# Package 'hierformR'

August 29, 2016

**Type** Package

**Title** Analysis of Dynamics Hierarchy Formation

**Version** 0.1.0

**Maintainer** James P. Curley <jc3181@columbia.edu>

**Description** Determine paths and states that social networks develop over time to form social hierarchies. Based upon algorithms described in W. Brent Lindquist & Ivan D. Chase (2009) <DOI:10.1007/s11538-008-9371-9>.

**Depends** R (>= 3.1.0)

**Imports** igraph, utils

**License** GPL-3

**LazyData** TRUE

**RoxygenNote** 5.0.0

**NeedsCompilation** no

**Author** James P. Curley [aut, cre], Ivan D. Chase [aut]

**Repository** CRAN

**Date/Publication** 2016-06-10 18:27:00

## R topics documented:

---

`addstates` *Add State ids and classes*

---

## Description

Add State ids and classes

## Usage

```
addstates(df)
```

## Arguments

| | |
|---|---|
| `df` | a dataframe with two variables, winners and losers |

## Value

the df with two extra rows containing state id and class

## Examples

```
df<-data.frame(winner=c(1,2,3,1,2,3,2,1,2,3,3,1,2,3,4,3,1,3,2,1,1,1,1,2,2),
loser=c(2,4,4,4,3,2,1,3,3,4,4,2,3,2,3,4,2,4,3,3,3,2,2,4,3)
)
addstates(df)
```

---

`csf` *Determine class-stability factor (CSF)*

---

## Description

Determine class-stability factor (CSF)

## Usage

```
csf(df)
```

## Arguments

| | |
|---|---|
| `df` | either a two-column dataframe of winners and losers or a dataframe produced by ‘addstates‘ with columns winner, loser, id and class |

## Value

a named vector of state ids with the CSF for all link numbers

## Examples

```
df<-data.frame(winner=c(1,2,3,1,2,3,2,1,2,3,3,1,2,3,4,3,1,3,2,1,1,1,1,2,2),
loser=c(2,4,4,4,3,2,1,3,3,4,4,2,3,2,3,4,2,4,3,3,3,2,2,4,3)
)
csf(df)

csf(addstates(wldf()))
```

---

`csf_n` *Determine class-stability factor (CSF) for specific number of links*

---

## Description

Determine class-stability factor (CSF) for specific number of links

## Usage

```
csf_n(df, links = 4)
```

## Arguments

| | |
|---|---|
| `df` | a dataframe produced by ‘addstates‘ with columns winner, loser, id and class |
| `links` | the number of links to determine the CSF for |

## Value

a named vector of state ids with the CSF for defined number of links

## Examples

```
csf_n(addstates(wldf()),links=6)
```

---

`delete_isolates` *Delete isolated nodes from network*

---

## Description

Delete isolated nodes from network

## Usage

```
delete_isolates(q)
```

## Arguments

| | |
|---|---|
| `q` | an igraph object |

## Value

an igraph object containing only connected nodes

## Examples

```
g=igraph::graph.edgelist(cbind(a=c(1,2,3,4), b=c(3,1,1,4)),directed=TRUE)
delete_isolates(g)
```

---

`lastints` *Get last interaction between each pair of nodes*

---

## Description

Get last interaction between each pair of nodes

## Usage

```
lastints(df)
```

## Arguments

| | |
|---|---|
| `df` | a dataframe with two variables, winners and losers |

## Value

a list with each element being the last interaction for each pair of nodes up to that row of the dataframe

## Examples

```
df<-data.frame(winner=c(1,2,3,1,2,3,2,1,2,3,3,1,2,3,4,3,1,3,2,1,1,1,1,2,2),
loser=c(2,4,4,4,3,2,1,3,3,4,4,2,3,2,3,4,2,4,3,3,3,2,2,4,3)
)
lastints(df)
```

## lastnet *Get last interaction between each pair of nodes*

### Description

Get last interaction between each pair of nodes

### Usage

```
lastnet(df)
```

### Arguments

| | |
|---|---|
| df | a dataframe with two variables, winners and losers |

### Value

a list with each element being the network object derived from the last interaction for each pair of nodes up to that row of the dataframe

### Examples

```
df<-data.frame(winner=c(1,2,3,1,2,3,2,1,2,3,3,1,2,3,4,3,1,3,2,1,1,1,1,2,2),
loser=c(2,4,4,4,3,2,1,3,3,4,4,2,3,2,3,4,2,4,3,3,3,2,2,4,3)
)
lastnet(df)
```

## netchar *Get characteristics of a directed graph*

### Description

Get characteristics of a directed graph

### Usage

```
netchar(g)
```

### Arguments

| | |
|---|---|
| g | a unweighted directed igraph object. |

### Value

a single row dataframe with characteristics of the graph

## Examples

```
g=igraph::graph.edgelist(cbind(a=c(1,2,3,3), b=c(4,1,1,2)),directed=TRUE)
netchar(g)
```

---

`rowmatch` *Match rows of dataframes*

---

## Description

Match rows of dataframes

## Usage

```
rowmatch(v1, v2, nomatch = NA)
```

## Arguments

| | |
|---|---|
| v1 | a dataframe or matrix |
| v2 | a row of a dataframe or matrix or a vector of length equal to ncol(v1) |
| nomatch | how to represent non matching indices of rows |

## Value

a vector with a 1 in the index of matched rows

## Examples

```
v1 <- data.frame(A=c(1,2,3),B=c(2,3,3),C=c(3,3,2),D=c(1,1,1))
v2 <- data.frame(A=3,B=3,C=2,D=1)
rowmatch(v1,v2,nomatch=NA)
```

---

`stateclass` *Find State Class*

---

## Description

Find State Class

## Usage

```
stateclass(x)
```

## Arguments

| | |
|---|---|
| x | a vector of network characteristics generated by get_statespace function |

## Value

the state class corresponding to the network characteristics

## Examples

```
g=igraph::graph.edgelist(cbind(a=c(1,2,3,3), b=c(4,1,1,2)),directed=TRUE)
gs=netchar(g)
stateclass(gs)
```

---

`stateclasses` *State classes of 41 network states*

---

## Description

Vector defining the classes of 41 network states

## Usage

```
stateclasses
```

## Format

A vector of length 41:

---

`stateid` *Find State ID*

---

## Description

Find State ID

## Usage

```
stateid(x)
```

## Arguments

| | |
|---|---|
| x | a vector of network characteristics generated by get_statespace function |

## Value

the state id corresponding to the network characteristics

## Examples

```
g=igraph::graph.edgelist(cbind(a=c(1,2,3,3), b=c(4,1,1,2)),directed=TRUE)
gs=netchar(g)
stateid(gs)
```

`states` *Characteristics of 41 network states for 4 node graphs*

## Description

Dataframe showing the transitivity and intransitivity of each of the four triads, the number of nodes, the number of edges, the number of other individuals dominated by the top-ranked individual, the mean distance of the network and the maximum difference between out-degree and in-degrees, in a four node network. Also gives the class of each state.

## Usage

```
states
```

## Format

A data frame with 41 rows and 14 variables:

**X003** Number of unconnected nodes

**X012** Number of triads with one single directed edge

**X021D** Number of triads with 2 edges going down

**X021U** Number of triads with 2 edges going up

**X021C** Number of triads with 2 edges that could be transitive/intransitive

**X030T** Number of transitive triads

**X030C** Number of intransitive triads

**maxdom** Number of individuals dominated by top ranked individual

**noedges** Number of edges in network

**nonodes** Number of nodes in network

**distance** Mean distance of the network

**degreediff** Max difference between each nodes out-degree and in-degree

**id** The id of the network

**class** The class of the network

wldf *Generate random winner-loser dataframe*

## Description

Generate random winner-loser dataframe

## Usage

```
wldf(ints = 100, actors = 4, intprob = NULL)
```

## Arguments

| | |
|---|---|
| ints | number of interactions |
| actors | number of individuals |
| intprob | vector of interaction probabilities of same length as ‘actors‘ |

## Value

a dataframe with two variables, winner and loser

## Examples

```
df<-data.frame(winner=c(1,2,3,1,2,3,2,1,2,3,3,1,2,3,4,3,1,3,2,1,1,1,1,2,2),
loser=c(2,4,4,4,3,2,1,3,3,4,4,2,3,2,3,4,2,4,3,3,3,2,2,4,3)
)
wldf()
wldf(ints=30, actors=3, intprob=c(0.8,0.1,0.1))
```

# Index