Package ‘rotl’

October 14, 2022

Title Interface to the 'Open Tree of Life' API

Version 3.0.14

Description An interface to the 'Open Tree of Life' API to retrieve phylogenetic trees, information about studies used to assemble the synthetic tree, and utilities to match taxonomic names to 'Open Tree identifiers'. The 'Open Tree of Life' aims at assembling a comprehensive phylogenetic tree for all named species.

License BSD_2_clause + file LICENSE

URL https://docs.ropensci.org/rotl/, https://github.com/ropensci/rotl

BugReports https://github.com/ropensci/rotl/issues

Depends R (>= 3.1.1)

Imports ape, curl (>= 3.0.0), httr, jsonlite, rentrez, rlang, rncl (>= 0.6.0)

Suggests knitr (>= 1.12), MCMCglmm, phylobase, readxl, rmarkdown (>= 0.7), RNeXML, testthat

VignetteBuilder knitr

Encoding UTF-8

X-schema.org-isPartOf https://ropensci.org

X-schema.org-keywords metadata, ropensci, phylogenetics, independant-contrasts, biodiversity

X-schema.org-relatedLink https://codemeta.github.io/codemetar

RoxygenNote 7.2.1

NeedsCompilation no

Author Francois Michonneau [aut, cre] (<https://orcid.org/0000-0002-9092-966X>),
Joseph Brown [aut] (<https://orcid.org/0000-0002-3835-8062>),
David Winter [aut] (<https://orcid.org/0000-0002-6165-0029>),
Scott Chamberlain [rev] (<https://orcid.org/0000-0003-1444-9135>)

Maintainer Francois Michonneau <francois.michonneau@gmail.com>

Repository CRAN

Date/Publication 2022-10-06 20:40:02 UTC

R topics documented:

`get_study` *Get all the trees associated with a particular study*

Description

Returns the trees associated with a given study

Usage

```
get_study(
  study_id = NULL,
  object_format = c("phylo", "nexml"),
  file_format,
  file,
  ...
)
```

Arguments

`study_id`	the study ID for the study of interest (character)
`object_format`	the class of the object the query should return (either `phylo` or `nexml`). Ignored if `file_format` is specified.
`file_format`	the format of the file to be generated (`newick`, `nexus`, `nexml` or `json`).
`file`	the file name where the output of the function will be saved.
`...`	additional arguments to customize the API request (see `rotl` package documentation).

Details

If `file_format` is missing, the function returns an object of the class `phylo` from the `ape` package (default), or an object of the class `nexml` from the `RNeXML` package.

Otherwise `file_format` can be either `newick`, `nexus`, `nexml` or `json`, and the function will generate a file of the selected format. In this case, a file name needs to be provided using the argument `file`. If a file with the same name already exists, it will be silently overwritten.

Value

if `file_format` is missing, an object of class `phylo` or `nexml`, otherwise a logical indicating whether the file was successfully created.

See Also

`get_study_meta`

Examples

```
## Not run:
that_one_study <- get_study(study_id="pg_719", object_format="phylo")
if (require(RNeXML)) { ## if RNeXML is installed get the object directly
   nexml_study <- get_study(study_id="pg_719", object_format="nexml")
} else { ## otherwise write it to a file
   get_study(study_id="pg_719", file_format="nexml", file=tempfile(fileext=".nexml"))
}

## End(Not run)
```

`get_study_subtree` *Study Subtree*

Description

Retrieve subtree from a specific tree in the Open Tree of Life data store

Usage

```
get_study_subtree(
  study_id,
  tree_id,
  subtree_id,
  object_format = c("phylo"),
  tip_label = c("original_label", "ott_id", "ott_taxon_name"),
  file_format,
  file,
  deduplicate = TRUE,
  ...
)
```

Arguments

`study_id`	the study identifier (character)
`tree_id`	the tree identifier (character)
`subtree_id,`	either a node id that specifies a subtree or “ingroup” which returns the ingroup for this subtree.
`object_format`	the class of the object returned by the function (default, and currently only possibility `phylo` from the `ape` package)
`tip_label`	the format of the tip labels. “`original_label`” (default) returns the original labels as provided in the study, “`ott_id`” labels are replaced by their ott IDs, “`ott_taxon_name`” labels are replaced by their Open Tree Taxonomy taxon name.
`file_format`	character, the file format to use to save the results of the query (possible values, ‘newick’ or ‘nexus’).
`file`	character, the path and file name where the output should be written.
`deduplicate`	logical (default `TRUE`). If the tree returned by the study contains duplicated taxon names, should they be made unique? It is normally illegal for NEXUS/Newick tree strings to contain duplicated tip names. This is a workaround to circumvent this requirement. If `TRUE`, duplicated tip labels will be appended `_1`, `_2`, etc.
`...`	additional arguments to customize the API request (see `rotl` package documentation).

Examples

```
## Not run:
small_tr <- get_study_subtree(study_id="pg_1144", tree_id="tree5800", subtree_id="node991044")
ingroup  <- get_study_subtree(study_id="pg_1144", tree_id="tree5800", subtree_id="ingroup")
nexus_file <- tempfile(fileext=".nex")
get_study_subtree(study_id="pg_1144", tree_id="tree5800", subtree_id="ingroup", file=nexus_file,
                  file_format="nexus")

## End(Not run)
```

`get_study_tree` *Study Tree*

Description

Returns a specific tree from within a study

Usage

```
get_study_tree(
  study_id = NULL,
  tree_id = NULL,
  object_format = c("phylo"),
  tip_label = c("original_label", "ott_id", "ott_taxon_name"),
  file_format,
  file,
  deduplicate = TRUE,
  ...
)
```

Arguments

`study_id`	the identifier of a study (character)
`tree_id`	the identifier of a tree within the study
`object_format`	the class of the object to be returned (default and currently only possible value `phylo` from the `ape` package).
`tip_label`	the format of the tip labels. “`original_label`” (default) returns the original labels as provided in the study, “`ott_id`” labels are replaced by their ott IDs, “`ott_taxon_name`” labels are replaced by their Open Tree Taxonomy taxon name.
`file_format`	the format of the file to be generated (`newick` default, `nexus`, or `json`).
`file`	the file name where the output of the function will be saved.
`deduplicate`	logical (default `TRUE`). If the tree returned by the study contains duplicated taxon names, should they be made unique? It is normally illegal for NEXUS/Newick tree strings to contain duplicated tip names. This is a workaround to circumvent this requirement. If `TRUE`, duplicated tip labels will be appended `_1`, `_2`, etc.

`...`	additional arguments to customize the API request (see `rotl` package documentation).

Value

if `file_format` is missing, an object of class `phylo`, otherwise a logical indicating whether the file was successfully created.

Examples

```
## Not run:
 tree <- get_study_tree(study_id="pg_1144", tree_id="tree2324")

 ## comparison of the first few tip labels depending on the options used
 head(get_study_tree(study_id="pg_1144", tree_id="tree2324", tip_label="original_label")$tip.label)
 head(get_study_tree(study_id="pg_1144", tree_id="tree2324", tip_label="ott_id")$tip.label)
 head(get_study_tree(study_id="pg_1144", tree_id="tree2324", tip_label="ott_taxon_name")$tip.label)

## End(Not run)
```

`get_tree_ids` *Study Metadata*

Description

Retrieve metadata about a study in the Open Tree of Life datastore.

Usage

```
get_tree_ids(sm)

get_publication(sm)

candidate_for_synth(sm)

get_study_year(sm)

## S3 method for class 'study_meta'
get_tree_ids(sm)

## S3 method for class 'study_meta'
get_publication(sm)

## S3 method for class 'study_meta'
candidate_for_synth(sm)

## S3 method for class 'study_meta'
get_study_year(sm)

get_study_meta(study_id, ...)
```

Arguments

`sm`	an object created by `get_study_meta`
`study_id`	the study identifier (character)
`...`	additional arguments to customize the API request (see `rotl` package documentation).

Details

`get_study_meta` returns a long list of attributes for the studies that are contributing to the synthetic tree. To help with the extraction of relevant information from this list, several helper functions exists:

- get_tree_ids The identifiers of the trees associated with the study
- get_publication The citation information of the publication for the study. The DOI (or URL) for the study is available as an attribute to the returned object (i.e., `attr(object, "DOI")`).
- candidate_for_synth The identifier of the tree(s) from the study used in the synthetic tree. This is a subset of the result of `get_tree_ids`.

  get_study_year The year of publication of the study.

Value

named-list containing the metadata associated with the study requested

Examples

```
## Not run:
req <- get_study_meta("pg_719")
get_tree_ids(req)
candidate_for_synth(req)
get_publication(req)
get_study_year(req)

## End(Not run)
```

`inspect.match_names` *Inspect and Update alternative matches for a name returned by tnrs_match_names*

Description

Taxonomic names may have different meanings in different taxonomic contexts, as the same genus name can be applied to animals and plants for instance. Additionally, the meaning of a taxonomic name may have change throughout its history, and may have referred to a different taxon in the past. In such cases, a given names might have multiple matches in the Open Tree Taxonomy. These functions allow users to inspect (and update) alternative meaning of a given name and its current taxonomic status according to the Open Tree Taxonomy.

Usage

```
## S3 method for class 'match_names'
inspect(response, row_number, taxon_name, ott_id, ...)

inspect(response, ...)

## S3 method for class 'match_names'
update(object, row_number, taxon_name, ott_id, new_row_number, new_ott_id, ...)
```

Arguments

`response`	an object generated by the `tnrs_match_names` function
`row_number`	the row number corresponding to the name to inspect
`taxon_name`	the taxon name corresponding to the name to inspect
`ott_id`	the ott id corresponding to the name to inspect
`...`	currently ignored
`object`	an object created by `tnrs_match_names`
`new_row_number`	the row number in the output of `inspect` to replace the taxa specified by `row_number`, `taxon_name`, or `ott_id`.
`new_ott_id`	the ott id of the taxon to replace the taxa specified by `row_number`, `taxon_name`, or `ott_id`.

Details

To inspect alternative taxonomic meanings of a given name, you need to provide the object resulting from a call to the tnrs_match_names function, as well as one of either the row number corresponding to the name in this object, the name itself (as used in the original query), or the ott_id listed for this name.

To update one of the name, you also need to provide the row number in which the name to be replaced appear or its ott id.

Value

a data frame

See Also

`tnrs_match_names`

Examples

```
## Not run:
 matched_names <- tnrs_match_names(c("holothuria", "diadema", "boletus"))
 inspect(matched_names, taxon_name="diadema")
 new_matched_names <- update(matched_names, taxon_name="diadema",
                             new_ott_id = 631176)
 new_matched_names
```

```
## End(Not run)
```

`is_in_tree` *Check that OTT ids occur in the Synthetic Tree*

Description

Some valid taxonomic names do not occur in the Synthetic Tree. This convenience function allows you to check whether a given Open Tree Taxonomy identifier (OTT id) is in the tree. A taxonomic name may not occur in the synthetic tree because (1) it is an extinct or invalid taxon, or (2) it is part of a group that is not monophyletic in the tree.

Usage

```
is_in_tree(ott_ids, ...)
```

Arguments

`ott_ids`	a vector of Open Tree Taxonomy identifiers
`...`	additional arguments to customize the API request (see `rotl` package documentation).

Value

A named logical vector. `TRUE` indicates that the OTT id is in the synthetic tree, and `FALSE` that it is not.

Examples

```
## Not run:
  plant_families <- c("Asteraceae", "Solanaceae", "Poaceae", "Amaranthaceae",
                      "Zamiaceae", "Araceae", "Juncaceae")
  matched_names <- tnrs_match_names(plant_families)
  ## This fails because some ott ids are not in the tree
  ## plant_tree <- tol_induced_subtree(ott_id(matched_names))
  ## So let's check which ones are actually in the tree first:
  in_tree <- is_in_tree(ott_id(matched_names))
  ## This now works:
  plant_tree <- tol_induced_subtree(ott_id(matched_names)[in_tree])

## End(Not run)
```

`list_trees` *List trees ids in objects returned by `studies_find_studies` and `studies_find_trees`.*

Description

`list_trees` returns all trees associated with a particular study when used on an object returned by `studies_find_studies`, but only the trees that match the search criteria when used on objects returned by `studies_find_trees`.

Usage

```
list_trees(matched_studies, ...)

## S3 method for class 'matched_studies'
list_trees(matched_studies, study_id, ...)
```

Arguments

`matched_studies`	an object created by `studies_find_trees` or `studies_find_studies`.
`...`	Currently unused
`study_id`	a `study_id` listed in the object returned by `studies_find_trees`

Value

`list_trees` returns a list of the tree_ids for each study that match the requested criteria. If a `study_id` is provided, then only the trees for this study are returned as a vector.

See Also

`studies_find_studies` and `studies_find_trees`. The help for these functions have examples demonstrating the use of `list_trees`.

`ott_id.match_names` `ott_id` *and* `flags` *for taxonomic names matched by* `tnrs_match_names`

Description

`rotl` provides a collection of functions that allows users to extract relevant information from an object generated by `tnrs_match_names` function.

Usage

```
## S3 method for class 'match_names'
ott_id(tax, row_number, taxon_name, ott_id, ...)

## S3 method for class 'match_names'
flags(tax, row_number, taxon_name, ott_id, ...)

flags(tax, ...)
```

Arguments

`tax`	an object returned by `tnrs_match_names`
`row_number`	the row number corresponding to the name for which to list the synonyms
`taxon_name`	the taxon name corresponding to the name for which to list the synonyms
`ott_id`	the ott id corresponding to the name for which to list the synonyms
`...`	currently ignored

Details

These methods optionally accept one of the arguments `row_number`, `taxon_name` or `ott_id` to retrieve the corresponding information for one of the matches in the object returned by the `tnrs_match_names` function.

If these arguments are not provided, these methods can return information for the matches currently listed in the object returned by `tnrs_match_names`.

Value

A list of the ott ids or flags for the taxonomic names matched with `tnrs_match_names`, for either one or all the names.

Examples

```
## Not run:
  rsp <- tnrs_match_names(c("Diadema", "Tyrannosaurus"))
  rsp$ott_id    # ott id for match currently in use
  ott_id(rsp)   # similar as above but elements are named

  ## flags() is useful for instance to determine if a taxon is extinct
  flags(rsp, taxon_name="Tyrannosaurus")

## End(Not run)
```

`rotl` *An Interface to the Open Tree of Life API*

Description

The Open Tree of Life is an NSF funded project that is generating an online, comprehensive phylogenetic tree for 1.8 million species. `rotl` provides an interface that allows you to query and retrieve the parts of the tree of life that is of interest to you.

Details

`rotl` provides function to most of the end points the API provides. The documentation of the API is available at: `https://github.com/OpenTreeOfLife/opentree/wiki/Open-Tree-of-Life-APIs`

Customizing API calls

All functions that use API end points can take 2 arguments to customize the API call and are passed as `...` arguments.

- `otl_v` This argument controls which version of the API your call is using. The default value for this argument is a call to the non-exported function `otl_version()` which returns the current version of the Open Tree of Life APIs (v2).
- `dev_url` This argument controls whether to use the development version of the API. By default, `dev_url` is set to `FALSE`, using `dev_url = TRUE` in your function calls will use the development version.

For example, to use the development version of the API, you could use: `tnrs_match_names("anas", dev_url=TRUE)`

Additional arguments can also be passed to the `GET` and `POST` methods.

Acknowledgments

This package was started during the Open Tree of Life Hackathon organized by OpenTree, the NESCent Hackathon Interoperability Phylogenetic group, and Arbor.

`source_list` *List of studies used in the Tree of Life*

Description

Retrieve the detailed information for the list of studies used in the Tree of Life.

Usage

```
source_list(tax, ...)

## S3 method for class 'tol_summary'
source_list(tax, ...)
```

Arguments

`tax`	a list containing a `source_id_map` slot.
`...`	additional arguments (currently unused)

Details

This function takes the object resulting from `tol_about(study_list = TRUE)`, `tol_mrca()`, `tol_node_info()`, and returns a data frame listing the `tree_id`, `study_id` and `git_sha` for the studies currently included in the Tree of Life.

Value

a data frame

`strip_ott_ids` *Strip OTT ids from tip labels*

Description

Strip OTT ids from tip labels

Usage

```
strip_ott_ids(tip_labels, remove_underscores = FALSE)
```

Arguments

`tip_labels`	a character vector containing tip labels (most likely the `tip.label` element from a tree returned by `tol_induced_subtree`
`remove_underscores`	logical (defaults to FALSE). If set to TRUE underscores in tip labels are converted to spaces

Value

A character vector containing the contents of `tip_labels` with any OTT ids removed.

Examples

```
## Not run:
genera <- c("Perdix", "Setophaga", "Cinclus", "Struthio")
tr <- tol_induced_subtree(ott_ids=c(102710, 285198, 267845, 292466))
tr$tip.label %in% genera
tr$tip.label <- strip_ott_ids(tr$tip.label)
tr$tip.label %in% genera

## End(Not run)
```

`studies_find_studies` *Find a Study*

Description

Return the identifiers of studies that match given properties

Usage

```
studies_find_studies(
  property = NULL,
  value = NULL,
  verbose = FALSE,
  exact = FALSE,
  detailed = TRUE,
  ...
)
```

Arguments

`property`	The property to be searched on (character)
`value`	The property value to be searched on (character)
`verbose`	Should the output include all metadata (logical default `FALSE`)
`exact`	Should exact matching be used? (logical, default `FALSE`)
`detailed`	If `TRUE` (default), the function will return a data frame that summarizes information about the study (see ‘Value’). Otherwise, it only returns the study identifiers.
`...`	additional arguments to customize the API request (see `rotl` package documentation).

Value

If `detailed=TRUE`, the function returns a data frame listing the study id (`study_ids`), the number of trees associated with this study (`n_trees`), the tree ids (at most 5) associated with the studies (`tree_ids`), the tree id that is a candidate for the synthetic tree if any (`candidate`), the year of publication of the study (`study_year`), the title of the publication for the study (`title`), and the DOI (Digital Object Identifier) for the study (`study_doi`).

If `detailed=FALSE`, the function returns a data frame with a single column containing the study identifiers.

See Also

`studies_properties` which lists properties against which the studies can be searched. `list_trees` that returns a list for all tree ids associated with a study.

Examples

```
## Not run:
## To match a study for which the identifier is already known
one_study <- studies_find_studies(property="ot:studyId", value="pg_719")
list_trees(one_study)

## To find studies pertaining to Mammals
mammals <- studies_find_studies(property="ot:focalCladeOTTTaxonName",
                                value="mammalia")
## To extract the tree identifiers for each of the studies
list_trees(mammals)
## ... or for a given study
list_trees(mammals, "ot_308")

## Just the identifiers without other information about the studies
mammals <- studies_find_studies(property="ot:focalCladeOTTTaxonName",
                                value="mammalia", detailed=FALSE)

## End(Not run)
```

`studies_find_trees` *Find Trees*

Description

Return a list of studies for which trees match a given set of properties

Usage

```
studies_find_trees(
  property = NULL,
  value = NULL,
  verbose = FALSE,
  exact = FALSE,
  detailed = TRUE,
  ...
)
```

Arguments

`property`	The property to be searched on (character)
`value`	The property-value to be searched on (character)
`verbose`	Should the output include all metadata? (logical, default `FALSE`)
`exact`	Should exact matching be used for the value? (logical, default `FALSE`)
`detailed`	Should a detailed report be provided? If `TRUE` (default), the output will include metadata about the study that include trees matching the property. Otherwise, only information about the trees will be provided.
`...`	additional arguments to customize the API request (see `rotl` package documentation).

Details

The list of possible values to be used as values for the argument `property` can be found using the function `studies_properties`.

Value

A data frame that summarizes the trees found (and their associated studies) for the requested criteria. If a study has more than 5 trees, the `tree_ids` of the first ones will be shown, followed by `...` to indicate that more are present.

If `detailed=FALSE`, the data frame will include the study ids of the study (`study_ids`), the number of trees in this study that match the search criteria (`n_matched_trees`), the tree ids that match the search criteria (`match_tree_ids`).

If `detailed=TRUE`, in addition of the fields listed above, the data frame will also contain the total number of trees associated with the study (`n_trees`), all the tree ids associated with the study (`tree_ids`), the tree id that is a potential candidate for inclusion in the synthetic tree (if any) (`candidate`), the year the study was published (`study_year`), the title of the study (`title`), the DOI for the study (`study_doi`).

See Also

`studies_properties` which lists properties the studies can be searched on. `list_trees` for listing the trees that match the query.

Examples

```
## Not run:
res <- studies_find_trees(property="ot:ottTaxonName", value="Drosophila",
                          detailed=FALSE)
## summary of the trees and associated studies that match this criterion
res
## With metadata about the studies (default)
res <- studies_find_trees(property="ot:ottTaxonName", value="Drosophila",
                          detailed=TRUE)
## The list of trees for each study that match the search criteria
list_trees(res)
## the trees for a given study
```

```
list_trees(res, study_id = "pg_2769")

## End(Not run)
```

`studies_properties` *Properties of the Studies*

Description

Return the list of study properties that can be used to search studies and trees used in the synthetic tree.

Usage

```
studies_properties(...)
```

Arguments

`...`	additional arguments to customize the API request (see `rotl` package documentation).

Details

The list returned has 2 elements `tree_properties` and `studies_properties`. Each of these elements lists additional arguments to customize the API request properties that can be used to search for trees and studies that are contributing to the synthetic tree. The definitions of these properties are available from `https://github.com/OpenTreeOfLife/phylesystem-api/wiki/NexSON`

Value

A list of the study properties that can be used to find studies and trees that are contributing to the synthetic tree.

See Also

`studies_find_trees`

Examples

```
## Not run:
 all_the_properties <- studies_properties()
 unlist(all_the_properties$tree_properties)

## End(Not run)
```

`study_external_IDs` *Get external identifiers for data associated with an Open Tree study*

Description

Data associated with studies contributing to the Open Tree synthesis may be available from other databases. In particular, trees and alignments may be available from treebase and DNA sequences and bibliographic information associated with a given study may be available from the NCBI. This function retrieves that information for a given study.

Usage

```
study_external_IDs(study_id)
```

Arguments

`study_id`	An open tree study ID

Value

A study_external_data object (which inherits from a list) which contains some of the following.

doi, character, the DOI for the paper describing this study

external_data_url, character, a URL to an external data repository (e.g. a treebase entry) if one exists.

pubmed_id character, the unique ID for this study in the NCBI's pubmed database

popset_ids character, vector of IDs for the NCBI's popset database

nucleotide_ids character, vector of IDs for the NCBI's nucleotide database

See Also

studies_find_studies (used to discover study IDs)

Examples

```
## Not run:
flies <- studies_find_studies(property="ot:focalCladeOTTTaxonName", value="Drosophilidae")
study_external_IDs(flies[2,]$study_ids)

## End(Not run)
```

`synonyms.match_names` *List the synonyms for a given name*

Description

When querying the Taxonomic Name Resolution Services for a particular taxonomic name, the API returns as possible matches all names that include the queried name as a possible synonym. This function allows you to explore other synonyms for an accepted name, and allows you to determine why the name you queried is returning an accepted synonym.

Usage

```
## S3 method for class 'match_names'
synonyms(tax, row_number, taxon_name, ott_id, ...)
```

Arguments

`tax`	a data frame generated by the `tnrs_match_names` function
`row_number`	the row number corresponding to the name for which to list the synonyms
`taxon_name`	the taxon name corresponding to the name for which to list the synonyms
`ott_id`	the ott id corresponding to the name for which to list the synonyms
`...`	currently ignored

Details

To list synonyms for a given taxonomic name, you need to provide the object resulting from a call to the `tnrs_match_names` function, as well as one of either the row number corresponding to the name in this object, the name itself (as used in the original query), or the ott_id listed for this name. Otherwise, the synonyms for all the currently matched names are returned.

Value

a list whose elements are all synonym names (as vectors of character) for the taxonomic names that match the query (the names of the elements of the list).

Examples

```
## Not run:
   echino <- tnrs_match_names(c("Diadema", "Acanthaster", "Fromia"))
   ## These 3 calls are identical
   synonyms(echino, taxon_name="Acanthaster")
   synonyms(echino, row_number=2)
   synonyms(echino, ott_id=337928)

## End(Not run)
```

`taxonomy_about` *Information about the Open Tree Taxonomy*

Description

Summary information about the Open Tree Taxonomy (OTT)

Usage

```
taxonomy_about(...)
```

Arguments

`...`	additional arguments to customize the API request (see `rotl` package documentation).

Details

Return metadata and information about the taxonomy itself. Currently, the available metadata is fairly sparse, but includes (at least) the version, and the location from which the complete taxonomy source files can be downloaded.

Value

A list with the following properties:

- weburl String. The release page for this version of the taxonomy.
- author String. The author string.
- name String. The name of the taxonomy.
- source String. The full identifying information for this version of the taxonomy.
- version String. The version number of the taxonomy.

Examples

```
## Not run:
taxonomy_about()

## End(Not run)
```

`taxonomy_mrca` *Taxonomic MRCA*

Description

Taxonomic Least Inclusive Common Ancestor (MRCA)

Usage

```
taxonomy_mrca(ott_ids = NULL, ...)

## S3 method for class 'taxon_mrca'
tax_rank(tax, ...)

## S3 method for class 'taxon_mrca'
tax_name(tax, ...)

## S3 method for class 'taxon_mrca'
ott_id(tax, ...)

## S3 method for class 'taxon_mrca'
unique_name(tax, ...)

## S3 method for class 'taxon_mrca'
tax_sources(tax, ...)

## S3 method for class 'taxon_mrca'
flags(tax, ...)

## S3 method for class 'taxon_mrca'
is_suppressed(tax, ...)
```

Arguments

`ott_ids`	a vector of ott ids for the taxa whose MRCA is to be found (numeric).
`...`	additional arguments to customize the API request (see `rotl` package documentation).
`tax`	an object generated by the `taxonomy_mrca` function

Details

Given a set of OTT ids, get the taxon that is the most recent common ancestor (the MRCA) of all the identified taxa.

Value

- `taxonomy_mrca` returns a list about the taxonomic information relating to the MRCA for the ott_ids provided.
- `tax_rank` returns a character vector of the taxonomic rank for the MRCA.
- `tax_name` returns a character vector the Open Tree Taxonomy name for the MRCA.
- `ott_id` returns a numeric vector of the ott id for the MRCA.

Examples

```
## Not run:
req <- taxonomy_mrca(ott_ids=c(515698,590452,643717))
tax_rank(req)
tax_name(req)
ott_id(req)

## End(Not run)
```

`taxonomy_subtree` *Taxonomy subtree*

Description

Given an ott id, return the inclusive taxonomic subtree descended from the specified taxon.

Usage

```
taxonomy_subtree(
  ott_id = NULL,
  output_format = c("taxa", "newick", "phylo", "raw"),
  label_format = NULL,
  file,
  ...
)
```

Arguments

`ott_id`	The ott id of the taxon of interest.
`output_format`	the format of the object to be returned. See the 'Return' section.
`label_format`	Character. Defines the label type; one of "`name`", "`id`", or "`name_and_id`" (the default).
`file`	the file name where to save the output of the function. Ignored unless `output_format` is set to "`phylo`".
`...`	additional arguments to customize the API request (see `rotl` package documentation).

Details

If the output of this function is exported to a file, the only possible value for the `output_format` argument is "`newick`". If the file provided already exists, it will be silently overwritten.

Value

If the `file` argument is missing:

- "`taxa`" a list of the taxa names (species) in slot `tip_label`, and higher-level taxonomy (e.g., families, genera) in slot `edge_label`, descending from the taxa corresponding to the `ott_id` provided.
- "`newick`" a character vector containing the newick formatted string corresponding to the taxonomic subtree for the `ott_id` provided.
- "`phylo`" an object of the class `phylo` from the `ape` package.
- "`raw`" the direct output from the API, i.e., a list with an element named 'newick' that contains the subtree as a newick formatted string.

If a `file` argument is provided (and `output_format` is set to "`phylo`"), a logical indicating whether the file was successfully created.

Examples

```
## Not run:
req <- taxonomy_subtree(ott_id=515698)
plot(taxonomy_subtree(ott_id=515698, output_format="phylo"))

## End(Not run)
```

`taxonomy_taxon_info` *Taxon information*

Description

Information about taxa.

Usage

```
taxonomy_taxon_info(
  ott_ids,
  include_children = FALSE,
  include_lineage = FALSE,
  include_terminal_descendants = FALSE,
  ...
)

## S3 method for class 'taxon_info'
tax_rank(tax, ...)
```

```
## S3 method for class 'taxon_info'
tax_name(tax, ...)

## S3 method for class 'taxon_info'
unique_name(tax, ...)

## S3 method for class 'taxon_info'
synonyms(tax, ...)

## S3 method for class 'taxon_info'
ott_id(tax, ...)

## S3 method for class 'taxon_info'
tax_sources(tax, ...)

## S3 method for class 'taxon_info'
is_suppressed(tax, ...)

## S3 method for class 'taxon_info'
flags(tax, ...)
```

Arguments

`ott_ids`	the ott ids of the taxon of interest (numeric or character containing only numbers)
`include_children`	whether to include information about all the children of this taxon. Default `FALSE`.
`include_lineage`	whether to include information about all the higher level taxa that include the `ott_ids`. Default `FALSE`.
`include_terminal_descendants`	whether to include the list of terminal `ott_ids` contained in the `ott_ids` provided.
`...`	additional arguments to customize the API request (see `rotl` package documentation).
`tax`	an object generated by the `taxonomy_taxon_info` function

Details

Given a vector of ott ids, `taxonomy_taxon_info` returns information about the specified taxa.

The functions `tax_rank`, `tax_name`, and `synonyms` can extract this information from an object created by the `taxonomy_taxon_info()`.

Value

`taxonomy_taxon_info` returns a list detailing information about the taxa. `tax_rank` and `tax_name` return a vector. `synonyms` returns a list whose elements are the synonyms for each of the `ott_id` requested.

See Also

`tnrs_match_names` to obtain `ott_id` from a taxonomic name.

Examples

```
## Not run:
req <- taxonomy_taxon_info(ott_id=515698)
tax_rank(req)
tax_name(req)
synonyms(req)

## End(Not run)
```

`taxon_external_IDs` *Get external identifiers for data associated with an Open Tree taxon*

Description

The Open Tree taxonomy is a synthesis of multiple reference taxonomies. This function retrieves identifiers to external taxonomic records that have contributed the rank, position and definition of a given Open Tree taxon.

Usage

```
taxon_external_IDs(taxon_id)
```

Arguments

`taxon_id`	An open tree study ID

Value

a data.frame in which each row represents a unique record in an external database. The column "source" provides and abbreviated name for the database, and "id" the unique ID for the record.

See Also

tnrs_matchnames, which can be used to search for taxa by name.

taxonomy_taxon, for more information about a given taxon.

Examples

```
## Not run:
   gibbon_IDs <- taxon_external_IDs(712902)

## End(Not run)
```

`tax_lineage` *Lineage of a taxon*

Description

Extract the lineage information (higher taxonomy) from an object returned by `taxonomy_taxon_info`.

Usage

```
tax_lineage(tax, ...)

## S3 method for class 'taxon_info'
tax_lineage(tax, ...)
```

Arguments

`tax`	an object created by `taxonomy_taxon_info` using the argument `include_lineage=TRUE`.
`...`	additional arguments (currently unused).

Details

The object passed to this function must have been created using the argument `include_lineage=TRUE`.

Value

A list with one slot per taxon that contains a data frame with 3 columns: the taxonomy rank, the name, and unique name for all taxa included in the lineage of the taxon up to the root of the tree.

`tax_rank` *Methods for Taxonomy*

Description

Methods for dealing with objects containing taxonomic information (Taxonomy, TNRS endpoints)

Usage

```
tax_rank(tax, ...)

ott_id(tax, ...)

synonyms(tax, ...)

tax_sources(tax, ...)
```

```
is_suppressed(tax, ...)

unique_name(tax, ...)

tax_name(tax, ...)
```

Arguments

`tax`	an object returned by `taxonomy_taxon_info`, `taxonomy_mrca`, or `tnrs_match_names`
`...`	additional arguments (see `tnrs_match_names`)

Details

This is the page for the generic methods. See the help pages for `taxonomy_taxon_info`, `taxonomy_mrca`, and `tnrs_match_names` for more information.

`tnrs_contexts` *TNRS contexts*

Description

This function returns a list of pre-defined taxonomic contexts (i.e. clades) which can be used to limit the scope of tnrs queries.

Usage

```
tnrs_contexts(...)
```

Arguments

`...`	additional arguments to customize the API request (see `rotl` package documentation).

Details

Taxonomic contexts are available to limit the scope of TNRS searches. These contexts correspond to uncontested higher taxa such as 'Animals' or 'Land plants'. This service returns a list containing all available taxonomic context names, which may be used as input (via the `context_name` argument in other functions) to limit the search scope of other services including `tnrs_match_names`.

Value

Returns invisibly a list for each major clades (e.g., animals, microbes, plants, fungi, life) whose elements contains the possible contexts.

`tnrs_infer_context` *Infer the taxonomic context from a list of names*

Description

Return a taxonomic context given a list of taxonomic names

Usage

```
tnrs_infer_context(names = NULL, ...)
```

Arguments

`names`	Vector of taxon names.
`...`	additional arguments to customize the API request (see `rotl` package documentation).

Details

Find the least inclusive taxonomic context that includes all the unambiguous names in the input set. Unambiguous names are names with exact matches to non-homonym taxa. Ambiguous names (those without exact matches to non-homonym taxa) are indicated in results.

Value

A list including the context name, the context ott id and possibly the names in the query that have an ambiguous taxonomic meaning in the query.

Examples

```
## Not run:
res <- tnrs_infer_context(names=c("Stellula calliope", "Struthio camelus"))

## End(Not run)
```

`tnrs_match_names` *Match names to the Open Tree Taxonomy*

Description

Match taxonomic names to the Open Tree Taxonomy.

Usage

```
tnrs_match_names(
  names = NULL,
  context_name = "All life",
  do_approximate_matching = TRUE,
  ids = NULL,
  include_suppressed = FALSE,
  ...
)
```

Arguments

`names`	taxon names to be queried. Currently limited to 10,000 names for exact matches and 2,500 names for approximate matches (character vector)
`context_name`	name of the taxonomic context to be searched (length-one character vector or `NULL`). Must match (case sensitive) one of the values returned by `tnrs_contexts`. Default to "All life".
`do_approximate_matching`	A logical indicating whether or not to perform approximate string (a.k.a. “fuzzy”) matching. Using `FALSE` will greatly improve speed. Default, however, is `TRUE`.
`ids`	A vector of ids to use for identifying names. These will be assigned to each name in the names array. If ids is provided, then ids and names must be identical in length.
`include_suppressed`	Ordinarily, some quasi-taxa, such as incertae sedis buckets and other non-OTUs, are suppressed from TNRS results. If this parameter is true, these quasi-taxa are allowed as possible TNRS results.
`...`	additional arguments to customize the API request (see `rotl` package documentation).

Details

Accepts one or more taxonomic names and returns information about potential matches for these names to known taxa in the Open Tree Taxonomy.

This service uses taxonomic contexts to disambiguate homonyms and misspelled names; a context may be specified using the `context_name` argument. If no context is specified, then the context will be inferred (i.e., the shallowest taxonomic context that contains all unambiguous names in the input). Taxonomic contexts are uncontested higher taxa that have been selected to allow limits to be applied to the scope of TNRS searches (e.g. ’match names only within flowering plants’). Once a context has been identified (either user-specified or inferred), all taxon name matches will performed only against taxa within that context. For a list of available taxonomic contexts, see `tnrs_contexts`.

A name is considered unambiguous if it is not a synonym and has only one exact match to any taxon name in the entire taxonomy.

Several functions listed in the ‘See also’ section can be used to inspect and manipulate the object generated by this function.

Value

A data frame summarizing the results of the query. The original query output is appended as an attribute to the returned object (and can be obtained using `attr(object, "original_response")`).

See Also

`inspect.match_names`, `update.match_names`, `synonyms.match_names`.

Examples

```
## Not run:
 deuterostomes <- tnrs_match_names(names=c("echinodermata", "xenacoelomorpha",
                                             "chordata", "hemichordata"))

## End(Not run)
```

`tol_about` *Information about the Tree of Life*

Description

Basic information about the Open Tree of Life (the synthetic tree)

Usage

```
tol_about(include_source_list = FALSE, ...)

## S3 method for class 'tol_summary'
tax_rank(tax, ...)

## S3 method for class 'tol_summary'
tax_sources(tax, ...)

## S3 method for class 'tol_summary'
unique_name(tax, ...)

## S3 method for class 'tol_summary'
tax_name(tax, ...)

## S3 method for class 'tol_summary'
ott_id(tax, ...)
```

Arguments

`include_source_list`	Logical (default = `FALSE`). Return an ordered list of source trees.
`...`	additional arguments to customize the API call (see `rotl` for more information).
`tax`	an object created with a call to `tol_about`.

Details

Summary information about the current draft tree of life, including information about the list of trees and the taxonomy used to build it. The object returned by `tol_about` can be passed to the taxonomy methods (`tax_name()`, `tax_rank()`, `tax_sources()`, `ott_id`), to extract relevant taxonomic information for the root of the synthetic tree.

Value

An invisible list of synthetic tree summary statistics:

- date_created String. The creation date of the tree.
- num_source_studies Integer. The number of studies (publications)used as sources.
- num_source_trees The number of trees used as sources (may be >1 tree per study).
- taxonomy_version The Open Tree Taxonomy version used as a source.
- filtered_flags List. Taxa with these taxonomy flags were not used in construction of the tree.
- root List. Describes the root node:
  - node_id String. The canonical identifier of the node.
  - num_tips Numeric. The number of descendent tips.
  - taxon A list of taxonomic properties:
    - ott_id Numeric. The OpenTree Taxonomy ID (ott_id).
    - name String. The taxonomic name of the queried node.
    - unique_name String. The string that uniquely identifies the taxon in OTT.
    - rank String. The taxonomic rank of the taxon in OTT.
    - tax_sources List. A list of identifiers for taxonomic sources, such as other taxonomies, that define taxa judged equivalent to this taxon.
- source_list List. Present only if `include_source_list` is `TRUE`. The sourceid ordering is the precedence order for synthesis, with relationships from earlier trees in the list having priority over those from later trees in the list. See `source_id_map` below for study details.
- source_id_map Named list of lists. Present only if `include_source_list` is `TRUE`. Names correspond to the 'sourceids' used in `source_list` above. Source trees will have the following properties:
  - git_sha String. The git SHA identifying a particular source version.
  - tree_id String. The tree id associated with the study id used.
  - study_id String. The study identifier. Will typically include a prefix ("pg_" or "ot_").
- synth_id The unique string for this version of the tree.

See Also

`source_list` to explore the list of studies used in the synthetic tree (see example).

Examples

```
## Not run:
res <- tol_about()
tax_sources(res)
ott_id(res)
studies <- source_list(tol_about(include_source_list=TRUE))
## End(Not run)
```

`tol_induced_subtree` *Subtree from the Open Tree of Life*

Description

Return the induced subtree on the synthetic tree that relates a list of nodes.

Usage

```
tol_induced_subtree(
  ott_ids = NULL,
  node_ids = NULL,
  label_format = NULL,
  file,
  ...
)
```

Arguments

`ott_ids`	Numeric vector. OTT ids indicating nodes to be used as tips in the induced tree.
`node_ids`	Character vector. Node ids indicating nodes to be used as tips in the induced tree.
`label_format`	Character. Defines the label type; one of “`name`”, “`id`”, or “`name_and_id`” (the default).
`file`	If specified, the function will write the subtree to a file in newick format.
`...`	additional arguments to customize the API call (see `rotl` for more information).

Details

Return a tree with tips corresponding to the nodes identified in the input set that is consistent with the topology of the current synthetic tree. This tree is equivalent to the minimal subtree induced on the draft tree by the set of identified nodes.

Value

If no value is specified to the `file` argument (default), a phylogenetic tree of class `phylo`.

Otherwise, the function returns invisibly a logical indicating whether the file was successfully created.

Note that the tree returned when specifying a file name with the `file` argument is the “raw” Newick string returned by Open Tree of Life. This string contains singleton nodes, and therefore will be different from the tree returned as a `phylo` object which will not contain these singleton nodes.

Examples

```
## Not run:
## Result as a `phylo` object
res <- tol_induced_subtree(ott_ids = c(292466, 267845, 316878, 102710))

## Raw Newick string from Open Tree of Life
tree_file <- tempfile(fileext = ".tre")
tol_induced_subtree(ott_ids = c(292466, 267845, 316878, 102710),
                    file=tree_file)


## End(Not run)
```

`tol_lineage` *Node info*

Description

Get summary information about a node in the synthetic tree

Usage

```
tol_lineage(tax, ...)

tol_node_info(ott_id = NULL, node_id = NULL, include_lineage = FALSE, ...)

## S3 method for class 'tol_node'
tax_rank(tax, ...)

## S3 method for class 'tol_node'
tax_sources(tax, ...)

## S3 method for class 'tol_node'
unique_name(tax, ...)

## S3 method for class 'tol_node'
tax_name(tax, ...)
```

```
## S3 method for class 'tol_node'
ott_id(tax, ...)

## S3 method for class 'tol_node'
source_list(tax, ...)

## S3 method for class 'tol_node'
tax_lineage(tax, ...)

## S3 method for class 'tol_node'
tol_lineage(tax, ...)
```

Arguments

`tax`	an object returned by `tol_node_info`.
`...`	additional arguments to customize the API call (see ?rotl for more information)
`ott_id`	Numeric. The OpenTree taxonomic identifier.
`node_id`	Character. The OpenTree node identifier.
`include_lineage`	Logical (default = FALSE). Whether to return the lineage of the node from the synthetic tree.

Details

Returns summary information about a node in the graph. The node of interest may be specified using either a node id or an taxon id, but not both. If the specified node or OTT id is not in the graph, an error will be returned.

If the argument `include_lineage=TRUE` is used, you can use `tax_lineage()` or `tol_lineage` to return the taxonomic information or the node information for all the ancestors to this node, down to the root of the tree.

Value

`tol_node_info` returns an invisible list of summary information about the queried node:

- node_id String. The canonical identifier of the node.
- num_tips Numeric. The number of descendent tips.
- partial_path_of List. The edge below this synthetic tree node is compatible with the edge below each of these input tree nodes (one per tree). Each returned element is reported as sourceid:nodeid.
- query The node id that resolved to this node. This can differ from the node_id field if the query id is not canonical.
- taxon A list of taxonomic properties. Only returned if the queried node is a taxon. Each source has:
  - ott_id Numeric. The OpenTree Taxonomy ID (ottID).
  - name String. The taxonomic name of the queried node.

  - unique_name String. The string that uniquely identifies the taxon in OTT.
  - rank String. The taxonomic rank of the taxon in OTT.
  - tax_sources List. A list of identifiers for taxonomic sources, such as other taxonomies, that define taxa judged equivalent to this taxon.

  The following properties list support/conflict for the node across synthesis source trees. All properties involve sourceid keys and nodeid values (see `source_id_map` below).

- supported_by List. Input tree nodes (one per tree) that support this synthetic tree node. Each returned element is reported as sourceid:nodeid.
- terminal List. Input tree nodes (one per tree) that are equivalent to this synthetic tree node (via an exact mapping, or the input tree terminal may be the only terminal descended from this synthetic tree node. Each returned element is reported as sourceid:nodeid.
- conflicts_with Named list of lists. Names correspond to sourceid keys. Each list contains input tree node ids (one or more per tree) that conflict with this synthetic node.

`tol_lineage` and `tax_lineage` return data frames. `tol_lineage` indicate for each ancestor its node identifier, the number of tips descending from that node, and whether it corresponds to a taxonomic level.

Examples

```
## Not run:
birds <- tol_node_info(ott_id=81461, include_lineage=TRUE)
source_list(birds)
tax_rank(birds)
ott_id(birds)
tax_lineage(birds)
tol_lineage(birds)
## End(Not run)
```

`tol_mrca` *MRCA of taxa from the synthetic tree*

Description

Most Recent Common Ancestor for a set of nodes

Usage

```
tol_mrca(ott_ids = NULL, node_ids = NULL, ...)

## S3 method for class 'tol_mrca'
tax_sources(tax, ...)

## S3 method for class 'tol_mrca'
unique_name(tax, ...)
```

```
## S3 method for class 'tol_mrca'
tax_name(tax, ...)

## S3 method for class 'tol_mrca'
tax_rank(tax, ...)

## S3 method for class 'tol_mrca'
ott_id(tax, ...)

## S3 method for class 'tol_mrca'
source_list(tax, ...)
```

Arguments

`ott_ids`	Numeric vector. The ott ids for which the MRCA is desired.
`node_ids`	Character vector. The node ids for which the MRCA is desired.
`...`	additional arguments to customize the API call (see `rotl` for more information).
`tax`	an object returned by `tol_mrca()`.

Details

Get the MRCA of a set of nodes on the current synthetic tree. Accepts any combination of node ids and ott ids as input. Returns information about the most recent common ancestor (MRCA) node as well as the most recent taxonomic ancestor (MRTA) node (the closest taxonomic node to the MRCA node in the synthetic tree; the MRCA and MRTA may be the same node). If they are the same, the taxonomic information will be in the `mrca` slot, otherwise they will be in the `nearest_taxon` slot of the list. If any of the specified nodes is not in the synthetic tree an error will be returned.

Taxonomic methods (`tax_sources()`, `ott_id()`, `unique_name()`, ...) are available on the objects returned by `tol_mrca()`. If the MRCA node is MRTA, the name of the object returned by these methods will start with ‘ott’, otherwise it will start with ‘mrca’.

Value

An invisible list of the MRCA node properties:

- mrca List of node properties.
  - node_id String. The canonical identifier of the node.
  - num_tips Numeric. The number of descendent tips.
  - taxon A list of taxonomic properties. Only returned if the queried node is a taxon. (If the node is not a taxon, a `nearest_taxon` list is returned (see below)).
    - ott_id Numeric. The OpenTree Taxonomy ID (ottID).
    - name String. The taxonomic name of the queried node.
    - unique_name String. The string that uniquely identifies the taxon in OTT.
    - rank String. The taxonomic rank of the taxon in OTT.
    - tax_sources List. A list of identifiers for taxonomic sources, such as other taxonomies, that define taxa judged equivalent to this taxon.

  The following properties list support/conflict for the node across synthesis source trees. All properties involve sourceid keys and nodeid values (see `source_id_map` below) Not all properties are are present for every node.

  - partial_path_of List. The edge below this synthetic tree node is compatible with the edge below each of these input tree nodes (one per tree). Each returned element is reported as sourceid:nodeid.
  - supported_by List. Input tree nodes (one per tree) that support this synthetic tree node. Each returned element is reported as sourceid:nodeid.
  - terminal List. Input tree nodes (one per tree) that are equivalent to this synthetic tree node (via an exact mapping, or the input tree terminal may be the only terminal descended from this synthetic tree node. Each returned element is reported as sourceid:nodeid.
  - conflicts_with Named list of lists. Names correspond to sourceid keys. Each list contains input tree node ids (one or more per tree) that conflict with this synthetic node.

- nearest_taxon A list of taxonomic properties of the nearest rootward taxon node to the MRCA node. Only returned if the MRCA node is a not taxon (otherwise the `taxon` list above is returned).

  - ott_id Numeric. The OpenTree Taxonomy ID (ottID).
  - name String. The taxonomic name of the queried node.
  - unique_name String. The string that uniquely identifies the taxon in OTT.
  - rank String. The taxonomic rank of the taxon in OTT.
  - tax_sources List. A list of identifiers for taxonomic sources, such as other taxonomies, that define taxa judged equivalent to this taxon.

- source_id_map Named list of lists. Names correspond to the sourceid keys used in the support/conflict properties of the `mrca` list above. Source trees will have the following properties:

  - git_sha The git SHA identifying a particular source version.
  - tree_id The tree id associated with the study id used.
  - study_id The study identifier. Will typically include a prefix ("pg_" or "ot_").

  The only sourceid that does not correspond to a source tree is the taxonomy, which will have the name "ott"+'taxonomy_version', and the value is the ott_id of the taxon in that taxonomy version. "Taxonomy" will only ever appear in `supported_by`.

Examples

```
## Not run:
birds_mrca <- tol_mrca(ott_ids=c(412129, 119214))
ott_id(birds_mrca)
tax_sources(birds_mrca)

## End(Not run)
```

`tol_subtree` *Extract a subtree from the synthetic tree*

Description

Extract a subtree from the synthetic tree from an Open Tree node id.

Usage

```
tol_subtree(ott_id = NULL, node_id = NULL, label_format = NULL, file, ...)
```

Arguments

`ott_id`	Numeric. The ott id of the node in the tree that should serve as the root of the tree returned.
`node_id`	Character. The node id of the node in the tree that should serve as the root of the tree returned.
`label_format`	Character. Defines the label type; one of “`name`”, “`id`”, or “`name_and_id`” (the default).
`file`	If specified, the function will write the subtree to a file in newick format.
`...`	additional arguments to customize the API call (see `rotl` for more information).

Details

Given a node, return the subtree of the synthetic tree descended from that node. The start node may be specified using either a node id or an ott id, but not both. If the specified node is not in the synthetic tree an error will be returned. There is a size limit of 25000 tips for this method.

Value

If no value is specified to the `file` argument (default), a phylogenetic tree of class `phylo`. Otherwise, the function returns invisibly a logical indicating whether the file was successfully created.

Examples

```
## Not run:
res <- tol_subtree(ott_id=241841)
## End(Not run)
```

Index