

# Package ‘NCIgraph’

November 22, 2024

**Title** Pathways from the NCI Pathways Database

**Version** 1.55.0

**Date** 2012-04-27

**Author** Laurent Jacob

**Maintainer** Laurent Jacob <laurent.jacob@gmail.com>

**Description** Provides various methods to load the pathways from the NCI Pathways Database in R graph objects and to re-format them.

**License** GPL-3

**LazyLoad** yes

**Imports** graph, KEGGgraph, methods, RBGL, RCy3, R.oo

**Depends** R (>= 4.0.0)

**Suggests** Rgraphviz

**Enhances** DEGraph

**biocViews** Pathways, GraphAndNetwork

**git\_url** <https://git.bioconductor.org/packages/NCIgraph>

**git\_branch** devel

**git\_last\_commit** bbee7a4

**git\_last\_commit\_date** 2024-10-29

**Repository** Bioconductor 3.21

**Date/Publication** 2024-11-21

## Contents

directedBFS . . . . .	2
edgesToMerge . . . . .	3
getNCIPathways . . . . .	3
getSubtype.NCIgraph . . . . .	5
is.NCIgraph . . . . .	6
mergeNodes . . . . .	6
NCI.demo.cyList . . . . .	7

NCIgraph . . . . .	8
parseNCInetwork . . . . .	8
propagateRegulation . . . . .	9
translateNCI2GeneID . . . . .	10
<b>Index</b>	<b>11</b>

---

directedBFS	<i>Uses a breadth first search on a directed graph to identify which genes are regulated by a particular node in the graph</i>
-------------	--

---

### Description

Uses a breadth first search on a directed graph to identify which genes are regulated by a particular node in the graph.

### Usage

```
directedBFS(g, node)
```

### Arguments

g	A <a href="#">graph</a> object.
node	A node of g.

### Value

A structured [list](#) containing the regulated genes and the type of interaction between node and each gene.

### Author(s)

Laurent Jacob

### See Also

[propagateRegulation\(\)](#)

---

edgesToMerge	<i>Identifies edges that should be merged to parse a NCI network</i>
--------------	--

---

**Description**

Identifies edges that should be merged to parse a NCI network.

**Usage**

```
edgesToMerge(g)
```

**Arguments**

`g` A [graph](#) object.

**Value**

A [list](#) of edges to be merged

**Author(s)**

Laurent Jacob

**See Also**

[parseNCInetwork\(\)](#)

---

getNCIPathways	<i>Loads networks from Cytoscape and parses them</i>
----------------	--

---

**Description**

Loads networks from Cytoscape and parses them.

**Usage**

```
getNCIPathways(cyList=NULL, parseNetworks=TRUE, entrezOnly=TRUE, verbose=FALSE)
```

**Arguments**

`cyList` a [list](#) providing the networks loaded from Cytoscape. If [NULL](#), the function will try to build the [list](#) from Cytoscape.

`verbose` If [TRUE](#), extra information is output.

`parseNetworks` A [logical](#). If [FALSE](#), the raw NCI networks are returned as [graphNEL](#) objects. If [TRUE](#), some additional parsing is performed by the [parseNCInetwork](#) function.

`entrezOnly` A [logical](#). If [TRUE](#), only keep nodes with an [entrezID](#) property.

**Value**

A **list** of two elements: pList, a **list** of graphNEL objects, and failedW a **list** containing the names of the networks that R failed to read from cytoscape.

**Author(s)**

Laurent Jacob

**See Also**

[parseNCInetwork\(\)](#)

**Examples**

```
##-----
## Load NCIGraph
##-----

library(NCIGraph)

##-----
## Example 1: with Cytoscape
##-----

## Must have Cytoscape running with some networks open and CyREST plugin started.

## In this case, getNCIPathways will both read the raw networks from Cytoscape and parse them.

## Not run:
grList <- getNCIPathways(cyList=NULL, parseNetworks=TRUE, entrezOnly=TRUE, verbose=TRUE)$pList

## End(Not run)

##-----
## Example 2: without Cytoscape
##-----

## Get some raw networks

data("NCIGraphVignette", package="NCIGraph")

## When passed a non null cyList argument (a list of networks),
## getNCIPathways will simply parse the list of networks

grList <- getNCIPathways(cyList=NCI.demo.cyList, parseNetworks=TRUE, entrezOnly=TRUE, verbose=TRUE)$pList
```

---

getSubtype.NCIgraph     *Returns a list of @KEGGEdgeSubType objects describing each edge of the NCI network*

---

### Description

Returns a list of @KEGGEdgeSubType objects describing each edge of the NCI network.

### Usage

```
getSubtype.NCIgraph(object)
```

### Arguments

object             An [NCIgraph](#) object.

### Value

A [list](#) of KEGGEdgeSubType objects.

### Author(s)

Laurent Jacob

### Examples

```
##-----  
## Load NCIgraph  
##-----  
  
library(NCIgraph)  
  
##-----  
## Get some raw networks  
##-----  
  
data("NCIgraphVignette", package="NCIgraph")  
  
##-----  
## Parse them  
##-----  
  
grList <- getNCIPathways(cyList=NCI.demo.cyList, parseNetworks=TRUE, entrezOnly=TRUE, verbose=TRUE)$pList  
  
##-----  
##  
## Get the subtype of the second network. Some activation and some  
## inhibition edges.  
##
```

```
##-----  
getSubtype.NCIgraph(grList[[2]])
```

---

is.NCIgraph	<i>Assess whether a graph is a NCI graph</i>
-------------	--

---

### Description

Assess whether a graph is a NCI graph.

### Usage

```
is.NCIgraph(gr)
```

### Arguments

gr            A [graph](#) object.

### Value

A [logical](#), [TRUE](#) if the graph is a NCI graph, [FALSE](#) otherwise.

### Author(s)

Laurent Jacob

### See Also

[parseNCInetwork\(\)](#)

---

mergeNodes	<i>Merges a given list of nodes in a graph</i>
------------	--

---

### Description

Merges a given list of nodes in a graph.

### Usage

```
mergeNodes(g, mEdges, separateEntrez=TRUE, entrezOnly=TRUE)
```

**Arguments**

`g` A [graph](#) object.  
`mEdges` A [list](#) of nodes to be merged.  
`separateEntrez` A [logical](#). If `TRUE`, don't merge two nodes with `entrezID`.  
`entrezOnly` A [logical](#). If `TRUE`, only keep nodes with an `entrezID` property.

**Value**

The updated [graph](#) object

**Author(s)**

Laurent Jacob

**See Also**

[parseNCInetwork\(\)](#)

---

NCI.demo.cyList	<i>10 raw NCI networks from Nature curated pathways and BioCarta imported as graphNEL objects, for demonstration purpose</i>
-----------------	--

---

**Description**

These are the ten first elements of the full list of raw networks that can be downloaded using the `downloadCyLists.R` script.

**Usage**

```
NCI.demo.cyList
```

**Format**

A list of 10 [graphNEL](#) objects.

**Author(s)**

Laurent Jacob

**Examples**

```
data("NCIgraphVignette")
length(NCI.demo.cyList)

library(Rgraphviz)
plot(NCI.demo.cyList[[1]])
```

---

NCIgraph	<i>Class NCIgraph</i>
----------	-----------------------

---

### Description

Package: NCIgraph

#### Class NCIgraph

public static class **NCIgraph**  
 extends graphNELObject

Class extending graphNEL fro graphs build from NCI gene networks.

### Author(s)

Laurent Jacob

---

parseNCInetwork	<i>Takes a NCI network and transforms it into a simpler graph only representing inhibition/activation relationships between genes</i>
-----------------	---

---

### Description

Takes a NCI network and transforms it into a simpler graph only representing inhibition/activation relationships between genes.

### Usage

```
parseNCInetwork(g, propagateReg=TRUE, separateEntrez=TRUE, mergeEntrezCopies=TRUE, entrezOnly=TRUE)
```

### Arguments

g	A <a href="#">graph</a> object.
propagateReg	A <a href="#">logical</a> . If <b>TRUE</b> , use propagateRegulation to transform the network before parsing it.
separateEntrez	A <a href="#">logical</a> . If <b>TRUE</b> , don't merge two nodes with entrezID.
mergeEntrezCopies	A <a href="#">logical</a> . If <b>TRUE</b> , merge resulting nodes that have the same entrezID.
entrezOnly	A <a href="#">logical</a> . If <b>TRUE</b> , only keep nodes with an entrezID property.

### Value

The new [graph](#) object.



**Author(s)**

Laurent Jacob

**Examples**

```
## Load NCIgraph
library(NCIgraph)

## Get some raw networks
data("NCIgraphVignette", package="NCIgraph")

## Parse the first of them

parsedNetwork <- parseNCInetwork(NCI.demo.cyList[[1]],propagateReg=TRUE,separateEntrez=TRUE,mergeEntrezCopies=T
```

---

`propagateRegulation` *Transforms the network in a way that each Biochemical Reaction node pointing to a Complex points to what is regulated by the complex and updates the interaction types accordingly*

---

**Description**

Transforms the network in a way that each Biochemical Reaction node pointing to a Complex points to what is regulated by the complex and updates the interaction types accordingly.

**Usage**

```
propagateRegulation(g)
```

**Arguments**

`g` A [graph](#) object.

**Value**

The updated [graph](#) object

**Author(s)**

Laurent Jacob

**See Also**

[parseNCInetwork\(\)](#)

---

translateNCI2GeneID *Gives the entrezID corresponding to the nodes of a graph*

---

**Description**

Gives the entrezID corresponding to the nodes of a graph.

**Usage**

```
translateNCI2GeneID(g)
```

**Arguments**

`g` A [graph](#) object.

**Value**

A vector of [character](#) giving the entrez ID of the nodes of `g`.

**Author(s)**

Laurent Jacob

**See Also**

[parseNCInetwork\(\)](#)

**Examples**

```
##-----  
## Load NCIgraph  
##-----  
  
library(NCIgraph)  
  
## Get some raw networks  
  
data("NCIgraphVignette", package="NCIgraph")  
  
## Parse them  
  
grList <- getNCIPathways(cyList=NCI.demo.cyList, parseNetworks=TRUE, entrezOnly=TRUE, verbose=TRUE)$pList  
  
## Get the gene ids for the first of them  
  
gids <- translateNCI2GeneID(grList[[1]])
```

# Index

- \* **classes**
  - NCIgraph, 8
- \* **datasets**
  - NCI.demo.cyList, 7
- \* **documentation**
  - NCIgraph, 8
- character, 10
- directedBFS, 2
- edgesToMerge, 3
- FALSE, 3, 6
- getNCIPathways, 3
- getSubtype.NCIgraph, 5
- graph, 2, 3, 6–10
- is.NCIgraph, 6
- list, 2–5, 7
- logical, 3, 6–8
- mergeNodes, 6
- NCI.demo.cyList, 7
- NCIgraph, 5, 8
- NCIgraph-class (NCIgraph), 8
- NULL, 3
- parseNCInetwork, 3, 4, 6, 7, 8, 9, 10
- propagateRegulation, 2, 9
- translateNCI2GeneID, 10
- TRUE, 3, 6–8