dDAGgeneSim

July 16, 2015

`dDAGgeneSim` *Function to calculate pair-wise semantic similarity between genes based on a direct acyclic graph (DAG) with annotated data*

Description

`dDAGgeneSim` is supposed to calculate pair-wise semantic similarity between genes based on a direct acyclic graph (DAG) with annotated data. It first calculates semantic similarity between terms and then derives semantic similarity between genes from terms-term semantic similarity. Parallel computing is also supported for Linux or Mac operating systems.

Usage

```
dDAGgeneSim(g, genes = NULL, method.gene = c("BM.average", "BM.max",
"BM.complete", "average", "max"), method.term = c("Resnik", "Lin",
"Schlicker", "Jiang", "Pesquita"), force = TRUE, fast = TRUE,
parallel = TRUE, multicores = NULL, verbose = TRUE)
```

Arguments

`g`	an object of class "igraph" or "graphNEL". It must contain a vertex attribute called 'annotations' for storing annotation data (see example for howto)
`genes`	the genes between which pair-wise semantic similarity is calculated. If NULL, all genes annotatable in the input dag will be used for calculation, which is very prohibitively expensive!
`method.gene`	the method used for how to derive semantic similarity between genes from semantic similarity between terms. It can be "average" for average similarity between any two terms (one from gene 1, the other from gene 2), "max" for the maximum similarity between any two terms, "BM.average" for best-matching (BM) based average similarity (i.e. for each term of either gene, first calculate maximum similarity to any term in the other gene, then take average of maximum similarity; the final BM-based average similiary is the pre-calculated average between two genes in pair), "BM.max" for BM based maximum similarity (i.e. the same as "BM.average", but the final BM-based maximum similiary is the maximum of the pre-calculated average between two genes in pair), "BM.complete" for BM-based complete-linkage similarity (inspired by complete-linkage concept: the least of any maximum similarity between a term

of one gene and a term of the other gene). When comparing BM-based similarity between genes, "BM.average" and "BM.max" are sensitive to the number of terms invovled; instead, "BM.complete" is much robust in this aspect. By default, it uses "BM.average".

`method.term` the method used to measure semantic similarity between terms. It can be "Resnik" for information content (IC) of most informative common ancestor (MICA) (see http://arxiv.org/pdf/cmp-lg/9511007.pdf), "Lin" for 2*IC at MICA divided by the sum of IC at pairs of terms (see http://webdocs.cs.ualberta.ca/~lindek/papers/sim.pdf), "Schlicker" for weighted version of 'Lin' by the 1-prob(MICA) (see http://www.ncbi.nlm.nih.gov/pubmed/16776819), "Jiang" for 1 - difference between the sum of IC at pairs of terms and 2*IC at MICA (see http://arxiv.org/pdf/cmp-lg/9709008.pdf), "Pesquita" for graph information content similarity related to Tanimoto-Jacard index (ie. summed information content of common ancestors divided by summed information content of all ancestors of term1 and term2 (see http://www.ncbi.nlm.nih.gov/pubmed/18460186))

`force` logical to indicate whether the only most specific terms (for each gene) will be used. By default, it sets to true. It is always advisable to use this since it is computationally fast but without compromising accuracy (considering the fact that true-path-rule has been applied when running `dDAGannotate`)

`fast` logical to indicate whether a vectorised fast computation is used. By default, it sets to true. It is always advisable to use this vectorised fast computation; since the conventional computation is just used for understanding scripts

`parallel` logical to indicate whether parallel computation with multicores is used. By default, it sets to true, but not necessarily does so. Partly because parallel backends available will be system-specific (now only Linux or Mac OS). Also, it will depend on whether these two packages "foreach" and "doMC" have been installed. It can be installed via: `source("http://bioconductor.org/biocLite.R"); biocLite(c("foreach","doMC"))`. If not yet installed, this option will be disabled

`multicores` an integer to specify how many cores will be registered as the multicore parallel backend to the 'foreach' package. If NULL, it will use a half of cores available in a user's computer. This option only works when parallel computation is enabled

`verbose` logical to indicate whether the messages will be displayed in the screen. By default, it sets to true for display

Value

It returns a sparse matrix containing pair-wise semantic similarity between input genes. This sparse matrix can be converted to the full matrix via the function `as.matrix`

Note

For the mode "shortest_paths", the induced subgraph is the most concise, and thus informative for visualisation when there are many nodes in query, while the mode "all_paths" results in the complete subgraph.

See Also

`dDAGtermSim`, `dDAGinduce`, `dDAGtip`, `dCheckParallel`

Examples

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

# 2) load human genes annotated by HPPA
org.Hs.egHPPA <- dRDataLoader(RData='org.Hs.egHPPA')

# 3) prepare for ontology and its annotation information
dag <- dDAGannotate(g, annotations=org.Hs.egHPPA,
path.mode="all_paths", verbose=TRUE)

# 4) calculate pair-wise semantic similarity between 5 randomly chosen genes
allgenes <- unique(unlist(V(dag)$annotations))
genes <- sample(allgenes,5)
sim <- dDAGgeneSim(g=dag, genes=genes, method.gene="BM.average",
method.term="Resnik", parallel=FALSE, verbose=TRUE)
sim
```