Package 'cisPath'

October 12, 2021

Title Visualization and management of the protein-protein interaction networks.

Version 1.32.0

Author Likun Wang <wanglk@hsc.pku.edu.cn>

Description cisPath is an R package that uses web browsers to visualize and manage protein-protein interaction networks.

Maintainer Likun Wang <wanglk@hsc.pku.edu.cn>

Depends R (>= 2.10.0)

Imports methods, utils

License GPL (>= 3)

biocViews Proteomics

Collate AllGenerics.R cisPath.R networkView.R getResults.R easyEditor.R getMappingFile.R formatPINAPPI.R formatSIFfile.R formatSTRINGPPI.R combinePPI.R formatiRefIndex.R

LazyLoad yes

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

git_branch RELEASE_3_13

git_last_commit 32610c8

git_last_commit_date 2021-05-19

Date/Publication 2021-10-12

R topics documented:

cisPath 2
combinePPI 6
easyEditor 7
formatiRefIndex 7
formatPINAPPI 9
formatSIFfile 10
formatSTRINGPPI 12
getMappingFile 14
networkView 16

Index 19

`cisPath` *Visualization of the shortest paths of functional interaction between proteins*

Description

This method is used to identify and visualize the shortest functional paths between proteins in the protein–protein interaction (PPI) network.

Usage

```
cisPath(infoFile, outputDir, proteinName=NULL, targetProteins=NULL, swissProtID=FALSE, sprotFile="",
     nodeColors=c("#1F77B4", "#FF7F0E", "#D62728", "#9467BD", "#8C564B", "#E377C2"), leafColor="#2CA0
## S4 method for signature 'character,character'
cisPath(infoFile, outputDir, proteinName=NULL, targetProteins=NULL, swissProtID=FALSE, sprotFile="",
     nodeColors=c("#1F77B4", "#FF7F0E", "#D62728", "#9467BD", "#8C564B", "#E377C2"), leafColor="#2CA0
```

Arguments

`infoFile`	File that contains PPI data (character(1)). Please see the file `PPI_Info.txt` as an example.
`outputDir`	Output directory (character(1)).
`proteinName`	Gene name or Swiss-Prot accession number of the source protein (character(1)). If `null`, identify the shortest functional paths in the web page.
`targetProteins`	Gene names or Swiss-Prot accession numbers of the target proteins (character vector). If `null`, treat all other proteins as potential targets.
`swissProtID`	A logical value. If `targetProteins` contains Swiss-Prot accession numbers, set as `TRUE`. If `targetProteins` contains gene names, set as `FALSE`.
`sprotFile`	Input: File downloaded from the UniProt database (UniProtKB/Swiss-Prot) (character(1)).
`tremblFile`	Input: File downloaded from the UniProt database (UniProtKB/TrEMBL) (character(1)).
`nodeColors`	Represents colors for main nodes in the graph. If there are fewer values than main nodes, it will be recycled in the standard manner. Form: "#RRGGBB", each of the pairs RR, GG, BB consist of two hexadecimal digits giving a value in the range 00 to FF.
`leafColor`	Represents color for leaf nodes in the graph. (character(1)) Form: "#RRGGBB", each of the pairs RR, GG, BB consist of two hexadecimal digits giving a value in the range 00 to FF.

byStep — A logical value. If users wish to identify the paths utilizing the shortest number of steps (instead of minimal cost), set byStep as TRUE.
In this situation, all the edge costs will be assigned as 1.
Note: If viewing more possible paths between two proteins is desired, we recommend this value be set as TRUE.

Details

The input PPI data file infoFile should follow the format as the output files of the method formatSTRINGPPI, formatPINAPPI, or formatiRefIndex. See files STRINGPPI.txt or PINAPPI.txt as examples. The first four fields contain the Swiss-Prot accession numbers and gene names for two interacting proteins. The PubMedID field should be stated to be NA if unavailable. The evidence field may present an introduction to the evidence. The edgeValue field should be assigned a value no less than 1. This value will be treated as the cost while identifying the shortest paths. If there is no method available to estimate this value, please give the value as 1.

The shortest functional paths between the proteins are calculated using Dijkstra's algorithm. The results are shown in an HTML file, and users can easily query them using a browser. Each shortest path is displayed as a force-directed graph (http://bl.ocks.org/4062045) with JavaScript library D3 (www.d3js.org). The HTML file follows HTML 4.01 Strict and CSS version 3 standards to maintain consistency across different browsers. Chrome, Firefox, Safari, and IE9 will all properly display the PPI view. Please contact us if the paths do not display correctly.

As an example, we have generated PPI interaction data for several species from the PINA database (http://cbg.garvan.unsw.edu.au/pina/), STRING database (http://string-db.org/), and iRefIndex database (http://www.irefindex.org/wiki/). Users can download these files from http://www.isb.pku.edu.cn/cisPath/. If you make use of these files, please cite PINA, STRING, and iRefIndex accordingly. Users can edit the PPI interactions generated with these two databases, or combine them with their private data to construct more complete PPI interaction networks. In this package, we select only a small portion of the available PPI interaction data as an example. An ID mapping file is also provided in this package, which was generated according to the data from the UniProt (http://www.uniprot.org/) database.

Using this method, a protein must be chosen as the source protein. However, target proteins need not be chosen. If target proteins are not provided, this method will identify the shortest paths between the source protein and all other relevant proteins. The user can query the results upon finding an interesting "target" protein. Although the output in such cases is large, we strongly suggest users give this method of selecting a source but not a target a try.

A protein often has several names, and some of these names have perhaps not been included in the input file infoFile. We therefore suggest users take a look at the output file targetIDs.txt to check whether the input protein names are valid. In order to avoid inputting invalid target protein names, the unique identifier Swiss-Prot accession numbers may alternatively be used as input. The Swiss-Prot accession numbers can be sought in the UniProt (http://www.uniprot.org/) database. We strongly suggest users provide the files from downloaded from the UniProt database (sprotFile and tremblFile).

All species: ftp://ftp.uniprot.org/pub/databases/uniprot/current_release/knowledgebase/complete/uniprot_sprot.dat.gz

ftp://ftp.uniprot.org/pub/databases/uniprot/current_release/knowledgebase/complete/uniprot_trembl.dat.gz
Taxonomic divisions: ftp://ftp.uniprot.org/pub/databases/uniprot/current_release/knowledgebase/taxonomic_divisions/

uniprot_sprot_archaea.dat.gz and uniprot_trembl_archaea.dat.gz contain all archaea entries.
uniprot_sprot_bacteria.dat.gz and uniprot_trembl_bacteria.dat.gz contain all bacteria entries.
uniprot_sprot_fungi.dat.gz and uniprot_trembl_fungi.dat.gz contain all fungi entries.
uniprot_sprot_human.dat.gz and uniprot_trembl_human.dat.gz contain all human entries.
uniprot_sprot_invertebrates.dat.gz and uniprot_trembl_invertebrates.dat.gz contain all invertebrate entries.
uniprot_sprot_mammals.dat.gz and uniprot_trembl_mammals.dat.gz contain all mammalian entries except human and rodent entries.
uniprot_sprot_plants.dat.gz and uniprot_trembl_plants.dat.gz contain all plant entries.
uniprot_sprot_rodents.dat.gz and uniprot_trembl_rodents.dat.gz contain all rodent entries.
uniprot_sprot_vertebrates.dat.gz and uniprot_trembl_vertebrates.dat.gz contain all vertebrate entries except mammals.
uniprot_sprot_viruses.dat.gz and uniprot_trembl_viruses.dat.gz contain all eukaryotic entries except those from vertebrates, fungi and plants.
We suggest you take a look at the README file before you download these files.

If you make use of these files, please cite the UniProt database.

Value

A list will be returned, and each element will contain the shortest paths from the source protein to a target protein.

The output directory contains the shortest paths from a source protein to the target proteins. Users can search for the paths easily using a browser. The file `validInputProteins.txt` contains the proteins that are valid as input to the HTML file. Please take a look at the output file `targetIDs.txt` to check whether the input protein names to this method are valid.

References

Cowley, M.J. and et al. (2012) PINA v2.0: mining interactome modules. *Nucleic Acids Res*, **40**, D862-865.

Wu, J. and et al. (2009) Integrated network analysis platform for protein-protein interactions. *Nature methods*, **6**, 75-77.

Razick S. and et al. (2008) iRefIndex: A consolidated protein interaction database with provenance. *BMC Bioinformatics*, **9**, 405

Aranda, B. and et al. (2011) PSICQUIC and PSISCORE: accessing and scoring molecular interactions, *Nat Methods*, **8**, 528-529.

Szklarczyk,D. and et al. (2011) The STRING database in 2011: functional interaction networks of proteins, globally integrated and scored. *Nucleic Acids Res*, **39**, D561-D568.

Franceschini,A. and et al. (2013) STRING v9.1: protein-protein interaction networks, with increased coverage and integration. *Nucleic Acids Res*, **41**, D808-D815.

UniProt Consortium and others. (2012) Reorganizing the protein space at the Universal Protein Resource (UniProt). *Nucleic Acids Res*, **40**, D71-D75.

See Also

formatSTRINGPPI, formatPINAPPI, formatSIFfile, formatiRefIndex, combinePPI, networkView, easyEditor.

Examples

```
    # examples
    infoFile <- system.file("extdata", "PPI_Info.txt", package="cisPath")

    # source protein: TP53
    # Identify the shortest functional paths from TP53 to all other relevant proteins
    outputDir <- file.path(tempdir(), "TP53_example1")
    results <- cisPath(infoFile, outputDir, "TP53", byStep=TRUE)

    # Identify the shortest paths from TP53 to proteins MAGI1 and GH1
    outputDir <- file.path(tempdir(), "TP53_example2")
   results <- cisPath(infoFile, outputDir, "TP53", targetProteins=c("MAGI1", "GH1"), byStep=TRUE)
    results

   # Identify the shortest paths from TP53 to proteins Q96QZ7 and P01241 (with the Swiss-Prot accession numbers)
    outputDir <- file.path(tempdir(), "TP53_example3")
   results <- cisPath(infoFile, outputDir, "TP53", targetProteins=c("Q96QZ7", "P01241"), swissProtID=TRUE, byStep

    # Identify the shortest functional paths in the web page
    outputDir <- file.path(tempdir(), "cisPath_example")
    results <- cisPath(infoFile, outputDir)

## Not run:
    # example of downloading PPI data from our website

    # Change to your own output directory
    outputDir <- file.path(getwd(), "TP53")
    # Create the output directory
    dir.create(outputDir, showWarnings=FALSE, recursive=TRUE)

    # infoFile: site where the PPI data file will be saved.
    infoFile <- file.path(outputDir, "PPIdata.txt")

    # Download PPI data from our website
   download.file("http://www.isb.pku.edu.cn/cispath/data/Homo_sapiens_PPI.txt", infoFile)
   download.file("http://www.isb.pku.edu.cn/cispath/data/Caenorhabditis_elegans_PPI.txt", infoFile)
   download.file("http://www.isb.pku.edu.cn/cispath/data/Drosophila_melanogaster_PPI.txt", infoFile)
   download.file("http://www.isb.pku.edu.cn/cispath/data/Mus_musculus_PPI.txt", infoFile)
   download.file("http://www.isb.pku.edu.cn/cispath/data/Rattus_norvegicus_PPI.txt", infoFile)
   download.file("http://www.isb.pku.edu.cn/cispath/data/Saccharomyces_cerevisiae_PPI.txt", infoFile)
```

```
    results <- cisPath(infoFile, outputDir, "TP53")
    outputDir <- file.path(getwd(), "cisPathWeb")
    results <- cisPath(infoFile, outputDir)

## End(Not run)
```

`combinePPI` *Combine PPI information from databases*

Description

This method is used to combine the PPI information generated from different databases.

Usage

```
combinePPI(input, output, mappingFile="", dbNames="", maxEdgeValue=-1)
## S4 method for signature 'character,character'
combinePPI(input, output, mappingFile="", dbNames="", maxEdgeValue=-1)
```

Arguments

`input`	Files that contain PPI information (character vector).
`output`	Output file (character(1)).
`mappingFile`	Identifier mapping file (character(1)). Generate this file with method `getMappingFile`.
`dbNames`	dbNames for input files (character vector).
`maxEdgeValue`	Filter out PPI information with edge cost greater than this value. (double(1)). Default: no filter.

Details

The input files should follow the format as the output files of the method `formatSTRINGPPI`, `formatPINAPPI`, `formatSIFfile`, or `formatiRefIndex`. See the files `STRINGPPI.txt` or `PINAPPI.txt` as examples. The first four fields contain the Swiss-Prot accession numbers and gene names for two interacting proteins. The `PubMedID` field should be stated to be `NA` if unavailable. The `evidence` field may present an introduction to the evidence. The `edgeValue` field should be given a value no less than `1`. This value will be treated as the cost while identifying the shortest paths. If there is no method available to estimate this value, please give the value as `1`.

In some cases, a protein may have several swiss-Prot accession numbers. The parameter `mappingFile` is used to unify the swiss-Prot accession numbers for proteins from different databases. The proteins whose swiss-Prot accession numbers are not included in the mappingfile will be ignored as well as the intereractions associated to them.

See Also

`formatSTRINGPPI`, `formatPINAPPI`, `formatSIFfile`, `formatiRefIndex`, `cisPath`, `getMappingFile`.

Examples

```
library(cisPath)
inputFile1 <- system.file("extdata", "PINA2PPI.txt", package="cisPath")
inputFile2 <- system.file("extdata", "iRefIndex.txt", package="cisPath")
inputFile3 <- system.file("extdata", "STRINGPPI.txt", package="cisPath")
output <- file.path(tempdir(), "allPPI.txt")
combinePPI(c(inputFile1, inputFile2, inputFile3), output, maxEdgeValue=1.4)
```

---

easyEditor *Easy editor for network graphs*

Description

This method is used to open the editor for network graphs.

Usage

```
easyEditor(outputDir)
## S4 method for signature 'character'
easyEditor(outputDir)
```

Arguments

outputDir	Output directory (character(1)).

Value

The output HTML file of this method is a editor for network graphs. Users can draw and edit their own network with this editor.

Examples

```
library(cisPath)
outputDir <- file.path(tempdir(), "easyEditor")
easyEditor(outputDir)
```

---

formatiRefIndex *Format PPI files downloaded from the iRefIndex database*

Description

This method is used to format the PPI file which is downloaded from the iRefIndex database.

Usage

```
formatiRefIndex(input, output, taxonId="")
## S4 method for signature 'character,character'
formatiRefIndex(input, output, taxonId="")
```

Arguments

input	File downloaded from the iRefIndex database (character(1)).
output	Output file (character(1)).
taxonId	NCBI taxonomy specie identifier (character(1)). Process only data for this specie. Examples: 9606: Homo sapiens 4932: Saccharomyces cerevisiae 6239: Caenorhabditis elegans 7227: Drosophila melanogaster 10090: Mus musculus 10116: Rattus norvegicus

Details

The input file is downloaded from the iRefIndex database (http://irefindex.org/wiki/).
Access http://irefindex.org/download/irefindex/data/archive/release_13.0/psi_mitab/MITAB2.6/ to download PPI files with the MITAB2.6 format for different species.
If you make use of this file, please cite the iRefIndex database.

Value

Each line of the output file contains Swiss-Prot accession numbers and gene names for two interacting proteins. The edge value will be assigned as 1 for each link between two interacting proteins. This may be treated as the "cost" while identifying the shortest paths between proteins. Advanced users can edit the file and change this value for each edge.

References

Razick S. and et al. (2008) iRefIndex: A consolidated protein interaction database with provenance. *BMC Bioinformatics*, **9**, 405

Aranda, B. and et al. (2011) PSICQUIC and PSISCORE: accessing and scoring molecular interactions, *Nat Methods*, **8**, 528-529.

See Also

cisPath, formatPINAPPI, formatSIFfile, formatSTRINGPPI, combinePPI.

Examples

```
    library(cisPath)
    input <- system.file("extdata", "9606.mitab.100.txt", package="cisPath")
    output <- file.path(tempdir(), "iRefIndex.txt")
    formatiRefIndex(input, output)

## Not run:
    outputDir <- file.path(getwd(), "cisPath_test")
    dir.create(outputDir, showWarnings=FALSE, recursive=TRUE)
```

```
    # Download iRefIndex PPI for humans only (compressed:~105M, decompressed:~757M)
    destfile <- file.path(outputDir, "9606.mitab.08122013.txt.zip")
    cat("Downloading...\n")
    httpURL <- "http://irefindex.org/download"
    filePath <- "irefindex/data/archive/release_13.0/psi_mitab/MITAB2.6"
    fileName <- "9606.mitab.08122013.txt.zip"
    URL <- paste(httpURL, filePath, fileName, sep="/")
    download.file(URL, destfile)
    unzip(destfile, overwrite=TRUE, exdir=outputDir)

    # Format iRefIndex PPI
    fileFromiRef <- file.path(outputDir, "9606.mitab.08122013.txt")
    iRefIndex <- file.path(outputDir, "iRefIndex.txt")
    formatiRefIndex(fileFromiRef, output=iRefIndex)

## End(Not run)
```

`formatPINAPPI` *Format PPI files downloaded from the PINA database (MITAB Format)*

Description

This method is used to format the PPI file which is downloaded from the PINA database (MITAB Format).

Usage

```
formatPINAPPI(input, output)
## S4 method for signature 'character,character'
formatPINAPPI(input, output)
```

Arguments

`input`	File downloaded from the PINA database (character(1)).
`output`	Output file (character(1)).

Details

The input file (MITAB Format) is downloaded from the PINA database (http://cbg.garvan.unsw.edu.au/pina/).
Access http://cbg.garvan.unsw.edu.au/pina/interactome.stat.do to download PPI files with the `MITAB format` for different species.
If you make use of this file, please cite the PINA database.

Value

Each line of the output file contains Swiss-Prot accession numbers and gene names for two interacting proteins. The edge value will be assigned as 1 for each link between two interacting proteins. This may be treated as the "cost" while identifying the shortest paths between proteins. Advanced users can edit the file and change this value for each edge.

References

Cowley, M.J. and et al. (2012) PINA v2.0: mining interactome modules. *Nucleic Acids Res*, **40**, D862-865.

Wu, J. and et al. (2009) Integrated network analysis platform for protein-protein interactions. *Nature methods*, **6**, 75-77.

See Also

cisPath, formatSIFfile, formatiRefIndex, formatSTRINGPPI, combinePPI.

Examples

```
    library(cisPath)
    input <- system.file("extdata", "Homo_sapiens_PINA100.txt", package="cisPath")
    output <- file.path(tempdir(), "PINAPPI.txt")
    formatPINAPPI(input, output)

## Not run:
    outputDir <- file.path(getwd(), "cisPath_test")
    dir.create(outputDir, showWarnings=FALSE, recursive=TRUE)

    # Download PINA PPI (MITAB format) for humans only (~96M)
    destfile <- file.path(outputDir, "Homo_sapiens.txt")
    cat("Downloading...\n")
   download.file("http://cbg.garvan.unsw.edu.au/pina/download/Homo%20sapiens-20121210.txt", destfile)

    # Format PINA PPI
    fileFromPINA <- file.path(outputDir, "Homo_sapiens.txt")
    PINAPPI <- file.path(outputDir, "PINAPPI.txt")
    formatPINAPPI(fileFromPINA, output=PINAPPI)

## End(Not run)
```

---

formatSIFfile *Format PPI file downloaded from the PINA2 database (SIF Format)*

Description

This method is used to format the PPI file which is downloaded from the PINA2 database (SIF Format).

Usage

```
formatSIFfile(input, mappingFile, output)
## S4 method for signature 'character,character,character'
formatSIFfile(input, mappingFile, output)
```

Arguments

input	File downloaded from the PINA2 database (SIF Format) (character(1)).
mappingFile	Identifier mapping file (character(1)). Generate this file with method getMappingFile.
output	Output file (character(1)).

Details

The input file (SIF Format) is downloaded from PINA2 database (http://cbg.garvan.unsw.edu.au/pina/).
Access http://cbg.garvan.unsw.edu.au/pina/interactome.stat.do to download PPI files with the SIF format for different species.
If you make use of this file, please cite the PINA database.

Value

Each line of the output file contains Swiss-Prot accession numbers and gene names for two interacting proteins. The edge value will be assigned as 1 for each link between two interacting proteins. This may be treated as the "cost" while identifying the shortest paths between proteins. Advanced users can edit the file and change this value for each edge.

References

Cowley, M.J. and et al. (2012) PINA v2.0: mining interactome modules. *Nucleic Acids Res*, **40**, D862-865.

Wu, J. and et al. (2009) Integrated network analysis platform for protein-protein interactions. *Nature methods*, **6**, 75-77.

See Also

cisPath, getMappingFile, formatPINAPPI, formatSTRINGPPI, formatiRefIndex, combinePPI.

Examples

```
library(cisPath)

# Generate the identifier mapping file
input <- system.file("extdata", "uniprot_sprot_human10.dat", package="cisPath")
mappingFile <- file.path(tempdir(), "mappingFile.txt")
getMappingFile(input, output=mappingFile, taxonId="9606")

# Format the file downloaded from STRING database
output <- file.path(tempdir(), "PINA2PPI.txt")
```

```
    fileFromPINA2 <- system.file("extdata", "Homo_sapiens_PINA2.sif", package="cisPath")
    formatSIFfile(fileFromPINA2, mappingFile, output)

## Not run:
    if (!requireNamespace("BiocManager", quietly=TRUE))
        install.packages("BiocManager")
    BiocManager::install("R.utils")
    library(R.utils)

    outputDir <- file.path(getwd(), "cisPath_test")
    dir.create(outputDir, showWarnings=FALSE, recursive=TRUE)

    # Generate the identifier mapping file
    fileFromUniProt <- file.path(outputDir, "uniprot_sprot_human.dat")
    mappingFile <- file.path(outputDir, "mappingFile.txt")
    getMappingFile(fileFromUniProt, output=mappingFile)

    # Download PINA2 PPI (SIF format) for humans only (~2.8M)
    destfile <- file.path(outputDir, "Homo_sapiens.sif")
    cat("Downloading...\n")
   download.file("http://cbg.garvan.unsw.edu.au/pina/download/Homo%20sapiens-20140521.sif", destfile)

    # Format PINA2 PPI
    fileFromPINA2 <- file.path(outputDir, "Homo_sapiens.sif")
    PINA2PPI <- file.path(outputDir, "PINA2PPI.txt")
    formatSIFfile(fileFromPINA2, mappingFile, output=PINA2PPI)

## End(Not run)
```

`formatSTRINGPPI` *Format PPI file downloaded from the STRING database*

Description

This method is used to format the PPI file which is downloaded from the STRING database.

Usage

```
formatSTRINGPPI(input, mappingFile, taxonId, output, minScore=700)
## S4 method for signature 'character,character,character,character'
formatSTRINGPPI(input, mappingFile, taxonId, output, minScore=700)
```

Arguments

`input`	File downloaded from the STRING database (character(1)).
`mappingFile`	Identifier mapping file (character(1)). Generate this file with method `getMappingFile`.

taxonId	NCBI taxonomy specie identifier (character(1)). Process only data for this specie. Examples: 9606: Homo sapiens 4932: Saccharomyces cerevisiae 6239: Caenorhabditis elegans 7227: Drosophila melanogaster 10090: Mus musculus 10116: Rattus norvegicus
output	Output file (character(1)).
minScore	Filter out PPI information with STRING scores less than this value. (integer(1)). Recommended default 700 (Only consider high confidence interactions).

Details

The input file is downloaded from the STRING database (http://string-db.org/). The URL of this file is http://string-db.org/newstring_download/protein.links.v9.1.txt.gz. Access http://string-db.org/newstring_download/species.v9.1.txt to determine the parameter taxonId. Access http://string-db.org/newstring_cgi/show_download_page.pl for more details.
If you make use of this file, please cite the STRING database.

Value

Each line of the output file contains Swiss-Prot accession numbers and gene names for two interacting proteins. An edge value is estimated for each link between two interacting proteins. This value is defined as max(1,log(1000-STRING_SCORE,100)). This may be treated as the "cost" while determining the shortest paths between proteins. Advanced users can edit the file and change this value for each edge.

References

Szklarczyk,D. and et al. (2011) The STRING database in 2011: functional interaction networks of proteins, globally integrated and scored. *Nucleic Acids Res*, **39**, D561-D568.

Franceschini,A. and et al. (2013) STRING v9.1: protein-protein interaction networks, with increased coverage and integration. *Nucleic Acids Res*, **41**, D808-D815.

UniProt Consortium and others. (2012) Reorganizing the protein space at the Universal Protein Resource (UniProt). *Nucleic Acids Res* **40**, D71-D75.

See Also

cisPath, getMappingFile, formatPINAPPI, formatSIFfile, formatiRefIndex, combinePPI.

Examples

```
library(cisPath)

# Generate the identifier mapping file
input <- system.file("extdata", "uniprot_sprot_human10.dat", package="cisPath")
```

```
    mappingFile <- file.path(tempdir(), "mappingFile.txt")
    getMappingFile(input, output=mappingFile, taxonId="9606")

    # Format the file downloaded from STRING database
    output <- file.path(tempdir(), "STRINGPPI.txt")
    fileFromSTRING <- system.file("extdata", "protein.links.txt", package="cisPath")
    formatSTRINGPPI(fileFromSTRING, mappingFile, "9606", output, 700)

## Not run:
    if (!requireNamespace("BiocManager", quietly=TRUE))
        install.packages("BiocManager")
    BiocManager::install("R.utils")
    library(R.utils)

    outputDir <- file.path(getwd(), "cisPath_test")
    dir.create(outputDir, showWarnings=FALSE, recursive=TRUE)

    # Generate the identifier mapping file
    fileFromUniProt <- file.path(outputDir, "uniprot_sprot_human.dat")
    mappingFile <- file.path(outputDir, "mappingFile.txt")
    getMappingFile(fileFromUniProt, output=mappingFile)

    # Download STRING PPI for Homo sapiens (compressed:~27M, decompressed:~213M)
    destfile <- file.path(outputDir, "9606.protein.links.v9.1.txt.gz")
    cat("Downloading...\n")
   download.file("http://string-db.org/newstring_download/protein.links.v9.1/9606.protein.links.v9.1.txt.gz", d
    cat("Uncompressing...\n")
    gunzip(destfile, overwrite=TRUE, remove=FALSE)

    # Format STRING PPI
    fileFromSTRING <- file.path(outputDir, "9606.protein.links.v9.1.txt")
    STRINGPPI <- file.path(outputDir, "STRINGPPI.txt")
    formatSTRINGPPI(fileFromSTRING, mappingFile, "9606", output=STRINGPPI, 700)

## End(Not run)
```

`getMappingFile` *Generate the identifier mapping file*

Description

This method is used to generate the identifier mapping file which is necessary for methods `formatSIFfile` and `formatSTRINGPPI`.

Usage

```
getMappingFile(sprotFile, output, tremblFile="", taxonId="")
## S4 method for signature 'character,character'
getMappingFile(sprotFile, output, tremblFile="", taxonId="")
```

Arguments

sprotFile	Input: File downloaded from the UniProt database (UniProtKB/Swiss-Prot) (character(1)).
output	Output file (character(1)).
tremblFile	Input: File downloaded from the UniProt database (UniProtKB/TrEMBL) (character(1)).
taxonId	NCBI taxonomy specie identifier (character(1)). This method will process only data for this specie. Default: process all data (recommended).

Details

UniProtKB/Swiss-Prot: fully annotated curated entries.
UniProtKB/TrEMBL: computer-generated entries enriched with automated classification and annotation.
sprotFile is mandatory, while tremblFile is optional. If users only want to process the reviewed proteins from the UniProt database, tremblFile should be ignored.

All species: ftp://ftp.uniprot.org/pub/databases/uniprot/current_release/knowledgebase/complete/uniprot_sprot.dat.gz
ftp://ftp.uniprot.org/pub/databases/uniprot/current_release/knowledgebase/complete/uniprot_trembl.dat.gz
Taxonomic divisions: ftp://ftp.uniprot.org/pub/databases/uniprot/current_release/knowledgebase/taxonomic_divisions/

uniprot_sprot_archaea.dat.gz and uniprot_trembl_archaea.dat.gz contain all archaea entries.
uniprot_sprot_bacteria.dat.gz and uniprot_trembl_bacteria.dat.gz contain all bacteria entries.
uniprot_sprot_fungi.dat.gz and uniprot_trembl_fungi.dat.gz contain all fungi entries.
uniprot_sprot_human.dat.gz and uniprot_trembl_human.dat.gz contain all human entries.
uniprot_sprot_invertebrates.dat.gz and uniprot_trembl_invertebrates.dat.gz contain all invertebrate entries.
uniprot_sprot_mammals.dat.gz and uniprot_trembl_mammals.dat.gz contain all mammalian entries except human and rodent entries.
uniprot_sprot_plants.dat.gz and uniprot_trembl_plants.dat.gz contain all plant entries.
uniprot_sprot_rodents.dat.gz and uniprot_trembl_rodents.dat.gz contain all rodent entries.
uniprot_sprot_vertebrates.dat.gz and uniprot_trembl_vertebrates.dat.gz contain all vertebrate entries except mammals.
uniprot_sprot_viruses.dat.gz and uniprot_trembl_viruses.dat.gz contain all eukaryotic entries except those from vertebrates, fungi and plants.
We suggest you take a look at the README file before you download these files.

If you make use of these files, please cite the UniProt database.

Value

The output file contains identifier mapping information which is necessary for methods formatSIFfile and formatSTRINGPPI. Each line contains both the Ensembl Genomes Protein identifier and the

Swiss-Prot accession number for a given protein.

References

UniProt Consortium and others. (2012) Reorganizing the protein space at the Universal Protein Resource (UniProt). *Nucleic Acids Res* **40**, D71-D75.

See Also

cisPath, formatSTRINGPPI, formatSIFfile, combinePPI.

Examples

```
    library(cisPath)
    sprotFile <- system.file("extdata", "uniprot_sprot_human10.dat", package="cisPath")
    output <- file.path(tempdir(), "mappingFile.txt")
    getMappingFile(sprotFile, output, taxonId="9606")

## Not run:
    if (!requireNamespace("BiocManager", quietly=TRUE))
        install.packages("BiocManager")
    BiocManager::install("R.utils")
    library(R.utils)

    outputDir <- file.path(getwd(), "cisPath_test")
    dir.create(outputDir, showWarnings=FALSE, recursive=TRUE)

    # Download protein information file for humans only from UniProt (decompressed:~246M)
    destfile <- file.path(outputDir, "uniprot_sprot_human.dat.gz");
    cat("Downloading...\n")
   download.file("ftp://ftp.uniprot.org/pub/databases/uniprot/current_release/knowledgebase/taxonomic_divisions
    gunzip(destfile, overwrite=TRUE, remove=FALSE)

    destfile <- file.path(outputDir, "uniprot_trembl_human.dat.gz");
    cat("Downloading...\n")
   download.file("ftp://ftp.uniprot.org/pub/databases/uniprot/current_release/knowledgebase/taxonomic_divisions
    gunzip(destfile, overwrite=TRUE, remove=FALSE)

    # Generate identifier mapping file
    sprotFile <- file.path(outputDir, "uniprot_sprot_human.dat")
    tremblFile <- file.path(outputDir, "uniprot_trembl_human.dat")
    mappingFile <- file.path(outputDir, "mappingFile.txt")
    getMappingFile(sprotFile, output=mappingFile, tremblFile)

## End(Not run)
```

---

networkView *Visualization of the interactions between input proteins*

Description

This method can visualize the input proteins in the protein–protein interaction (PPI) network and display the evidence that supports the specific interactions among these proteins.

Usage

```
networkView(infoFile, proteinNames, outputDir, swissProtID=FALSE, mainNode=c(1), nodeColors=c("#1F77B
## S4 method for signature 'character,character,character'
networkView(infoFile, proteinNames, outputDir, swissProtID=FALSE, mainNode=c(1), nodeColors=c("#1F77B
```

Arguments

`infoFile`	File that contains PPI data (character(1)). Please see the file `PPI_Info.txt` as an example.
`proteinNames`	Gene names or Swiss-Prot accession numbers of the proteins (character vector).
`outputDir`	Output directory (character(1)).
`swissProtID`	A logical value. If `proteinNames` contains Swiss-Prot accession numbers, set as `TRUE`. If `proteinNames` contains gene names, set as `FALSE`.
`mainNode`	A vector of value `0` and `1`. `0`: display the corresponding protein as a leaf node. `1`: display the corresponding protein as a main node. If there are fewer values than proteins, it will be recycled in the standard manner.
`nodeColors`	Represents colors for proteins. If there are fewer values than proteins, it will be recycled in the standard manner. Form: "#RRGGBB", each of the pairs RR, GG, BB consist of two hexadecimal digits giving a value in the range 00 to FF.
`displayMore`	A logical value. If `TRUE`, other proteins will be displayed which can interact with at least `two` proteins in `proteinNames`.
`leafColor`	Only used while `displayMore=TRUE`. This is the color for additional proteins. (character(1)) Form: "#RRGGBB", each of the pairs RR, GG, BB consist of two hexadecimal digits giving a value in the range 00 to FF.

Details

As an example, we have generated PPI interaction data for several species from the PINA database (`http://cbg.garvan.unsw.edu.au/pina/`) and the STRING database (`http://string-db.org/`). Users can download these files from `http://www.isb.pku.edu.cn/cisPath/`. If you make use of these files, please cite PINA or STRING accordingly. Users can edit the PPIs generated with these two databases, or combine them with their private data to construct more complete PPI networks. In this package, we select only a small portion of the available PPI data as an example.

A protein often has several names, and some of these names have perhaps not been included in the input file `infoFile`. We therefore suggest users take a look at the output file `proteinIDs.txt`

to check whether the input protein names are valid. In order to avoid inputting invalid target protein names, the unique identifier Swiss-Prot accession numbers may alternatively be used as input. The Swiss-Prot accession numbers can be sought in the UniProt (http://www.uniprot.org/) database.

Value

The output HTML file contains the PPIs and related evidence supporting them. Users can view the PPI network and the evidence supporting these interactions easily using a browser. Please take a look at the output file `proteinIDs.txt` to check whether the input protein names are valid.

References

Cowley, M.J. and et al. (2012) PINA v2.0: mining interactome modules. *Nucleic Acids Res*, **40**, D862-865.

Wu, J. and et al. (2009) Integrated network analysis platform for protein-protein interactions. *Nature methods*, **6**, 75-77.

Szklarczyk,D. and et al. (2011) The STRING database in 2011: functional interaction networks of proteins, globally integrated and scored. *Nucleic Acids Res*, **39**, D561-D568.

Franceschini,A. and et al. (2013) STRING v9.1: protein-protein interaction networks, with increased coverage and integration. *Nucleic Acids Res*, **41**, D808-D815.

UniProt Consortium and others. (2012) Reorganizing the protein space at the Universal Protein Resource (UniProt). *Nucleic Acids Res* **40**, D71-D75.

See Also

cisPath, easyEditor.

Examples

```
 library(cisPath)
 infoFile <- system.file("extdata", "PPI_Info.txt", package="cisPath")
 outputDir <- file.path(tempdir(), "networkView")
networkView(infoFile, c("MAGI1","TP53BP2","TP53", "PTEN"), outputDir, FALSE, c(1,1,1,0), displayMore=TRUE)

 outputDir2 <- file.path(tempdir(), "networkView2")
 inputFile <- system.file("extdata", "networkView.txt", package="cisPath")
 rt <- read.table(inputFile, sep=",", comment.char="", header=TRUE)
 proteins <- as.vector(rt[,1])
 sizes <- as.vector(rt[,2])
 nodeColors <- as.vector(rt[,3])
networkView(infoFile, proteins, outputDir2, FALSE, sizes, nodeColors, displayMore=FALSE)
```

Index

∗ **methods**
    cisPath, 2
    combinePPI, 6
    easyEditor, 7
    formatiRefIndex, 7
    formatPINAPPI, 9
    formatSIFfile, 10
    formatSTRINGPPI, 12
    getMappingFile, 14
    networkView, 16

cisPath, 2, *6*, *8*, *10, 11*, *13*, *16*, *18*
cisPath,character,character-method
        *(*cisPath*)*, 2
combinePPI, *5*, 6, *8*, *10, 11*, *13*, *16*
combinePPI,character,character-method
        *(*combinePPI*)*, 6

easyEditor, *5*, 7, *18*
easyEditor,character-method
        *(*easyEditor*)*, 7

formatiRefIndex, *3*, *5, 6*, 7, *10, 11*, *13*
formatiRefIndex,character,character-method
        *(*formatiRefIndex*)*, 7
formatPINAPPI, *3*, *5, 6*, *8*, 9, *11*, *13*
formatPINAPPI,character,character-method
        *(*formatPINAPPI*)*, 9
formatSIFfile, *5, 6*, *8*, *10*, 10, *13–16*
formatSIFfile,character,character,character-method
        *(*formatSIFfile*)*, 10
formatSTRINGPPI, *3*, *5, 6*, *8*, *10, 11*, 12, *14–16*
formatSTRINGPPI,character,character,character,character-method
        *(*formatSTRINGPPI*)*, 12

getMappingFile, *6*, *11–13*, 14
getMappingFile,character,character-method
        *(*getMappingFile*)*, 14

networkView, *5*, 16
networkView,character,character,character-method
        *(*networkView*)*, 16