

# Package: hasseDiagram (via r-universe)

January 15, 2025

**Type** Package

**Title** Drawing Hasse Diagram

**Version** 0.2.0

**Date** 2021-06-10

**Author** Krzysztof Ciomek

**Maintainer** Krzysztof Ciomek <k.ciomek@gmail.com>

**URL** <https://github.com/kciomek/hasseDiagram>

**Depends** Rgraphviz (>= 2.6.0), grid (>= 3.0.2), graph

**Imports** methods

**Description** Drawing Hasse diagram - visualization of transitive reduction of a finite partially ordered set.

**License** MIT + file LICENSE

**RoxygenNote** 7.1.1

**NeedsCompilation** no

**Date/Publication** 2021-06-10 06:10:02 UTC

**Additional\_repositories** <https://cranhaven.r-universe.dev>

**Repository** <https://cranhaven.r-universe.dev>

**RemoteUrl** <https://github.com/cranhaven/cranhaven.r-universe.dev>

**RemoteRef** package/hasseDiagram

**RemoteSha** deddca4a7ea8d9d1c03c53b18305896cdeaf5025

**RemoteSubdir** hasseDiagram

## Contents

hasseDiagram-package . . . . .	2
generateRandomData . . . . .	2
hasse . . . . .	3

<b>Index</b>	<b>5</b>
--------------	----------

---

hasseDiagram-package    *Drawing Hasse Diagram*

---

### Description

Drawing Hasse diagram - visualization of transitive reduction of a finite partially ordered set.

### Details

Package: hasseDiagram  
Type: Package  
Version: 0.2.0  
Date: 2021-06-10  
License: MIT

### Author(s)

Krzysztof Ciomek  
Maintainer: Krzysztof Ciomek <k.ciomek@gmail.com>

### See Also

[hasse](#)

---

generateRandomData    *Generate random data for hasse function*

---

### Description

This function generates random data for [hasse](#) function.

### Usage

```
generateRandomData(nrNodes, minGraphs = 1, density = 0.5)
```

### Arguments

nrNodes	Numer of nodes ( $0 < \text{nrNodes}$ ).
minGraphs	Minimal number of graphs to generate ( $0 < \text{minGraphs} \leq \text{nrNodes}$ ).
density	Value which determines number of edges and shape of graphs (density in $[0.0; 1.0]$ ).

**Value**

nrNodes x nrNodes matrix.

**Examples**

```
data0_0 <- generateRandomData(15, 2, 0.0)
data0_5 <- generateRandomData(15, 2, 0.5)
data1_0 <- generateRandomData(15, 2, 1.0)
```

```
hasse(data0_0)
hasse(data0_5)
hasse(data1_0)
```

---

hasse	<i>Draw Hasse diagram</i>
-------	---------------------------

---

**Description**

This function draws Hasse diagram – visualization of transitive reduction of a finite partially ordered set.

**Usage**

```
hasse(data, labels = c(), parameters = list())
```

**Arguments**

- |            |  |
|------------|--|
| data       | $n \times n$ matrix, which represents partial order of $n$ elements in set. Each cell $[i, j]$ has value TRUE iff $i$ -th element precedes $j$ -th element.  |
| labels     | Vector containing labels of elements. If missing or NULL then data row names will be used as labels. If rownames(data) are not present, the labels will be generated as ('a' + element index).   |
| parameters | List with named elements: <ul style="list-style-type: none"> <li>• arrow – direction of arrows: "forward", "backward", "both" or "none" (default "forward"),</li> <li>• cluster – whether to cluster elements which have the same parents and children and are connected all to all (see first commented example) (default TRUE),</li> <li>• clusterMerge – merge clustered nodes within single node frame (default FALSE),</li> <li>• clusterNonAdjacent – to allow clustering elements that are not mutually adjacent (default FALSE),</li> <li>• edgeColor – edge color, from colors() (default "black"),</li> <li>• newpage – whether to call grid.newpage() before drawing (default TRUE),</li> <li>• nodeColor – node frame color, from colors() (default "black"),</li> </ul> |

- `margin` – node margins, a list with 4 numerical items: "tb" for top-bottom margin, "rl" for right-left margin, "otb" and "orl" for outer margin when multiple labels are present,
- `shape` – shape of diagram nodes: "roundrect", "rect" or "none" (default "roundrect"),
- `transitiveReduction` – whether to perform transitive reduction (default TRUE).

## Examples

```
randomData <- generateRandomData(15, 2, 0.5)
hasse(randomData)

# Clustering example
data <- matrix(data = FALSE, ncol = 4, nrow = 4)
data[1, 2] = data[1, 3] = data[2, 4] = data[3, 4] = TRUE
data[2, 3] = data[3, 2] = TRUE
hasse(data, c(), list(cluster = TRUE))
hasse(data, c(), list(cluster = FALSE))

# Hasse to pdf example
# randomData <- generateRandomData(15, 2, 0.5)
# pdf("path-for-diagram.pdf")
# hasse(randomData, NULL, list(newpage = FALSE))
# dev.off()
```

# Index

\* **hasse diagram finite partially ordered set**

hasseDiagram-package, [2](#)

generateRandomData, [2](#)

hasse, [2](#), [3](#)

hasseDiagram (hasseDiagram-package), [2](#)

hasseDiagram-package, [2](#)