ontoTools
April 19, 2009

A.csr

Description

demonstration sparse matrix matrix.csr form

Usage

data(A.csr)

Format

The format is: list() - attr("ra")= num [1:27] 2.894 -0.610 -0.714 0.546 0.901 ...
- attr("ja")= int [1:27] 3 4 5 2 4 1 3 4 5 3 ...
- attr("ia")= int [1:11] 1 4 6 10 11 14 18 21 22 26 ...
- attr("dimension")= int [1:2] 10 5
- attr("class")= atomic [1:1] matrix.csr
- attr("package")= chr ".GlobalEnv"

Source

SparseM library matrix.csr example generated it

GDI_NCIThesaurus

Description

Structures for working with formal nomenclatures

Usage

data(GDI_NCIThesaurus)
parents(term, nom)
children(term, nom)
getDefs(term, nom)
Arguments

term character string, term
nom instance of class taggedHierNomenclature

Details

DAG-structured nomenclatures are in wide use. For any term, one can seek parents (generalizations) or children (specializations). These resources prototype tools for dealing with such structures, including provenance information.

Author(s)

Vince Carey <stvjc@channing.harvard.edu>

Examples

data(GDI_NCIThesaurus)
parents("Mesna", GDI_NCIThesaurus)
parents("Actinomycin_Antibiotic", GDI_NCIThesaurus)

Description

concept probabilities for GO MF terms applied to human LocusLink entries

Usage

data(LLGOMFcp)

Format

The format is: Named num [1:5399] 0 0 0 0 0 ... - attr(*, "names")= chr [1:5399] "GO:0000005" "GO:0000006" "GO:0000007" "GO:0000008" ...

Source

derived from Bioconductor packages humanLL.Mappings and GO
OOC-class

Class "OOC" object-ontology complex

Description

Object that binds ontology (structured vocabulary) with an object-term map.

Objects from the Class

Objects can be created by calls of the form `new("OOC", ...)`.

Slots

- **ontology**: Object of class "ontology" instance of ontoTools::ontology
- **OOmap**: Object of class "namedSparse" SparseM::matrix.csr bound with dimnames facilities

Methods

- `coverageMat` signature(x = "OOC") : return a namedSparse incidence matrix with r,c element indicating whether term c covers object r
- `ontology` signature(x = "OOC") : accessor
- `OOmap` signature(x = "OOC") : accessor
- `show` signature(object = "OOC") : concise printer

Author(s)

VJ Carey <stvjc@channing.harvard.edu>

Examples

data(litOnto)
g1 <- new("rootedDAG", DAG=litOnto, root="A")
o1 <- new("ontology", name="demo", version="0.1", rDAG=g1)
kvlist <- list(W="E", X="K", Y="B", Z=c("D","G"))
litMap <- otkvList2namedSparse( names(kvlist), LETTERS[1:12], kvlist )
print(litMap)
ooc1 <- makeOOC( o1, litMap )
show(ooc1)
print(coverageMat(ooc1))
# note the following will be slow with large OOCs
print(conceptProbs(ooc1))
# for larger OOCs it is useful to precompute the accessibility
# matrix of the ontology and the map from objects to terms -- these
# can be supplied as additional arguments to conceptProbs
SGDIvocab  
**Vocabulary for genomic data integration**

**Description**

Vocabulary for genomic data integration

**Usage**

```r
data(SGDIvocab)
```

**Details**

Currently unites some English terms with a formal tag to NCI Metathesaurus. Additional content to be added, with bridges to NCI EVS.

**Value**

This is an instance of a taggedHierNomenclature derived from an informal specification of terms about breast cancer provided by S. Ramaswamy.

**Author(s)**

Vince Carey <stvjc@channing.harvard.edu>

**Examples**

```r
data(SGDIvocab)
grep("differ", SGDIvocab)
getTerms(SGDIvocab)
```

---

STMA  
**Vocabulary from statistics theory and methods abstracts**

**Description**

Vocabulary from statistics theory and methods abstracts

**Usage**

```r
data(STMA)
```

**Author(s)**

Vince Carey <stvjc@channing.harvard.edu>

**Examples**

```r
data(STMA)
grep("exponential", STMA)
parents("Exponential families", STMA)
children("PROBABILITY DISTRIBUTIONS", STMA)
```
**accessMat**

*matrix utilities for ontoTools.*

**Description**

A variety of matrix utilities used in ontoTools

**Usage**

`accessMat(object)`

**Arguments**

- `object` object

**Author(s)**

Vince Carey <stvjc@channing.harvard.edu>

---

**buildGOgraph**

*build graphNEL corresponding to bioc GO environment*

**Description**

build graphNEL corresponding to bioc GO environment

**Usage**

`buildGOgraph(useenv=GOMFPARENTS)`

**Arguments**

- `useenv` `useenv`: environment to be used

**Details**

all GO MF tags are nodes, edges drawn from node to parent

**Value**

graphNEL instance

**Author(s)**

Vince Carey <stvjc@channing.harvard.edu>

**See Also**

Biograph package

**Examples**

```
# takes a while; trivial invocation
```
compoundGraph-class

Class "compoundGraph" list representation of multiple graph::graph objects

Description

Class "compoundGraph" list+list representation of multiple graph::graph objects

Objects from the Class

Objects can be created by calls of the form `new("compoundGraph", ...)`. 

Slots

- `grList`: Object of class "list" list of graph::graph objects
- `between`: Object of class "list" list of node-to-node connections across graphs

Methods

- `adjMat` signature(cg = "compoundGraph"): ...
- `between` signature(object = "compoundGraph"): ...
- `grList` signature(object = "compoundGraph"): ...
- `toDot` signature(G = "compoundGraph", outDotFile = "character", renderList = "list", optList = "missing"): ...
- `toDot` signature(G = "compoundGraph", outDotFile = "character", renderList = "list", optList = "list"): ...
- `toDot` signature(G = "compoundGraph", outDotFile = "missing", renderList = "list", optList = "missing"): ...

Note

Should be supplanted by Rgraphviz facilities before too long.

Author(s)

VJ Carey <stvjc@channing.harvard.edu>

References

~put references to the literature/web site here ~

See Also

Rgraphviz::subgraph
depthStruct

tools for manipulating depth concepts for rooted DAGs

Description

return a list of environments giving mapping from node name to rooted DAG depth and from depth to vector of names of nodes at that depth

Usage

depthStruct(rg)
ontoDepth(rg)
DMdepth(g, maxd)

Arguments

rg
rg: instance of class rootedDAG
g
rg: instance of class depth
maxd
maxd: bound on depth to be measured

Value

depthStruct: a list of two environments (see examples).

Note

ontoDepth is the workhorse for depthStruct. DMdepth is a function that works on a plain graph, creating the ‘daughter matrix’ and computing depths.

Author(s)

Vince Carey <stvjc@channing.harvard.edu>

Examples

data(litOnto)
print(litOnto)
g1 <- new("rootedDAG", DAG=litOnto, root="A")
o1 <- new("ontology", name="demo", version="0.1", rDAG=g1)
print(ds <- depthStruct(g1))
ds$tag2depth("A")
ds$tag2depth("H")
ds$depth2tag(2)
litOnto

litOnto: graph illustrating the ontology concept; litObj: matrix illustrating the object-ontology mapping

Description
litOnto: graph illustrating the ontology concept; litObj: matrix illustrating the object-ontology mapping

Usage
`data(litOnto)`

Format
The format is: list() - attr(*, "nodes")= chr [1:12] "A" "B" "C" "D" ... - attr(*, "edgeL")= List of 12 ..A : Listof1.... edges: NULL ..B : Listof1.... edges: int 1 ..C : Listof1.... edges: int 1 ..D : Listof1.... edges: int 2 ..E : Listof1.... edges: int 2 ..F : Listof1.... edges: int 3 ..G : Listof1.... edges: int 3 ..H : Listof1.... edges: int 1:2 4 5 ..I : Listof1.... edges: int [1:3] 4 3 5 ..J : Listof1.... edges: int 6 ..K : Listof1.... edges: int 6 ..L : Listof1.... edges: int 7 - attr(*, "edgemode")= chr "directed" - attr(*, "class")= chr "graphNEL"
Class "namedSparse" adds margin names to sparse matrices

Description

manages margin names for sparse matrices

Objects from the Class

Objects can be created by calls of the form `new("namedSparse", ...)`. These are S4 objects that include a `SparseM::matrix.csr`, associated dimnames in the customary form, and two lists of name-to-index mapping environments (for row and column name resolution, with forward (name to index) and backward (index to name) mapping).

Slots

- **Dimnames**: Object of class "list" ordinary dimnames matrix metadata
- **mat**: Object of class "matrix.csr" sparse matrix

Methods

- **Arith** signature(e1 = "namedSparse", e2 = "namedSparse"): ...
- **show** signature(object = "namedSparse"): ...

Note

A constructor `makeNamedSparse` is illustrated in the example.

Author(s)

VJ Carey <stvj@channing.harvard.edu>

Examples

data(A.csr)
namedA <- mkNS(A.csr) # installs default dimnames R(1:nrow), C(1:ncol)
print(namedA)
print(colSums(namedA))
dimnames(namedA) <- list(paste("A",1:10,sep=""),paste("B",1:5,sep=""))
print(namedA)
print(namedA %*% t(namedA))
ontology-class

Class "ontology" wraps a rooted DAG with some ontology metadata

Description

instances of class ontology are used to represent things like Gene Ontology

Objects from the Class

Objects can be created by calls of the form new("ontology", ...). This simply possesses information on name and version of ontology.

Slots

name: Object of class "character" name of ontology
version: Object of class "character" version tag
rDAG: Object of class "rootedDAG" the rooted DAG representing the terminology hierarchy

Methods

accessMat signature(object = "ontology"): returns square matrix with 1 in element r,c if term corresponding to r can be reached from term corresponding to c
name signature(x = "ontology"): access name
OVersion signature(x = "ontology"): access version
rDAG signature(x = "ontology"): access the rooted DAG
show signature(object = "ontology"): concise report

Note

This class was written to deal with ontologies that are representable as rooted DAGs. It is not clear that this is a good use of the term 'ontology', which has broader implications. However this does work for Gene Ontology.

Author(s)

~~who you are~~

Examples

data(litOnto)
print(litOnto)
g1 <- new("rootedDAG", DAG=litOnto, root="A")
o1 <- new("ontology", name="demo", version="0.1", 
rDAG=g1)
# can also use
olb <- makeOntology( name="demo", version="0.1",
graph=litOnto, root="A")
show(ol)
print(accessMat(ol))
print(OVersion(ol))
**ooMapLL2GOMFdemo**  
object-term mapping for human locuslink entries and GO MF

**Description**

object-term mapping for human locuslink entries and GO MF

**Usage**

```r
data(ooMapLL2GOMFdemo)
```

**Format**

The format is:  
```plaintext
list() - attr(*, "Dimnames")=List of 2 ..: chr[1 : 10776]"1""10""100""1000".....
: chr [1:5399] "GO:00000005" "GO:00000006" "GO:00000007" "GO:00000008" ... - attr(*, "mat")=
list() ..- attr(*, "ra")= num [1:19679] 0 1 1 1 1 1 1 1 ... - attr(*, "ia")= int [1:19679] 1 1842
541 3528 3753 485 3799 1594 1804 1098 ... - attr(*, "ia")= int [1:19779] 1 3 6 8 10 13 14 17
23 28 ... - attr(*, "dimension")= int [1:2] 10776 5399 ... - attr(*, "class")= chr "matrix.csr" - attr(*,
"rowindex")=List of 2 ..n2i : length0 < environment > .. i2n:length 0 <environment> - attr(*,
"colindex")=List of 2 ..n2i : length0 < environment > .. i2n:length 0 <environment> - attr(*,
"class")= chr "namedSparse"
```

**Source**

bioconductor GO, humanLLMapping and ontoTools otkv tools.

---

**otkvEnv2namedSparse**  
obtain sparse matrix representation of key-value structures

**Description**

obtain sparse matrix representation of key-value structures

**Usage**

```r
otkvEnv2namedSparse(obs, tms, otkvEnv)
```

**Arguments**

- `obs`: vector of object tags
- `tms`: vector of terms to which objects are mapped
- `otkvEnv`: otkvEnv: environment encoding the key-value mapping
- `otkvlist`: otkvlist: list encoding the key-value mapping

**Author(s)**

Vince Carey <stvjc@channing.harvard.edu>
Examples

```
okvList2namedSparse(c("A","B","D","E"), letters[1:7],
    list("A"=c("a","b"), "B"=c("b","d"), "E"="c"))
```

---

**rootedDAG-class**

Class "rootedDAG"

**Description**

wraps a graph that can be shown to be a DAG and has a root (one node with no ancestor)

**Objects from the Class**

Objects can be created by calls of the form `new("rootedDAG", ...)`. Does not extend graph but probably should.

**Slots**

- **root**: Object of class "character" name of root
- **DAG**: Object of class "graph" DAG

**Methods**

- **DAG** signature(x = "rootedDAG"): extract the graph
- **getMatrix** signature(g = "rootedDAG", type = "character", mode = "character"): Currently only 'child2parent' can be used for type, meaning that row corresponds to child, column corresponds to ancestor and mat[row,column] is 1 if node corresponding to row is a child of node corresponding to ancestor. Type can be 'sparse' (return sparse representation) or 'dense'.
- **root** signature(x = "rootedDAG"): extract name of root

**Author(s)**

VJ Carey <stvjc@channing.harvard.edu>

---

**semsim**

*Compute semantic similarity measure for terms in an object-ontology complex*

**Description**

Compute semantic similarity measure for terms in an object-ontology complex

**Usage**

```
semsim(c1, c2, ooc, acc=NULL, pc=NULL)
conceptProbs(ooc, acc=NULL, inds=NULL)
subsumers(c1, c2, ont, acc=NULL)
pms(c1, c2, ooc, acc=NULL, pc=NULL)
usageCount(map, acc, inds)
```
semsim

Arguments

- c1, c2: "character" terms to be compared
- ooc: an object of class "OOC": object-ontology complex
- ont: an object of class "ontology": annotated rooted DAG
- acc: optional (sparse) accessibility matrix for the ontology
- pc: optional vector of concept probabilities, if pre-computed
- map: OOmap component of an ooc
- inds: vector of numeric indices, row indices of object-ontology map to be processed

Details

For large ontologies, computation of the term accessibility relationships and term probabilities can be costly. Once these are computed to support one semsim calculation, they should be saved. The acc and pc parameters allow use of this saved information.

Value


Author(s)

Vince Carey <stvjc@channing.harvard.edu>

References

PW Lord et al, Bioinformatics, 19(10)2003:1275

Examples

```
# we are given a graph of GOMF and the OOmap between LL and GOMF
# derived from humanLLMappings and stored as data resources in
# ontoTools -- these will have to be updated regularly
# data(goMFgraph.1.15)
data(LL2GOMFOoMap.1.15)
#
# build the rooted DAG, the ontology, and the OOC objects
# gomfrDAG <- new("rootedDAG", root="GO:0003674", DAG=goMFgraph.1.15)
GOMFonto <- new("ontology", name="GOMF", version="bioc GO 1.15", rDAG=gomfrDAG)
LLGOMFOOC <- makeOOC(GOMFonto, LL2GOMFOoMap.1.15)
#
# we are given the accessibility matrix for the GO MF graph as a
# data resource, and we can compute some term probabilities
# data(goMFamat.1.15)
pc <- conceptProbs(LLGOMFOOC, goMFamat.1.15, inds=1:20)
#
# now we will get a sample of GO MF terms and compute the
# semantic similarities of pairs of terms in the sample
```
data(LL2GOMFcp.1.15) # full set of precomputed concept probabilities
library(GO.db)
library(Biobase)
library(combinat)
library(annotate)
GO() # get the GO environments
GOTags <- ls(GOTERM)
GOlabs <- mget(GOTags, GOTERM, ifnotfound=NA)
GOMFtags <- GOTags[sapply(GOLabs, Ontology) == "MF"]
GOMFtags <- GOMFtags[!is.na(GOMFtags)]
GOMFtermObs <- mget(GOMFtags, env=GOTERM)
GOMFterms <- sapply( GOMFtermObs, Term )
ntags <- length(GOMFtags)
if (any(duplicated(GOMFterms))){
dups <- (1:ntags)[duplicated(GOMFterms)]
GOMFterms[dups] <- paste(GOMFterms[dups], ".2", sep="")
}
set.seed(1234)
# does not lead to common samples across platforms...
st <- sample(names(GOMFterms), size=50) # take the sample
st <- intersect(st, names(LL2GOMFcp.1.15))[1:10] # use only those terms available in bioc GO 1.15
# thus ...
require(gbs)
pst <- combn(st,2) # get a matrix with the pairs of terms in columns
bad = c(4L, 12L, 19L, 25L, 31L, 32L, 33L, 34L, 35L) # can't use 8640
pst = pst[,-bad]
npst <- ncol(pst)
ss <- rep(NA,npst)
for (i in 1:npst) # compute semantic similarities
{
cat(i)
ss[i] <- semsim( pst[1,i], pst[2,i], ooc=LLGOMFOOC, acc=gomFamat.1.15, pc=LL2GOMFcp.1.15 )
}
print(summary(ss))
top <- (1:npst)[ss==max(ss,na.rm=TRUE)][1] # index of the most similar pair
# note -- must come to an understanding of the NAs
print( GOMFterms[ as.character(pst[,top]) ] )
pen <- (1:npst)[ss==max(ss[-top],na.rm=TRUE)][1] # second most similar
print( GOMFterms[ as.character(pst[,pen]) ] )
Objects from the Class

Objects can be created by calls of the form `new("taggedHierNomenclature", ...). See example

Slots

- **tags**: Object of class "character"; formal tags, often semantically opaque
- **parents**: Object of class "character"; terms regarded as generalizations of the given term
- **delim**: Object of class "character"; the parent strings are decomposed using this delimiter
- **rootToken**: Object of class "character"; token used to indicate root of DAG
- **name**: Object of class "character" name of nomenclature
- **provenance**: Object of class "provStruct" information on origins of vocabulary
- **inMappings**: Object of class "character" list of mappings in which the term is employed
- **terms**: Object of class "character" actual subject matter terms being organized
- **definitions**: Object of class "character" verbal definitions of terms

Extends

Class "nomenclature", directly.

Methods

- **children** signature(term = "character", nom = "taggedHierNomenclature"): ...
- **parents** signature(term = "character", nom = "taggedHierNomenclature"): ...
- **show** signature(object = "taggedHierNomenclature"): ...

Author(s)

VJ Carey <stvjc@channing.harvard.edu>

See Also

GDI_NCIThesaurus

Examples

data(GDI_NCIThesaurus)
GDI_NCIThesaurus
**Methods for Function toDot in Package  `ontoTools` – should be replaced by Rgraphviz facilities soon**

**Description**

These methods write graphviz dot language for various graph structures encountered with ontologies. This activity should be moved to Rgraphviz ASAP, but there are aspects of representation and portability that need to be resolved.

**Methods**

- \( G = \text{"graphNEL"}, \text{outDotFile} = \text{"character"}, \text{renderList} = \text{"list"}, \text{optList} = \text{"list"} \) create dot language description of graph
- \( G = \text{"graphNEL"}, \text{outDotFile} = \text{"character"}, \text{renderList} = \text{"missing"}, \text{optList} = \text{"missing"} \) create dot language description of graph
- \( G = \text{"graphNEL"}, \text{outDotFile} = \text{"character"}, \text{renderList} = \text{"missing"}, \text{optList} = \text{"list"} \) create dot language description of graph
- \( G = \text{"graphNEL"}, \text{outDotFile} = \text{"missing"}, \text{renderList} = \text{"missing"}, \text{optList} = \text{"missing"} \) create dot language description of graph
- \( G = \text{"graphNEL"}, \text{outDotFile} = \text{"missing"}, \text{renderList} = \text{"missing"}, \text{optList} = \text{"list"} \) create dot language description of graph
- \( G = \text{"graphNEL"}, \text{outDotFile} = \text{"missing"}, \text{renderList} = \text{"missing"}, \text{optList} = \text{"missing"} \) create dot language description of graph
- \( G = \text{"graphNEL"}, \text{outDotFile} = \text{"missing"}, \text{renderList} = \text{"list"}, \text{optList} = \text{"list"} \) create dot language description of graph
- \( G = \text{"graphNEL"}, \text{outDotFile} = \text{"missing"}, \text{renderList} = \text{"missing"}, \text{optList} = \text{"missing"} \) create dot language description of graph
- \( G = \text{"compoundGraph"}, \text{outDotFile} = \text{"character"}, \text{renderList} = \text{"list"}, \text{optList} = \text{"missing"} \) create dot language description of graph
- \( G = \text{"compoundGraph"}, \text{outDotFile} = \text{"character"}, \text{renderList} = \text{"list"}, \text{optList} = \text{"list"} \) create dot language description of graph
- \( G = \text{"compoundGraph"}, \text{outDotFile} = \text{"missing"}, \text{renderList} = \text{"list"}, \text{optList} = \text{"missing"} \) create dot language description of graph
- \( G = \text{"compoundGraph"}, \text{outDotFile} = \text{"missing"}, \text{renderList} = \text{"list"}, \text{optList} = \text{"missing"} \) create dot language description of graph

**Examples**

```r
eexample(randomGraph)
tmp <- tempfile()
toDot(g1, tmp)
readLines(tmp)
unlink(tmp)
```
Index

*Topic classes
  compoundGraph-class, 6
  namedSparse-class, 9
  ontology-class, 10
  OOC-class, 3
  rootedDAG-class, 12
  taggedHierNomenclature-class, 14

*Topic datasets
  gomfAmat, 8
  litOnto, 8
  LLGOMFcpp, 2
  ooMapLL2GOMFdemo, 11

*Topic methods
  toDot-methods, 16

*Topic models
  A.csr, 1
  accessMat, 5
  buildGOgraph, 5
  depthStruct, 7
  GDI_NCIThesaurus, 1
  otkvEnv2namedSparse, 11
  semsim, 12
  SGDIvocab, 4
  STMA, 4
  +, namedSparse, namedSparse-method (namedSparse-class), 9
  [~, namedSparse-method (namedSparse-class), 9
  %*%, namedSparse, namedSparse-method (namedSparse-class), 9
  %+% (namedSparse-class), 9
  A.csr, 1
  accessMat, 5
  accessMat, ontology-method (ontology-class), 10
  adjMat (compoundGraph-class), 6
  adjMat, compoundGraph-method (compoundGraph-class), 6
  AMN (namedSparse-class), 9
  annoSource-class (SGDIvocab), 4
  annotationResource-class (SGDIvocab), 4
  Arith, namedSparse, namedSparse-method (namedSparse-class), 9
  as.matrix, namedSparse-method (namedSparse-class), 9
  as.matrix.ok (namedSparse-class), 9
  between (compoundGraph-class), 6
  between, compoundGraph-method (compoundGraph-class), 6
  buildGOgraph, 5
  child2parentMatDense (accessMat), 5
  child2parentMatSparse (accessMat), 5
  children (GDI_NCIThesaurus), 1
  children, character, taggedHierNomenclature-method (taggedHierNomenclature-class), 14
  colinds (namedSparse-class), 9
  colinds, namedSparse, character-method (namedSparse-class), 9
  colSums (namedSparse-class), 9
  colSums, namedSparse, missing, missing-method (namedSparse-class), 9
  colSumsSp (namedSparse-class), 9
  compoundGraph-class, 6
  conceptProbs (semsim), 12
  coverageMat (OOC-class), 3
  coverageMat, OOC-method (OOC-class), 3
  coverMat (accessMat), 5
  DAG (rootedDAG-class), 12
  DAG, rootedDAG-method (rootedDAG-class), 12
  daughterMat (accessMat), 5
  daughterSpMat (accessMat), 5
  depthStruct, 7
  dimnames, namedSparse-method (namedSparse-class), 9
  dimnames<-, namedSparse, list-method (namedSparse-class), 9
INDEX

DMdepth (depthStruct), 7
exns2 (namedSparse-class), 9
exptArchive-class (SGDIvocab), 4
exptSample-class (SGDIvocab), 4
GDI_NCIThesaurus, 1, 15
GDILabel (GDI_NCIThesaurus), 1
GDILabel-class (SGDIvocab), 4
GDontology (GDI_NCIThesaurus), 1
GDontology-class (SGDIvocab), 4
GDIplatform-class (SGDIvocab), 4
getDefs (GDI_NCIThesaurus), 1
getMatrix, character, nomenclature-method
(GDI_NCIThesaurus), 1
getMatrix (accessMat), 5
getMatrix, rootedDAG, character, character-method
(rootedDAG-class), 12
getTerms (GDI_NCIThesaurus), 1
getTerms, taggedHierNomenclature-method
(SGDIvocab), 4
gol.15DAG (LLGOMFcp), 2
GOMF1.15 (LLGOMFcp), 2
gomfAmat, 8
gomfAmat.1.15 (LLGOMFcp), 2
gomfAmat.1.4 (LLGOMFcp), 2
gomfGraph.1.15 (LLGOMFcp), 2
gomfGraph.1.4 (LLGOMFcp), 2
gomfGraphDemo (gomfAmat), 8
grep, character, taggedHierNomenclature-method
(SGDIvocab), 4
grList (compoundGraph-class), 6
grList, compoundGraph-method
(compoundGraph-class), 6
Iyer517 (ooMapLL2GOMFdemo), 11
litObj (litOnto), 8
litOnto, 8
LL2GOMFcp1.15 (LLGOMFcp), 2
LL2GOMFcp1.4 (LLGOMFcp), 2
LL2GOMFoop1.15 (LLGOMFcp), 2
LL2GOMFoop1.15 (LLGOMFcp), 2
LL2GOMFcp, 2
makeNamedSparse
(namedSparse-class), 9
makeOntology (ontology-class), 10
makeOOC (OOC-class), 3
makeSparseZero
(namedSparse-class), 9
mapNamesInds (namedSparse-class), 9
mat (namedSparse-class), 9
mat, namedSparse-method
(namedSparse-class), 9
maxval (namedSparse-class), 9
maxval, matrix.csr-method
(namedSparse-class), 9
mkNS (namedSparse-class), 9
name (ontology-class), 10
name, ontology-method
(ontology-class), 10
namedSparse-class, 9
cmp, namedSparse-method
(namedSparse-class), 9
newadj (accessMat), 5
nomenclature-class (SGDIvocab), 4
OOC-method, namedSparse-method
(namedSparse-class), 9
nsparse (namedSparse-class), 9
ontoDepth (depthStruct), 7
ontology (ontology-class), 10
ontology, OOC-method (OOC-class), 3
ontology-class, 10
OOC-class, 3
OOMap (OOC-class), 3
OOMap, OOC-method (OOC-class), 3
ooMapLL2GOMFdemo, 11
otkvEnv2namedSparse, 11
otkvList2namedSparse
(otkvEnv2namedSparse), 11
OVersion (ontology-class), 10
OVersion, ontology-method
(ontology-class), 10
parents (GDI_NCIThesaurus), 1
parents, character, taggedHierNomenclature-method
(taggedHierNomenclature-class), 14
pms (semsim), 12
provStruct (GDI_NCIThesaurus), 1
provStruct-class (SGDIvocab), 4
rDAG (ontology-class), 10
rDAG, ontology-method
(ontology-class), 10
revArcs (accessMat), 5
root (rootedDAG-class), 12
root, rootedDAG-method
(rootedDAG-class), 12
rootedDAG-class, 12
rowinds (namedSparse-class), 9
rowinds, namedSparse, character-method
(namedSparse-class), 9
INDEX

rowSums (namedSparse-class), 9
rowSums, namedSparse, missing, missing-method (namedSparse-class), 9
rowSumsSp (namedSparse-class), 9
semsim, 12
SGDIvocab, 4
show, exptArchive-method (SGDIvocab), 4
show, exptSample-method (SGDIvocab), 4
show, GDIPlatform-method (SGDIvocab), 4
show, namedSparse-method (namedSparse-class), 9
show, nomenclature-method (GDI_NCIThesaurus), 1
show, ontology-method (ontology-class), 10
show, OOC-method (OOC-class), 3
show, provStruct-method (SGDIvocab), 4
show, symMapping-method (SGDIvocab), 4
show, taggedHierNomenclature-method (taggedHierNomenclature-class), 14
STMA, 4
subsumers (semsim), 12
sumSp (namedSparse-class), 9
sumSpSLOW (namedSparse-class), 9
symMapping-class (SGDIvocab), 4
t, namedSparse-method (namedSparse-class), 9
taggedHierNomenclature, 2, 4
taggedHierNomenclature (taggedHierNomenclature-class), 14
taggedHierNomenclature-class, 14
thesList (GDI_NCIThesaurus), 1
toDot (toDot-methods), 16
toDot, compoundGraph, character, list, list-method (compoundGraph-class), 6
toDot, compoundGraph, character, list, missing-method (compoundGraph-class), 6
toDot, compoundGraph, missing, list, missing-method (compoundGraph-class), 6
toDot, graphNEL, character, list, list-method (toDot-methods), 16
toDot, graphNEL, character, missing, list-method (toDot-methods), 16
toDot, graphNEL, missing, character, missing-method (toDot-methods), 16
toDot, graphNEL, missing, list, list-method (toDot-methods), 16
toDot, graphNEL, missing, list, missing-method (toDot-methods), 16
toDot, graphNEL, missing, missing, list-method (toDot-methods), 16
toDot, graphNEL, missing, missing, missing-method (toDot-methods), 16
toDot-methods, 16
typedSymMapping-class (SGDIvocab), 4
usageCount (semsim), 12

toDot, graphNEL, character, missing, missing-method (toDot-methods), 16

toDot, graphNEL, missing, character, missing-method (toDot-methods), 16

toDot, graphNEL, missing, list, list-method (toDot-methods), 16

toDot, graphNEL, missing, list, missing-method (toDot-methods), 16

toDot, graphNEL, missing, missing, list-method (toDot-methods), 16

toDot, graphNEL, missing, missing, missing-method (toDot-methods), 16