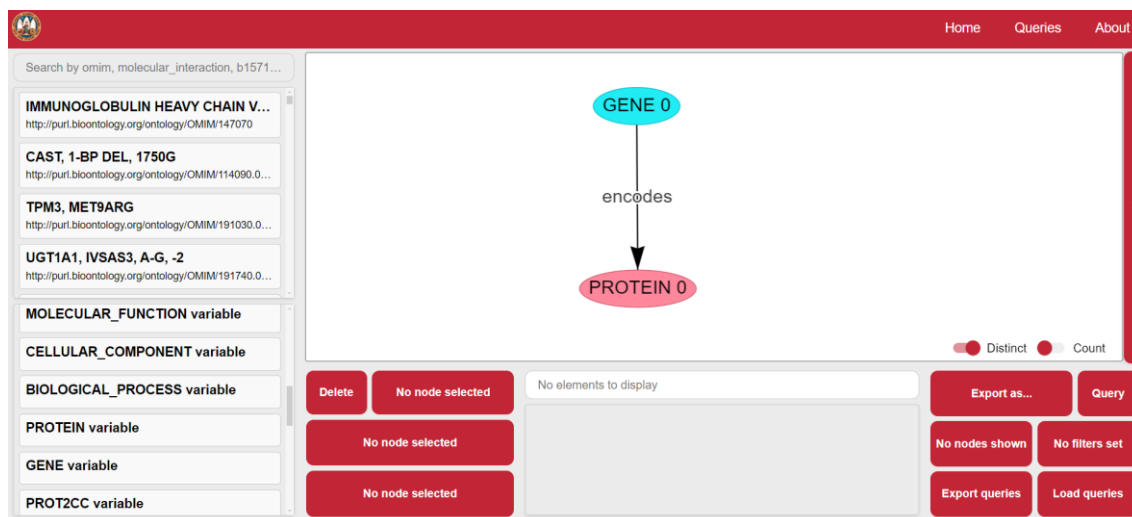


1. Introduction

INTUITION is a web application for user friendly SPARQL query building. In this way, users can exploit RDF knowledge graphs without advanced knowledge in SPARQL query language.

INTUITION analyses the knowledge network of an accessible endpoint, in this case BioGateway (<http://ssb4.nt.ntnu.no:23122/sparql>), and allows building queries graphically by representing nodes (entities) and edges (properties).

RDF knowledge graphs represent entities through uniform resource identifiers (URIs), and information through triples or statements that represent a directional relationship between two entities, similar to a sentence: <subject> <predicate> <object>. For example: <Gene> <encodes> <Protein>. The SPARQL query language also uses this pattern to build queries. These queries can be complex by linking multiple triples and including operations. A tutorial for building SPARQL queries in BioGateway (BGW) is available in this repository (<https://github.com/juan-mulero/cisreg>). INTUITION uses this same design for query development.



2. Design

In INTUITION we distinguish different sections:

- A- Query building screen.
- B- Entity browser (Currently deactivated).
- C- Graph builder (Used for queries that use union clauses).
- D- Variable selection (Main types of entities in the network).
- E- Properties selection:
 - Object properties: relations between entities. Their inclusion acts as a filter.
 - Data properties: attributes of the entities. Their inclusion acts as a filter.
 - Optional properties: relations between entities. Their inclusion does not act as a filter (optional).
- F- Output screen.
- G- Query builder:

- Query: runs the query.
- Export as: exports the query results.
- Nodes shown: output editor. Allow to indicate which variables are shown in the output screen, and to create bindings, i.e., to define variables.
- Filters set: allow adding filters.
- Export query: export the designed query.
- Load query: to load a previously designed query.

The screenshot shows a web application for querying biological data. The interface is divided into several panels:

- Panel A:** Central workspace showing a graph with a blue node labeled "GENE 0" and a red node labeled "PROTEIN 0", connected by a black arrow labeled "encodes".
- Panel B:** Search bar with the text "Search by omim, molecular_interaction, b1571..." and a list of search results including "IMMUNOGLOBULIN HEAVY CHAIN V...", "CAST, 1-BP DEL, 1750G", "TPM3, MET9ARG", and "UGT1A1, IVSAS3, A-G, -2".
- Panel C:** Graph management controls including a "Define new graph" button, a "Default" button, and a red trash icon.
- Panel D:** Variable selection list with options: "MOLECULAR_FUNCTION variable", "CELLULAR_COMPONENT variable", "BIOLOGICAL_PROCESS variable", "PROTEIN variable", "GENE variable", and "PROT2CC variable".
- Panel E:** Action buttons: "Delete", "Set 'gene' properties...", "Set 'gene' optional properties...", and "Set 'gene' data properties...".
- Panel F:** Table of search results with columns: "Gene 0 Label", "Gene 0 URI", "Protein 0 Label", and "Protein 0 URI". The table contains several rows of data, including "TBC1D19", "D6RA60_HUMAN", "D6RA74_HUMAN", "D6RD52_HUMAN", "D6RD21_HUMAN", "TBC19_HUMAN", "TBD2A_HUMAN", and "TRC1D20".
- Panel G:** Export and query controls including "Export as...", "Query", "2 nodes shown", "No filters set", "Export queries", and "Load queries".

3. Variables and properties

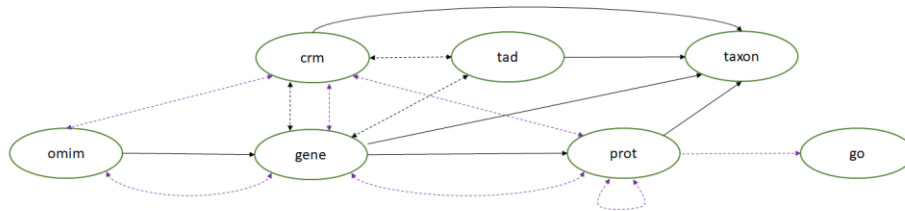
Variables:

- CRM variable: cis regulatory module.
- Gene variable: genes.
- Protein variable: proteins.
- OMIM variable: entities from OMIM ontology (mainly phenotypes).
- Molecular_interaction: entities from Molecular Interactions ontology (MI).
- crm2gene variable: relation between CRM and gene.
- gene2phen variable: relation between gene and phenotype.
- crm2phen variable: relation between CRM and phenotype.
- crm2tfac variable: relation between CRM and protein (transcription factor).
- Transcription factor variable: transcription factors (currently only proteins that interact with CRM).
- reg2targ variable: regulatory relation between proteins.
- prot2prot: molecular interaction relation between proteins.
- tfac2gene variable: relation between gene and protein.
- Database variable: databases.
- Chromosome variable: chromosomes.
- Reference_genome variable: genome assembly.
- TAD variable: topologically associated domain.
- Cellular_component variable: cellular components from Gene Ontology (GO).
- prot2cc variable: relation between protein and its cellular components.
- Molecular_function variable: molecular functions from GO.
- prot2mf: relation between protein and its molecular functions.

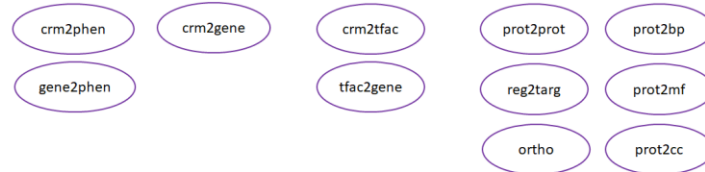
- Biological_process variable: biological processes from GO.
- prot2bp variable: relation between protein and its biological processes.
- Ortho variable: orthology relation between proteins.
- Root variable: top hierarchical class of NCBITaxon Ontology.
- Taxonomic_rank variable: top hierarchical class of NCBITaxon Ontology.

The properties are detailed with examples and their domains [here](#).

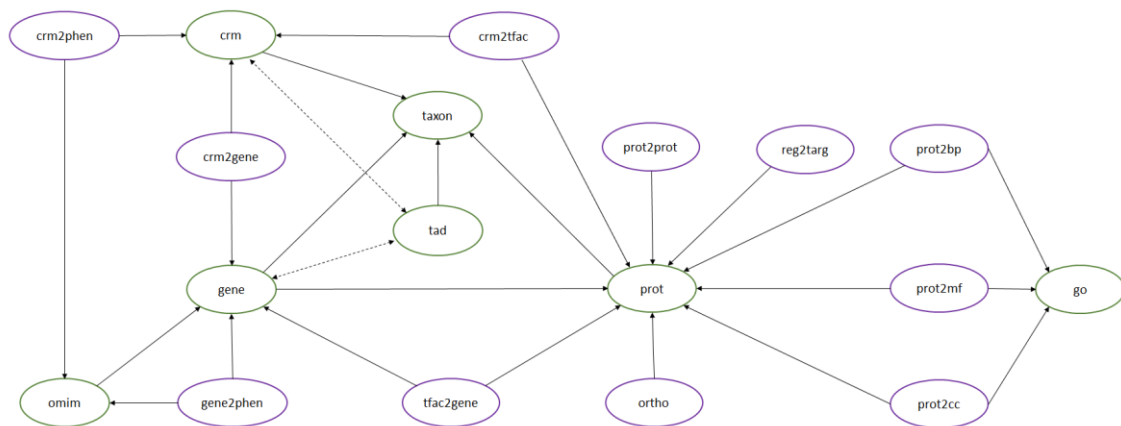
A-type grahs
(entities)



B-type graphs
(relations)



BGW
network



ctm: cis regulatory domains (enhancers)
tad: topologically associating domains
gene: protein-coding genes
prot: proteins
omim: Online Mendelian Inheritance in Man
go: Gene Ontology
taxon: NCBITaxon Ontology

—→ links through properties
 - - - - - links through coordinate operations
 links through relation graphs

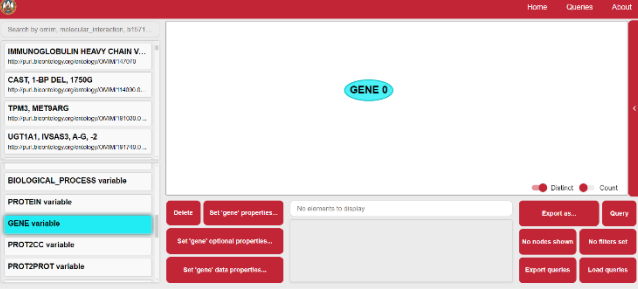
ctm2gene: ctm-target gene relations
ctm2tfac: ctm-dbTF interactions
ctm2phen: ctm-phenotype relations
gene2phen: gene-phenotype relations
prot2prot: protein interactions
prot2bp: protein-biological process relations
prot2mf: protein-molecular function relations
prot2cc: protein-cellular component relations
tfac2gene: transcription factor-target gene relations
reg2targ: protein-protein regulatory relations
ortho: protein-protein orthology relations

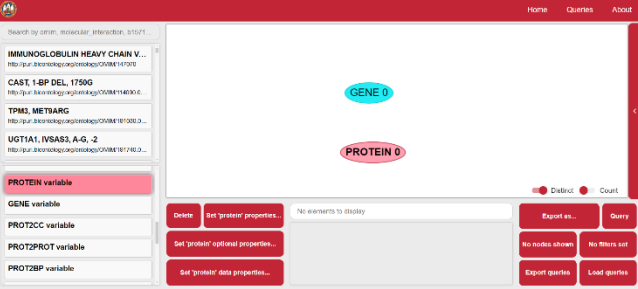
4. How to build a query

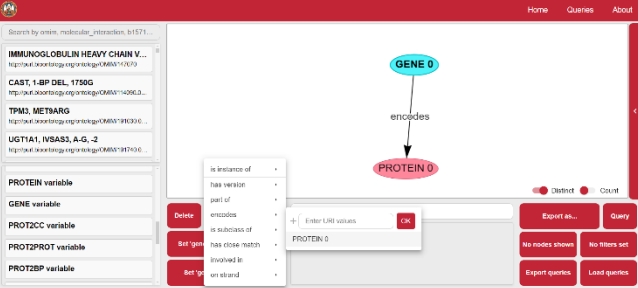
We take as an example the previous case: <Gene> <encodes> <Protein>.

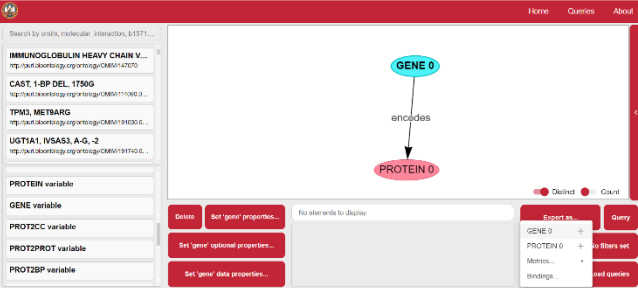
1. Select the first node or subject, in this case Gene, in the variables section.
2. Select the second node or object, in this case Protein, in the variables section.
3. Select the object property that relates both entities, in this case "encodes".

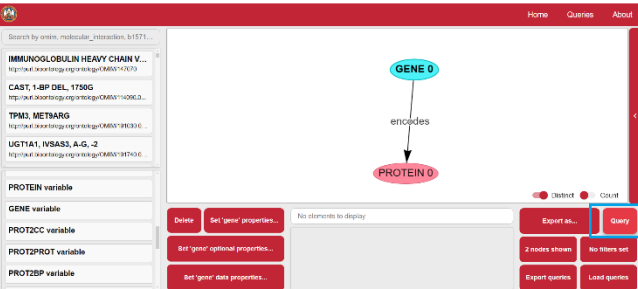
4. Select in "Nodes shown" the data we want to show in the output, in this case Gene and Protein.
5. Click on "Query" to launch the query.
6. Click on "Export as" to download the data obtained. Click on "Export query" if you want to save the query as well.

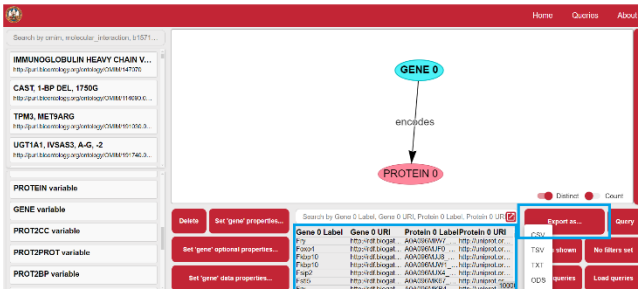
1. 

2. 

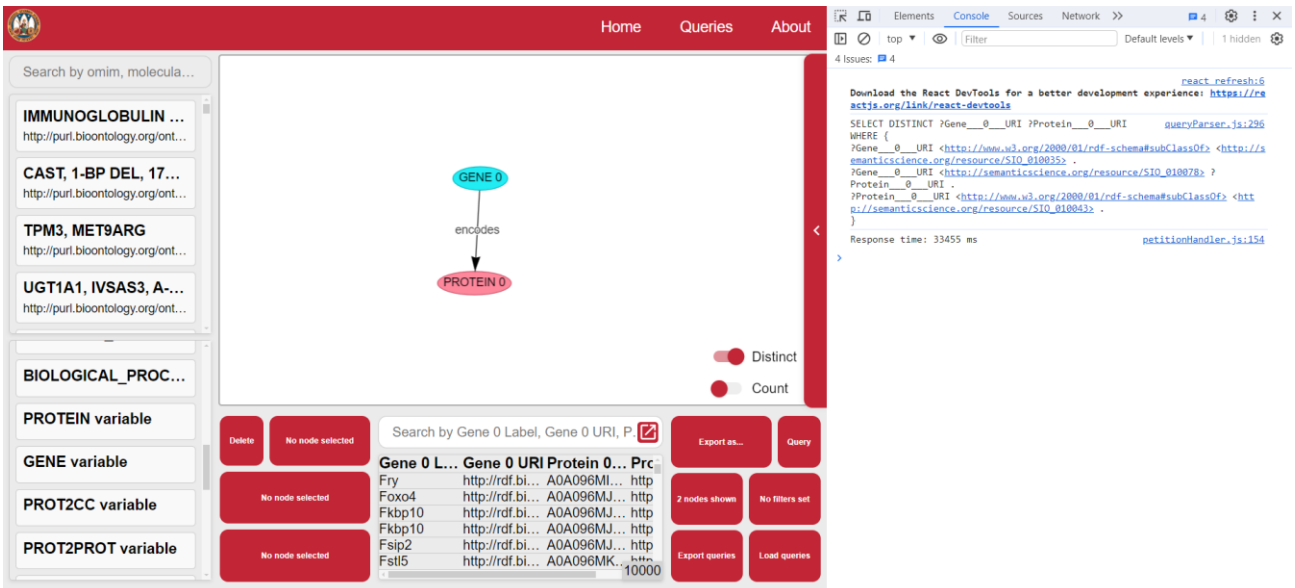
3. 

4. 

5. 

6. 

The generated SPARQL query can also be found by accessing the Console.



```

SELECT DISTINCT ?Gene_0_URI ?Protein_0_URI
WHERE {
  ?Gene_0_URI <http://www.w3.org/2000/01/rdf-schema#subClassOf> <http://semanticscience.org/resource/SIO_010035> .
  ?Gene_0_URI <http://semanticscience.org/resource/SIO_010078> ?Protein_0_URI .
  ?Protein_0_URI <http://www.w3.org/2000/01/rdf-schema#subClassOf> <http://semanticscience.org/resource/SIO_010083> .
}

```

Response time: 33455 ms

5. Filters and other clauses

5.1. DISTINCT and COUNT

By default, the queries include the DISTINCT clause to avoid duplicate results. This can be deactivated clicking on the Distinct button, in the Query building screen. You can also activate the Count button if you want to include this clause in the query.

The screenshot shows the query building interface. On the left, there is a search bar and a list of variables including 'IMMUNOGLOBULIN HEAVY CHAIN V...', 'CAST, 1-BP DEL, 1750G', 'TPM3, MET9ARG', 'UGT1A1, IVSAS3, A-G, -2', 'BIOLOGICAL_PROCESS variable', 'PROTEIN variable', 'GENE variable', 'PROT2CC variable', and 'PROT2PROT variable'. The main area displays a query graph with 'GENE 0' (blue oval) and 'PROTEIN 0' (pink oval) connected by an 'encodes' relationship. Below the graph, there are buttons for 'Delete', 'No node selected', and 'Search by Gene 0 URI count, Protein 0 URI count'. A table shows the results for 'Gene 0 URI count' (255461) and 'Protein 0 URI count' (525012). On the right, there are buttons for 'Export as...', 'Query', '2 nodes shown', 'No filters set', 'Export queries', and 'Load queries'. A red box highlights the 'Distinct' and 'Count' buttons.

5.2. VALUES

This clause permits the assignment of one or more values to a specific variable. For example, cis regulatory modules identified in two tissues of interest. First we indicate the subject node (CRM variable), then we include in the corresponding object property (observed in) the values to be included. Click on "+" to include values and click on OK when all values are listed.

The screenshot shows the query building interface. On the left, there is a search bar and a list of variables including 'IMMUNOGLOBULIN HEAVY CHAIN V...', 'CAST, 1-BP DEL, 1750G', 'TPM3, MET9ARG', 'UGT1A1, IVSAS3, A-G, -2', 'ROOT variable', 'TAXONOMIC_RANK variable', 'TFAC2GENE variable', 'CRM2GENE variable', and 'CRM variable'. The main area displays a query graph with 'CRM 0' (blue oval) and 'observed in' relationship. A dropdown menu is open, showing options like 'is instance of', 'has source', 'has evidence origin', 'is defined by', 'observed in', 'has evidence', 'involved in positive regulation of', 'molecularly interacts with', 'part of', 'has version', 'is subclass of', and 'involved in'. The 'observed in' option is selected, and a list of URI values is displayed: 'http://purl.obolibrary.org/obo/UBERON_0002107', 'http://purl.obolibrary.org/obo/UBERON_0000948', and 'http://purl.obolibrary.org/obo/UBERON_0002107'. A red box highlights the 'Enter URI values' input field and the 'OK' button.

5.3. Filters of properties

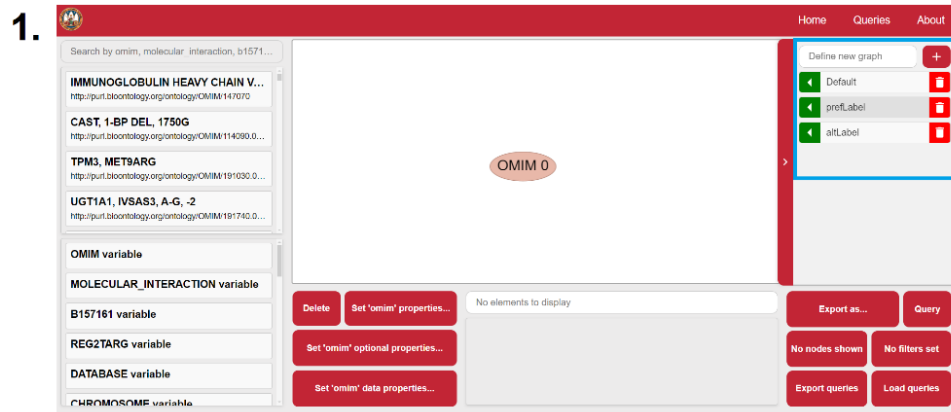
Clicking on "data properties" we can also filter attributes. If the value is a string we can use the operator "=" to indicate an exact value, or the operator "⊆" to indicate a substring contained in the string. If the value is numeric we can use the operators =, >, ≥, <, ≤. For example, we filter the variables chromosome and CRM to obtain the enhancers that overlap at certain coordinates.

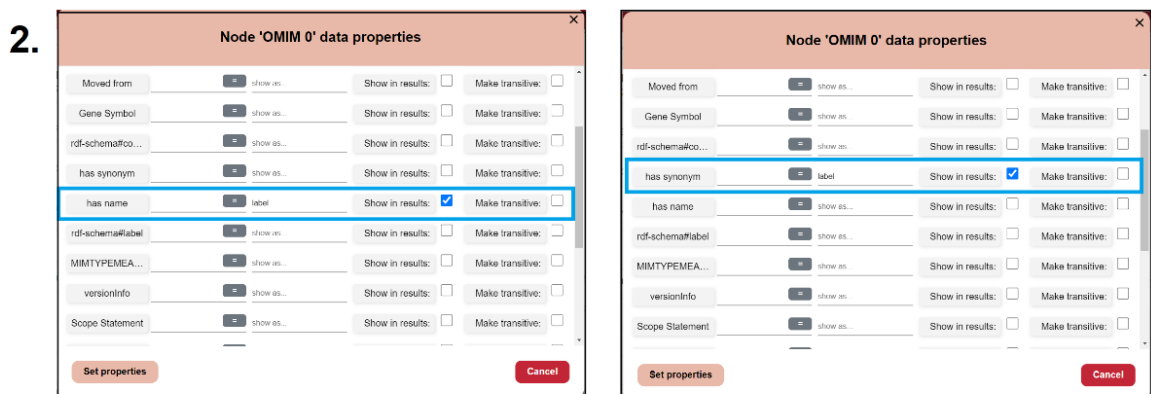
The screenshot displays the INTUITION web interface. At the top, there's a navigation bar with 'Home', 'Queries', and 'About' links. Below it, a search bar contains the text 'Search by omim, molecular_interaction, b1571...'. A list of search results is shown on the left, including 'IMMUNOGLOBULIN HEAVY CHAIN V...', 'CAST, 1-BP DEL, 1750G', 'TPM3, MET9ARG', and 'UGT1A1, IVSAS3, A-G, -2'. The main area shows a graph with two nodes: 'CHROMOSOME 0' (yellow oval) and 'CRM 0' (blue oval), connected by a 'part of' relationship. Below the graph, there are buttons for 'Delete', 'No node selected', and 'Search by Crm 0 Label, Crm 0 URI'. A table lists 'Crm 0 Label' and 'Crm 0 URI' with various identifiers and URLs. On the right, there are buttons for 'Export as...', 'Query', '1 nodes shown', 'No filters set', 'Export queries', and 'Load queries'. Below the main interface, two dialog boxes are shown. The first is titled 'Node 'CHROMOSOME 0' data properties' and contains fields for 'is instance of', 'has name', and 'category', each with a 'show as...' button and a 'Make transitive' checkbox. The second dialog is titled 'Node 'CRM 0' data properties' and contains fields for 'in taxon', 'is instance of', 'involved in pos...', 'has definition', 'start position', 'has name', 'end position', and 'is defined by', each with a 'show as...' button and a 'Make transitive' checkbox. Both dialogs have 'Set properties' and 'Cancel' buttons.

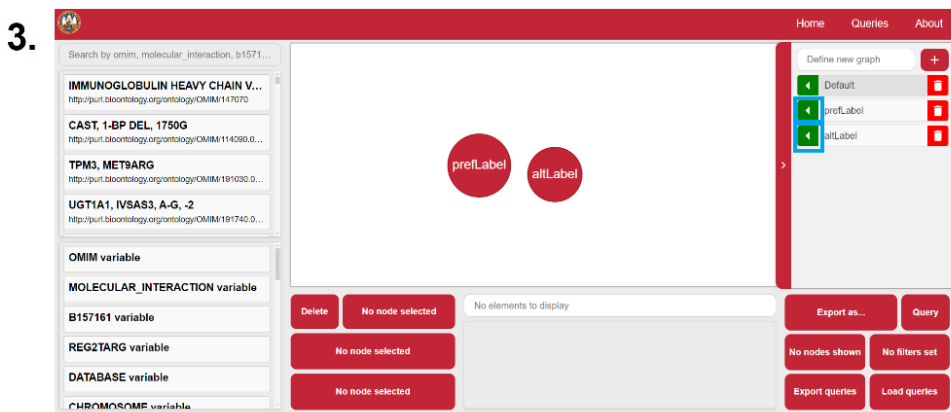
Variables can also be filtered clicking on "Filters set" button.

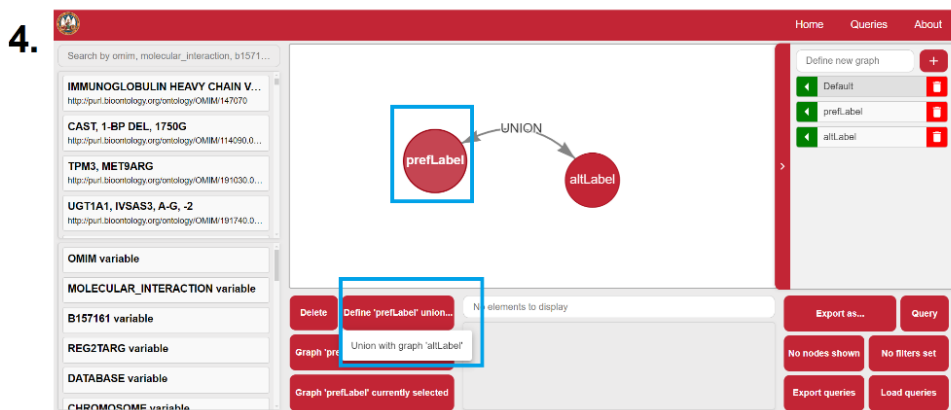
5.4. Creating and filtering variables

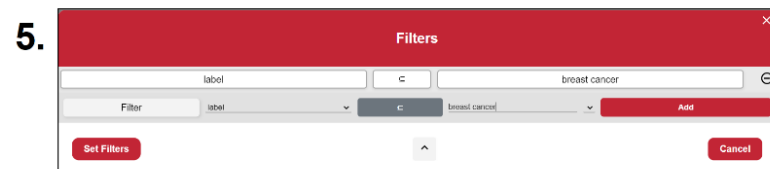
INTUITION supports the generation and filtering of new variables. This functionality is implemented in "Nodes shown" > "Bindings" button. For example, by subtracting the end and start positions of the CRMs we obtain the length of the sequences in a new variable. Then, we can filter this new variable in the "Filters set" button.

1. 

2. 

3. 

4. 

5. 

6.

The screenshot displays the OMIM database interface. On the left, a sidebar lists various OMIM variables including 'IMMUNOGLOBULIN HEAVY CHAIN V...', 'CAST, 1-BP DEL, 1750G', 'TPM3, MET9ARG', 'UGT1A1, IVSAS3, A-G, -2', 'OMIM variable', 'MOLECULAR_INTERACTION variable', 'B157161 variable', 'REG2TARG variable', 'DATABASE variable', and 'CHROMOSOME variable'. The main area shows a graph with two red circular nodes labeled 'prefLabel' and 'altLabel' connected by a 'UNION' relationship. Below the graph, there are buttons for 'Delete', 'Define 'prefLabel' union...', and 'Graph 'prefLabel' currently selected'. A search bar at the bottom center allows searching by OMIM ID, Label, or URI. Below the search bar, a table lists search results with columns for 'Omim ID', 'Label', and 'Omim ID URI'. On the right side, there are buttons for 'Export as...', 'Query', 'OMIM ID', '1 filter set', 'Metrics...', 'Load queries', and 'Load queries'.

6. Use Cases

The following Use Cases were developed in the paper "*Analysis of the landscape of human enhancer sequences in biological databases*". The corresponding queries are attached for reproducibility and as examples of use.

1. Use case 1: json files to load [here](#).
2. Use case 2: json files to load [here](#).
3. Use case 3: json files to load [here](#).