Package ‘pkgndep’

October 13, 2023

Type Package

Title Analyze Dependency Heaviness of R Packages

Version 1.99.3

Date 2023-10-12

Depends R (>= 4.0.0)

Imports ComplexHeatmap (>= 2.6.0), GetoptLong, GlobalOptions, utils, grid, hash, methods, BiocManager, brew, BiocVersion

Suggests knitr, rmarkdown, svglite, callr, rjson, Rook, igraph, ggplot2, ggrepel, base64, testthat, cowplot

Description A new metric named 'dependency heaviness' is proposed that measures the number of additional dependency packages that a parent package brings to its child package and are unique to the dependency packages imported by all other parents. The dependency heaviness analysis is visualized by a customized heatmap. The package is described in <doi:10.1093/bioinformatics/btac449>. We have also performed the dependency heaviness analysis on the CRAN/Bioconductor package ecosystem and the results are implemented as a web-based database which provides comprehensive tools for querying dependencies of individual R packages. The systematic analysis on the CRAN/Bioconductor ecosystem is described in <doi:10.1016/j.jss.2023.111610>. From 'pkgndep' version 2.0.0, the heaviness database includes snapshots of the CRAN/Bioconductor ecosystems for many old R versions.

URL `https://github.com/jokergoo/pkgndep`

VignetteBuilder knitr

License MIT + file LICENSE

NeedsCompilation no

Author Zuguang Gu [aut, cre] (<`https://orcid.org/0000-0002-7395-8709`>)

Maintainer Zuguang Gu <`z.gu@dkfz.de`>

Repository CRAN

Date/Publication 2023-10-13 05:50:02 UTC

R topics documented:

`ALL_BIOC_RELEASES` *All Bioconductor releases*

Description

All Bioconductor releases

Usage

```
ALL_BIOC_RELEASES
```

Value

A data frame

Examples

```
ALL_BIOC_RELEASES
```

`all_pkg_stat_snapshot` *The complete table of dependency heaviness for all CRAN/Bioconductor packages*

Description

The complete table of dependency heaviness for all CRAN/Bioconductor packages

Usage

```
all_pkg_stat_snapshot()
```

Value

The returned data frame is directly from `load_pkg_stat_snapshot`, but with only a subset of columns of heaviness metrics.

Examples

```
# There is no example
NULL
```

`check_pkg` *Check whether a package is available*

Description

Check whether a package is available

Usage

```
check_pkg(pkg, bioc = FALSE)
```

Arguments

`pkg`	The name of the package.
`bioc`	Whether it is a Bioconductor package.

Details

One of the suggestions to avoid heavy dependencies is to put parent packages that are not frequently used to 'Suggests' and to load them when the corresponding functions are used. Here the `check_pkg` function helps to check whether these parent packages are avaiable and if not, it prints messages to guide users to install the corresponding packages.

Examples

```
# There is no example
NULL
```

`child_dependency` *Get child dependency for a package*

Description

Get child dependency for a package

Usage

```
child_dependency(package, fields = NULL, online = FALSE)
```

Arguments

`package`	Package name.
`fields`	Which fields in DESCRIPTION? Values should be in `Depends`, `Imports`, `LinkingTo`, `Suggests` and `Enhances`. The value can also be set to `strong` or `weak`.
`online`	Whether use the newest package database directly from CRAN/Bioconductor or the pre-computated package database? The version of the pre-computated package database can be set via `pkgndep_opt`$heaviness_db_version.

Value

A data frame with child packages as well as its heaviness on its child packages. If `snapshot` is set to `FALSE`, heaviness on child packages is set to NA.

Examples

```
## Not run:
child_dependency("ComplexHeatmap")

## End(Not run)
```

`co_heaviness` *Co-heaviness for pairs of parent packages*

Description

Co-heaviness for pairs of parent packages

Usage

```
co_heaviness(x, rel = FALSE, a = 10, jaccard = FALSE)
```

Arguments

`x`	An object returned by pkgndep.
`rel`	Whether to return the absolute measure or the relative measure.
`a`	A constant added for calculating the relative measure.
`jaccard`	Whether to return Jaccard coeffcient?

Details

Denote a package as P and its two strong parent packages as A and B, i.e., parent packages in "Depends", "Imports" and "LinkingTo", the co-heaviness for A and B is calculated as follows.

Denote S_A as the set of reduced dependency packages when only moving A to "Suggests" of P, and denote S_B as the set of reduced dependency packages when only moving B to "Suggests" of P, denote S_AB as the set of reduced dependency packages when moving A and B together to "Suggests" of P, the co-heaviness of A, B on P is calculatd as `length(setdiff(S_AB, union(S_A, S_B)))`, which is the number of reduced package only caused by co-action of A and B.

Note the co-heaviness is only calculated for parent packages in "Depends", "Imports" and "LinkingTo".

When `jaccard` is set to `TRUE`, the function returns jaccard coeffcient. `setdiff(S_AB, union(S_A, S_B))` is actually the set of dependencies imported by and only by two parent packages A and B. Thus the jaccard coeffcient is calculated as `length(setdiff(S_AB, union(S_A, S_B)))/length(S_AB)`.

Examples

```
## Not run:
# DESeq version 1.36.0, the dependencies have been changed in later versions.
x = readRDS(system.file("extdata", "DESeq2_dep.rds", package = "pkgndep"))
hm = co_heaviness(x)
ComplexHeatmap::Heatmap(hm)
co_heaviness(x, jaccard = TRUE)

## End(Not run)
```

`dependency_database` *Database of package dependency heaviness of all R packages*

Description

Database of package dependency heaviness of all R packages

Usage

```
dependency_database(version = pkgndep_opt$heaviness_db_version)
```

Arguments

`version`	Version of the heaviness database. See `pkgndep_opt`$heaviness_db_version.

Examples

```
if(interactive()) {
    dependency_database()
}
```

`dependency_heatmap` *Make the dependency heatmap*

Description

Make the dependency heatmap

Usage

```
dependency_heatmap(x, pkg_fontsize = 10*cex, title_fontsize = 12*cex,
    legend_fontsize = 10*cex, fix_size = !dev.interactive(), cex = 1,
    help = TRUE, file = NULL, res = 144)
```

Arguments

`x`	An object from `pkgndep`.
`pkg_fontsize`	Font size for the package names.
`title_fontsize`	Font size for the title.
`legend_fontsize`	Font size for the legends.
`fix_size`	Should the rows and columns in the heatmap have fixed size?
`cex`	A factor multiplicated to all font sizes.
`help`	Whether to print help message?
`file`	A path of the figure. The size of the figure is automatically calculated.
`res`	Resolution of the figure (only for png and jpeg).

Details

If `fix_size` is set to `TRUE`. The size of the whole plot can be obtained by:

```
size = dependency_heatmap(x, fix_size = TRUE)
```

where `size` is a numeric vector of length two which are the width and height of the whole heatmap. If `file` argument is set, the size of the figure is automatically calculated.

If there are no dependency packages stored in `x`, `NULL` is returned.

Value

A vector of two numeric values (in inches) that correspond to the width and height of the plot.

Examples

```
# See examples in `pkgndep()`.
```

`dependency_report` *HTML report for package dependency heaviness analysis*

Description

HTML report for package dependency heaviness analysis

Usage

```
dependency_report(...)
```

Arguments

`...`	Pass to `heaviness_report`.

Details

It is the same as `heaviness_report`.

Examples

```
# There is no example
NULL
```

`dependency_website` *Database of package dependency heaviness of all R packages*

Description

Database of package dependency heaviness of all R packages

Usage

```
dependency_website(version = pkgndep_opt$heaviness_db_version)
```

Arguments

`version`	Version of the heaviness database. See `pkgndep_opt`$heaviness_db_version.

Examples

```
if(interactive()) {
    dependency_website()
}
```

`downstream_dependency` *Get downstream dependency for a package*

Description

Get downstream dependency for a package

Usage

```
downstream_dependency(package, online = FALSE)
```

Arguments

`package`	Package name.
`online`	Whether use the newest package database directly from CRAN/Bioconductor or the pre-computated package database? The version of the pre-computated package database can be set via `pkgndep_opt`$heaviness_db_version.

Details

Downstream packages with relations of `Depends`, `Imports` and `LinkingTo` are retrieved.

Value

A data frame with all downstream packages.

Examples

```
## Not run:
downstream_dependency("ComplexHeatmap")

## End(Not run)
```

`get_all_functions_imported_to_children`

Get functions that are imported to its child packages

Description

Get functions that are imported to its child packages

Usage

```
get_all_functions_imported_to_children(package)
```

Arguments

`package`	Package name.

Details

The information is based on pre-computated results for a specific CRAN/Bioconductor snapshot. See `pkgndep`$heaviness_db_version for how to set the version of the snapshot.

Value

It returns a list of function names that are imported to every of its child packages.

Examples

```
## Not run:
get_all_functions_imported_to_children("circlize")

## End(Not run)
```

`gini_index` *Gini index*

Description

Gini index

Usage

```
gini_index(v)
```

Arguments

`v`	A numeric vector.

Examples

```
x = readRDS(system.file("extdata", "ComplexHeatmap_dep.rds", package = "pkgndep"))
gini_index(x$heaviness[x$which_required])
```

`heaviness` *Heaviness from parent packages*

Description

Heaviness from parent packages

Usage

```
heaviness(x, rel = FALSE, a = 10, only_strong_dep = FALSE)
```

Arguments

`x`	An object returned by pkgndep.
`rel`	Whether to return the absolute measure or the relative measure.
`a`	A constant added for calculating the relative measure.
`only_strong_dep`	Whether to only return the heaviness for strong parents.

Details

The heaviness from a parent package is calculated as follows: If package B is in the `Depends`/`Imports`/`LinkingTo` fields of package A, which means, package B is necessary for package A, denote `v1` as the total numbers of packages required for package A, and `v2` as the total number of required packages if moving package B to `Suggests` (which means, now B is not necessary for A). The absolute measure is simply `v1 - v2` and relative measure is `(v1 + a)/(v2 + a)`.

In the second scenario, if B is in the `Suggests`/`Enhances` fields of package A, now `v2` is the total number of required packages if moving B to `Imports`, the absolute measure is `v2 - v1` and relative measure is `(v2 + a)/(v1 + a)`.

Value

A numeric vector.

Examples

```
x = readRDS(system.file("extdata", "ComplexHeatmap_dep.rds", package = "pkgndep"))
heaviness(x)
heaviness(x, rel = TRUE)
```

`heaviness_database` *Database of package dependency heaviness of all R packages*

Description

Database of package dependency heaviness of all R packages

Usage

```
heaviness_database(version = pkgndep_opt$heaviness_db_version)
```

Arguments

`version`	Version of the heaviness database. See `pkgndep_opt`$heaviness_db_version.

Examples

```
if(interactive()) {
    heaviness_database()
}
```

`heaviness_from_upstream` *Heaviness from all upstream packages*

Description

Heaviness from all upstream packages

Usage

```
heaviness_from_upstream(package)
```

Arguments

`package`	A package name.

Details

It is calculated based on a specific CRAN/Bioconductor snapshot. The version is set via `pkgndep_opt`$heaviness_db_version.

Value

A named vector.

Examples

```
# There is no example
NULL
```

`heaviness_on_children` *Heaviness on all child packages*

Description

Heaviness on all child packages

Usage

```
heaviness_on_children(package, add_values_attr = FALSE, total = FALSE)
```

Arguments

`package`	A package name.
`add_values_attr`	Whether to include "values" attribute? Internally used.
`total`	Whether to return the total heaviness?

Details

It is calculated based on a specific CRAN/Bioconductor snapshot. The version is set via pkgndep_opt$heaviness_db_version.

Value

The value is the mean heaviness of the package on all its child packages.

Examples

```
## Not run:
heaviness_on_children("ComplexHeatmap")

## End(Not run)
```

---

heaviness_on_downstream

Heaviness on all downstream packages

Description

Heaviness on all downstream packages

Usage

```
heaviness_on_downstream(package, add_values_attr = FALSE, via = NULL,
    total = FALSE, internal = FALSE)
```

Arguments

`package`	A package name.
`add_values_attr`	Whether to include "values" attribute? Internally used.
`via`	Whether to only consider downstream packages via a intermediate package?
`total`	Whether to return the total heaviness?
`internal`	Whether to use internally calculated heaviness?

Details

It is calculated based on a specific CRAN/Bioconductor snapshot. The version is set via pkgndep_opt$heaviness_db_version.

Value

The value is the mean heaviness of the package on all its downstream packages. Denote `n` as the number of all its downstream packages, `k_i` as the number of required packages for package i, `v_1` as the total number of required packages for all downstream packages, i.e. `v_1 = sum_i^n {k_i}`. Denote `p_i` as the number of required packages if moving `package` to `Suggests`, and `v_2` as the total number of required packages, i.e. `v_1 = sum_i^n {p_i}`. The final heaviniss on downstream packages is `(v_1 - v_2)/n`.

Note since the interaction from `package` to its downstream packages may go through several intermediate packages, which means, the reduction of required packages for a downstream package might be joint effects from all its upstream packages, thus, to properly calculate the heaviness of a package to its downstream packages, we first make a copy of the package database and move `package` to `Suggests` for all packages which depends on `package`. Then for all downstream packages of `package`, dependency analysis by `pkgndep` is redone with the modified package database. Finally, the heaviness on downstream packages is collected and the mean heaviness is calculated.

Examples

```
## Not run:
heaviness_on_downstream("ComplexHeatmap")

## End(Not run)
```

`heaviness_report` *HTML report for package dependency heaviness analysis*

Description

HTML report for package dependency heaviness analysis

Usage

```
heaviness_report(pkg, file = NULL)
```

Arguments

`pkg`	An object from `pkgndep`.
`file`	The path of the html file. If it is not specified, the report will be automatically opened in the web browser.

Value

The path of the HTML file of the report.

Examples

```
if(interactive()) {
    x = readRDS(system.file("extdata", "ComplexHeatmap_dep.rds", package = "pkgndep"))
    heaviness_report(x)
}
```

`is_parent` *Test the parent-child relationship*

Description

Test the parent-child relationship

Usage

```
is_parent(parent, child, ...)
```

Arguments

`parent`	A vector of package names.
`child`	A single package name.
`...`	Pass to `parent_dependency`.

Value

A logical vector.

Examples

```
# There is no example
NULL
```

`is_upstream` *Test upstream-downstream relationship*

Description

Test upstream-downstream relationship

Usage

```
is_upstream(upstream, package, ...)
```

Arguments

`upstream`	A vector of package names.
`package`	A single package name.
`...`	Pass to `upstream_dependency`.

Value

A logical vector.

Examples

```
# There is no example
NULL
```

`loaded_packages` *Loaded packages*

Description

Loaded packages

Usage

```
loaded_packages(pkg, verbose = TRUE)
```

Arguments

`pkg`	A package name.
`verbose`	Whether to print messages.

Details

It loads pkg into a new R session and collects which other packages are loaded by parsing the output from `sessionInfo`.

Value

A data frame.

Examples

```
loaded_packages("ComplexHeatmap")
```

`load_all_pkg_dep` *Load dependency analysis results of all packages*

Description

Load dependency analysis results of all packages

Usage

```
load_all_pkg_dep(hash = TRUE)
```

Arguments

`hash`	Whether to convert the named list to a hash table by `hash`.

Details

It is calculated based on a specific CRAN/Bioconductor snapshot. The version is set via `pkgndep_opt`$heaviness_db_version.

Value

A list (as a hash table) of `pkgndep` objects where each element corresponds to the analysis on one package.

Examples

```
## Not run:
lt = load_all_pkg_dep()
length(lt)
head(names(lt))
lt[["ggplot2"]]

## End(Not run)
```

`load_from_heaviness_db` *Load pre-computed objects*

Description

Load pre-computed objects

Usage

```
load_from_heaviness_db(file)
```

Arguments

`file`	File name.

Details

The pathway of the file can be set via `pkgndep_opt`$db_file_template.

Internally used.

Examples

```
# There is no example
NULL
```

`load_heaviness_timeline` *Load heaviness statistics at all time points*

Description

Load heaviness statistics at all time points

Usage

```
load_heaviness_timeline()
```

Details

Used internally.

Value

A list of data frames.

Examples

```
# There is no example
NULL
```

`load_pkg_db` *Load package database*

Description

Load package database

Usage

```
load_pkg_db(lib = NULL, online = TRUE, db = NULL, verbose = TRUE)
```

Arguments

`lib`	Local library path. If the value is `NA`, only remote package database is used.
`online`	If the value is `TRUE`, it will directly use the newest package database file from CRAN/Bioconductor. If the value is `FALSE`, it uses the pre-computated package database on a specific CRAN/Bioconductor snapshot. The version of the pre-computated package database can be set via `pkgndep_opt`$heaviness_db_version.
`db`	A pre-computed pkg_db object.
`verbose`	Whetehr to print messages.

Details

It loads the package database from CRAN/Bioconductor and locally installed packages.

The database object internaly is cached for repeated use of other functions in this package.

Value

A pkg_db class object. See `reformat_db` for how to use the pkg_db object.

Examples

```
## Not run:
pkg_db = load_pkg_db(lib = NA)
pkg_db

## End(Not run)
```

`load_pkg_description` *Load DESCRIPTION files of all packages*

Description

Load DESCRIPTION files of all packages

Usage

```
load_pkg_description()
```

Details

It is calculated based on a specific CRAN/Bioconductor snapshot. The version is set via `pkgndep_opt`$heaviness_db_version.

Value

A list of character vectors.

Examples

```
## Not run:
lt = load_pkg_description()
lt[1:2]

## End(Not run)
```

`load_pkg_downstream_dependency_path_snapshot` *Load downstream dependency paths for all packages*

Description

Load downstream dependency paths for all packages

Usage

```
load_pkg_downstream_dependency_path_snapshot()
```

Details

It is calculated based on a specific CRAN/Bioconductor snapshot. The version is set via `pkgndep_opt`$heaviness_db_version.

Value

A list.

Examples

```
## Not run:
downstream_path_list = load_pkg_downstream_dependency_path_snapshot()
downstream_path_list[["ComplexHeatmap"]]

## End(Not run)
```

`load_pkg_namespace` *Load NAMESPACE files of all packages*

Description

Load NAMESPACE files of all packages

Usage

```
load_pkg_namespace()
```

Details

It is calculated based on a specific CRAN/Bioconductor snapshot. The version is set via `pkgndep_opt`$heaviness_db_version.

Value

A list of character vectors.

Examples

```
## Not run:
lt = load_pkg_namespace()
lt[1:2]

## End(Not run)
```

`load_pkg_stat_snapshot` *Load all package dependency statistics*

Description

Load all package dependency statistics

Usage

```
load_pkg_stat_snapshot()
```

Details

It is calculated based on a specific CRAN/Bioconductor snapshot. The version is set via `pkgndep_opt`$heaviness_db_version.

Value

A data frame.

Examples

```
## Not run:
df = load_pkg_stat_snapshot()
head(df)

## End(Not run)
```

`parent_dependency` *Get parent dependency for a package*

Description

Get parent dependency for a package

Usage

```
parent_dependency(package, fields = NULL, online = FALSE)
```

Arguments

`package`	Package name.
`fields`	Which fields in DESCRIPTION? Values should be in `Depends`, `Imports`, `LinkingTo`, `Suggests` and `Enhances`. The value can also be set to `strong` or `weak`.
`online`	Whether use the newest package database directly from CRAN/Bioconductor or the pre-computated package database? The version of the pre-computated package database can be set via `pkgndep_opt`$heaviness_db_version.

Value

A data frame with parent packages as well as their heaviness on `pacakge`. If `snapshot` is set to `FALSE`, heaviness on child packages is set to NA.

Examples

```
## Not run:
parent_dependency("ComplexHeatmap")

## End(Not run)
```

`pkgndep` *Package dependency analysis*

Description

Package dependency analysis

Usage

```
pkgndep(package, verbose = TRUE, online = TRUE, load = FALSE, parse_namespace = TRUE)
```

Arguments

`package`	Package name. The value can be 1. a CRAN/Bioconductor package, 2. an installed package, 3. a path of a local package, 4. URL of a GitHub repository.
`verbose`	Whether to show messages.
`online`	If the value is `TRUE`, it will directly use the newest package database file from CRAN/Bioconductor. Note the default Bioconductor version is the one corresponding to the current R version. If you want to use a higher bioc version, you need to set the global option `pkgndep_opt$bioc_version`. If the value of `online` is `FALSE`, a snapshot of the CRAN/Bioconductor package database will be used. The version of the package database snapshot can be via the global option `pkgndep_opt`$heaviness_db_version.
`load`	If the value is `TRUE`, the package is loaded into a fresh new R session and the function will check which upstream packages are also loaded into the R session. Note it is possible that an "Imports" package is not loaded or a "Suggests" package is loaded in the R session when loading package.
`parse_namespace`	Whether to also parse the NAMESPACE file. It is only used internally.

Value

A pkgndep object.

Examples

```
## Not run:
x = pkgndep("ComplexHeatmap")

## End(Not run)
# The `x` variable generated by `pkgndep()` is already saved in this package.
x = readRDS(system.file("extdata", "ComplexHeatmap_dep.rds", package = "pkgndep"))
x
dependency_heatmap(x)
```

`pkgndep_opt` *Global parameters for pkgndep*

Description

Global parameters for pkgndep

Usage

```
pkgndep_opt(..., RESET = FALSE, READ.ONLY = NULL, LOCAL = FALSE, ADD = FALSE)
```

Arguments

`...`	Arguments for the parameters, see "details" section
`RESET`	Reset to default values.
`READ.ONLY`	Please ignore.
`LOCAL`	Pllease ignore.
`ADD`	Please ignore.

Details

There are following parameters:

- `bioc_version` The bioconductor version. By default it is the version corresponding to the R version under use. Please note this option is only for switching between bioc release version and development version, while not for switching to very old bioc versions.
- `heaviness_db_version` The version of the heaviness database. The value can be the corresponding bioc version, the R version or the corresponding date for the bioc release. All supported values are in the object `ALL_BIOC_RELEASES`.

Examples

```
pkgndep_opt
```

`plot.pkgndep` *Make the dependency heatmap*

Description

Make the dependency heatmap

Usage

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

Arguments

`x`	An object from pkgndep.
`...`	Other arguments.

Details

Please use `dependency_heatmap` instead.

Examples

```
# There is no example
NULL
```

`print.pkgndep` *Print method*

Description

Print method

Usage

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

Arguments

`x`	An object from pkgndep.
`...`	Other arguments.

Value

No value is returned.

Examples

```
# See examples in `pkgndep()`.
```

`reformat_db` *Format the package database*

Description

Format the package database

Usage

```
reformat_db(db, version = NULL)
```

Arguments

`db`	A data frame returned from `available.packages` or `installed.packages`.
`version`	Version of the database, a self-defined text.

Details

It reformats the data frame of the package database into a `pkg_db` class object.

Value

A pkg_db class object. There are the following methods:

- `pkg_db$get_meta(package,field=NULL)` `field` can take values in "Package", "Version" and "Repository".
- `pkg_db$get_dependency_table(package)` Get the dependency table.
- `pkg_db$get_rev_dependency_table(package)` Get the reverse dependency table.
- `pkg_db$package_dependencies(package,recursive=FALSE,reverse=FALSE,which="strong",simplify=FALSE)` All the arguments are the same as in `package_dependencies`. Argument `simplify` controls whether to return a data frame or a simplied vector.

Examples

```
## Not run:
db = available.packages()
db2 = reformat_db(db)

# a pkg_db object generated on 2021-10-28 can be loaded by load_pkg_db()
db2 = load_pkg_db(online = FALSE)
db2
db2$get_meta("ComplexHeatmap")
db2$get_dependency_table("ComplexHeatmap")
db2$get_rev_dependency_table("ComplexHeatmap")
db2$package_dependencies("ComplexHeatmap")
db2$package_dependencies("ComplexHeatmap", recursive = TRUE)

## End(Not run)
```

`required_dependency_packages` *Required dependency packages*

Description

Required dependency packages

Usage

```
required_dependency_packages(x, all = FALSE)
```

Arguments

`x`	An object from `pkgndep`.
`all`	Whether to include the packages required if also including packages from "Suggests"/"Enhances" field.

Details

The function returns all upstream packages.

Value

A vector of package names.

Examples

```
## Not run:
x = readRDS(system.file("extdata", "ComplexHeatmap_dep.rds", package = "pkgndep"))
required_dependency_packages(x)

## End(Not run)
```

`upstream_dependency` *Get upstream dependency for a package*

Description

Get upstream dependency for a package

Usage

```
upstream_dependency(package, online = FALSE)
```

Arguments

`package`	Package name.
`online`	Whether use the newest package database directly from CRAN/Bioconductor or the pre-computated package database? The version of the pre-computated package database can be set via `pkgndep_opt`$heaviness_db_version.

Details

Upstream packages with relations of "Depends", "Imports" and "LinkingTo" are retrieved.

Value

A data frame with all upstream packages.

Examples

```
## Not run:
upstream_dependency("ComplexHeatmap")

## End(Not run)
```

Index