API Documentation

API Documentation

December 5, 2017

Contents

1 Module EveryGO

EveryGO workfow. In order to perform GO-terms enrichment analysis for one or more nodes/comparisons of the phylogeny stored in the XML file.

Requires:

- python 2.7[1] (tested with 2.7.6)
- lxml[2]
- GOtrimmer_xml
- GOwalker
- GoView

1.1 Functions

get_parser()

Arguments setting and parsing

Return Value

arguments list

(type=parser object)

makeuniverse(*xmlname*)

Make universe of GO-terms from the XML file

Parameters

`xmlname`: the xml filename

(type=string)

everygo(*compvalues*, *view*, *xmlname*, *Rpath*)

Make enrichment analysis for each comparison values

Parameters

`compvalues`: comparisons values returned by getcomp()

(type=string)

`view`: -view argument –> if view != 0 , graphical representation of enrichment will be processed

(type=int)

`xmlname`: the xml filename

(type=string)

main()

[1] https://www.python.org/downloads/

[2] https://github.com/lxml/lxml

1.2 Variables

Name	**Description**
__doc__	**Value:** `...`
__package__	**Value:** `None`

2 Module GOslimmer_xml

GOtrimmer script. In order to keep prokaryotic terms only in VCFtoGO universe output file, based on GOslim (GO-basic.obo)

Requires:

- VCFtoGO output file (GO-terms universe)
- GO-basic.obo file (GOslim)

2.1 Functions

get_parser()

Parse arguments

Return Value

arguments list

(type=parser object)

add_object(*d*, *outputFile*)

main()

2.2 Variables

Name	Description
__doc__	**Value:** `...`
__package__	**Value:** `None`

3 Module GOtrimmer

3.1 Functions

get_parser()

Arguments setting and parsing

Return Value

arguments list

(type=parser object)

add_object(*d*)

collect prokaryotic terms in GO-basic.obo

Parameters

`d`: go-term related data collected from GO-basic.obo

(type=object data)

3.2 Variables

Name	Description
__doc__	**Value:** `...`
parser	**Value:** `get_parser()`
Arguments	**Value:** `parser.parse_args()`
fname	**Value:** `"../db/go-basic.obo"`
term_head	**Value:** `"[Term]"`
all_objects	**Value:** `{}`
output	**Value:** `open("goprok.txt", "w")`
gofull	**Value:** `str(Arguments.input)`
outputuniv	**Value:** `open("VCFtoGOresults/univers.txt", "w")`
current	**Value:** `defaultdict(list)`

4 Module GetGOxML

GoXML Wrapper.

Requires:

- python 2.7[3] (tested with 2.7.6)
- PyVCF[4]
- lxml[5]
- GOslimmer_xml
- GoXML

4.1 Functions

get_parser()

Arguments setting and parsing

Return Value

arguments list

(type=parser object)

main()

4.2 Variables

Name	Description
__doc__	**Value:** `...`
__package__	**Value:** `None`

[3] https://www.python.org/downloads/
[4] https://github.com/jamescasbon/PyVCF
[5] https://github.com/lxml/lxml

5 Module GoXML

GoXML workfow + GOslimmer_xml wrapper. In order to add GO-terms from a XML file and generate a non-enriched universe of GO-terms.

Requires:

- python 2.7[6] (tested with 2.7.6)
- PyVCF[7]
- lxml[8]
- GOslimmer_xml

5.1 Functions

get_parser()

Arguments setting and parsing

Return Value

arguments list

(type=parser object)

readDBFile(*fichier*)

Read NP-EBI local storage file

Parameters

`fichier`: NP-EBI_table.tsv file

(type=string)

Return Value

dictionnary of NP identifiers

(type=dictionnary)

readDBGOFile(*fichier*)

Read EBI-GO local storage file

Parameters

`fichier`: EBI-GO_table.tsv file

(type=string)

Return Value

dictionnary of GO identifiers

(type=dictionnary)

[6] https://www.python.org/downloads/

[7] https://github.com/jamescasbon/PyVCF

[8] https://github.com/lxml/lxml

extractNP(*path*, *withHomoplasy*)

Retrieve NP ids from XML file

Parameters

`path`: relative path to the XML file

(type=string)

Return Value

dictionnary of NP identifiers

(type=dictionnary)

idmapping(*nplist*, *dicoNP*, *NPfile*)

Make ID mapping from NP identifiers to Uniprot identifiers requesting Online(Uniprot database) of Offline(NP-EBI_databases.tsv)

Parameters

`nplist`: list used in ONLINE request –> values = NP identifiers

(type=list)

`dicoNP`: dictionnary used in OFFLINE request –> keys = NP identifiers , values = uniprot id retrieved

(type=dictionnary)

`NPfile`: Storage of uniprot identifiers to NP identifiers (NP-EBI_databases.tsv)

(type=string)

Return Value

list of Uniprot identifiers

(type=list)

UNIPROTtoGO(*path*, *output*, *protlist*, *dicoGO*, *GOFile*, *GO_prok*)

Make ID mapping from Uniprot identifiers to GO identifiers requesting Online(Quick-GO database) of Offline(EBI-GO_databases.tsv)

Parameters

`path`: relative path to the XML file

(type=string)

`output`: Universe of GO identifiers (univers.txt)

(type=string)

`protlist`: list used in ONLINE request –> values = UNIPROT identifiers

(type=list)

`dicoGO`: dictionnary used in ONLINE requests –> keys = Uniprot identifiers , values = GO identifiers list

(type=dictionnary)

`GOFile`: Storage of uniprot identifiers to GO identifiers (EBI-GO_databases.tsv)

(type=string)

main()

5.2 Variables

Name	Description
__doc__	**Value:** `...`
__package__	**Value:** `None`

6 Module VCFtoGO

VCFtoGO workfow. In order to generate non-enriched universe of GO-terms from a VCF.

Requires:

- python 2.7[9] (tested with 2.7.6)
- PyVCF[10]
- GOtrimmer

6.1 Functions

get_parser()

Arguments setting and parsing

Return Value

arguments list

(type=parser object)

readDBFile(*fichier*)

Read NP-EBI local storage file

Parameters

`fichier`: NP-EBI_databases.tsv file

(type=.tsv file)

Return Value

dictionnary of NP identifiers

(type=dictionnary)

readDBGOFile(*fichier*)

Read EBI-GO local storage file

Parameters

`fichier`: EBI-GO_databases.tsv file

(type=.tsv file)

Return Value

dictionnary of GO identifiers

(type=dictionnary)

[9] https://www.python.org/downloads/

[10] https://github.com/jamescasbon/PyVCF

extractinfopos(*fichier*, *pos*)

Parse NP identifiers from VCF file

Parameters

`fichier`: VCF file

(type=.vcf file)

Return Value

dictionnary of NP identifiers

(type=dictionnary)

idmapping(*dico*, *dicoNP*, *NPfile*)

Make ID mapping from NP identifiers to Uniprot identifiers requesting Online(Uniprot database) of Offline(NP-EBI_databases.tsv)

Parameters

`dico`: dictionnary used in ONLINE request –> keys = NP identifiers , values = uniprot id retrieved

(type=dictionnary)

`dicoNP`: dictionnary used in OFFLINE request –> keys = NP identifiers , values = uniprot id retrieved

(type=dictionnary)

`NPfile`: Storage of uniprot identifiers to NP identifiers (NP-EBI_databases.tsv)

(type=.tsv file)

UNIPROTtoGO(*dico*, *output*, *pos*, *dicoGO*, *GOfile*)

Make ID mapping from Uniprot identifiers to GO identifiers requesting Online(Quick-GO database) of Offline(EBI-GO_databases.tsv)

Parameters

`dico`: dictionnary used in OFFLINE requests –> keys = Uniprot identifiers , values = GO identifiers list

(type=dictionnary)

`output`: Universe of GO identifiers (univers.txt)

(type=.txt file)

`pos`: positions requested if -pos argument was used

(type=int)

`dicoGO`: dictionnary used in ONLINE requests –> keys = Uniprot identifiers , values = GO identifiers list

(type=dictionnary)

`GOfile`: Storage of uniprot identifiers to GO identifiers (EBI-GO_databases.tsv)

(type=.tsv file)

main()

6.2 Variables

Name	Description
__doc__	**Value:** `...`
__package__	**Value:** `None`

Index