Package 'centiserve'

July 15, 2017

Type Package

Title Find Graph Centrality Indices

Version 1.0.0

Depends igraph (>= 0.7.1), Matrix (>= 1.1-4)

Date 2017-07-15

Author Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

Maintainer Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

Description Calculates centrality indices additional to the 'igraph' package centrality functions.

License GPL (>= 2)

Suggests expm (>= 0.99-1.1), linkcomm (>= 1.0-11)

URL http://www.centiserver.org/

BugReports http://www.centiserver.org/?q1=contact

NeedsCompilation no

Repository CRAN

Date/Publication 2017-07-15 09:34:41 UTC

R topics documented:

`centiserve-package` *Functions to find graph centrality indices*

Description

Find centrality indices (measures) additional to the igraph package centrality functions. The centiserve is a part of www.CentiServer.org project which is a comprehensive centrality measures resource and a web based application for finding graph centrality measures.

Details

Package:	centiserve
Type:	Package
Version:	1.0.0
Date:	2014-12-30
License:	GPL (>= 2)

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>
Adapted algorithms and sources are referenced in function document.

Maintainer: Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Csardi G, Nepusz T: The igraph software package for complex network research, InterJournal, Complex Systems 1695. 2006. http://igraph.org

Adapted algorithms and sources are referenced in function document.

`averagedis` *Find the average distance of a node*

Description

This function return average distance of a node in a strongly connected and loop free graph.

Usage

```
averagedis(graph, vids = V(graph), mode = c("all", "out", "in"),
  weights = NULL)
```

Arguments

`graph`	The input graph as igraph object
`vids`	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
`mode`	Character constant, gives whether the shortest paths to or from the given vertices should be calculated for directed graphs. If out then the shortest paths from the vertex, if in then to it will be considered. If all, the default, then the corresponding undirected graph will be used, ie. not directed paths are searched. This argument is ignored for undirected graphs.
`weights`	Possibly a numeric vector giving edge weights. If this is NULL, the default, and the graph has a weight edge attribute, then the attribute is used. If this is NA then no weights are used (even if the graph has a weight attribute).

Details

Average distance of node u to the rest of nodes in the net defined as:

$$C_u = \frac{\sum_{w \in V} dis(u, w)}{n - 1}$$

It is invers of closeness centrality.
More detail at Average Distance

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

del Rio, Gabriel, Dirk Koschutzki, and Gerardo Coello. "How to identify essential genes from molecular networks?." BMC systems biology 3.1 (2009): 102.

Examples

```
g <- graph(c(1,2,2,3,3,4,4,2), directed=FALSE)
averagedis(g)
```

---

barycenter *Find the barycenter centrality score*

Description

Barycenter scores are calculated as 1 / (total distance from vertex v to all other vertices) in a strongly connected network.

Usage

```
barycenter(graph, vids = V(graph), mode = c("all", "out", "in"),
  weights = NULL)
```

Arguments

graph	The input graph as igraph object
vids	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
mode	Character constant, gives whether the shortest paths to or from the given vertices should be calculated for directed graphs. If out then the shortest paths from the vertex, if in then to it will be considered. If all, the default, then the corresponding undirected graph will be used, ie. not directed paths are searched. This argument is ignored for undirected graphs.
weights	Possibly a numeric vector giving edge weights. If this is NULL, the default, and the graph has a weight edge attribute, then the attribute is used. If this is NA then no weights are used (even if the graph has a weight attribute).

Details

There are 2 types of distance centrality scores, Closeness Centrality and Barycenter Centrality. Barycenter Centrality for vertex v defined as:

$$1/(totaldistancefromvtoallothervertices)$$

Closeness scores are calculated using the formula $1/(averagedistancefromvertexvtoallothervertices)$ and Barycenter scores are calculated as $1/(totaldistancefromvertexvtoallothervertices)$.
More detail at Barycenter Centrality

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Viswanath, Meghana. Ontology-based automatic text summarization. Diss. University of Georgia, 2009.

Examples

```
g <- graph(c(1,2,2,3,3,4,4,2), directed=FALSE)
barycenter(g)
```

`bottleneck` *Find the BottleNeck centrality score*

Description

BottleNeck Centrality for vertex v defined as:

$$BN(v) = \sum_{s \in v} P_s(v)$$

Let T_s be a shortest path tree rooted at node s. $P_s(v) = 1$ if more than $|V(Ts)|/4$ paths from node s to other nodes in T_s meet at the vertex v, otherwise $P_s(v) = 0$.

Usage

```
bottleneck(graph, vids = V(graph), mode = c("all", "out", "in"))
```

Arguments

graph	The input graph as igraph object
vids	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
mode	Character constant, gives whether the shortest paths to or from the given vertices should be calculated for directed graphs. If out then the shortest paths from the vertex, if in then to it will be considered. If all, the default, then the corresponding undirected graph will be used, ie. not directed paths are searched. This argument is ignored for undirected graphs.

Details

For each node v in the graph, construct a tree T_v of shortest paths from that node to all other nodes in the graph. For a node v, n_v is the number of nodes that are directly or indirectly connected to node v (i.e. the tree T_v contains n_v nodes). So extract all nodes w on the above defined tree T_v of shortest paths from node v, such that more than $n_v/4$ paths from v to other nodes in the tree meet at node w. Nodes w extracted in this way represent 'bottle necks' of the shortest path tree T_v rooted at node v, since at least $n_v/4$ paths of the $n_v - node$ tree T_v 'meet' at w.
More detail at BottleNeck

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Przulj, N., Dennis A. Wigle, and Igor Jurisica. "Functional topology in a network of protein interactions." Bioinformatics 20.3 (2004): 340-348.

Examples

```
g <- graph(c(1,2,2,3,3,4,4,2))
bottleneck(g)
```

---

centroid *Find the centroid value of graph vertices*

Description

Centroid value C_cen(v) for node v defined as:

$$C_{cen}(v) := min f(v,w) : w \in Vv$$

where $f(v,w) := \gamma_v(w) - \gamma_w(v)$, and $\gamma_v(w)$ is the number of vertex closer to v than to w.

Usage

```
centroid(graph, vids = V(graph), mode = c("all", "out", "in"),
  weights = NULL)
```

Arguments

graph	The input graph as igraph object
vids	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
mode	Character constant, gives whether the shortest paths to or from the given vertices should be calculated for directed graphs. If out then the shortest paths from the vertex, if in then to it will be considered. If all, the default, then the corresponding undirected graph will be used, ie. not directed paths are searched. This argument is ignored for undirected graphs.
weights	Possibly a numeric vector giving edge weights. If this is NULL, the default, and the graph has a weight edge attribute, then the attribute is used. If this is NA then no weights are used (even if the graph has a weight attribute).

Details

Centroid value computed by focusing the calculus on couples of nodes (v, w) and systematically counting the nodes that are closer (in term of shortest path) to v or to w. The calculus proceeds by comparing the node distance from other nodes with the distance of all other nodes from the others, such that a high centroid value indicates that a node v is much closer to other nodes. Thus, the centroid value provides a centrality index always weighted with the values of all other nodes in the graph.
More detail at Centroid value

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

Algorithm adapted from CentiLib (Grabler, Johannes, 2012).

References

Scardoni, Giovanni, Michele Petterlini, and Carlo Laudanna. "Analyzing biological network parameters with CentiScaPe." Bioinformatics 25.21 (2009): 2857-2859.

Grabler, Johannes, Dirk Koschutzki, and Falk Schreiber. "CentiLib: comprehensive analysis and exploration of network centralities." Bioinformatics 28.8 (2012): 1178-1179.

Examples

```
g <- graph(c(1,2,2,3,3,4,4,2), directed=FALSE)
centroid(g)
```

`closeness.currentflow` *Find current-flow closeness centrality*

Description

Current-flow closeness centrality is defined by:

$$C_{cc}(s) = \frac{n-1}{\sum_{s \neq t} p_{st}(s) - p_{st}(t)} forall : s \in V$$

where $(n-1)$ is a normalizing factor, $p_{st}(s)$ is the absolute electrical potential of vertex s based on the electrical current supply from vertex s to vertex t, and $p_{st}(s) - p_{st}(t)$ corresponds to the effective resistance typically measured as voltage, which can be interpreted as an alternative measure of distance between s and t.

Usage

```
closeness.currentflow(graph, vids = V(graph), weights = NULL)
```

Arguments

`graph`	The input graph as igraph object
`vids`	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
`weights`	Possibly a numeric vector giving edge weights. If this is NULL, the default, and the graph has a weight edge attribute, then the attribute is used. If this is NA then no weights are used (even if the graph has a weight attribute).

Details

The closeness index based on shortest paths can also be transformed to a measure based on electrical current. For the electrical current model set, Brandes et al. developed an alternative measure of the distance between two vertices s and t, which is defined as the difference of their electrical potentials. More detail at Current-Flow Closeness Centrality

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

Note: This implementation is based on Daniel Fleischer's implementation for yFiles and JMP which convert to java implementation for CentiLib by Johannes Graessler and Dirk Koschuetzki.

References

Brandes, Ulrik, and Daniel Fleischer. Centrality measures based on current flow. Springer Berlin Heidelberg, 2005.

Grabler, Johannes, Dirk Koschutzki, and Falk Schreiber. "CentiLib: comprehensive analysis and exploration of network centralities." Bioinformatics 28.8 (2012): 1178-1179.

Examples

```
g <- graph(c(1,2,2,3,3,4,4,2), directed=FALSE)
closeness.currentflow(g)
```

`closeness.freeman` *Find the closeness centrality in a strongly connected graph*

Description

Freeman closeness centrality defined as:

$$\frac{1}{\sum_{i \neq v} d(v,i)}$$

Usage

```
closeness.freeman(graph, vids = V(graph), mode = c("all", "out", "in"),
  weights = NULL, normalized = FALSE)
```

Arguments

`graph`	The input graph as igraph object
`vids`	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
`mode`	Character string, defined the types of the paths used for measuring the distance in directed graphs. 'in' measures the paths to a vertex, 'out' measures paths from a vertex, all uses undirected paths. This argument is ignored for undirected graphs.
`weights`	Possibly a numeric vector giving edge weights. If this is NULL, the default, and the graph has a weight edge attribute, then the attribute is used. If this is NA then no weights are used (even if the graph has a weight attribute).
`normalized`	Logical scalar, whether to calculate the normalized score.

Details

Because closeness is infinite if there is no path between two vertex so freeman closeness require a strongly connected graph. In igraph if there is no (directed) path between vertex v and i then the total number of vertices is used in the formula instead of the path length.
More detail at Closeness Centrality

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

Use igraph package closeness function.

References

Freeman, Linton C. "Centrality in social networks conceptual clarification." Social networks 1.3 (1979): 215-239.

Examples

```
g <- graph(c(1,2,2,3,3,4,4,2), directed=FALSE)
closeness.freeman(g)
```

`closeness.latora` *Find the variant (Latora) closeness centrality in a disconnected graph*

Description

Variant (Latora) closeness centrality defined as:

$$\sum_{i \neq v} \frac{1}{d(v,i)}$$

Usage

```
closeness.latora(graph, vids = V(graph), mode = c("all", "out", "in"),
  weights = NULL, normalized = FALSE)
```

Arguments

`graph`	The input graph as igraph object
`vids`	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
`mode`	Character constant, gives whether the shortest paths to or from the given vertices should be calculated for directed graphs. If out then the shortest paths from the vertex, if in then to it will be considered. If all, the default, then the corresponding undirected graph will be used, ie. not directed paths are searched. This argument is ignored for undirected graphs.
`weights`	Possibly a numeric vector giving edge weights. If this is NULL, the default, and the graph has a weight edge attribute, then the attribute is used. If this is NA then no weights are used (even if the graph has a weight attribute).
`normalized`	Logical scalar, whether to calculate the normalized score.

Details

This variant (sum of inversed distances to all other nodes instead of the inversed of the sum of distances to all other nodes) applicable to both connected and unconnected graphs.
More detail at Closeness Centrality

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Latora V., Marchiori M., Efficient behavior of small-world networks, Physical Review Letters, V. 87, p. 19, 2001.

Opsahl, Tore, Filip Agneessens, and John Skvoretz. "Node centrality in weighted networks: Generalizing degree and shortest paths." Social Networks 32.3 (2010): 245-251.

Examples

```
g <- graph(c(1,2,2,3,3,4,4,2))
closeness.latora(g)
```

---

closeness.residual *Find the residual closeness centrality*

Description

Residual closeness centrality defined as:

$$C_i = \sum_{j \neq i} \frac{1}{2^{d(i,j)}}$$

Usage

```
closeness.residual(graph, vids = V(graph), mode = c("all", "out", "in"),
  weights = NULL)
```

Arguments

graph	The input graph as igraph object
vids	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.

mode	Character constant, gives whether the shortest paths to or from the given vertices should be calculated for directed graphs. If out then the shortest paths from the vertex, if in then to it will be considered. If all, the default, then the corresponding undirected graph will be used, ie. not directed paths are searched. This argument is ignored for undirected graphs.
weights	Possibly a numeric vector giving edge weights. If this is NULL, the default, and the graph has a weight edge attribute, then the attribute is used. If this is NA then no weights are used (even if the graph has a weight attribute).

Details

This function calculate closeness of a vertex as Dangalchev defination.
More detail at Residual Closeness Centrality

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Dangalchev, Chavdar. "Residual closeness in networks." Physica A: Statistical Mechanics and its Applications 365.2 (2006): 556-564.

Examples

```
g <- graph(c(1,2,2,3,3,4,4,2))
closeness.residual(g)
```

---

closeness.vitality *Find the closeness vitality centrality in a strongly connected graph*

Description

Closeness vitality of a node is the change in the sum of distances between all node pairs when excluding that node.

Usage

```
closeness.vitality(graph, vids = V(graph), mode = c("all", "out", "in"),
  weights = NULL)
```

Arguments

graph	The input graph as igraph object
vids	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
mode	Character string, defined the types of the paths used for measuring the distance in directed graphs. 'in' measures the paths to a vertex, 'out' measures paths from a vertex, all uses undirected paths. This argument is ignored for undirected graphs.
weights	Possibly a numeric vector giving edge weights. If this is NULL, the default, and the graph has a weight edge attribute, then the attribute is used. If this is NA then no weights are used (even if the graph has a weight attribute).

Details

More detail at Closeness Vitality

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Brandes, U. & Erlebach, T. 2005. Network Analysis: Methodological Foundations, U.S. Government Printing Office.

Examples

```
g <- graph(c(1,2,2,3,3,4,4,2,1,4), directed=FALSE)
closeness.vitality(g)
```

---

clusterrank *Find the ClusterRank ranks in a graph*

Description

Mathematically, the ClusterRank score s_i of node i is defined as:

$$s_i = f(c_i) \sum_{j \in \tau_i} (k^j_{out} + 1)$$

where the term f(c_i) accounts for the effect of i's local clustering and the term '+1' results from the contribution of j itself.
Here $f(c_i) = 10^{-c_i}$

Usage

```
clusterrank(graph, vids = V(graph), directed = TRUE, loops = TRUE)
```

Arguments

graph	The input graph as igraph object
vids	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
directed	Logical scalar, whether to directed graph is analyzed. This argument is ignored for undirected graphs.
loops	Logical; whether the loop edges are also counted.

Details

ClusterRank is a local ranking algorithm which takes into account not only the number of neighbors and the neighbors' influences, but also the clustering coefficient.
ClusterRank can also be applied to undirected networks where the superiority of ClusterRank is significant compared with degree centrality and k-core decomposition.
More detail at ClusterRank

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Chen, Duan-Bing, et al. "Identifying influential nodes in large-scale directed networks: the role of clustering." PloS one 8.10 (2013): e77455.

Examples

```
g <- graph(c(1,2,2,3,3,4,4,2,2,5,5,3,4,1,4,3,1,6,6,3,3,6,2,6,5,6))
clusterrank(g)
```

communibet *Find the communicability betweenness centrality*

Description

The communicability betweenness of a node r is:

$$\omega_r = \frac{1}{C}\sum_p \sum_q \frac{G_{prq}}{G_{pq}}, p \neq q, p \neq r, q \neq r$$

where where $G_{prq} = (e^A)_{pq} - (e^{A+E(r)})_{pq}$ is the number of walks involving node r, $G_{pq} = (e^A)_{pq}$ is the number of closed walks starting at node p and ending at node q, and $C = (n-1)^2 - (n-1)$ is a normalization factor equal to the number of terms in the sum.

Usage

```
communibet(graph, vids = V(graph), normalized = FALSE)
```

Arguments

graph	The input graph as igraph object
vids	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
normalized	Logical scalar, whether to calculate the normalized score.

Details

Communicability betweenness measure makes use of the number of walks connecting every pair of nodes as the basis of a betweenness centrality measure.
The resulting ω_r takes values between zero and one. The lower bound cannot be attained for a connected graph, and the upper bound is attained in the star graph.
More detail at Communicability Betweenness Centrality

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

Algorithm adapted from NetworkX 1.9 (Hagberg, A. 2008).

References

Estrada, Ernesto, Desmond J. Higham, and Naomichi Hatano. "Communicability betweenness in complex networks." Physica A: Statistical Mechanics and its Applications 388.5 (2009): 764-774.

Hagberg, Aric, Pieter Swart, and Daniel S Chult. Exploring network structure, dynamics, and function using NetworkX. No. LA-UR-08-05495; LA-UR-08-5495. Los Alamos National Laboratory (LANL), 2008.

Examples

```
## Not run:
g <- graph(c(1,2,2,3,2,6,6,5,3,5,3,4,5,4,4,7), directed=FALSE)
communibet(g)

## End(Not run)
```

`communitycent` *Find the community-based node centrality*

Description

This function returns community-based node centrality measures.

Usage

```
communitycent(graph, vids = V(graph), type = c("commweight", "commconn"),
  normalise = TRUE)
```

Arguments

`graph`	The input graph as igraph object
`vids`	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
`type`	A character string naming the community centrality measure. Can be one of "commweight" or "commconn"
`normalise`	Logical, whether to normalise community connectedness for "commconn". Defaults to TRUE. Will be ignored for "commweight".

Details

The "commweight" type weights each community that a node belongs to by how similar that community is to each of the other communities to which the node also belongs. For node i the community centrality is:

$$C_c(i) = \sum_{i \in j}^{N} (1 - \frac{1}{m} \sum_{i \in j \cap k}^{m} S(j,k))$$

where the main sum is over the N communities to which node i belongs, and $S(j,k)$ refers to the similarity between community j and k, calculated as the Jaccard coefficient for the number of shared nodes between each community pair, and this is averaged over the m communities paired with community j and in which node i jointly belongs.
The "commconn" type weights each community that a node belongs to by how many connections the community forms outside of itself relative to how many connections the community has within

itself (the inverse of modularity), so that nodes that belong to more highly connecting communitites will receive a higher community centrality score. For node i the community centrality is:

$$C_c(i) = \sum_{i \in j}^{N} e_{ij} \frac{\check{e}_{B(j)}}{\check{e}_{W(j)}}$$

where e_{ij} is the number of edges node i has in community j, $\check{e}_{B(j)} = \frac{e_{B(j)}}{n_j d}$ is the number of edges community j makes outside of itself normalised by the number of nodes in community j multiplied by the average degree in the network, and $\check{e}_{W(j)} = \frac{e_{W(j)}}{n(n-1)/2}$ is the number of edges within community j normalised by the total number possible.
For more detail see 'linkcomm' package and Community Centrality

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

Code obtained from 'linkcomm' package.

References

Kalinka, Alex T., and Pavel Tomancak. "linkcomm: an R package for the generation, visualization, and analysis of link communities in networks of arbitrary size and type." Bioinformatics 27.14 (2011).

Examples

```
## Not run:
g <- random.graph.game(20, 3/10)
communitycent(g)

## End(Not run)
```

`crossclique` *Find the cross-clique connectivity (centrality)*

Description

The cross-clique connectivity $X(v)$ of a node is the number of cliques to which belongs. A node with a high $X(v)$ value is called a highly cross-connected node.

Usage

```
crossclique(graph, vids = V(graph))
```

Arguments

graph	The input graph as igraph object
vids	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.

Details

Note: Directed graph considered as undirected ones and multiple edges and loops are ignored.
More detail at Cross-Clique Connectivity

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Faghani, M., and U. Nguyen. "A Study of XSS Worm Propagation and Detection Mechanisms in Online Social Networks." (2013): 1-1.

Examples

```
g <- graph(c(1,2,2,3,3,4,4,2), directed=FALSE)
crossclique(g)
```

---

decay *Find the decay centrality of a given vertex*

Description

Decay centrality of a given vertex x of a graph G is define as:

$$\sum_{y \in V(G)} \sigma^{d(x,y)}$$

where $d(x, y)$ denotes the distance between x and y and $\sigma \in (0, 1)$ is a parameter.

Usage

```
decay(graph, vids = V(graph), mode = c("all", "out", "in"),
  weights = NULL, decay = 0.5)
```

Arguments

graph	The input graph as igraph object
vids	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
mode	Character constant, gives whether the shortest paths to or from the given vertices should be calculated for directed graphs. If out then the shortest paths from the vertex, if in then to it will be considered. If all, the default, then the corresponding undirected graph will be used, ie. not directed paths are searched. This argument is ignored for undirected graphs.
weights	Possibly a numeric vector giving edge weights. If this is NULL, the default, and the graph has a weight edge attribute, then the attribute is used. If this is NA then no weights are used (even if the graph has a weight attribute).
decay	A decay parameter which the default is 0.5.

Details

Decay centrality is a centrality measure based on the proximity between a choosen vertex and every other vertex weighted by the decay.
More detail at Decay Centrality

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Jana Hurajova, Silvia Gago and Tomas Madaras, Decay Centrality, 15th Conference of Kosice Mathematicians. Herl'ny 2.-5. aprila 2014.

Examples

```
g <- graph(c(1,2,2,3,3,4,4,2), directed=FALSE)
decay(g)
```

---

diffusion.degree *Find the variant (Latora) closeness centrality in a disconnected graph*

Description

The diffusion degree of a node is defined as the cumulative contribution score of the node itself and its neighbors.

Usage

```
diffusion.degree(graph, vids = V(graph), mode = c("all", "out", "in"),
  loops = TRUE, lambda = 1)
```

Arguments

graph	The input graph as igraph object
vids	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
mode	Character constatnt, it specifies how to use the direction of the edges if a directed graph is analyzed. For 'out' only the outgoing edges are followed. For 'in' all vertices from which the source vertex is reachable in at most order steps are counted. 'all' ignores the direction of the edges. This argument is ignored for undirected graphs.
loops	Logical; whether the loop edges are also counted.
lambda	Possibly a numeric vector giving propagation probability of vertices. The default is 1 for all vertices.

Details

Diffusion degree C_{DD} of node v defined as:

$$C_{DD}(v) = \lambda_v * C_D(v) + \sum_{i \in neighbors(v)} \lambda_i * C_D(i)$$

where C_D is degree of of vertex and λ is propagation probability of vertex.
In a diffusion process, a node v with propagation probability λ_v, can activate its neighbor u with probability λ_v.
When the diffusion process propagates to the next level, active neighbors of v will try to activate their inactive neighbors. Thus the cumulative contribution in the diffusion process by neighbors of v will be maximized when all of its neighbors will be activated in the previous step.
More detail at Diffusion Degree

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Pal, Sankar K., Suman Kundu, and C. A. Murthy. "Centrality Measures, Upper Bound, and Influence Maximization in Large Scale Directed Social Networks." Fundamenta Informaticae 130.3 (2014): 317-342.

Examples

```
g <- graph(c(1,2,2,3,3,4,4,2))
diffusion.degree(g)
```

`dmnc` *Find the density of maximum neighborhood component (DMNC) in a graph*

Description

The score of node v, $DMNC(v)$, is defined to be $\frac{E}{N^{\epsilon}}$:

$$\frac{|E(MNC(v))|}{|V(MNC(v))|^{\epsilon}}$$

where for some $1 \leq \epsilon \leq 2$.

Usage

```
dmnc(graph, vids = V(graph), mode = c("all", "out", "in"), epsilon = 1.67)
```

Arguments

`graph`	The input graph as igraph object
`vids`	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
`mode`	Character constatnt, it specifies how to use the direction of the edges if a directed graph is analyzed. For 'out' only the outgoing edges are followed. For 'in' all vertices from which the source vertex is reachable in at most order steps are counted. 'all' ignores the direction of the edges. This argument is ignored for undirected graphs.
`epsilon`	ϵ parameter which default is 1.67.

Details

See Maximum Neighborhood Component (MNC)
More detail at DMNC-Density of Maximum Neighborhood Component

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Lin, Chung-Yen, et al. "Hubba: hub objects analyzer-a framework of interactome hubs identification for network biology." Nucleic acids research 36.suppl 2 (2008): W438-W443.

Examples

```
g <- random.graph.game(20, 3/10)
dmnc(g)
```

`entropy` *Find the entropy centrality in a graph*

Description

Entropy centrality measures centrality of nodes depending on their contribution to the entropy of the graph.

Usage

```
entropy(graph, vids = V(graph), mode = c("all", "out", "in"),
  weights = NULL)
```

Arguments

`graph`	The input graph as igraph object
`vids`	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
`mode`	Character constant, gives whether the shortest paths to or from the given vertices should be calculated for directed graphs. If out then the shortest paths from the vertex, if in then to it will be considered. If all, the default, then the corresponding undirected graph will be used, ie. not directed paths are searched. This argument is ignored for undirected graphs.
`weights`	Possibly a numeric vector giving edge weights. If this is NULL, the default, and the graph has a weight edge attribute, then the attribute is used. If this is NA then no weights are used (even if the graph has a weight attribute).

Details

The centrality entropy measures H_{ce} of a graph G, defined as:

$$H_{ce}(G) = -\sum_{i=1}^{n} \gamma(v_i) \times log_2 \gamma(v_i)$$

where $\gamma(v_i) = \frac{paths(v_i)}{paths(v_1, v_2, ..., v_M)}$ where $paths(v_i)$ is the number of geodesic paths from node v_i to all the other nodes in the graph and $paths(v_1, v_2, ..., v_M)$ is the total number of geodesic paths M that exists across all the nodes in the graph.

The centrality entropy provides information on the degree of centrality for a node in the graph. Those nodes that will split the graph in two or that will reduce substantially the number of paths available to reach other nodes when removed, will have a higher impact in decreasing the total centrality entropy of a graph.
More detail at Entropy Centrality

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Ortiz-Arroyo, Daniel, and DM Akbar Hussain. "An information theory approach to identify sets of key players." Intelligence and Security Informatics. Springer Berlin Heidelberg, 2008. 15-26.

Examples

```
g <- erdos.renyi.game(10, 1/10)
entropy(g)
```

---

epc *Find the edge percolated component (EPC) in a graph*

Description

For a node v in G, $EPC(v)$ is defined as:

$$EPC(v) = \frac{1}{|v|} \sum_{k=1}^{1000} \sum_{t \in e} \delta_{vt}^{k}$$

Given a threshold $(0 \leq thethreshold \leq 1)$, we create 1000 reduced network by asigning a random number between 0 and 1 to every edge and remove edges if their associated random numbers are less than the threshold.
Let the G_k be the reduced network generated at the k_{th} time reduced process. If nodes u and v are connected in G_k, set δ_{vt}^{k} to 1; otherwise $\delta_{vt}^{k} = 0$.

Usage

```
epc(graph, vids = V(graph), threshold = 0.5)
```

Arguments

graph	The input graph as igraph object
vids	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
threshold	The threshold parameter, for filter graph and create reduced one, which must be between 0 and 1. The default is 0.5.

Details

For an interaction network G, assign a removing probability p to every edge. Let G'be a realization of the random edge removing from G. If nodes v and w are connected in G', set d_{vw} be 1, otherwise set d_{vw} be 0. The percolated connectivity of v and w, c_{vw}, is defined to be the average of d_{vw} over realizations. The size of percolated component containing node v, s_v, is defined to be the sum of c_{vw} over nodes w. The score of node v, $EPC(v)$, is defined to be s_v.
More detail at EPC-Edge Percolated Component

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Lin, Chung-Yen, et al. "Hubba: hub objects analyzer-a framework of interactome hubs identification for network biology." Nucleic acids research 36.suppl 2 (2008): W438-W443.

Chen, Shu-Hwa, et al. "cyto-Hubba: A Cytoscape plug-in for hub object analysis in network biology." 20th International Conference on Genome Informatics. 2009.

Examples

```
g <- graph(c(1,2,2,3,3,4,4,2))
epc(g)
```

---

geokpath *Find the geodesic k-path centrality*

Description

Geodesic K-path centrality counts neighbours as those that are on a geodesic path less than "k" away.

Usage

```
geokpath(graph, vids = V(graph), mode = c("all", "out", "in"),
  weights = NULL, k = 3)
```

Arguments

graph	The input graph as igraph object
vids	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
mode	Character constant, gives whether the shortest paths to or from the given vertices should be calculated for directed graphs. If out then the shortest paths from the vertex, if in then to it will be considered. If all, the default, then the corresponding undirected graph will be used, ie. not directed paths are searched. This argument is ignored for undirected graphs.
weights	Possibly a numeric vector giving edge weights. If this is NULL, the default, and the graph has a weight edge attribute, then the attribute is used. If this is NA then no weights are used (even if the graph has a weight attribute).
k	The k parameter. The default is 3.

Details

More detail at Geodesic K-Path Centrality

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Borgatti, Stephen P., and Martin G. Everett. "A graph-theoretic perspective on centrality." Social networks 28.4 (2006): 466-484.

Examples

```
g <- barabasi.game(100)
geokpath(g)
```

hubbell *Find the Hubbell centrality or the Hubbell Index*

Description

Hubbell centrality defined as:

$$C_h = E + WC_h$$

where E is some exogeneous input and W is a weight matrix derived from the adjancancy matrix A.

Usage

```
hubbell(graph, vids = V(graph), weights = NULL, weightfactor = 0.5)
```

Arguments

graph	The input graph as igraph object
vids	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
weights	Possibly a numeric vector giving edge weights. If this is NULL, the default, and the graph has a weight edge attribute, then the attribute is used. If this is NA then no weights are used (even if the graph has a weight attribute).
weightfactor	The weight factorLogical which must be greater than 0. The defualt is 0.5.

Details

This centrality value is defined by means of a weighted and loop allowed network. The weighted adjacency matrix W of a network G is asymmetric and contains real-valued weights for each edge. More detail at Hubbell Index

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

Algorithm adapted from CentiLib (Grabler, Johannes, 2012).

References

Hubbell, Charles H. "An input-output approach to clique identification." Sociometry (1965): 377-399.

Grabler, Johannes, Dirk Koschutzki, and Falk Schreiber. "CentiLib: comprehensive analysis and exploration of network centralities." Bioinformatics 28.8 (2012): 1178-1179.

Examples

```
g <- barabasi.game(100)
hubbell(g)
```

`katzcent` *Find the Katz centrality (Katz Status Index)*

Description

The Katz centrality for node i is:

$$x_i = \alpha \sum_j A_{ij} x_j + \beta$$

where A is the adjacency matrix of the graph G with eigenvalues λ. The parameter β controls the initial centrality and $\alpha < \frac{1}{\lambda_{max}}$.

Usage

```
katzcent(graph, vids = V(graph), alpha = 0.1)
```

Arguments

`graph`	The input graph as igraph object
`vids`	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
`alpha`	The alpha parameter, which must be between 0.0-0.2. The default is 0.1.

Details

Katz centrality computes the relative influence of a node within a network by measuring the number of the immediate neighbors (first degree nodes) and also all other nodes in the network that connect to the node under consideration through these immediate neighbors.
More detail at Katz Centrality

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

Algorithm adapted from CentiBin with thanks Dirk Koschutzki. (Junker, Bjorn H. 2006).

References

Newman, Mark. Networks: an introduction. Oxford University Press, 2010.

Junker, Bjorn H., Dirk Koschutzki, and Falk Schreiber. "Exploration of biological network centralities with CentiBiN." BMC bioinformatics 7.1 (2006): 219.

Examples

```
g <- barabasi.game(20)
katzcent(g)
```

`laplacian` *Find the laplacian centrality*

Description

The Laplacian centrality with respect to v is:

$$C_v^L = (\Delta E)_v = d_G^2(v) + d_G(v) + 2 \sum_{v_i \in N(v)} d_G(v_i)$$

where G is a graph of n vertices, $N(v)$ is the set of neighbors of v in G and $d_G(v_i)$ is the degree of v_i in G.

Usage

```
laplacian(graph, vids = V(graph), mode = c("all", "out", "in"),
  loops = TRUE)
```

Arguments

`graph`	The input graph as igraph object
`vids`	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
`mode`	Character constatnt, it specifies how to use the direction of the edges if a directed graph is analyzed. For 'out' only the outgoing edges are followed. For 'in' all vertices from which the source vertex is reachable in at most order steps are counted. 'all' ignores the direction of the edges. This argument is ignored for undirected graphs.
`loops`	Logical; whether the loop edges are also counted.

Details

Laplacian centrality is a simple centrality measure that can be calculated in linear time. It is defined as the drop in the Laplacian energy (i.e. sum of squares of the eigenvalues in the Laplacian matrix) of the graph when the vertex is removed.
More detail at Laplacian Centrality

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Qi, Xingqin, et al. "Laplacian centrality: A new centrality measure for weighted networks." Information Sciences 194 (2012): 240-253.

Examples

```
g <- graph(c(1,2,2,3,3,4,4,2))
laplacian(g)
```

`leaderrank` *Find the LeaderRank in a directed graph*

Description

This function find the LeaderRank in a directed graph

Usage

```
leaderrank(graph, vids = V(graph))
```

Arguments

`graph`	The input graph as igraph object
`vids`	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.

Details

Given a network consisting of N nodes and M directed links, a ground node connected with every node by a bidirectional link is added. Then, the network becomes strongly connected and consists of N+1 nodes and M+2N links (a bidirectional link is counted as two links with inverse directions). LeaderRank directly applies the standard random walk process to determine the score of every node. Accordingly, if the score of node i at time step t is $si(t)$, the dynamics can be described by an iterative process as:

$$s_i(t+1) = \sum_{j=1}^{N+1} \frac{a_{ji}}{k_j^{out}} s_j(t)$$

where a_{ji} is the element of the corresponding (N + 1)-dimensional adjacency matrix, which equals 1 if there is a directed link from j to i and 0 otherwise, and k_j^{out} is the out-degree of node j. The process starts with the initialization where all node scores are 1 and will soon converge to a unique steady state denoted as $s_i^{\infty}, (i = 1, 2, ..., N, N + 1)$. LeaderRank ranks all nodes according to s_i^{∞}, and the nodes with larger final scores are considered to be more influential in spreading.
More detail at LeaderRank

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Lu, Linyuan, et al. "Leaders in social networks, the delicious case." PloS one 6.6 (2011): e21202.

Examples

```
g <- graph(c(1,2,2,3,3,4,4,2))
leaderrank(g)
```

---

leverage *Find the leverage centrality*

Description

Leverage centrality considers the degree of a node relative to its neighbors and operates under the principle that a node in a network is central if its immediate neighbors rely on that node for information.

Usage

```
leverage(graph, vids = V(graph), mode = c("all", "out", "in"),
  loops = TRUE)
```

Arguments

graph	The input graph as igraph object
vids	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
mode	Character constatnt, it specifies how to use the direction of the edges if a directed graph is analyzed. For 'out' only the outgoing edges are followed. For 'in' all vertices from which the source vertex is reachable in at most order steps are counted. 'all' ignores the direction of the edges. This argument is ignored for undirected graphs.
loops	Logical; whether the loop edges are also counted.

Details

Leverage centrality of vertex i defined as:

$$l_i = \frac{1}{k_i} \sum_{N_i} \frac{k_i - k_j}{k_i + k_j}$$

where k_i is degree of a given node i, k_j is degree of each of its neighbors and N_i is all neighbors. A node with negative leverage centrality is influenced by its neighbors, as the neighbors connect and

interact with far more nodes. A node with positive leverage centrality, on the other hand, influences its neighbors since the neighbors tend to have far fewer connections.
More detail at Leverage Centrality

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Joyce, Karen E., et al. "A new measure of centrality for brain networks." PLoS One 5.8 (2010): e12200.

Examples

```
g <- graph(c(1,2,2,3,3,4,4,2))
leverage(g)
```

`lincent` *Find the lin centrality in a graph*

Description

Lin centrality of a node x with a nonempty coreachable set is:

$$\frac{|\{y|d(x,y)<\infty\}|^2}{\sum_{d(x,y)<\infty} d(x,y)}$$

where

Usage

```
lincent(graph, vids = V(graph), mode = c("all", "out", "in"),
  weights = NULL)
```

Arguments

`graph`	The input graph as igraph object
`vids`	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
`mode`	Character constant, gives whether the shortest paths to or from the given vertices should be calculated for directed graphs. If out then the shortest paths from the vertex, if in then to it will be considered. If all, the default, then the corresponding undirected graph will be used, ie. not directed paths are searched. This argument is ignored for undirected graphs.

`weights` Possibly a numeric vector giving edge weights. If this is NULL, the default, and the graph has a weight edge attribute, then the attribute is used. If this is NA then no weights are used (even if the graph has a weight attribute).

Details

Lin centrality consider closeness not the inverse of a sum of distances, but rather the inverse of the average distance, which entails a first multiplication by the number of coreachable nodes. This change normalizes closeness across the graph. Now, however, we want nodes with a larger coreachable set to be more important, given that the average distance is the same, so we multiply again by the number of coreachable nodes. Nodes with an empty coreachable set have centrality 1 by definition.
More detail at Lin Centrality

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Lin, Nan. Foundations of social research. New York: McGraw-Hill, 1976.

Boldi, Paolo, and Sebastiano Vigna. "Axioms for centrality." Internet Mathematics just-accepted (2014): 00-00.

Examples

```
g <- graph(c(1,2,2,3,3,4,4,2))
lincent(g)
```

`lobby` *Find the lobby index (centrality)*

Description

The l-index or lobby index of a node x is the largest integer k such that x has at least k neighbors with a degree of at least k.

Usage

```
lobby(graph, vids = V(graph), mode = c("all", "out", "in"), loops = TRUE)
```

Arguments

graph	The input graph as igraph object
vids	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
mode	Character constatnt, it specifies how to use the direction of the edges if a directed graph is analyzed. For 'out' only the outgoing edges are followed. For 'in' all vertices from which the source vertex is reachable in at most order steps are counted. 'all' ignores the direction of the edges. This argument is ignored for undirected graphs.
loops	Logical; whether the loop edges are also counted.

Details

For more detail at Lobby Index

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Korn, A., A. Schubert, and A. Telcs. "Lobby index in networks." Physica A: Statistical Mechanics and its Applications 388.11 (2009): 2221-2226.

Examples

```
g <- random.graph.game(20, 3/10)
lobby(g)
```

---

markovcent *Find the markov centrality score*

Description

The Markov centrality score uses the concept of a random walk through the graph to calculate the centrality of each vertex.

Usage

```
markovcent(graph, vids = V(graph))
```

Arguments

graph	The input graph as igraph object
vids	Vertex sequence, the vertices for which the markov centrality values are returned.

Details

The method uses the mean first-passage time from every vertex to every other vertex to produce a score for each vertex.
More detail at Markov Centrality

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

Original code from Bioconductor SANTA package (Cornish AJ, 2014)

References

White, S. & Smyth, P. Algorithms for estimating relative importance in networks. Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, 2003. ACM, 266-275.

Cornish AJ and Markowetz F (2014). "SANTA: Quantifying the Functional Content of Molecular Networks." PLOS Computational Biology, 10(9), pp. e1003808. http://dx.doi.org/10.1371/journal.pcbi.1003808.

Examples

```
g <- graph(c(1,2,2,3,3,4,4,2))
markovcent(g)
```

---

mnc *Find the maximum neighborhood component (MNC)*

Description

Maximum Neighborhood Component defined as:

$$MNC(v) = |V(MC(v))|$$

where where MC(v) is a maximum connected component of the $G[N(v)]$ and $G[N(v)]$ is the induced subgraph of G by $N(v)$ and $N(v)$ is neighborhoods of node v.

Usage

```
mnc(graph, vids = V(graph), mode = c("all", "out", "in"))
```

Arguments

graph	The input graph as igraph object
vids	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
mode	Character constatnt, it specifies how to use the direction of the edges if a directed graph is analyzed. For 'out' only the outgoing edges are followed. For 'in' all vertices from which the source vertex is reachable in at most order steps are counted. 'all' ignores the direction of the edges. This argument is ignored for undirected graphs.

Details

The neighborhood of a node v, nodes adjacent to v, induce a subnetwork $N(v)$. The score of node v, $MNC(v)$, is defined to be the size of the maximum connected component of $N(v)$. The neighborhood $N(v)$ is the set of nodes adjacent to v and does not contain node v.
More detail at MNC-Maximum Neighborhood Component

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Lin, Chung-Yen, et al. "Hubba: hub objects analyzer-a framework of interactome hubs identification for network biology." Nucleic acids research 36.suppl 2 (2008): W438-W443.

Examples

```
g <- random.graph.game(20, 3/10)
mnc(g)
```

---

pairwisedis *Find the pairwise disconnectivity index*

Description

The pairwise disconnectivity index of vertex v, $Dis(v)$ defined as:

$$Dis(v) = \frac{N_0 - N_{-v}}{N_0} = 1 - \frac{N_{-v}}{N_0}$$

where N_0 is the total number of ordered pairs of vertices in a network that are connected by at least one directed path of any length. It is supposed that $N_0 > 0$, i.e., there exists at least one edge in the network that links two different vertices. N_{-v} is the number of ordered pairs that are still connected after removing vertex v from the network, via alternative paths through other vertices.

Usage

```
pairwisedis(graph, vids = V(graph))
```

Arguments

graph	The input graph as igraph object
vids	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.

Details

The pairwise disconnectivity defined as index of vertex v, $Dis(v)$, as the fraction of those initially connected pairs of vertices in a network which become disconnected if vertex v is removed from the network. The pairwise disconnectivity index quantifies how crucial an individual element is for sustaining the communication ability between connected pairs of vertices in a network that is displayed as a directed graph.
More detail at Pairwise Disconnectivity Index

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Potapov, Anatolij P., Bjorn Goemann, and Edgar Wingender. "The pairwise disconnectivity index as a new metric for the topological analysis of regulatory networks." BMC bioinformatics 9.1 (2008): 227.

Examples

```
g <- graph(c(1,2,2,3,3,4,4,2))
pairwisedis(g)
```

---

radiality *Find the radiality centrality in a graph*

Description

The radiality is a node centrality index and will give high centralities to vertices that are a short distance to every other vertex in its reachable neighborhood compared to its diameter.

Usage

```
radiality(graph, vids = V(graph), mode = c("all", "out", "in"),
  weights = NULL)
```

Arguments

graph	The input graph as igraph object
vids	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
mode	Character constant, gives whether the shortest paths to or from the given vertices should be calculated for directed graphs. If out then the shortest paths from the vertex, if in then to it will be considered. If all, the default, then the corresponding undirected graph will be used, ie. not directed paths are searched. This argument is ignored for undirected graphs.
weights	Possibly a numeric vector giving edge weights. If this is NULL, the default, and the graph has a weight edge attribute, then the attribute is used. If this is NA then no weights are used (even if the graph has a weight attribute).

Details

Radiality centrality defined as:

$$C_{rad}(v) = \frac{\sum_{w \in V}(d + 1 - d(v, w))}{n - 1}$$

where d is diameter of graph G with n vertices and $d(v, w)$ is distance between vertex v and w.
The radiality of a node v is calculated by computing the shortest path between the node v and all other nodes in the graph. The value of each path is then subtracted by the value of the diameter +1 (G+1) and the resulting values are summated. Finally, the obtained value is divided for the number of nodes -1 (n-1). The radiality should be always compared to the closeness and to the eccentricity: a node with high eccentricity + high closeness+ high radiality is a consistent indication of a high central position in the graph.
More detail at Radiality Centrality

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Wolfram Research, Inc., Mathematica, Version 10.0, Champaign, IL (2014). http://reference.wolfram.com/language/ref/Radi

Scardoni, G., Laudanna, C., Tosadori, G., Fabbri, F. & Faizaan, M. CentiScaPe: Network centralities for Cytoscape. http://www.cbmc.it/~scardonig/centiscape/CentiScaPefiles/CentralitiesTutorial.pdf

Examples

```
g <- graph(c(1,2,2,3,3,4,4,2))
radiality(g)
```

---

salsa *Find the SALSA as 'hub' or 'authority' score*

Description

The Stochastic Approach for Link-Structure Analysis (SALSA) is combination of HITS and PageRank which creates a neighborhood graph using authority and hub pages and links and create a bipartite graph of the authority and hub pages in the neighborhood graph.

Usage

```
salsa(graph, vids = V(graph), score = c("hub", "authority"))
```

Arguments

graph	The input graph as igraph object
vids	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
score	Character constant, gives which score should be calculated and must be one of 'hub' or 'authority'. The default is 'hub'.

Details

More detail at SALSA

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Lempel, Ronny, and Shlomo Moran. "SALSA: the stochastic approach for link-structure analysis." ACM Transactions on Information Systems (TOIS) 19.2 (2001): 131-160.

Examples

```
g <- barabasi.game(10)
salsa(g)
```

`semilocal` *Find the semi local centrality (or local centrality)*

Description

The local centrality $CL(v)$ of node v is defined as:

$$C_L(v) = \sum_{u \in \Gamma(v)} Q(u)$$

where

$$Q(u) = \sum_{w \in \Gamma(u)} N(w)$$

and $\Gamma(u)$ is the set of the nearest neighbors of node u and $N(w)$ is the number of the nearest and the next nearest neighbors of node w.

Usage

```
semilocal(graph, vids = V(graph), mode = c("all", "out", "in"))
```

Arguments

`graph`	The input graph as igraph object
`vids`	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.
`mode`	Character constatnt, it specifies how to use the direction of the edges if a directed graph is analyzed. For 'out' only the outgoing edges are followed. For 'in' all vertices from which the source vertex is reachable in at most order steps are counted. 'all' ignores the direction of the edges. This argument is ignored for undirected graphs.

Details

The local centrality is proposed aiming at identifying the influencers in undirected network, it can be applied to directed network as well with a modified definition of $N(w)$. Of course, for directed network, $N(w)$ should be the number of the nearest and next nearest upstream nodes of node w.
Local centrality measure is likely to be more effective to identify influential nodes than degree centrality measure as it utilizes more information, while it has much lower computational complexity than the betweenness and closeness centralities.
More detail at Semi_Local Centrality

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Chen, Duanbing, et al. "Identifying influential nodes in complex networks." Physica a: Statistical mechanics and its applications 391.4 (2012): 1777-1787.

Examples

```
g <- barabasi.game(10)
semilocal(g)
```

`topocoefficient` *Find the topological coefficient of a node in a undirected graph*

Description

The topological coefficient is a relative measure for the extent to which a node shares neighbors with other nodes.

Usage

```
topocoefficient(graph, vids = V(graph))
```

Arguments

`graph`	The input graph as igraph object
`vids`	Vertex sequence, the vertices for which the centrality values are returned. Default is all vertices.

Details

Topological coefficient T_n of a node n with k_n neighbors defined as:

$$T_n = \frac{avg(J(n,m))}{k_n}$$

where $J(n,m)$ is defined for all nodes m that share at least one neighbor with n. The value $J(n,m)$ is the number of neighbors shared between the nodes n and m, plus one if there is a direct link between n and m.
Nodes that have one or no neighbors are assigned a topological coefficient of zero.
More detail at Topological Coefficient

Value

A numeric vector contaning the centrality scores for the selected vertices.

Author(s)

Mahdi Jalili <m_jalili@farabi.tums.ac.ir>

References

Assenov, Yassen, et al. "Computing topological parameters of biological networks." Bioinformatics 24.2 (2008): 282-284.

Examples

```
g <- graph(c(1,2,2,3,3,4,4,2), directed=FALSE)
topocoefficient(g)
```

Index