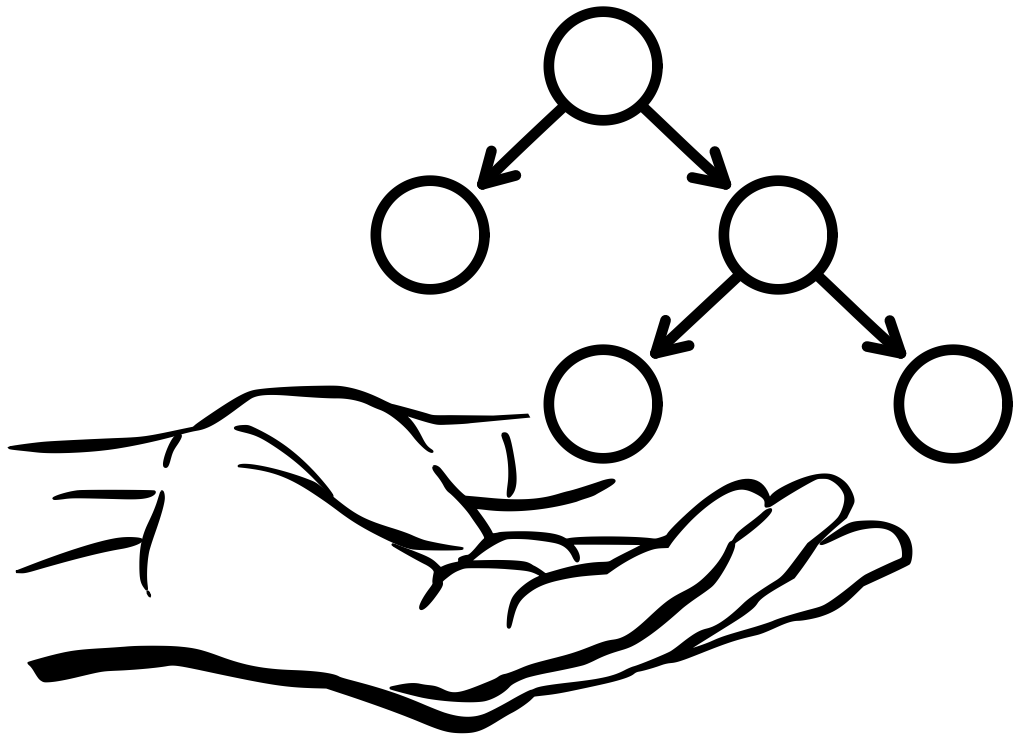
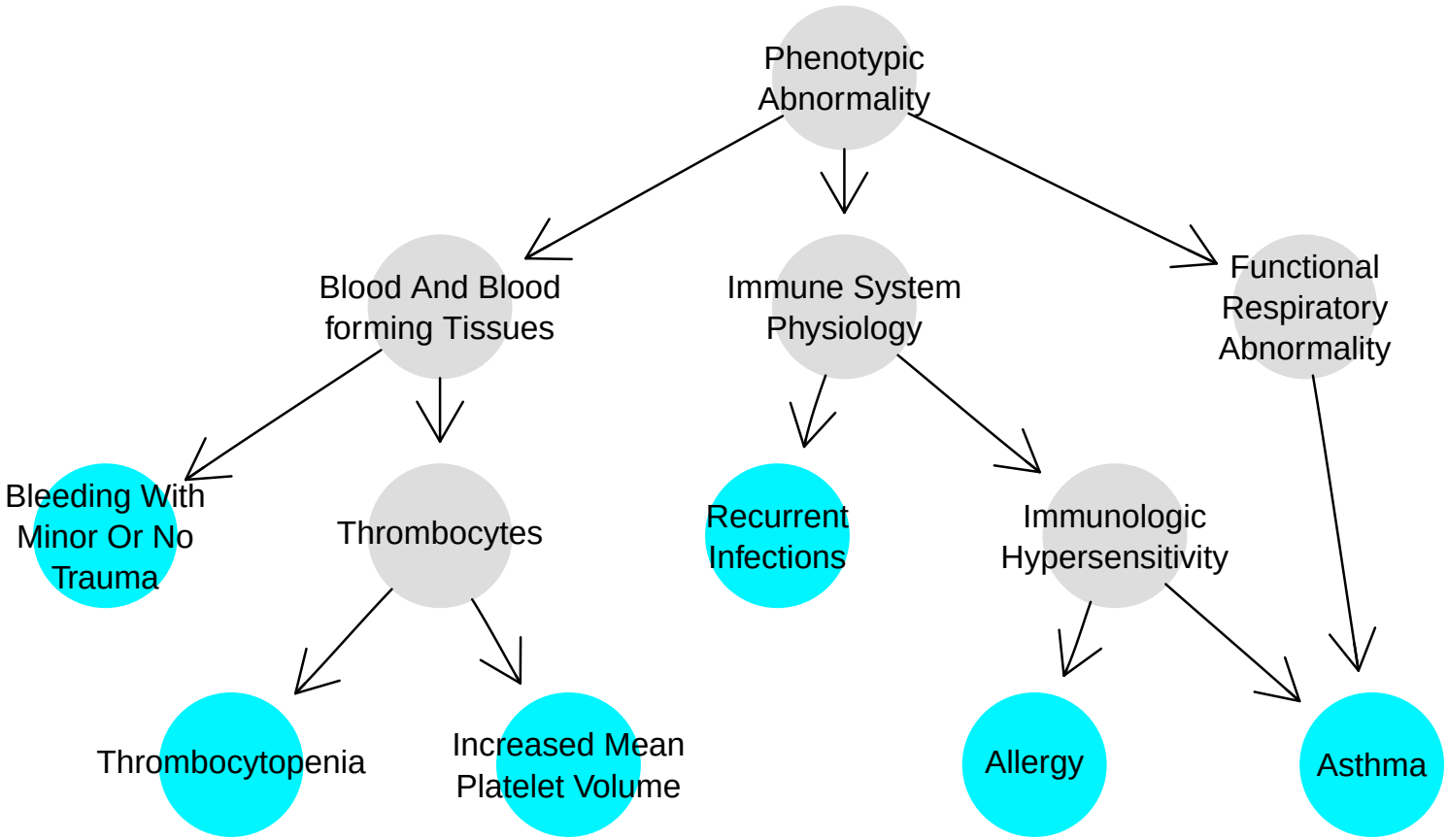
ontologyX



Bringing Scientific minds and
Biomedical Ontologies together

Ontologies

- Sets of "terms" and the relationships between them... typically including "is a" relations.
- Terms often used to "annotate" objects.



Data

Availability

- Over 100 ontologies available from
  `http://www.obofoundry.org/`
  inc. GO, HPO, MPO, PRO...
- Publicly available annotation sets include:
  - Human genes annotated with GO terms
    `http://geneontology.org`
  - Diseases and genes annotated with HPO terms
    `http://human-phenotype-ontology.github.io`

Analysis

- 'Semantic similarity' between ontologically annotated entities.
- Select sets of genes/diseases/samples based on annotation,
  e.g. for case groups, GSEA, ...

Software

- There is software enabling relational reasoning and queries for arbitrary ontologies.
- R has ontoCAT but a bit slow for large datasets.
- No software (in R) for semantic similarity for calculations with arbitrary ontologies.
- No simple way of visualising relations between terms or visualising annotation sets for arbitrary ontologies.

ontologyX

A suite of R packages: ontologyIndex, ontologyPlot and ontologySimilarity, which simplifies and improves processing, analysis and visualisation of ontological data through:

- enabling arbitrary ontologies to be read into R,
- representation of ontological objects by native R types,
- providing a simple set of quick functions for querying ontologies,
- simple plotting functions for ontological objects familiar to R's `plot`,
- semantic similarity calculations which are generally applicable and fast,
- enabling evaluation of statistical significance in semantic similarity,
- providing a base for extension to complex ontological functionality.

ontologyIndex

```
library(ontologyIndex)
data(hpo) # loads the 'hpo' object: an 'ontology_index' for the HPO
names(hpo)
```

```
## [1] "id"        "name"      "parents"   "children"  "ancestors" "obsolete"
```

```
tcp <- hpo$id[ hpo$name=="Thrombocytopenia" ]
tcp
```

```
##   HP:0001873
## "HP:0001873"
```

```
hpo$name[ hpo$ancestors[[tcp]] ]
```

```
##                                       HP:0000001
##                                             "All"
##                                       HP:0000118
##                           "Phenotypic abnormality"
##                                       HP:0001871
## "Abnormality of blood and blood-forming tissues"
##                                       HP:0001872
##                     "Abnormality of thrombocytes"
##                                       HP:0011873
##                         "Abnormal platelet count"
##                                       HP:0001873
##                                "Thrombocytopenia"
```

Examples

```
data(go)
library(ontologySimilarity)
data(gene_GO_terms)
gene_GO_terms[c("TUBB1", "FLI1", "NBEAL2")]
```

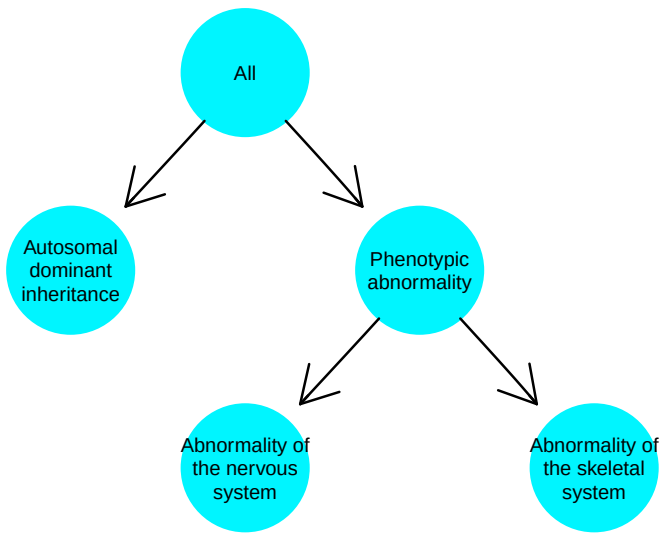
```
## $TUBB1
## [1] "GO:0003924" "GO:0005200" "GO:0005525" "GO:0005737" "GO:0005874"
## [6] "GO:0051225" "GO:0070062"
##
## $FLI1
##  [1] "GO:0000978" "GO:0000980" "GO:0001077" "GO:0003682" "GO:0005634"
##  [6] "GO:0006366" "GO:0007599" "GO:0008015" "GO:0009887" "GO:0030154"
## [11] "GO:0035855" "GO:0045944"
##
## $NBEAL2
## [1] "GO:0005543" "GO:0005783" "GO:0019898" "GO:0030220"
```

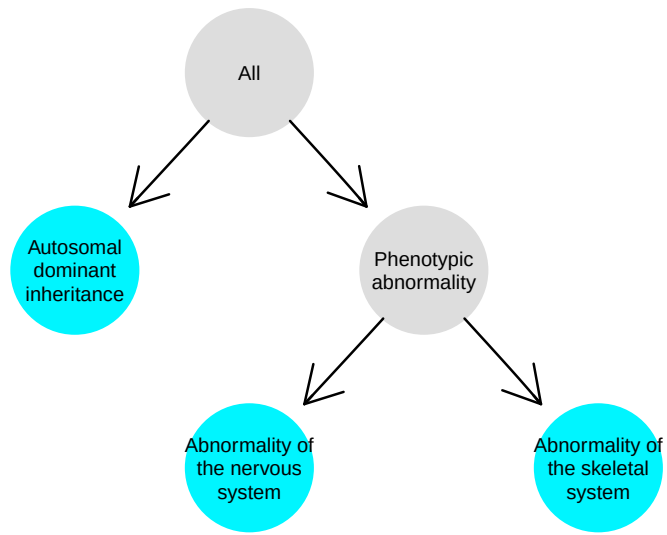
```
go$name[gene_GO_terms$NBEAL2]
```

```
##                        GO:0005543                        GO:0005783
##            "phospholipid binding"          "endoplasmic reticulum"
##                        GO:0019898                        GO:0030220
## "extrinsic component of membrane"             "platelet formation"
```

Examples

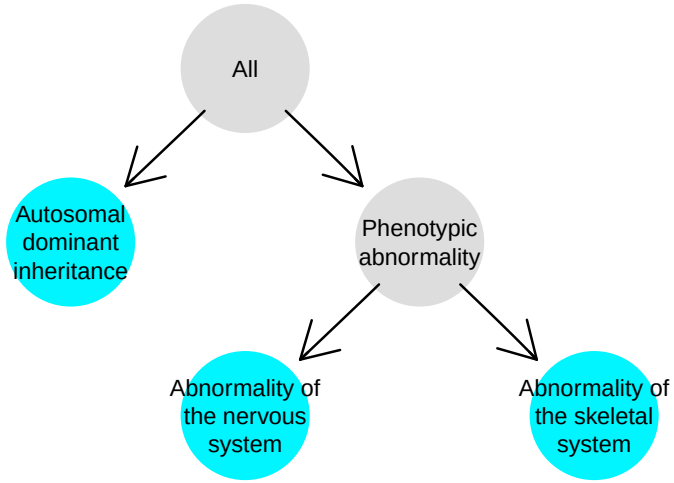
ontologyIndex has functions for operating with respect to relations between terms, e.g. `minimal_set`



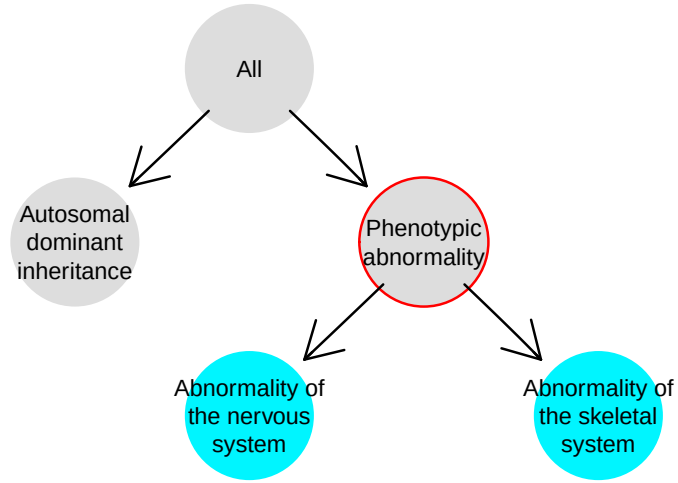


Examples

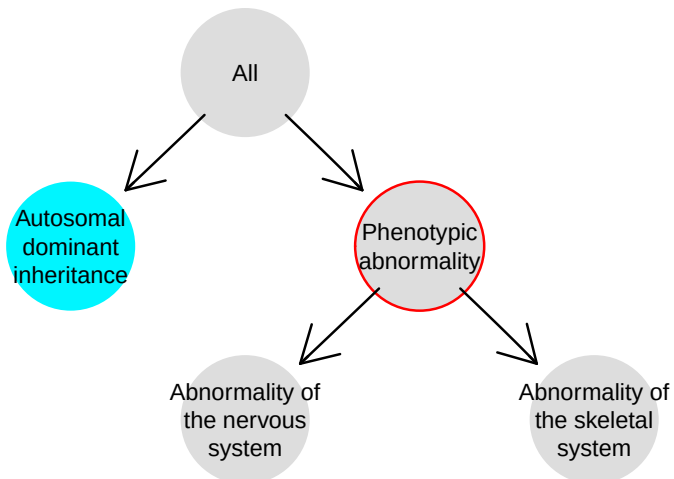
Original set



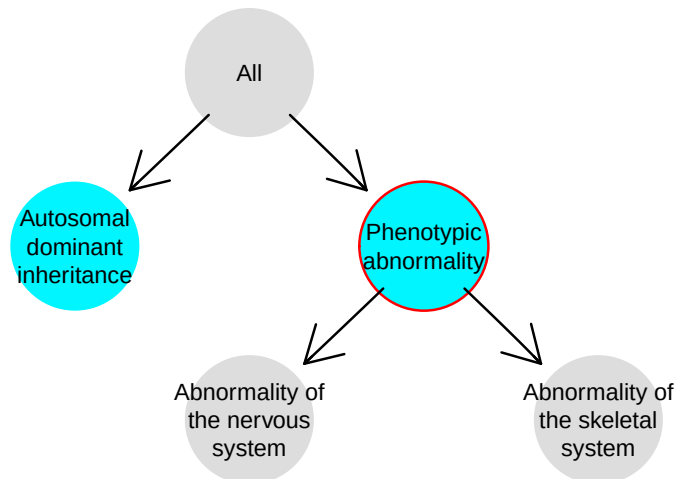
intersection_with_descendants of 'Phenotypic abnormality'



exclude_descendants of 'Phenotypic abnormality'



prune_descendants of 'Phenotypic abnormality'



Examples

```
HPO_table <- read.table("phenotypes.txt", stringsAsFactors=FALSE)
head(HPO_table)
```

personID	HPO
sample1	HP:0011889,HP:0001873,HP:0011877,HP:0002719,HP:0002099,HP:0012393
sample2	HP:0009815,HP:0010329,HP:0012718
sample3	HP:0100370,HP:0030680,HP:0008518,HP:0005879,HP:0001872,HP:0000929,HP:0006699
sample4	HP:0000235,HP:0000153,HP:0001939,HP:0011894,HP:0011883,HP:0040064
sample5	HP:0100921,HP:0010166,HP:0001909,HP:0004374,HP:0000234
sample6	HP:0012529,HP:0000235,HP:0010185,HP:0011878,HP:0000163,HP:0002250

```
phenotypes <- strsplit(HPO_table$HPO, split=",")
names(phenotypes) <- HPO_table$personID
phenotypes$sample1
```

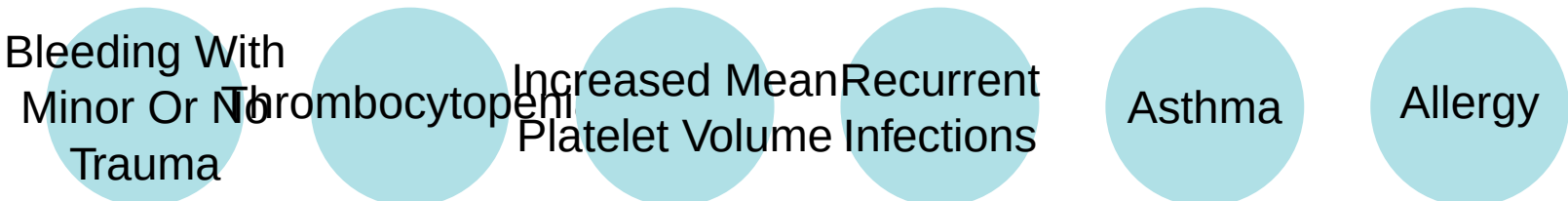
```
## [1] "HP:0011889" "HP:0001873" "HP:0011877" "HP:0002719" "HP:0002099"
## [6] "HP:0012393"
```

```
hpo$name[phenotypes$sample1]
```

```
##                          HP:0011889                          HP:0001873
## "Bleeding with minor or no trauma"              "Thrombocytopenia"
##                          HP:0011877                          HP:0002719
##   "Increased mean platelet volume"          "Recurrent infections"
##                          HP:0002099                          HP:0012393
##                            "Asthma"                            "Allergy"
```

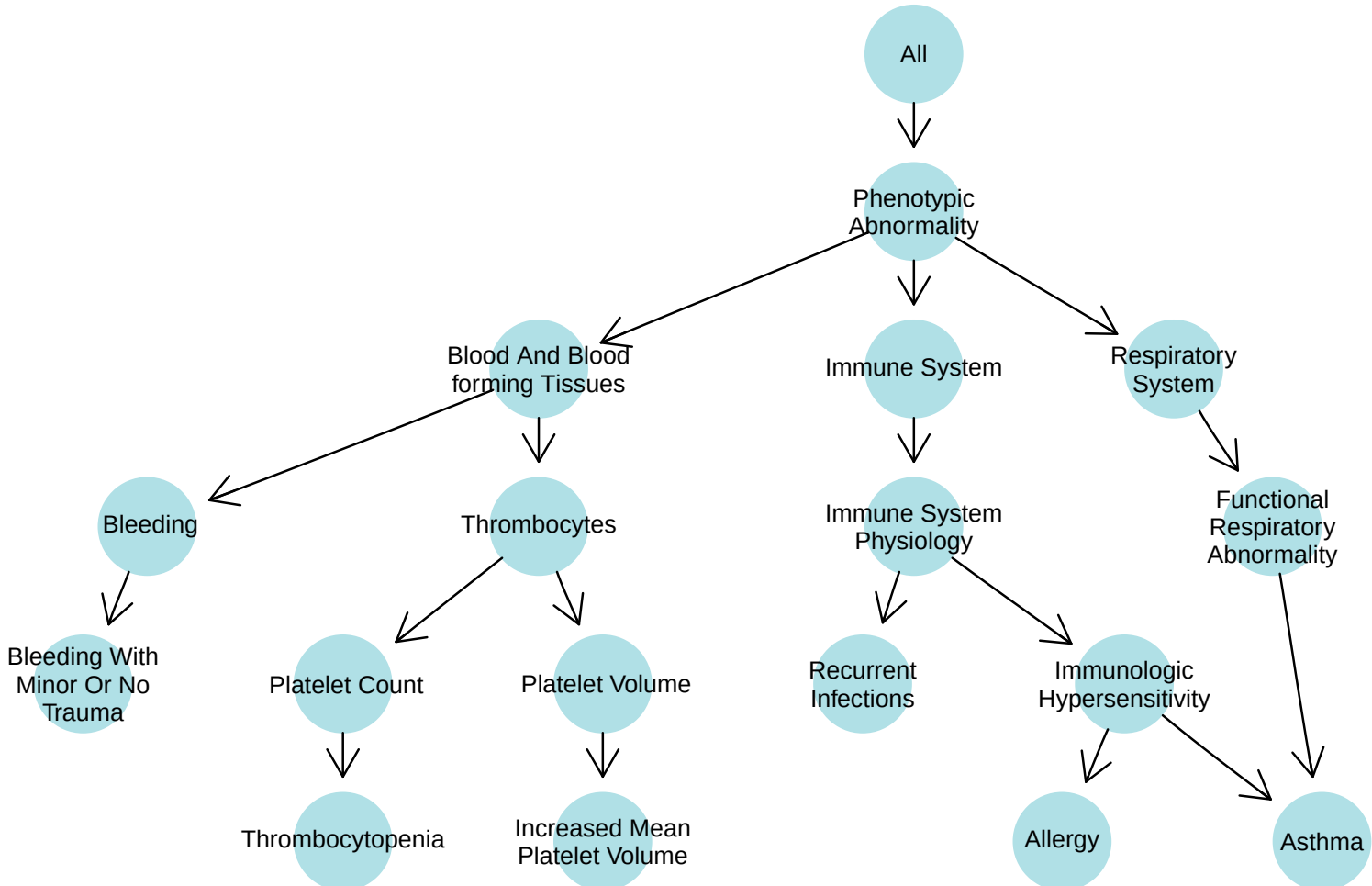
ontologyPlot

```
library(ontologyPlot)
onto_plot(ontology=hpo, terms=phenotypes$sample1)
```



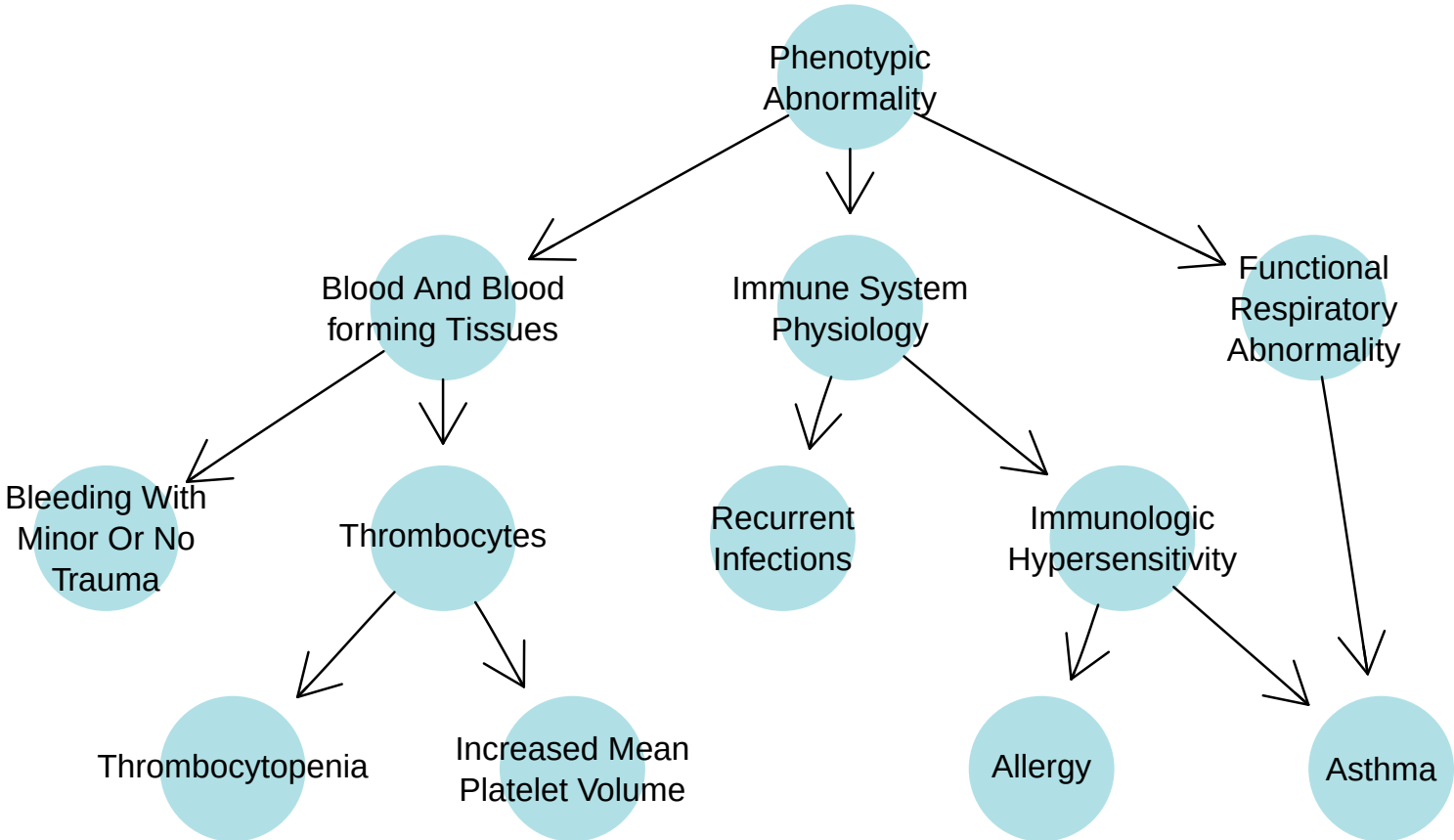
Examples

```
with_ancestors <- get_ancestors(hpo, phenotypes$sample1)
onto_plot(ontology=hpo, terms=with_ancestors)
```



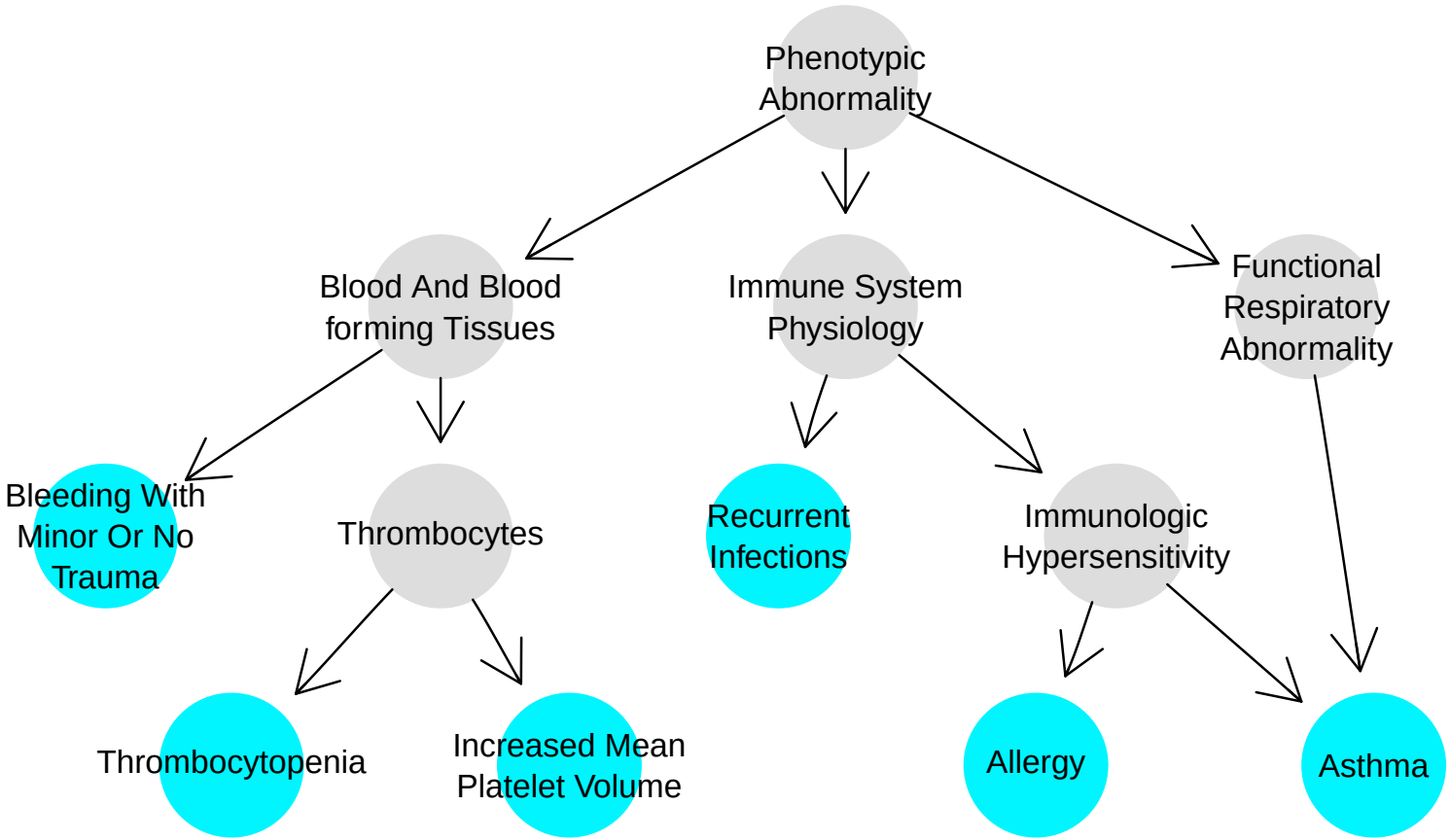
Examples

```
no_links <- remove_links(hpo, with_ancestors)
onto_plot(ontology=hpo, terms=no_links)
```



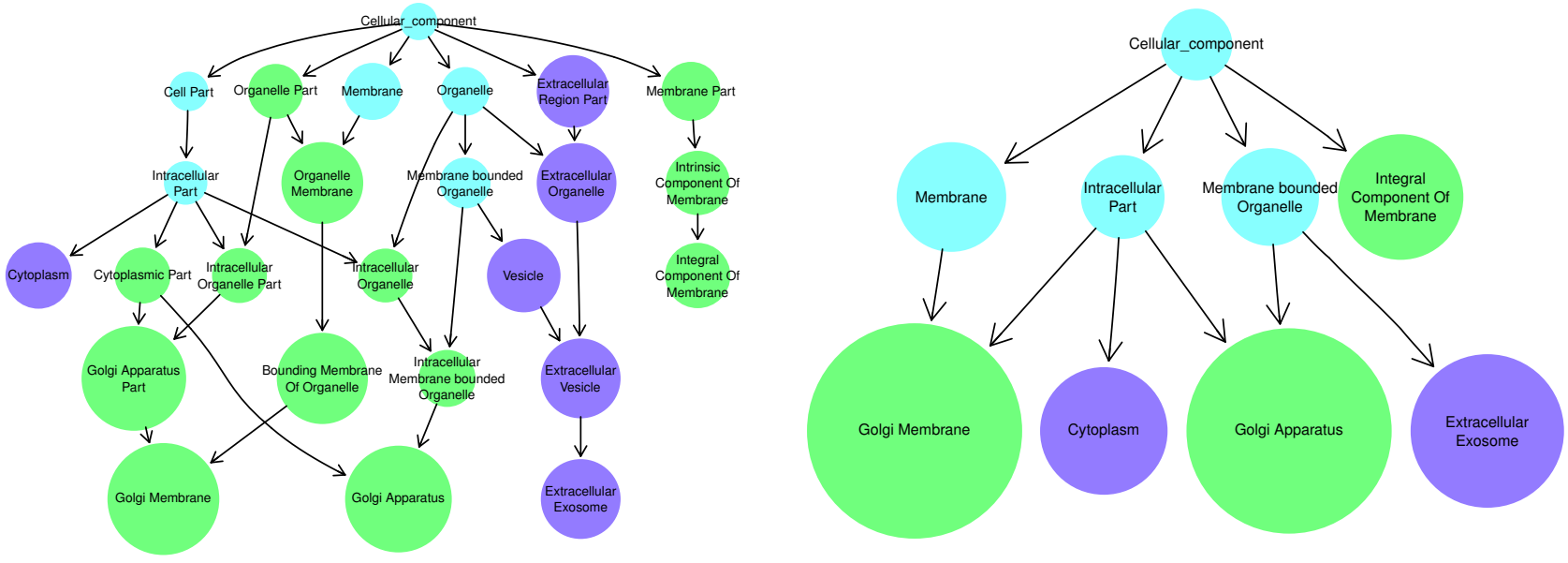
Examples

```
colours <- ifelse(no_links %in% phenotypes$sample1, "turquoise1", "#DDDDDD")
onto_plot(ontology=hpo, terms=no_links, fillcolor=colours)
```




```
# other graphical params available with graphviz names by default, e.g. width, border, ...
# can also pass functions to set values
```

Examples



Terms annotated to genes *QPCTL* and *CRNN* descending from the "`cellular_component`" term in the GO. Left: all ancestors, right: those remaining after `remove_uninformative_terms` has been called. Terms in both genes blue, only *QPCTL* green and *CRNN* purple.

ontologySimilarity

- Enables semantic similarity calculation between terms or between entities annotated with ontological terms.
- Similarity between terms defined with respect to 'population frequency' and shared ancestry.
- Similarity between two annotated entities is calculated by averaging over between-term similarites.
- Annotations represented by `lists` of `character` vectors of term IDs.

Table : Execution times for computing pairwise similarities matrices for 1000 randomly selected GO terms and 100 randomly selected gene GO annotation sets.

	Term sim (s)	Gene sim (s)
GOSim	1075.43	298.34
GOSemSim	1.71	116.72
ontologySimilarity	0.31	0.06

Examples

```
sim_matrix <- get_sim_grid(ontology=hpo, term_sets=phenotypes)

neoplasm    <- hpo$id[hpo$name=="Neoplasm"]
bleeding    <- hpo$id[hpo$name=="Abnormal bleeding"]
abnormality <- hpo$id[hpo$name=="Phenotypic abnormality"]

congenital_phenotype <- function(terms) {
    abnormalities <- intersection_with_descendants(hpo, roots=abnormality, terms=terms)
    no_cancer <- exclude_descendants(hpo, roots=neoplasm, terms=abnormalities)
    no_specific_bleeding <- prune_descendants(hpo, roots=bleeding, terms=no_cancer)
    return(no_specific_bleeding)
}

congenital <- lapply(phenotypes, congenital_phenotype)
sim_matrix <- get_sim_grid(ontology=hpo, term_sets=congenital)

bleeds <- function(terms) {
    has_is_a_bleeding_term <- any(get_ancestors(hpo, terms) == bleeding)
    return(has_is_a_bleeding_term)
}

bleeders <- which(sapply(phenotypes, bleeds))

get_sim_p(sim_matrix, bleeders)


## [1] 0.05484038
```

References

Greene D, Richardson S and Turro E (2017). "ontologyX: a suite of R packages for working with ontological data." *Bioinformatics*, pp. btw763.

`https://cran.r-project.org/web/packages/`

- `ontologyIndex`
- `ontologyPlot`
- `ontologySimilarity`