a node with assigned layer and attributes to a graph

Usage

```
addNode(g, nodeName, layerName, attributes = NA)
```

Arguments

g	The input graph.
nodeName	The name of the node to add.
layerName	The name of the layer to be assigned to the node.
attributes	The attributes of the node to add. This argument must be a named list.

Value

The mully graph, with the new node.

Examples

```
g=mully::demo()
attributes=list("specie"="Homo Sapiens")
addNode(g = g,nodeName = "g3",layerName = "Gene",attributes = attributes)
```

---

demo *A demo function to test the package*

Description

A demo function to test the package

Usage

```
demo(name = "MyFirstMully")
```

Arguments

name	Optional name for the graph to be created. By default set to "MyFirstMully"

Value

A mully test graph with few layers, nodes and edges

---

exportCSV *Export mully into CSV files*

Description

Export mully into CSV files

Usage

```
exportCSV(g, target)
```

Arguments

g	The input graph
target	The target file in which the files will be generated.

Value

No return value. Exports the given graph into different CSV files saved in the target folder

Examples

```
## Not run:
g=mully::demo()
exportCSV(g,directory)

## End(Not run)
```

`getEdgeAttributes` *Get the attributes of the edges connecting two nodes*

Description

Get the attributes of the edges connecting two nodes

Usage

```
getEdgeAttributes(g, nodeStart, nodeDest)
```

Arguments

`g`	The input graph
`nodeStart`	The first endpoint of the edge
`nodeDest`	The second endpoint of the edge

Value

A dataframe containing the edges with their attributes. If both nodes' arguments are missing, it returns all the edges with their attributes.

Examples

```
g=mully::demo()
#Print all Edges
getEdgeAttributes(g)
#Get a Single Edge
getEdgeAttributes(g,"d2","g1")
```

getIDEdge *Get the ids of the edges connecting two nodes*

Description

Get the ids of the edges connecting two nodes

Usage

```
getIDEdge(g, nodeStart, nodeDest)
```

Arguments

g	The input graph
nodeStart	The first endpoint of the edge
nodeDest	The second endpoint of the edge

Value

A list containing the ids of the edges connecting the nodes

Examples

```
g=mully::demo()
getIDEdge(g,"d2","dr1")
```

getIDNode *Get the id of a node*

Description

Get the id of a node

Usage

```
getIDNode(g, nameNode)
```

Arguments

g	The input graph
nameNode	The name of the node

Value

The id of the specified node as an integer

Examples

```
g=mully::demo()
getIDNode(g,"g1")
```

`getLayer` *Get the nodes on a layer in a graph*

Description

Get the nodes on a layer in a graph

Usage

```
getLayer(g, nameLayer)
```

Arguments

`g`	The input graph.
`nameLayer`	The name of the layer.

Value

A List of the nodes on the given layer.

Examples

```
g = mully::demo()
getLayer(g,"gene")
```

`getLayersCount` *Get the number of layers in a graph*

Description

Get the number of layers in a graph

Usage

```
getLayersCount(g)
```

Arguments

`g`	The input graph.

Value

The count of the layers.

Examples

```
g = mully("MyFirstMully",direct = FALSE)
g = addLayer(g, c("Gene", "Drug", "Disease"))
getLayersCount(g)
```

---

getNode *Get a node from a graph*

Description

Get a node from a graph

Usage

```
getNode(g, nameNode)
```

Arguments

g	The input graph.
nameNode	The name of the node.

Value

The node as igraph.vs

Examples

```
g=mully::demo()
getNode(g,"g1")
```

---

getNodeAttributes *Get the attributes of a node*

Description

Get the attributes of a node

Usage

```
getNodeAttributes(g, nameNode, layerByName = FALSE)
```

Arguments

g	The input graph
nameNode	The name of the node
layerByName	A boolean to specify whether to export the layers by name or by ID

Value

A dataframe containing the attributes of the specified node

Examples

```
g=mully::demo()
getNodeAttributes(g,layerByName = TRUE)
```

`importEdgesCSV` *Import Edges to a mully graph from a CSV file*

Description

Import Edges to a mully graph from a CSV file

Usage

```
importEdgesCSV(g, file)
```

Arguments

`g`	The mully graph to which the nodes will be added. The graph should already have the layers and the nodes.
`file`	The path to the CSV file containing the edges' information

Value

The mully graph with the added edges

`importGraphCSV` *Import a mully graph from CSV files*

Description

Import a mully graph from CSV files

Usage

```
importGraphCSV(name = NA, direct = FALSE, layers, nodes, edges)
```

Arguments

`name`	The name of the graph
`direct`	A boolean to indicate if the graph is directed or not
`layers`	The path to the CSV file containing the layers' information
`nodes`	The path to the CSV file containing the nodes' information
`edges`	The path to the CSV file containing the edges' information

Value

A new mully graph

`importLayersCSV` *Import Layers to a mully graph from a CSV file*

Description

Import Layers to a mully graph from a CSV file

Usage

```
importLayersCSV(g, file)
```

Arguments

`g`	The mully graph to which the layers will be added. If missing, a new mully graph is created
`file`	The path to the CSV file containing the layers' information

Value

The mully graph with the added layers

`importNodesCSV` *Import Nodes to a mully graph from a CSV file*

Description

Import Nodes to a mully graph from a CSV file

Usage

```
importNodesCSV(g, file, name = "name")
```

Arguments

`g`	The mully graph to which the nodes will be added. The graph should already have the layers.
`file`	The path to the CSV file containing the nodes' information
`name`	The name of the column containing the names of the nodes

Value

The mully graph with the added nodes

is.mully *Is this a mully graph?*

Description

Is this a mully graph?

Usage

```
is.mully(g)
```

Arguments

g	The input graph

Value

A boolean whether the graph is or not a mully object

isLayer *Verify if the layer exists in a graph*

Description

Verify if the layer exists in a graph

Usage

```
isLayer(g, name)
```

Arguments

g	The input graph.
name	The name of the layer.

Value

A boolean value.

Examples

```
g = mully("MyFirstMully",direct = FALSE)
g = addLayer(g, c("Gene", "Drug", "Disease"))
isLayer(g,"Drug")
```

merge *Merge or unite two graphs*

Description

Merge or unite two graphs

Usage

```
merge(g1, g2)
```

Arguments

g1	The first graph to merge. This is the base of the merge.
g2	The second graph to merge. All of its elements are added to the first graph.

Value

The merge of the two graphs as a mully object. The merge is based on the first given graph

mully *Create an empty multilayered graph*

Description

Create an empty multilayered graph

Usage

```
mully(name = NA, direct = TRUE)
```

Arguments

name	The name to be assigned to the graph.
direct	A boolean value, if the graph is directed or not. By default TRUE.

Value

The created multilayered graph, a mully object.

Examples

```
g = mully("MyFirstMully",direct = FALSE)
```

plot.mully *Plot the graph in 2D*

Description

Plot the graph in 2D

Usage

```
## S3 method for class 'mully'
plot(x, layout, ...)
```

Arguments

x	The input graph
layout	The layout. Can either be random or scaled
...	Other arguments to be passed to plot.igraph

Value

No return value. A 2D plot of the given graph is created.

Examples

```
g=mully::demo()
plot(g,"Scaled")
```

plot3d *Plot the graph in 3D using rgl*

Description

Plot the graph in 3D using rgl

Usage

```
plot3d(
  g,
  layers = TRUE,
  vertex.label = NA,
  vertex.label.color = NA,
  vertex.plac = "circle",
  edge.color = NA,
  edge.width = 5,
  edge.arrow.size = 10,
  edge.arrow.width = 1
)
```

Arguments

g	The input graph
layers	A boolean whether to add the layers or not
vertex.label	The vertices' labels
vertex.label.color	The vertices' colors. If not specified, the colors will be chosen randomly
vertex.plac	The placement form of the vertices on the layer. Can either be "circle" which will place them on a circle, or "disc" which will place them randomly on a disc. The default is "circle"
edge.color	The edges' colors. If not specified, inter-edges are black, and intra-edges have the same color as the nodes on the layer
edge.width	The edge width. Default set to 5.
edge.arrow.size	The edges' arrow size. Default set to 10
edge.arrow.width	The edges' arrow width. Default set to 1

Value

No return value. A 3D plot of the given graph is created.

Note

This function can take the following arguments supported and not ignored by rglplot: vertex.label, vertex.label.color, edge.color, edge.width, edge.arrow.size,edge.arrow.width.

Examples

```
g=mully::demo()
labels=getNodeAttributes(g)$name
plot3d(g, layers=TRUE, vertex.label=labels,edge.width=6)
```

---

print.mully *Print function*

Description

Print function

Usage

```
## S3 method for class 'mully'
print(x, ...)
```

Arguments

x	The input graph
...	Other arguments to be passed to print

Value

No return value. The given graph is printed

Examples

```
g=mully::demo()
print(g)
```

---

removeEdge *Delete an edge*

Description

Delete an edge

Usage

```
removeEdge(g, nodeStart, nodeDest, attributes = NA, multi = FALSE)
```

Arguments

g	The input graph
nodeStart	The first endpoint of the edge
nodeDest	The second endpoint of the edge
attributes	The attributes of the edge to delete. Required if the nodes are multi-connected
multi	A boolean. Specifies whether to delete multiple edges or not, in case they exist.

Value

The mully graph with the deleted edges

Examples

```
g=mully::demo()
removeEdge(g,"dr1","d2",multi=TRUE)
```

removeLayer *Delete a layer or a set of layers from a graph*

Description

Delete a layer or a set of layers from a graph

Usage

```
removeLayer(g, name, trans = FALSE)
```

Arguments

g	The input graph.
name	The name or the list of the names of the layers to be deleted.
trans	A boolean whether to insert transitive edges or not

Value

The mully graph, with the given layer and its corresponding nodes and edges removed.

Examples

```
g = mully::demo()
removeLayer(g,"gene",trans=TRUE)
```

removeNode *Delete a node or a set of nodes from a graph*

Description

Delete a node or a set of nodes from a graph

Usage

```
removeNode(g, name, trans = FALSE)
```

Arguments

g	The input graph.
name	The name or the list of names of the nodes to be deleted.
trans	A boolean whether to insert transitive edges or not

Value

The mully graph, with the nodes deleted.

Examples

```
g=mully::demo()
removeNode(g,"dr1",trans=TRUE)
```

Index