R igraph manual pages

Use this if you are using igraph from R

Various methods for creating graphs

Description

These method can create various (mostly regular) graphs: empty graphs, graphs with the given edges, graphs from adjacency matrices, star graphs, lattices, rings, trees.

Usage

```
graph.empty(n=0, directed=TRUE)
graph(edges, n=max(edges), directed=TRUE)
graph.star(n, mode = c("in", "out", "mutual", "undirected"), center = 1)
graph.lattice(dimvector = NULL, length = NULL, dim = NULL, nei = 1,
              directed = FALSE, mutual = FALSE, circular = FALSE, ...)
graph.ring(n, directed = FALSE, mutual = FALSE, circular=TRUE)
graph.tree(n, children = 2, mode=c("out", "in", "undirected"))
graph.full(n, directed = FALSE, loops = FALSE)
graph.full.citation(n, directed = TRUE)
graph.atlas(n)
graph.edgelist(el, directed=TRUE)
graph.extended.chordal.ring(n, w)
```

Arguments

edges	Numeric vector defining the edges, the first edge points from the first element to the second, the second edge from the third to the fourth, etc.
directed	Logical, if TRUE a directed graph will be created. Note that for while most constructors the default is TRUE, for graph.lattice and graph.ring it is FALSE. For graph.star the mode argument should be used for creating an undirected graph.
n	The number of vertices in the graph for most functions. For graph this parameter is ignored if there is a bigger vertex id in edges . This means that for this function it is safe to supply zero here if the vertex with the largest id is not an isolate. For graph.atlas this is the number (id) of the graph to create.
mode	For graph.star it defines the direction of the edges, in : the edges point *to* the center, out : the edges point *from* the center, mutual : a directed star is created with mutual edges, undirected : the edges are undirected. For igraph.tree this parameter defines the direction of the edges. out indicates that the edges point from the parent to the children, in indicates that they point from the children to their parents, while undirected creates an undirected graph.

`center`	For `graph.star` the center vertex of the graph, by default the first vertex.
`dimvector`	A vector giving the size of the lattice in each dimension, for `graph.lattice`.
`nei`	The distance within which (inclusive) the neighbors on the lattice will be connected. This parameter is not used right now.
`mutual`	Logical, if TRUE directed lattices will be mutually connected.
`circular`	Logical, if TRUE the lattice or ring will be circular.
`length`	Integer constant, for regular lattices, the size of the lattice in each dimension.
`dim`	Integer constant, the dimension of the lattice.
`children`	Integer constant, the number of children of a vertex (except for leafs) for `graph.tree`.
`loops`	If TRUE also loops edges (self edges) are added.
`graph`	An object.
`el`	An edge list, a two column matrix, character or numeric. See details below.
`w`	A matrix which specifies the extended chordal ring. See details below.
`...`	Currently ignored.

Details

All these functions create graphs in a deterministic way.

`graph.empty` is the simplest one, this creates an empty graph.

`graph` creates a graph with the given edges.

`graph.star` creates a star graph, in this every single vertex is connected to the center vertex and nobody else.

`graph.lattice` is a flexible function, it can create lattices of arbitrary dimensions, periodic or unperiodic ones. It has two forms. In the first form you only supply `dimvector`, but not `length` and `dim`. In the second form you omit `dimvector` and supply `length` and `dim`.

`graph.ring` is actually a special case of `graph.lattice`, it creates a one dimensional circular lattice.

`graph.tree` creates regular trees.

`graph.full` simply creates full graphs.

`graph.full.citation` creates a full citation graph. This is a directed graph, where every i->j edge is present if and only if j<i. If `directed=FALSE` then the graph is just a full graph.

`graph.atlas` creates graphs from the book An Atlas of Graphs by Roland C. Read and Robin J. Wilson. The atlas contains all undirected graphs with up to seven vertices, numbered from 0 up to 1252. The graphs are listed:

1. in increasing order of number of nodes;
2. for a fixed number of nodes, in increasing order of the number of edges;

3. for fixed numbers of nodes and edges, in increasing order of the degree sequence, for example 111223 < 112222;
4. for fixed degree sequence, in increasing number of automorphisms.

`graph.edgelist` creates a graph from an edge list. Its argument is a two-column matrix, each row defines one edge. If it is a numeric matrix then its elements are interpreted as vertex ids. If it is a character matrix then it is interpreted as symbolic vertex names and a vertex id will be assigned to each name, and also a `name` vertex attribute will be added.

`graph.extended.chordal.ring` creates an extended chordal ring. An extended chordal ring is regular graph, each node has the same degree. It can be obtained from a simple ring by adding some extra edges specified by a matrix. Let p denote the number of columns in the 'W' matrix. The extra edges of vertex `i` are added according to column `i mod p` in 'W'. The number of extra edges is the number of rows in 'W': for each row `j` an edge `i->i+w[ij]` is added if `i+w[ij]` is less than the number of total nodes. See also Kotsis, G: Interconnection Topologies for Parallel Processing Systems, PARS Mitteilungen 11, 1-6, 1993.

Value

Every function documented here returns a `graph` object.

Author(s)

Gabor Csardi csardi.gabor@gmail.com (mailto:csardi.gabor@gmail.com)

See Also

`graph.adjacency (graph.adjacency.html)` to create graphs from adjacency matrices, `graph.formula (graph.formula.html)` for a handy way to create small graphs, `graph.data.frame (graph.data.frame.html)` for an easy way to create graphs with many edge/vertex attributes.

Examples

```
g1 <- graph.empty()
g2 <- graph( c(1,2,2,3,3,4,5,6), directed=FALSE )
g5 <- graph.star(10, mode="out")
g6 <- graph.lattice(c(5,5,5))
g7 <- graph.lattice(length=5, dim=3)
g8 <- graph.ring(10)
g9 <- graph.tree(10, 2)
g10 <- graph.full(5, loops=TRUE)
g11 <- graph.full.citation(10)
g12 <- graph.atlas(sample(0:1252, 1))
el <- matrix( c("foo", "bar", "bar", "foobar"), nc=2, byrow=TRUE)
g13 <- graph.edgelist(el)
g15 <- graph.extended.chordal.ring(15, matrix(c(3,12,4,7,8,11), nr=2))
```

[Package *igraph* version 0.7.1 Index (00Index.html)]