dDAGlevel

January 19, 2018

`dDAGlevel`	*Function to define/calculate the level of nodes in a direct acyclic graph (DAG)*

Description

`dDAGlevel` is supposed to calculate the level of nodes, given a direct acyclic graph (DAG; an ontology). The input is a graph of "igraph" or "graphNET" object, and the definition of the node level. The return can be the level for each node or the nodes for each level.

Usage

```
dDAGlevel(g, level.mode = c("longest_path", "shortest_path"),
return.mode = c("node2level", "level2node"))
```

Arguments

`g`	an object of class "igraph" or "graphNEL"
`level.mode`	the mode of how to define the level of nodes in DAG. It can be "longest_path" for defining the node level as the length of the longest path from the node to the root, and "shortest_paths" for defining the node level as the length of the shortest path from the node to the root
`return.mode`	the mode of how to return the node level information. It can be "node2level" for returning a named vector (i.e. the level for each node), and "level2node" for returning a named list (i.e. nodes for each level)

Value

When "return.mode" is "node2level", it returns a named vector: for each named node (i.e. Term ID), it stores its level When "return.mode" is "level2node", it returns a named list: for each named level, it contains the names (i.e. Term ID) of nodes belonging to this level

Note

The level for the root is 1. The level based on the longest path will ensure that nodes at the same level will never be reachable (i.e. in the same path), while the level based on the shortest path will not be necessary. The "longest path" based level can be useful in visiting nodes from the tipmost level to the root: 1) for the current node, all children have been visited; 2) nodes at the same level can be looked at independantly. The "shortest path" based level can be useful in deriving nodes according to their closeness to the root.

See Also

dDAGroot, dDAGreverse

Examples

```
# 1) load HPPA as igraph object
ig.HPPA <-dRDataLoader(RData='ig.HPPA')
g <- ig.HPPA

# 2) randomly select vertices as the query nodes
nodes_query <- sample(V(g),5)$name

# 3) obtain the complete subgraph induced
subg <- dDAGinduce(g, nodes_query)

# 4) calculate the node levels
# 4a) definition based on the longest path
dDAGlevel(subg, level.mode="longest_path")
# 4b) definition based on the shortest path
dDAGlevel(subg, level.mode="shortest_path")
# 4c) definition based on the longest path, and return nodes for each level
dDAGlevel(subg, level.mode="longest_path", return.mode="level2node")
```