

Ondex tutorial and user guide

contact@ondex.org

September 2010

Contents

1	Data visualisation	5
1.1	Loading Networks and Basics	6
1.2	Analysing Networks	12
1.2.1	Filtering	13
1.2.2	Searching	19
1.2.3	Right-clicking	20
1.2.4	Annotating	23
1.2.5	Exercise	23
2	Data integration	28
2.1	Using Ondex Integrator	28
2.1.1	Using Ondex Integrator - an Exercise	29
2.2	Using the scripting console	38
2.2.1	Basic functionality and parsing of delimited files	38
2.2.2	Concepts	41
2.2.3	Relations	42
2.3	Accession based mapping examples	43
2.3.1	No parameters	43
2.3.2	EquivalentCC parameter (equivalent concept class)	43
2.3.3	EquivalentCV parameter (equivalent accession type)	43
2.3.4	WithinCVMMapping parameter (within data source)	43
2.3.5	IgnoreAmbiguity parameter (use ambiguous accessions)	44
2.3.6	CCrestriction parameter (concept class restriction)	44
2.3.7	CVrestriction parameter (accession type restriction)	44
2.3.8	WithinCVMMapping parameter (within data source) and EquivalentCV parameter (equivalent accession type)	44
2.4	Exercise	45
2.4.1	Scripting Console	45
2.4.2	Integrator	45
3	GUI Reference	46
3.1	Installation	46
3.1.1	Requirements	46
3.1.2	Installer	46

3.1.3	Testing the requirements	47
3.1.4	How to set the PATH variable	47
3.1.5	To the attention of sysadmins	48
3.1.6	Using Ondex behind a firewall	48
3.2	Loading Networks	48
3.2.1	Loading Existing Biological Networks into Ondex	49
3.2.2	Creating an empty network and manually adding concepts and relations	49
3.2.3	Saving Files	49
3.3	Metagraph	50
3.4	Manipulating Networks	50
3.5	Buttons in toolbar	51
3.6	Search bar in toolbar	53
3.7	Menu system	54
3.7.1	File	54
3.7.2	Edit	54
3.7.3	View	55
3.7.4	Appearance	55
3.7.5	Tools	57
3.7.6	Help	59
3.8	Integrator	59
3.9	Console	61
3.10	Statistics	61
4	How do I compare genomes in Ondex?	63
5	How do I analyse micro-array data using Ondex?	76
6	How do I look at QTLs in Ondex?	90
7	How do I study protein-protein interactions in Ondex?	96
A	FAQs	105
B	List of accessions	107
C	List of relation types	124
D	Importing data in Ondex	136
D.1	Parsers	136
D.1.1	OXL import	136
D.1.2	Aracyc	136
D.1.3	Atregnet	136
D.1.4	Atregnet2	137
D.1.5	Biogrid	137
D.1.6	BioCyc	137
D.1.7	Brenda	138

D.1.8	Coex	138
D.1.9	Correlationtab	139
D.1.10	Customtab	140
D.1.11	Drastic	140
D.1.12	EC	140
D.1.13	Fasta	140
D.1.14	Genericobo	141
D.1.15	Generif	141
D.1.16	Go	142
D.1.17	Goa	142
D.1.18	Gramene	142
D.1.19	Kegg52 (experimental package)	142
D.1.20	Kegg52 (stable)	143
D.1.21	Kegg53 (stable)	144
D.1.22	Medline	144
D.1.23	Nodelist	145
D.1.24	Nwb	146
D.1.25	Omim	146
D.1.26	Pdb	146
D.1.27	Pfam	146
D.1.28	Plntfdb	146
D.1.29	Poplar	147
D.1.30	Psimi	147
D.1.31	Psimi2	147
D.1.32	Sbml	147
D.1.33	Sbml2	147
D.1.34	Simplegraph	148
D.1.35	Tab	148
D.1.36	Tableparser	149
D.1.37	TAIR9	150
D.1.38	Taxonomy	150
D.1.39	Tigrricefasta	151
D.1.40	Timeseries	151
D.1.41	Transfac	151
D.1.42	Transpath	151
D.1.43	Uniprot	152
D.1.44	Wordnet	152

E Mapping data in Oindex 153

E.1	Mapping	153
E.1.1	Concept accession-based mapping (Memory-efficient) . . .	153
E.1.2	BLAST based	154
E.1.3	Cross-species	155
E.1.4	GDS equality	156
E.1.5	Inparanoid	156
E.1.6	Name based	157

E.1.7	Ortholog prediction	157
E.1.8	Sequence2pfam	158
E.1.9	StructAlign	159
E.1.10	Tmbased	159

Chapter 1

Data visualisation

A network consists of genes/proteins/metabolites as concepts and interactions represented as links i.e. relations between concepts. Firstly, we will look at its basic user interface. Then we will load up a network to show the commonly used features and some of the core functionality such as layouts, annotators and filters.

Help on starting Ondex is available in Section 3.1. Once launched, you should see a window that looks like this:

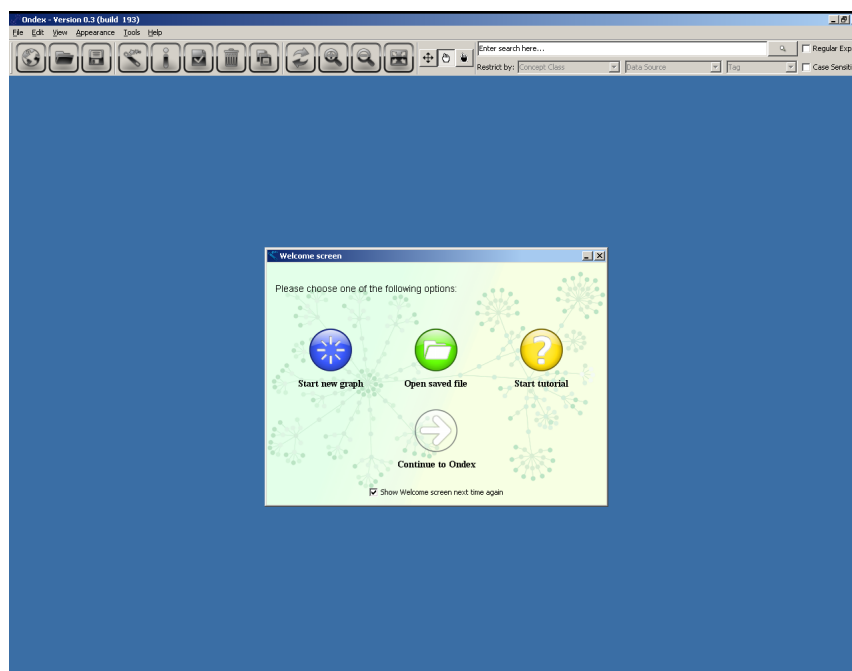


Figure 1.1: Launching Ondex

- At the top of the Ondex window there is a toolbar which contains the menus, icons and search bar. The name of each icon is shown when the mouse pointer hovers over it.
- The rest of the window is used to show visualization windows and pop-up dialog boxes to set up filters, annotators, etc.
- When Ondex is launched, a welcome screen is displayed in the middle of the main window. This welcome screen window offers users four buttons which are shortcuts to opening an empty network, opening an existing network, starting the tutorial or continuing to Ondex without taking any immediate action. Finally, at the bottom of the window, users are given the option of unticking a box if they do not wish this welcome screen window to pop up automatically next time the tool is started.

1.1 Loading Networks and Basics

We will now load a network (a subset of AraCyc) to show the commonly used features of Ondex's user interface. More information on loading and saving networks is available in Section 3.2.

- Go to File -> Open (or click on the second button in the icon bar)
- You should see an Open File Dialog
- Open the Tutorial_files/main_part folder, select aracyc_subset.xml.gz and click on open

You should see the following:

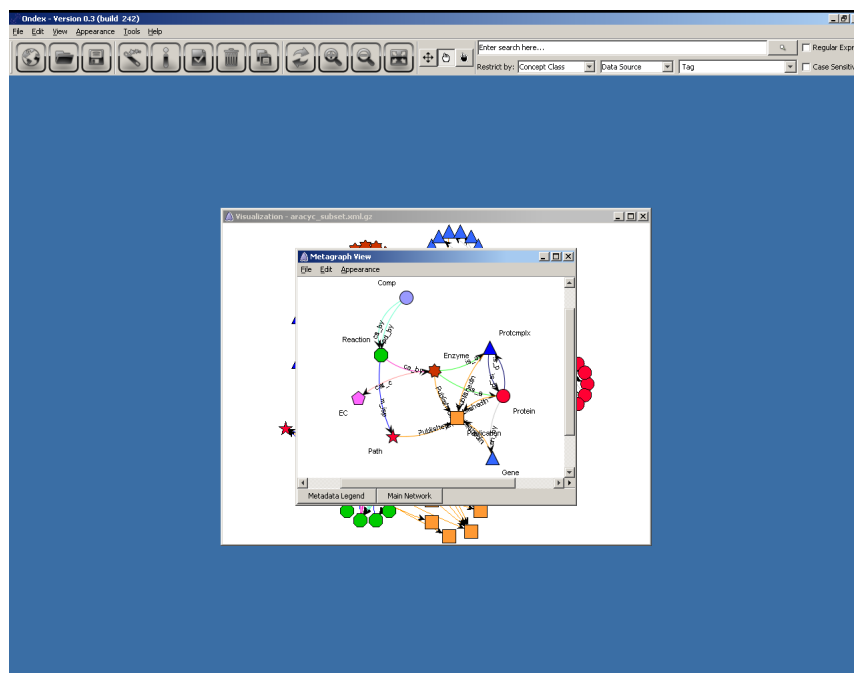


Figure 1.2: Metagraph - Aracyc

The first window that shows is the “Metagraph View”. Note that the main network is minimized in the bottom left-hand side corner if the whole graph takes long to load (in which case it is a good idea to filter it first, see Section 1.2), otherwise it will appear behind the metagraph (as it is here). The metagraph gives an overview of all the different types of concepts present on the main network, as well as all the different types of relations which link them.

The metagraph view has 2 buttons at the bottom of its window (for more information, Section 3.3 explains its menu):

- Metadata Legend: will pop up a window which displays a modifiable colour/shape legend as well as the number of concepts and relations in the network for concept classes, data sources, relation types and evidence types
- Main Network: will open the main network visualization window if it was minimized

Exercise Trying to understand the metagraph before opening the main network usually helps. The main network is opened with the “Circular” layout which is a circular arrangements of all the concept classes in the network.

Moving concepts around in the metagraph should help you to make sense of it. Figure 1.3 is an example of how the metagraph can be laid out.

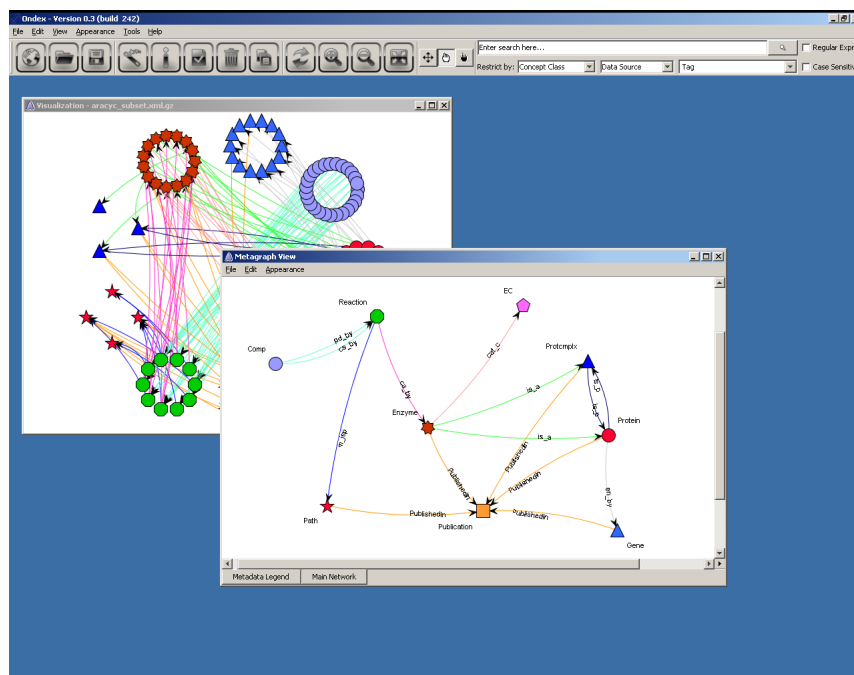


Figure 1.3: Rearranged metagraph of Aracyc

A protein is encoded by a gene (en_by). A protein is part of a protein complex (is_p). A protein is consumed by or produced by a reaction (cs_by, pd_by). An enzyme is a protein or a protein complex (is_a). An enzyme has a catalysing class EC (cat_c). An enzyme is co-factored (or activated) by a compound (co_by). A compound is consumed by or produced by a reaction (cs_by, pd_by). A reaction is a member, is part of a pathway (m_isp). A reaction is catalysed by an enzyme (ca_by). Finally, most concepts can be mentioned in publications or published in (pub_in).

Right-clicking on concepts/relations on the metagraph shows the number of concepts/relations of that type in the network. It also gives users the opportunity to untick “Visible”. This will make concepts/relations of that type invisible on the main network. The concept/relation in the metagraph will appear in a paler colour.

Note: Making concepts invisible in the network will not delete them from the network altogether. If you wish to do so, use Edit -> Delete Hidden Items. A window will pop up to ask you for confirmation.

Exercise In the metadata legend, clicking on colours/shapes will allow you to pick different colours/shapes for concept classes, data sources, relation types and evidence types (see 4 tabs). In order to see it take effect, refresh the metadata (bottom button in the metadata legend) and click on use Appearance -> Update Display.

For example, a protein complex is of the same colour as a gene concept classes. In order to make it more distinct, we can change its colour by clicking on the coloured rectangle (first column). Figure 1.4 is obtained.

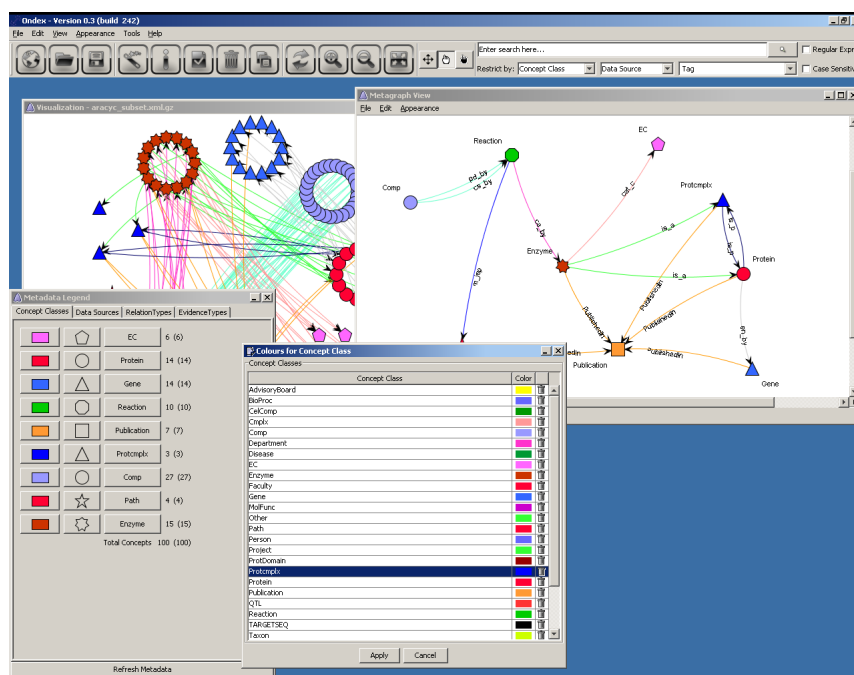


Figure 1.4: Click on a colour in the metadata legend. A window pops open to change colours

Clicking on the coloured rectangle in the new window will pop open a “Pick a colour” window (see Figure 1.5).

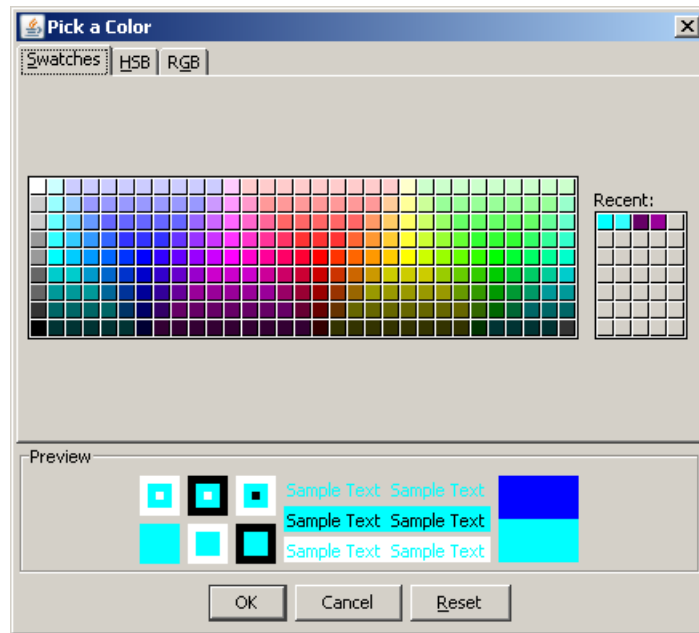


Figure 1.5: Select a colour of your choice

Select a colour, click on OK, Apply, Refresh Metadata and Appearance -> Update Display to finally obtain Figure 1.6.

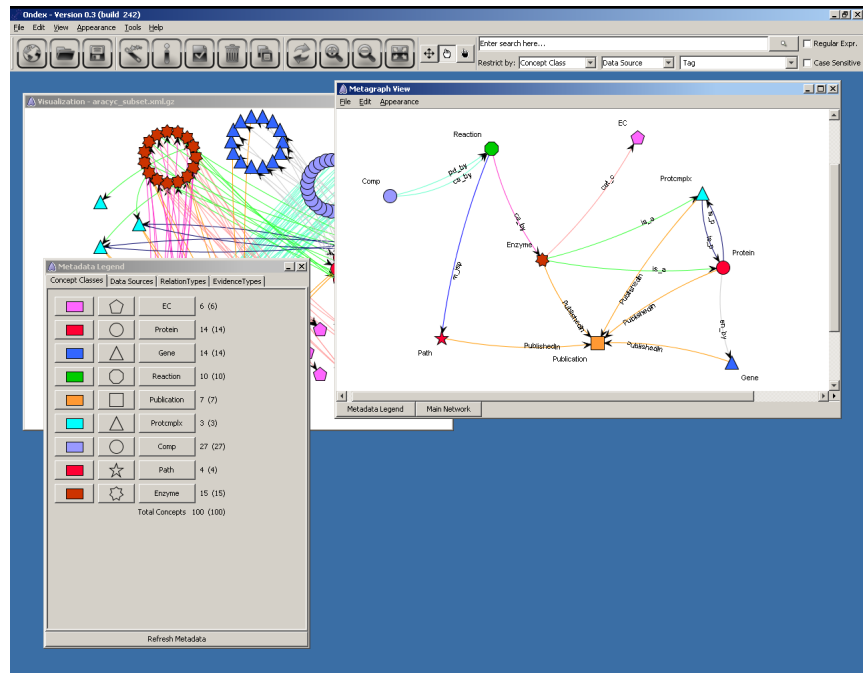


Figure 1.6: The colour of the protein complex metaconcept has changed to turquoise

The metadata legend has four tabs: Concept Classes, Data Sources, Relation Types and Evidence Types. Figure 1.7 shows the Relation Types tab for this example.

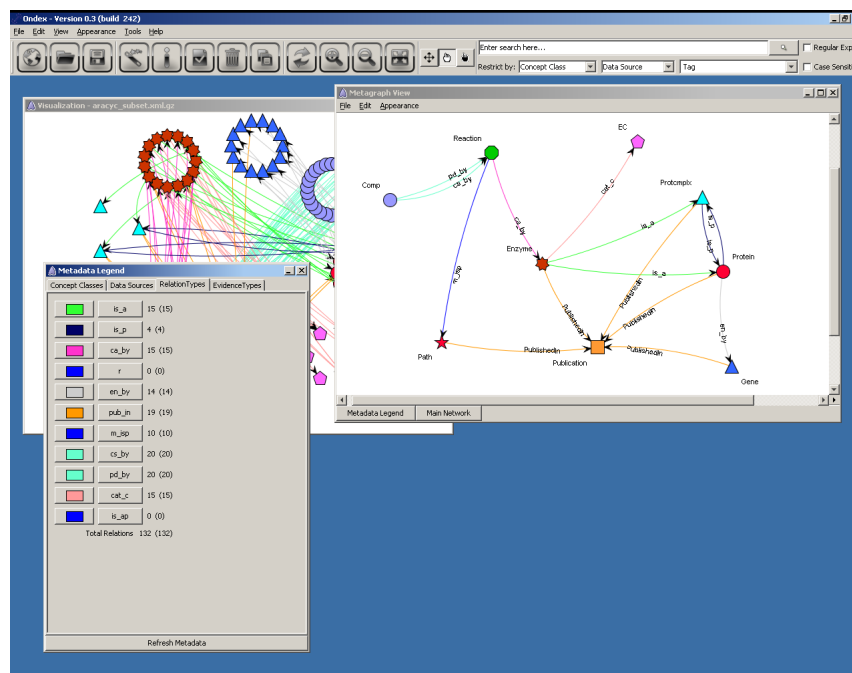


Figure 1.7: The Relation Types tab in the metadata

1.2 Analysing Networks

If the main network is large and wasn't opened behind the metagraph, you can click on "Main Network" in the "Metagraph View" window or maximize the visualization window that has been sitting in the bottom left-hand side corner all along. Opening the visualization window can take a while when there are a lot of concepts and relations to draw. Two options are possible at this stage:

1. if your computer is fast or you are patient, you can open the main network
2. otherwise you can use filters (under Tools) or a search (search bar, top-right corner) to open a smaller subset of the data. If the data contains "tags" (predefined subsets of the data) then using the Tag Filter is usually interesting.

1.2.1 Filtering

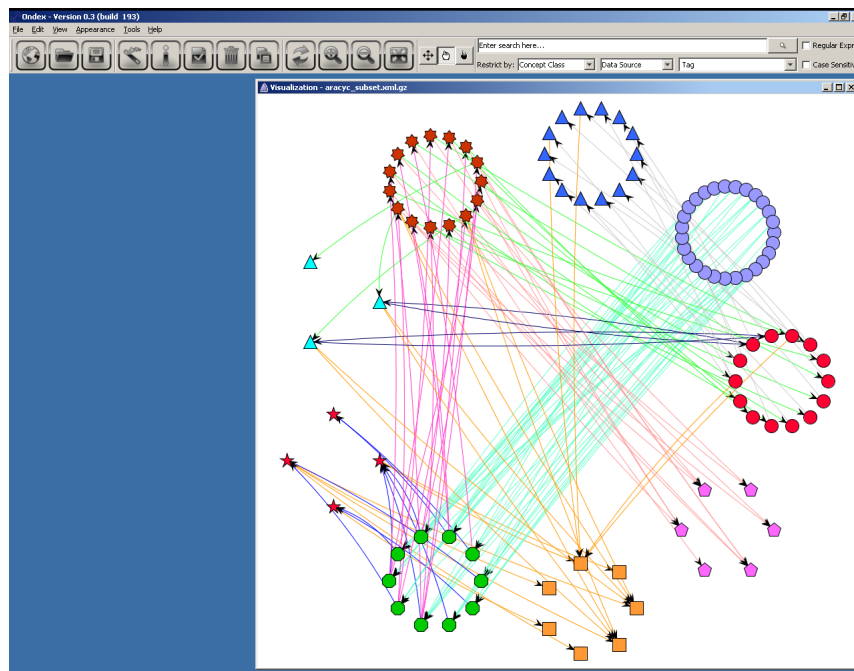


Figure 1.8: Subset of AraCyc drawn with the circular (default) layout

For example here, rather than looking at the whole subset of AraCyc in Figure 1.8, let us select a single pathway to analyse. In order to do so, select the Tag Filter under Tools -> Filters. This filter is fully explained in the F1 help documentation of Ondex. It offers the possibility of combining different tags. For now, we will simply filter down the network to a single pathway. Let us select GDP-L-fucose biosynthesis II as shown in Figure 1.9 (click on the view tag cell and select GDP-L-fucose biosynthesis II from the list of possible tags).

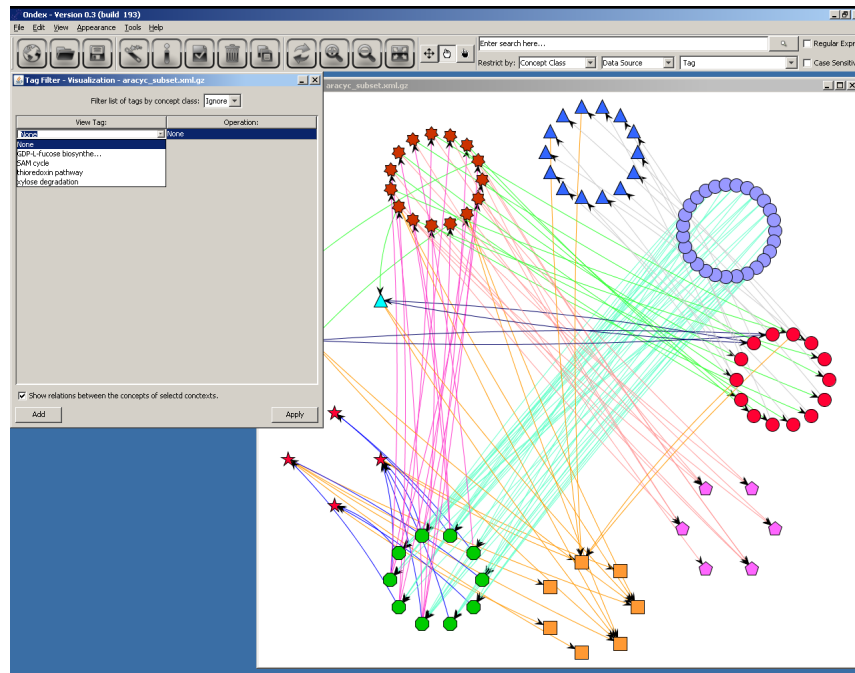


Figure 1.9: Pick a tag from the drop-down list

Figure 1.10) shows the resulting network.

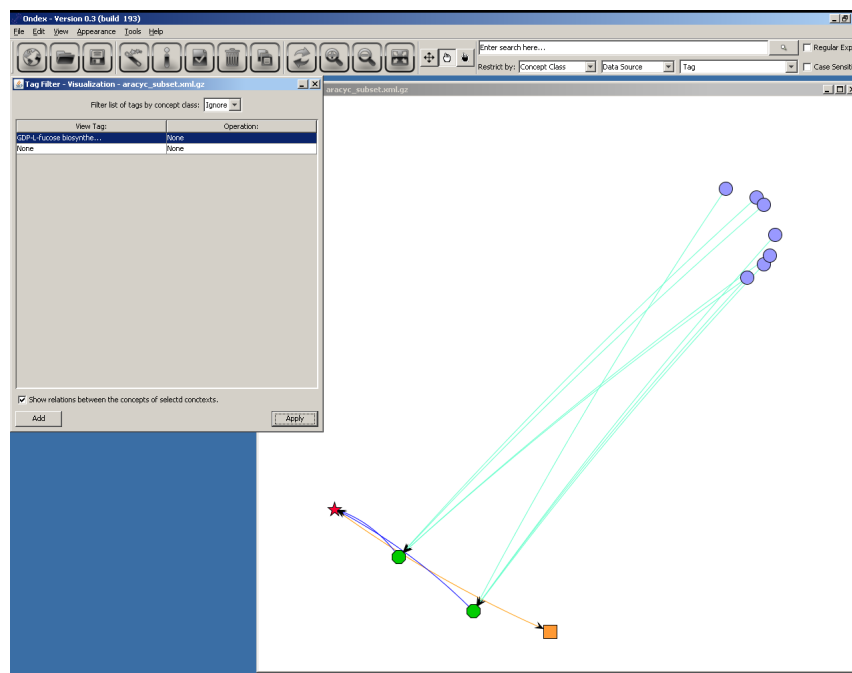


Figure 1.10: The GDP-L-fucose biosynthesis II pathway from the Aracyc database

We can use Appearance -> Layouts -> More -> Hierarchical for Onxex to organise the concepts hierarchically (see Figure 1.11).

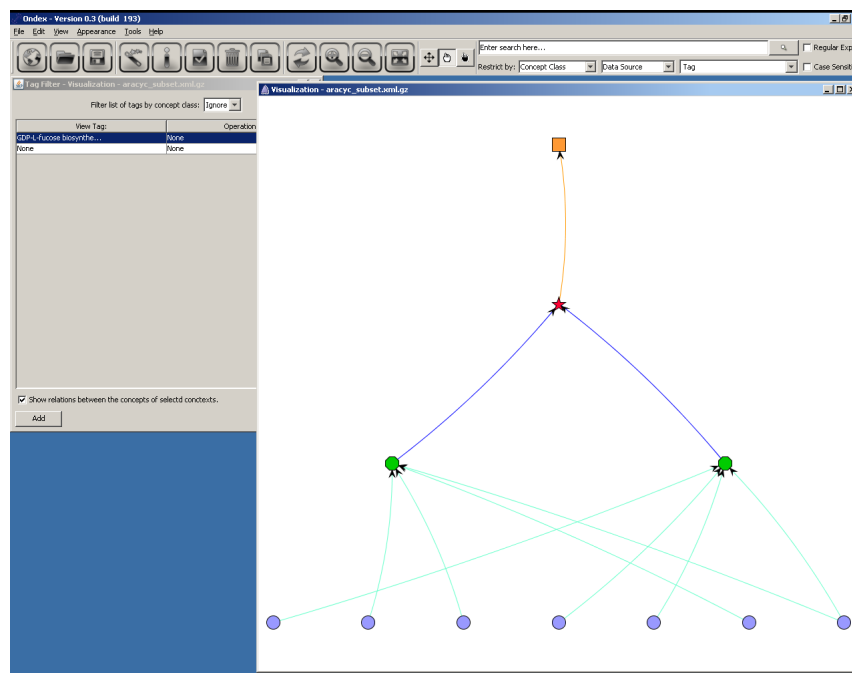
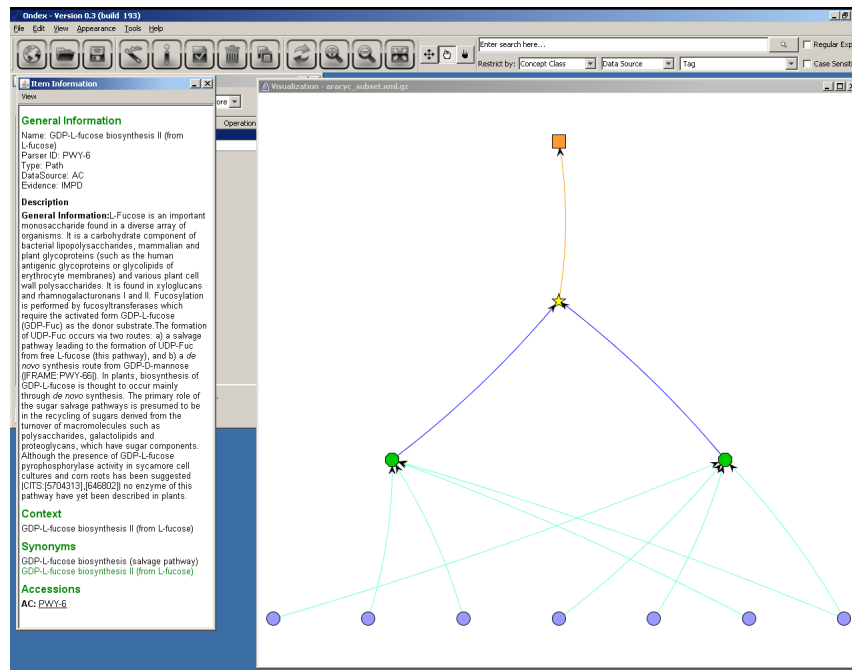


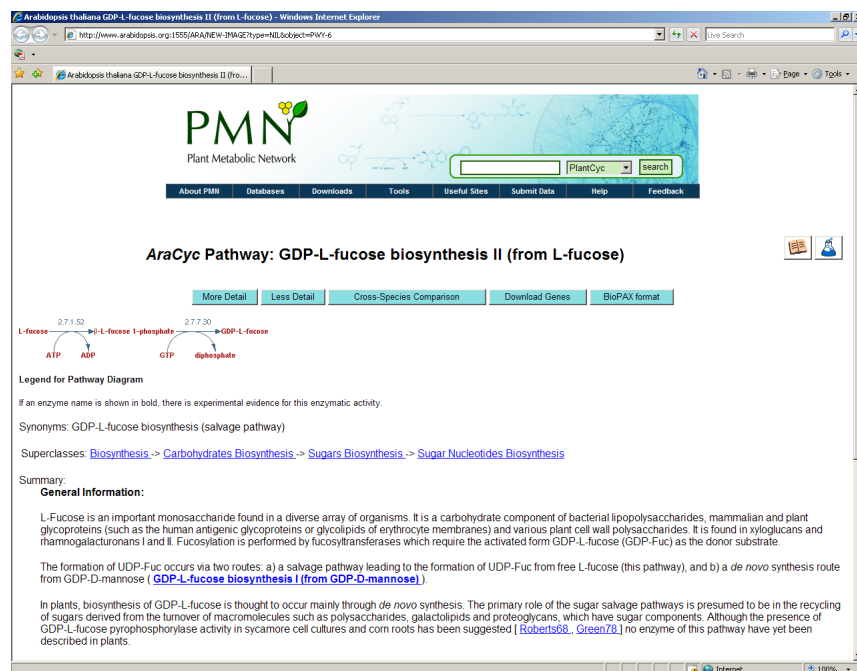
Figure 1.11: Hierarchical layout of the GDP-L-fucose biosynthesis II in Aracyc

You can then select a concept, for example, the pathway itself (click on it, it will become yellow). Then you can click on the “i” icon (a tooltip shows when you mouse over the icon) to see information about the selected pathway, as shown in Figure 1.13(a).

The information gives a link to a website. Clicking on this link opens a web browser as shown in 1.12.



(a) Item Information icon launches an information window



(b) The Aracyc website showing the selected pathway

Figure 1.12: Item information and its links to websites

Exercise Study the reaction shown in the website and try to understand how it is represented in Oindex.

Help: The first thing you might want to do is use the View menu to add concept and relation labels on the network so you know what is what without needing the “Item Information” window. Use Appearance -> Labels to show labels of concepts and relations.

Help: To move concepts around, you can either pick them (in picking mode - last icon before search bar in the toolbar or Edit -> Mouse Mode) and move them one by one. You may also press shift to select a few concepts with the left mouse button and using the left mouse button again you can drag and drop these group of concepts.

Figure 1.13 shows what can be obtained by following this process. Note: the “Metagraph View” window has been minimized on this screenshot. The reactions previously shown on the webpage have been reconstituted. Here, the compound that is a product of the first reaction and a substrate of the second has been selected and is automatically shown in the “Item Information” window. Note: SMILES stand for “simplified molecular input line entry specification”. Note: If you have closed the Metadata Legend (previously launched from the metagraph view window), you may launch it again using Edit -> Legend.

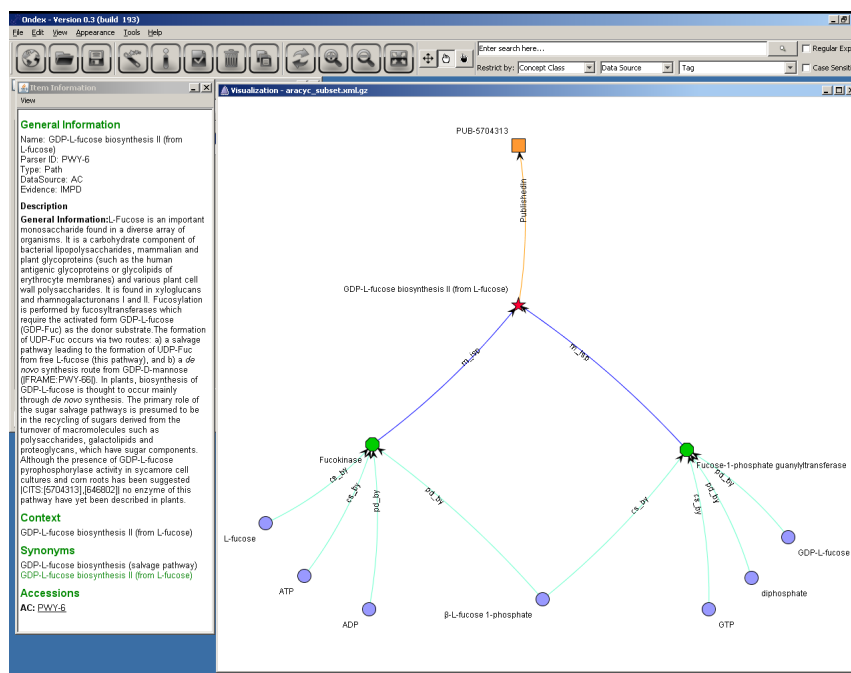


Figure 1.13: The GDP-L-fucose biosynthesis II as shown in Figure 1.13(b)

Other filters and annotators (as well as layouts) are available and documented in Ondex. Users can press F1 to access this documentation at any time. They can also place their mouse over tooltips for a short description to pop up, or click on the tooltip for the documentation window to open. In “Tutorial_files”, “Annotators_and_Filters”, a “readme_annotators_filters.txt” explains which file (within this directory) is used for each example in the documentation’s screenshots (to allow users to play with the data themselves).

1.2.2 Searching

Ondex includes a Search feature (see Figure 3.7), which enables you to quickly find concepts and relations.

Figure 1.14: The Ondex search bar

For example, searching for fucose in our current graph will give use the results shown in Figure 1.15

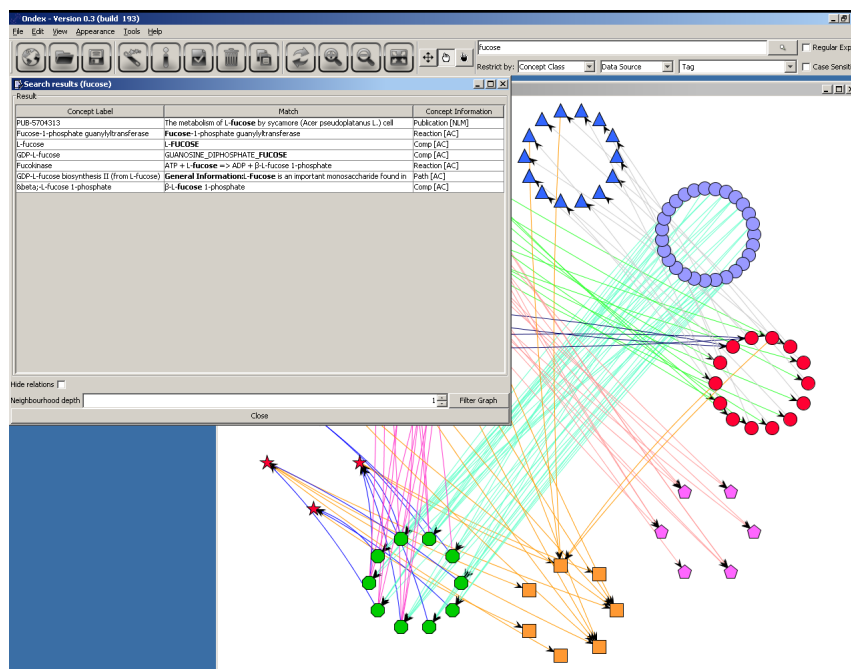


Figure 1.15: Search results for “fucose” in Aracyc

If you select (single click) one item in the search results, you will need to single click on the icon “Zoom In” in order for Ondex to zoom in on this particular concept. You can then use the mouse scroll to get to the concept. If you select several items (using the Control key) in the search results, Ondex will automatically zoom in to show all those items as close as possible.

Configuring an Ondex Search

- At the end of the toolbar, there are 2 options that can be configured for each search. If you tick “Regular Expression”, you may enter a java regular expression. An example would be: `AT\dG`. It will match “AT” followed by any digit and “G”. For more information on regular expressions in java, please visit <http://java.sun.com/docs/books/tutorial/essential/regex/>. If you tick “Case Sensitive”, the search will be sensitive to lower/upper case of each character.
- Under the search box, there are 3 drop-down lists which allow users to restrict their search (by Concept Class, Data Source or Tag). An example is provided below.

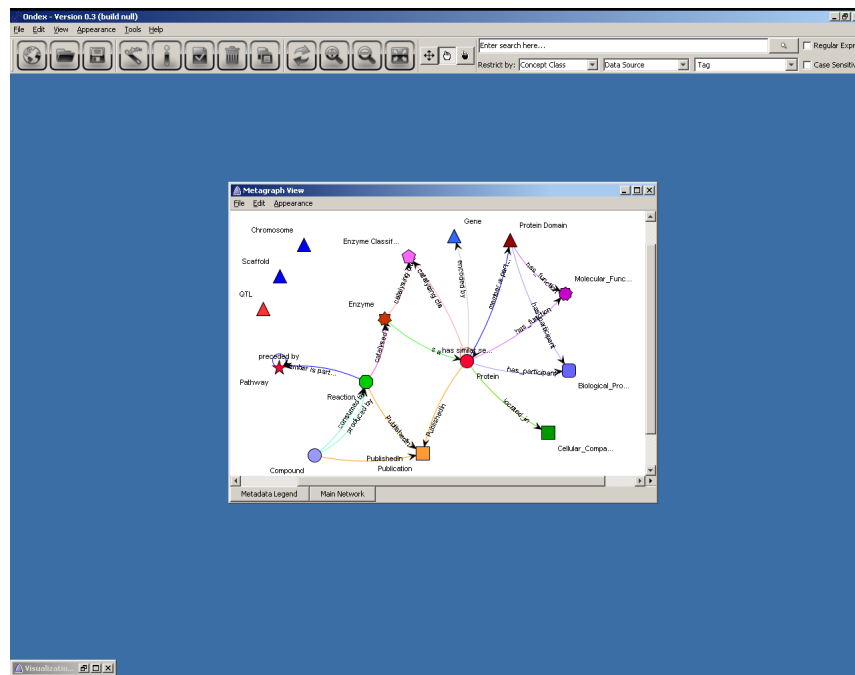
1.2.3 Right-clicking

Right-clicking on one or a selection of concepts/relations also offers users the possibility to narrow down the graph to what they are interested in. Close all the windows you have opened in Ondex (View -> Windows -> Close All).

- Go to File -> Open (or click on the second button in the icon bar)
- You should see an Open File Dialog
- Open the Tutorial_files/main_part folder, select poplar_subset.xml.gz and click on open

This graph results from a data integration (see Chapter 2) pipeline using information from JGI, UniProtKB, TAIR, Gramene, Gene Ontology (GO), GOA (GO Annotations), TraitOnt, PlantOnt, Medline, AraCyc, PoplarCyc and Pfam which aims to identify genes controlling biomass production in willow. The willow genome is not yet sequenced. Poplar is the first tree with fully sequenced genome but not much is known yet about the function of poplar genes. This study (see Chapter 6) uses comparative genomics to compare poplar to Arabidopsis.

The metagraph shows a lot of different concept classes and relation types, including “Molecular Function”, “Biological Process” and “Cellular Compartment” - the 3 GO categories (see Figure 6.2).



Exercise In this exercise, we will look at the gene named JGI-767106. As it is lacking functional annotation, we wish to study the poplar protein it encodes, its homologous proteins and the GO terms they have been associated with. Use the search bar and a combination of right-clicks to get to Figure 1.17 where only these concepts are visible on the graph.

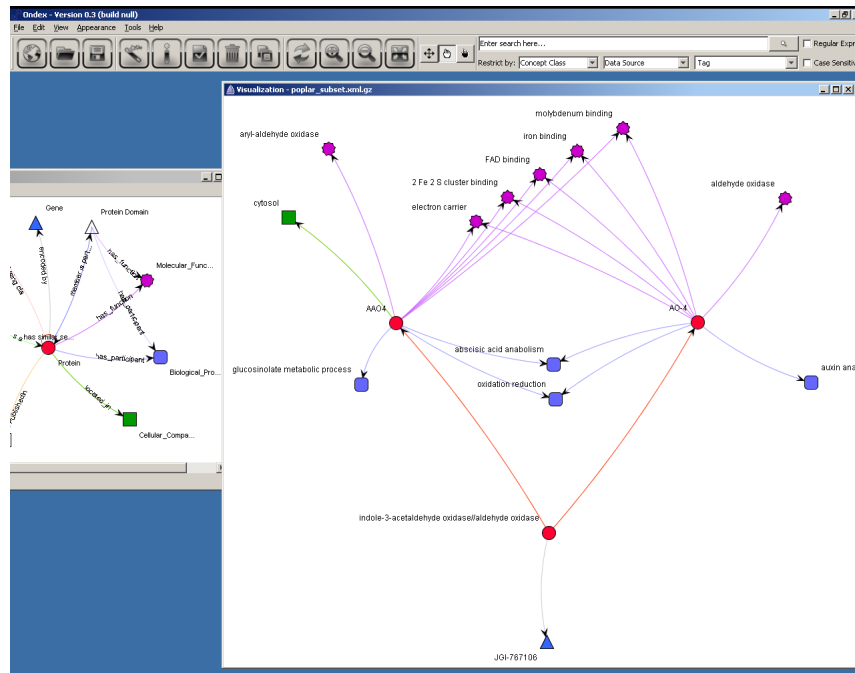


Figure 1.17: Visualisation window (and part of metagraph) at the end of the exercise

Help:

- In the search bar, type 767106 and restrict by Gene. You should obtain only one result.
- In the search results, select a neighbourhood depth of 1 and click on “Filter Graph”.
- Click on Main Network (from the metagraph or from the minimized window), you should see the gene you had searched for and the protein encoded by that gene.
- Click on Appearance -> Labels -> Concepts to add labels.

We are now interested in looking at homologous proteins which might be annotated with GO terms.

- Right-click on the protein and select Show -> Immediate Neighbourhood. Two proteins should appear as well as some protein domains and enzymes.
- Select the homologous proteins and check their item information using the “i” icon or View -> Item Info. You will find one is from Rice and one Arabidopsis.

- As we are not interested in the protein domains and enzymes, select one protein domain, right-click and select Hide -> Same Concept Class to get rid of all of them (same for enzymes).
- Select the two homologous proteins (click on one, it becomes yellow, press shift and select the second one)
- Right-click and select Show -> Immediate Neighbours by Concept Class -> and select the 3 categories of GO terms one by one so that neighbours of concept class Protein and Publication do not appear. Publications would be easy to hide again (right-click on any publication and select Hide -> Same Concept Class) but proteins would be more complex as we wish to keep our orthologous proteins (one would need to select the two poplar proteins added, right-click and select Hide -> Selected Concept(s)).

1.2.4 Annotating

Once a graph is filtered down to a region of interest, annotators (Tools -> Annotators) allow you to scale your concepts/relations, add shapes and colours to them based on the attributes they hold. We will first work on a data integration problem in Section 2.1 then we will analyse the results in Chapter 4 using a few annotators. Other application cases also show case various annotators.

1.2.5 Exercise

This exercise allows you to practice analysing networks in Oindex using a combination of the tools available. Close all the windows you have opened in Oindex (View -> Windows -> Close All).

- Go to File -> Open (or click on the second button in the icon bar)
- You should see an Open File Dialog
- Open the Tutorial_files/main_part folder, select poplar_exercise.xml.gz and click on open

This graph contains the same data that was in the file opened in Section 1.2.3 but a different subset. Firstly, given a candidate gene (652033) we wish to study its annotations and their GO hierarchy. We also wish to annotate the GO terms based on their information content. To do so:

- Search for 652033
- Select the protein found in the search results (the corresponding concept becomes yellow on the graph)

- Run the shortest path filter. The results are shown on Figure 1.18.

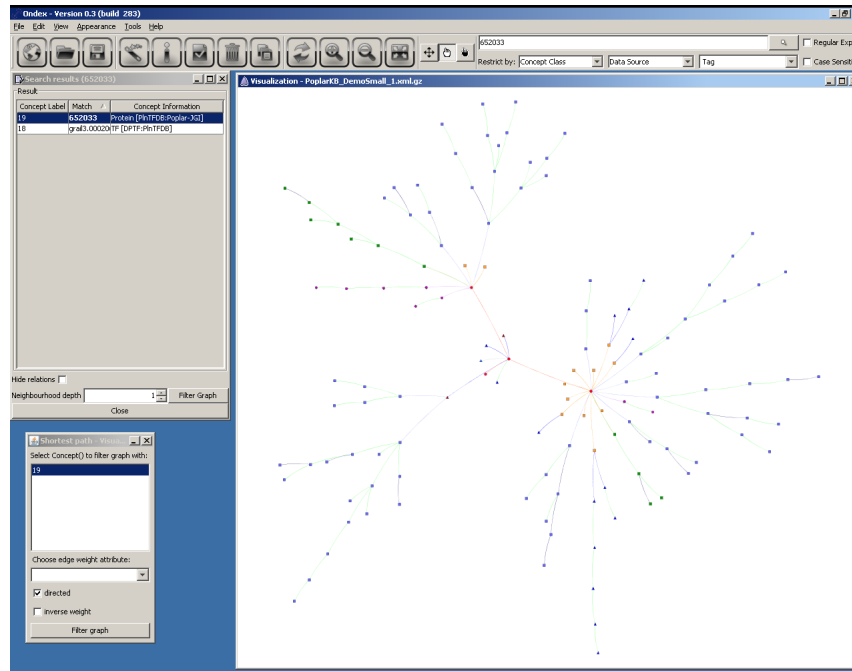


Figure 1.18: Screenshot after applying the shortest path filter

Help:

- Tools -> Filters -> More -> Shortest Paths -> Shortest Path
- Select the protein (its ID is 19)
- Do not select any relation weight attribute (from the drop-down list)
- Tick the box for directed (so that you can see the GO hierarchy)
- Click on “Filter Graph”
- Use the Gem layout (Appearance -> Layout -> Gem)
- Use Appearance -> Smooth Relations to make relations anti-aliased thereby smoother
- run an annotator to scale GO terms concepts based on their attribute called “Information Content” (which is added to the data by running a Transformer in the experimental package called “Annotate Information Content”, see Section 2.1). The results are shown on Figure 1.19.

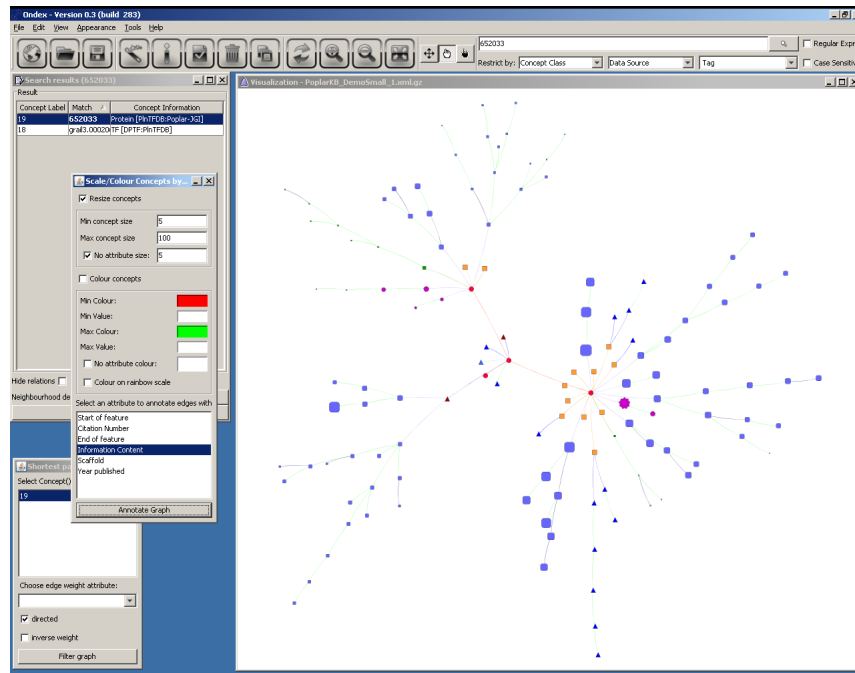


Figure 1.19: Screenshot after annotating GO terms with their information content

Help:

- Tools -> Annotators -> Scale/Colour Concepts by Numerical Value
- Tick the box “No attribute size” to select a default value for concepts which do not hold an Information Content attribute
- Change the sizes to 5, 100 and 5 (or experiment by yourself)
- Do not tick the box “Colour Concepts” (we want to keep the colour by concept class colour legend)
- Select “Information Content” in the list of attributes available
- Click on “Annotate Graph”

Secondly, given a trait ontology term (“shoot branching”) we wish to show all Poplar proteins which are related to it in the network. To do so:

- Search for shoot branching

Help:

- You may restrict the search to a particular concept class (here Trait Ontology)

- Select the Trait Ontology term found in the search results (the corresponding concept becomes yellow on the graph)
- Look at the immediate neighbours of this selected concept
 - Use the neighbourhood filter in the search results with a depth of 1 (click on “Filter Graph”) or right-click on the concept and use Show -> Immediate Neighbourhood
 - Use the Gem layout (Appearance -> Layout -> Gem)
- Find a publication which contains information of interest (Hint: PubMed ID 13130077)
- Show the immediate neighbours of this selected concept
- Find the protein mentioned in the article amongst the neighbours (LAX)
- Show the immediate neighbours of this selected concept.
- Colour proteins by species (results shown on Figure 1.20).

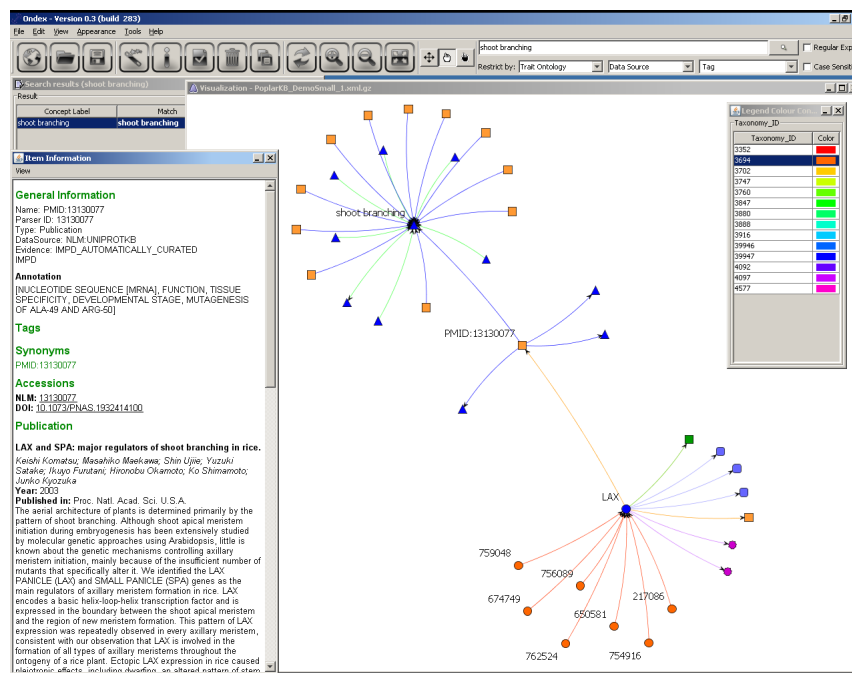


Figure 1.20: Screenshot after annotating neighbouring proteins by taxonomy

Help:

- Tools -> Annotators -> Colour Concepts by General Attribute
- Select Taxonomy_ID and click on “Annotate Graph”
- Change the relations’ width to reflect the BLAST result between LAX and these homologous proteins. The results are shown on Figure 1.21.

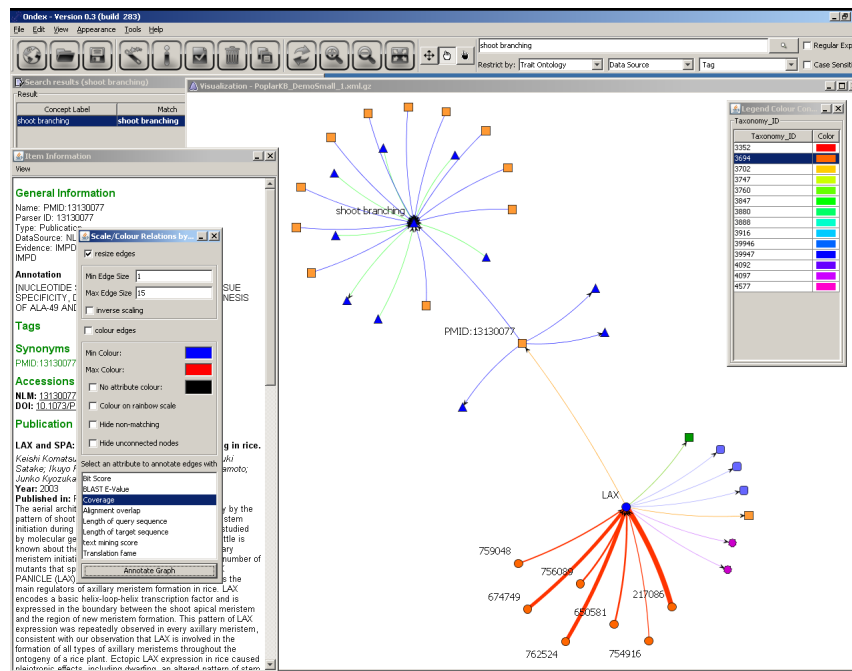


Figure 1.21: Screenshot after annotating neighbouring proteins by taxonomy

Help:

- Tools -> Annotators -> Scale/Colour Relations by Numerical Value
- Do not tick colour relations
- Select Coverage (to reflect the BLAST result between LAX and these homologous proteins)
- Click on “Annotate Graph”

Chapter 2

Data integration

Ondex is used to visualize networks produced by running an Ondex workflow. Originally, the data integration was done by writing a workflow in XML format and by running it in the back-end. In the following section, we are going to introduce the Integrator which allows users to prepare and run an Ondex workflow within Ondex (for more information see Section 3.8). Typically, a simple example for a workflow would be to first select parsers (to import data) then to select mapping methods, some filtering methods to improve the quality of the graph and, finally, an export method (*e.g.*, OXL export to open the results in Ondex).

2.1 Using Ondex Integrator

The Integrator is available from Tools -> Integrator in the menu of Ondex. Figure 3.8 shows it when it first opens up. For each category showing in the left pane, stable plugins will be displayed by default. To browse more plugins (under Windows, these experimental plugins will only be available if selected during installation), please untick the option in the Configure menu of the integrator. As explained above, composing a workflow using the Integrator will require selecting various steps or Ondex plugins - each of which offer parameters to be set if needed. N.B.: Grayed out parameters are optional.

2.1.1 Using Ondex Integrator - an Exercise

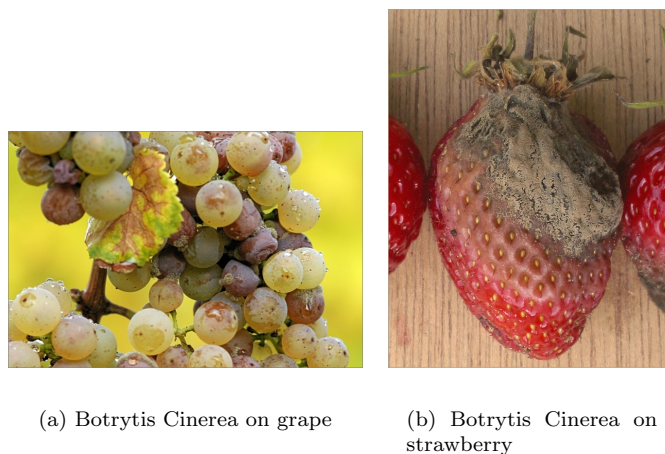


Figure 2.1: Botrytis Cinerea

Botrytis Cinerea (http://www.broad.mit.edu/annotation/genome/botrytis_cinerea/) is a necrotrophic fungus that affects many plant species. Its most notable hosts are wine grapes (see Figure 2.2(a)) and strawberries (see Figure 2.2(b)). Genomic data are available for this fungus, however we do not have any phenotypic data. The PHI-base database (<http://www.phi-base.org/>) is a reference database of virulence and pathogenicity genes validated by gene disruption experiments. It contains information on experimentally verified pathogenicity, virulence and effector genes from bacterial, fungal and Oomycete pathogens identified from the scientific literature.

For this example, techniques of comparative genomics were combined with data integration methods to predict pathogenicity genes in pathogenic organisms. The InParanoid algorithm¹ was implemented as part of the Ondex system for the prediction of orthologous groups of proteins. The workflow we are working on in this section is illustrated in Figure 2.2.

¹Remm M., Storm C.E., Sonnhammer E.L. (2001). "Automatic clustering of orthologs and in-paralogs from pairwise species comparisons". *Journal of molecular biology*, 314(5):1041-52, PubMed ID: 11743721.

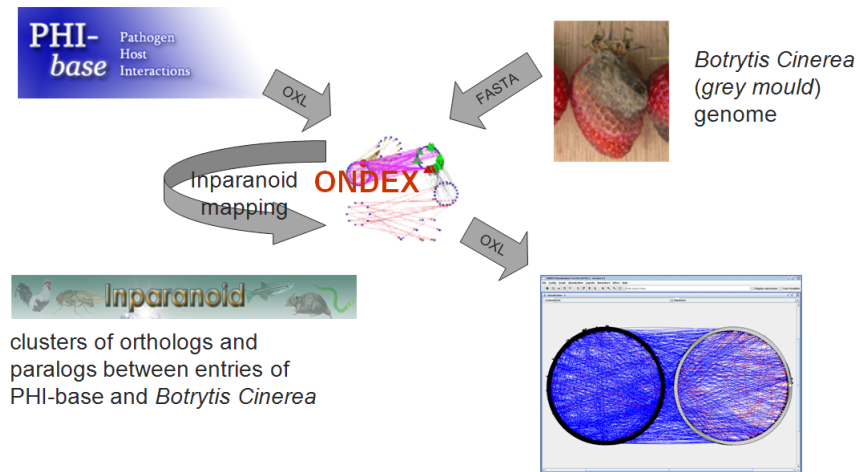


Figure 2.2: Integrating PHI-base and the genome sequence of Botrytis Cinerea

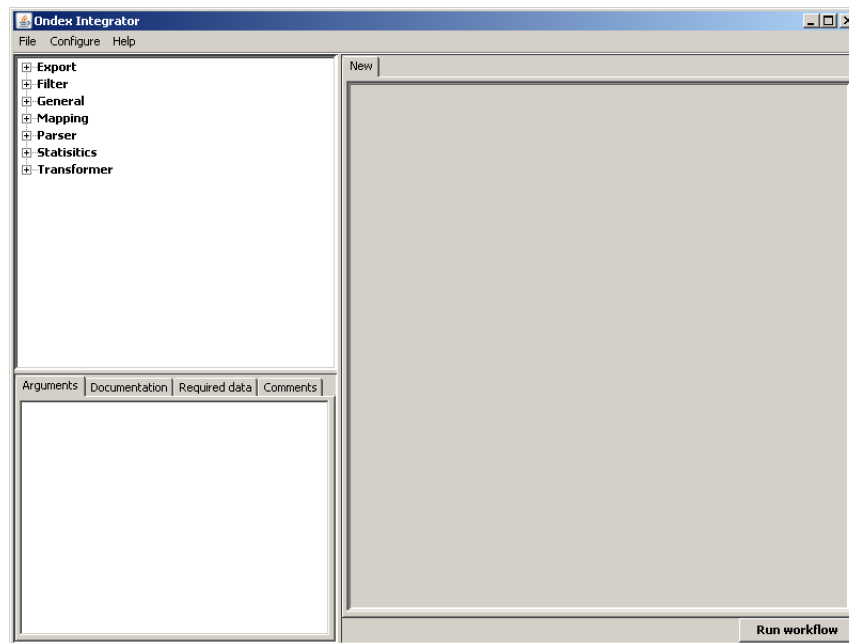


Figure 2.3: Data integration using the graphical interface - Ondex Integrator

Note: Running this workflow with the whole genome sequence from Botrytis will take about an hour. If you wish to see quick results during this hands-on tutorial, we suggest you use a fasta file which has been prepared for this purpose:

short.fasta which can be found under Tutorial_files/Data_integration/Integrator. This file contains a tenth of the entire genome and will take about 3 minutes. The results show a smaller graph which can be analysed nevertheless. In Chapter 4, we will load up the results for the whole genome and study them using Ondex annotators (these results are saved under Tutorial_files/Application_cases/botrytis_results.xml.gz).

Note: Before running a workflow in the integrator, make sure you save your current workflow so you can re-open it as you might not get all the parameters right the first time around. For this example, the steps needed in the workflow are as follows (*All the files needed here are saved under Tutorial_files/Data_integration/Integrator*):

- Parsing the file containing the PHI-base database (Parser - Oxl import, see Figure 2.4). Note: A “Create new graph” pre-step is automatically loaded.

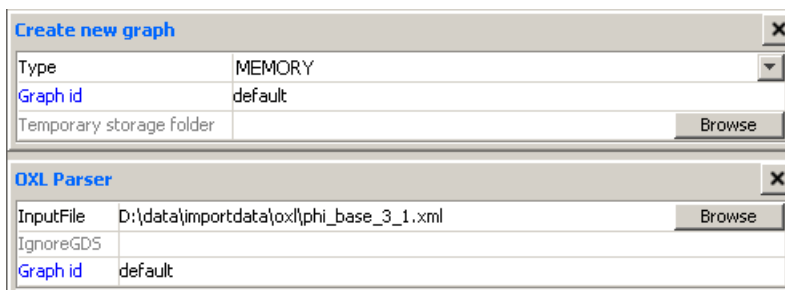


Figure 2.4: Parser - Oxl import

Opening this OXL file directly would produce the metagraph shown in Figure 2.5.

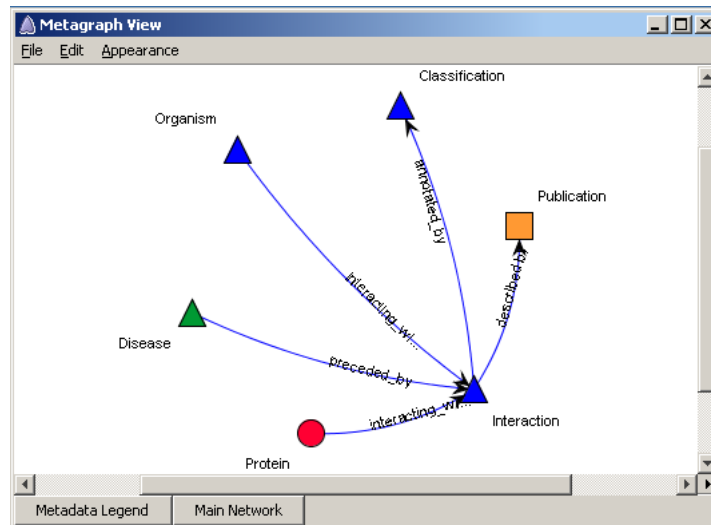


Figure 2.5: Metagraph - PHI-base

If you have time, have a look at the main graph. In particular, try to find which concepts hold the phenotypic information we are actually interested in for this workflow. This will later determine which concept classes you can filter out, which ones you should keep and/or combine.

- Parsing the file downloaded from the Broad institute website for the sequence of Botrytis Cinerea (Parser - Fasta, see Figure 2.6) Note: The taxonomy ID for Botryotinia fuckeliana was found on <http://www.ncbi.nlm.nih.gov/Taxonomy/> (40559). CC stands for Concept Class and is therefore “Protein”. SeqType stands for Sequence Type and is in this case “AA”, amino acid.

FASTA file parser	
FastaFiles	D:\data\importdata\fasta\short.fasta
FastaFileType	simple
TaxId	40559
CC	Protein
CV	unknown
POS_TO_ACCESSION	
SeqType	AA
Separator	
AccessionRegex	
Graph id	default

Figure 2.6: Parser - Fasta

An export (select Export in the drop-down list and OXL export in the list

of exporters) after this step alone would produce the metagraph shown in Figure 2.7.

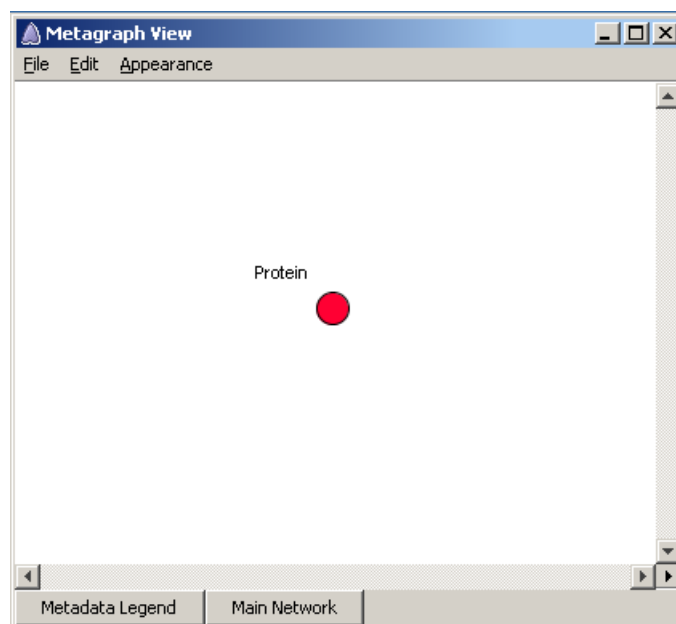


Figure 2.7: Metagraph - Botrytis

- After parsing the two data sources, they are mapped to each other using the “inparanoid” mapping which uses pair-wise sequence alignment results generated by BLAST <http://www.ncbi.nlm.nih.gov/blast/download.shtml> (Mapping - InParanoid, see Figure 2.8). The Eval parameter is passed on to BLAST. “cutoff” and “overlap” are post filter parameters which specify a bit score cutoff and the minimum length of the match compared to the longest sequence. (*If Blast is not installed on your machine, please run the appropriate installer from Tutorial_files/Data_integration/Integrator/blast_installers. The inparanoid mapping method expects to be given the bin directory of your blast installation.*)

InParanoid	
PathToBlast	D:\blast_installers\bin
Evalue	1.0E-6
SeqGDS	AA
SeqType	AA
Cutoff	30
Overlap	0.5
Graph id	default

Figure 2.8: Mapping - InParanoid

An export at this stage would produce the metagraph shown in Figure 2.9.

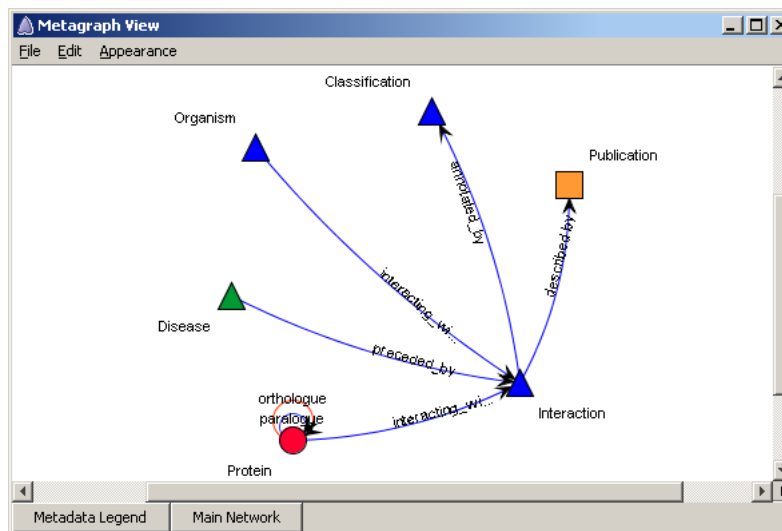
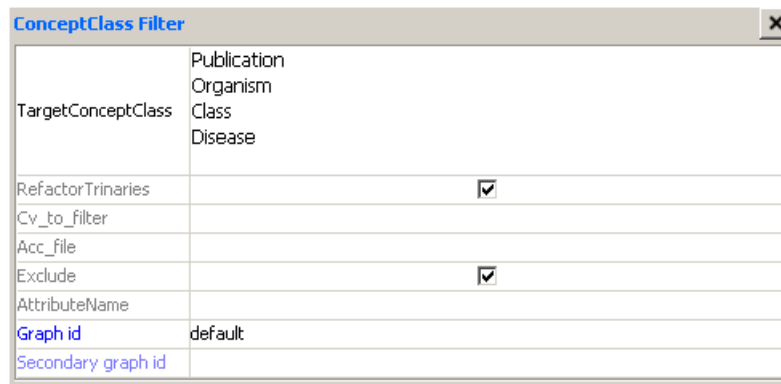


Figure 2.9: Metagraph - Results of the InParanoid mapping method

- To reduce the complexity of the network several concept classes we are not interested in are filtered out (Filter - ConceptClass, see Figure 2.10).

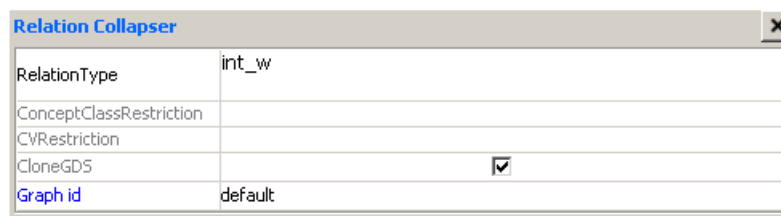


The 'ConceptClass Filter' dialog box contains the following fields and values:

TargetConceptClass	Publication Organism Class Disease
RefactorTrinaries	<input checked="" type="checkbox"/>
Cv_to_filter	
Acc_file	
Exclude	<input checked="" type="checkbox"/>
AttributeName	
Graph id	default
Secondary graph id	

Figure 2.10: Filter - ConceptClass

- In PHI-base a protein is always linked to an “Interaction” which carries the specific phenotype. To merge both information from protein and Interaction the relation type set “int_w” is collapsed and the connected concepts are merged into a single concept of concept class “Interaction:Protein”. (Transformer - Relation Collapser, see Figure 2.11)



The 'Relation Collapser' dialog box contains the following fields and values:

RelationType	int_w
ConceptClassRestriction	
CVRestriction	
CloneGDS	<input checked="" type="checkbox"/>
Graph id	default

Figure 2.11: Transformer - Relation Collapser

An export at this stage would produce the metagraph shown in Figure 2.12.

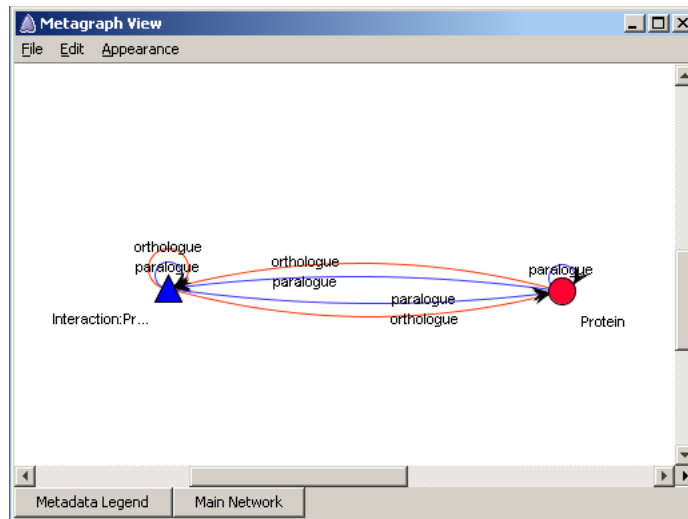


Figure 2.12: Metagraph - Results of the ConceptClass Filter and the Relation-Collapser Transformer

- An export at this stage would produce the graph shown in Figure 2.13.

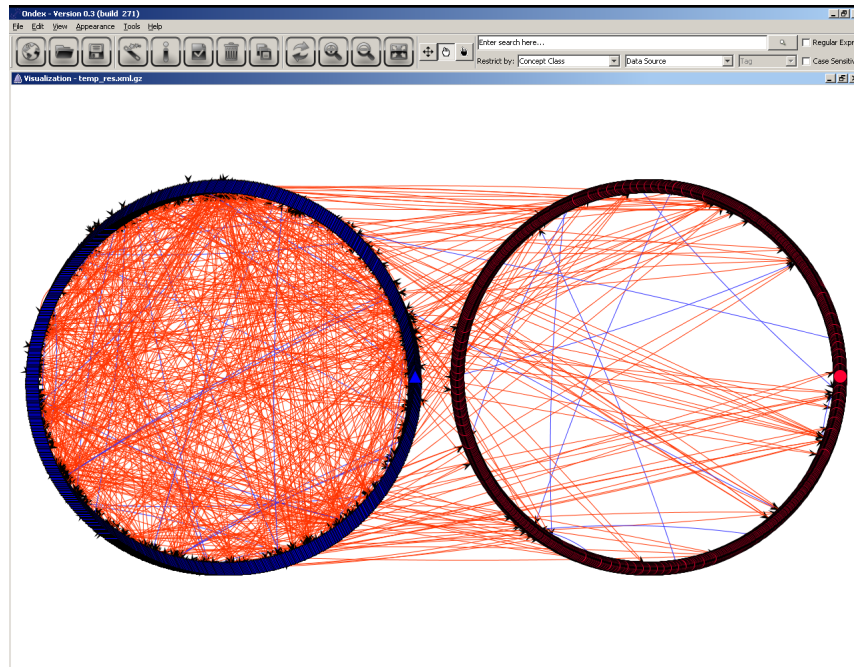


Figure 2.13: Main graph - Need for the Unconnected Filter

The unconnected filter is used to remove any isolated concepts (i.e. without any relations to the rest of the network). (Filter - Unconnected, see Figure 2.14)

Unconnected Filter	
RemoveContextDependencies	<input checked="" type="checkbox"/>
ConceptClassRestriction	
Graph id	default
Secondary graph id	

Figure 2.14: Filter - Unconnected

- An export at this stage would produce the graph shown in Figure 2.15.

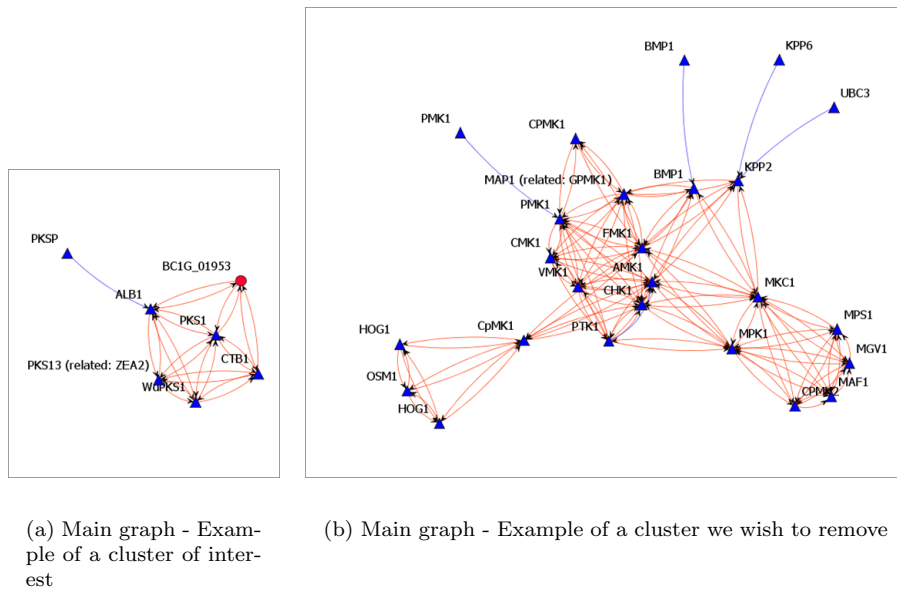


Figure 2.15: Two kinds of resulting clusters

Therefore the next step will only keep clusters of concepts in the network which contain at least one concept of concept class Protein from *Botryotinia fuckeliana*. (Filter - Isolateclusters, available in the experimental package - please untick the option in the Configure menu of the integrator, see Figure 2.16)

TargetConceptClass	Protein
TargetDataSource	
Graph id	default
Secondary graph id	

Figure 2.16: Filter - Isolateclusters

- The resulting network is exported as OXL (Export - Oxl export, see Figure 2.17) to a Gzip compressed XML file called short_botrytis_results.xml

ExcludeConceptsOfConceptClass	
ExcludeRelationsOfRelationType	
ExcludeGDSWithAttribute	
IncludeOnlyGDSAttribute	
IncludeOnlyConceptClass	
IncludeOnlyRelationType	
Pretty	<input checked="" type="checkbox"/>
GZip	<input checked="" type="checkbox"/>
ExportFile	D:\data\results_short_botrytis.xml.gz Browse
ExportIsolatedConcepts	<input checked="" type="checkbox"/>
Graph id	default

Figure 2.17: Export - Oxl export

Loading short_botrytis_results.xml in Ondex will look similar to Figure 2.18.

2.2 Using the scripting console

When Ondex does not provide any specific parser for the data a user is interested and this data is in a tab delimited format, it is possible to use a general tab delimited parser. This section explains how to use it.

2.2.1 Basic functionality and parsing of delimited files

Prototype based parsing API aims to simplify and speed up the import of custom user formats into Ondex graphs. As proprietary formats are highly variable it is impossible to write a universal parser to deal with all of the possible options. This approach aims to provide the middle ground way that retains all of the flexibility but takes care of the complexity involved with using the Ondex API directly. Most of delimited files can be parsed in about 5 lines. The following guide is primarily for using the API through the scripting interface, although it can also be used directly from Java.

Steps for example 1 (explained below):

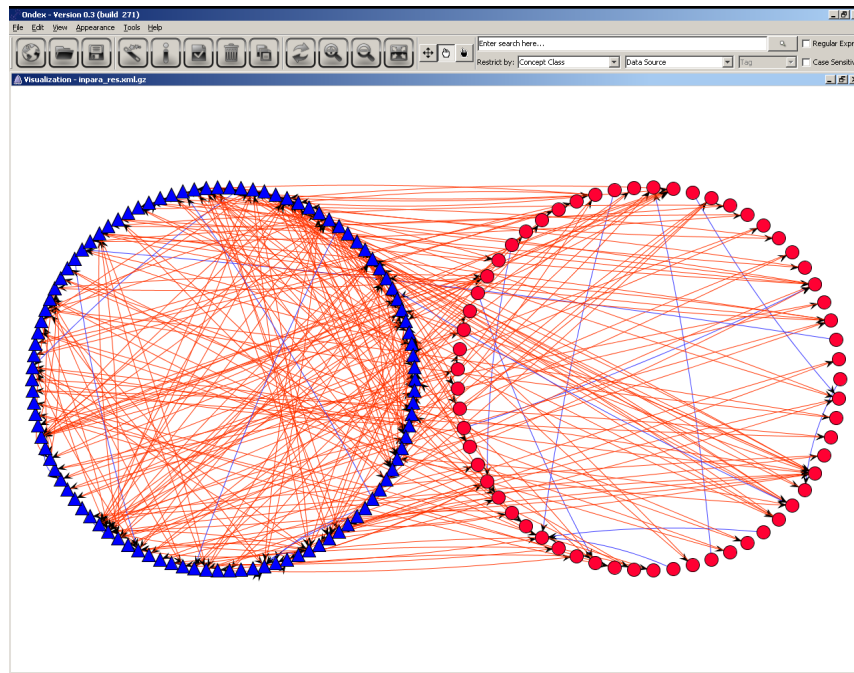


Figure 2.18: Resulting graph after integration

- Open a new empty graph
- Tools -> Console
- Type commands in the console or copy and paste the prepared script available at Tutorial_files/Data_integration/Console/readme_scripting.txt (change the path to test.tab so that it points to the correct file, make sure to keep forward slashes)
- Press enter (a quarter of a concept should appear in the top-left corner)
- Appearance -> Layouts -> Gem

Example 1 The tab file in Figure 2.19 produces the network on its right-hand side (using the commands underneath).

```
p = new PathParser(getActiveGraph(),
new DelimitedFileReader("C:/test.tab", " "));
```

This statement creates a new parser. `getActiveGraph()` call gives the reference to the last active viewer frame in Ondex (at least one must be open). The delimited file reader takes a path to file and a delimiter (in this case tab) as an argument.

Q9VU72	P23654		Nrt	0.86
Q9VI89	P23654		Nrt	0.34
P83949	P23654	Ubx	Nrt	0.85
P15364	P23654	Ama	Nrt	0.57
P23654	Q9VVC3	Nrt	Cpr73D	0.86
Q7KNS3	P23654	Lis-1	Nrt	0.75
Q9VDJ8	P23654		Nrt	0.86
Q9VB81	P23654	Cpr97Eb	Nrt	0.23
P19538	P23654	ci	Nrt	0.64

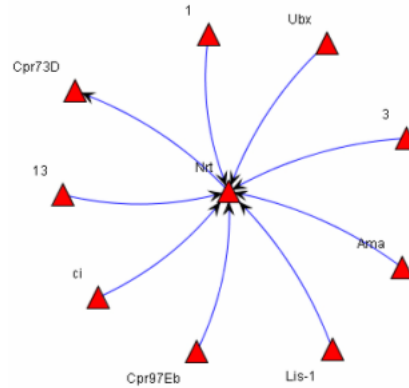


Figure 2.19: Tab-delimited file and generated graph

```

c1 = p.newConceptPrototype(
defAccession(0, "UNIPROT"), defCC("Protein"), defName(2));
c2 = p.newConceptPrototype(
defAccession(1, "UNIPROT"), defCC("Protein"), defName(3));

```

Here new concept prototypes are added to the parser. This method can take any number of arguments in any order. The arguments specify which user data (or which column) will be converted to which argument. In the above example, the integers 0, 1, 2, 3 represent the number of the column where the information can be found.

```

p.newRelationPrototype(
c1, c2, defGDS(4, "P-value", "NUMBER"));

```

This statement creates a relation prototype that will create a relation between the two concepts created from the prototypes c1 and c2. The source and target concepts are the only two required arguments. There can be any number of relation attribute arguments that can be in any order. Attributes get specified using defGDS.

```

p.parse();

```

Parses the file and creates the graph.

Example 2

```

p = new PathParser(getActiveGraph(), new DelimitedFileReader
("D:/workspace/core/data/importdata/Uniprot2AGI.tab", " "));
p.newConceptPrototype(defCC("Protein"),
defAccession(0, "UNIPROTKB"), defAccession(1, "TAIR"));
p.setProcessingOptions();

```

Sets the processing options to none - by default the concepts created will always be merged on accessions, but the parsing process will take much longer - even if no concepts actually satisfy this requirement.

```
p.parse();
```

Attribute specification arguments All attribute specification arguments are optional and can be supplied in any order. The type of the argument corresponds to the name of the function and the first argument is always either a position in the data vector (e.g. column in the case of the delimited files) or a string (in double quotes) when it is desired to have an argument as a constant value, rather than have it parsed from the data file.

2.2.2 Concepts

The diagram below outlines the data structure of an Ondex concept. Fields in green hold actual data, whereas fields in blue are compound attributes that group sub-attributes of their own. Three overlaid attribute boxes indicate that multiple instances of attributes of this type can exist. Attributes represent a special case because any number of attributes may exist as long as attribute name specified for each of them is unique. The boxes with dashed outline are Boolean values that should be specified as strings "true" and "false".

The only required parameter is the first one that can be either an integer or string, the rest of the parameters are optional.

Attribute types The valid options are:

```
"NUMBER", "TEXT", "INTEGER", "COLOR", "DOUBLE" and "OBJECT"
```

. e.g.:

```
defGDS("3.14", "Pi", "NUMBER");
```

Processing options To specify no processing options set empty processing options:

```
p.setProcessingOptions();
```

Disabling the post-processing options will increase the parsing speed considerably. The following processing options are available, from none to all tree may be set at the same time:

```
"MERGE_ON_ACCESSIONS"
```

```
(on by default),
```

```
"MERGE_ON_NAMES"
```

```
,
```

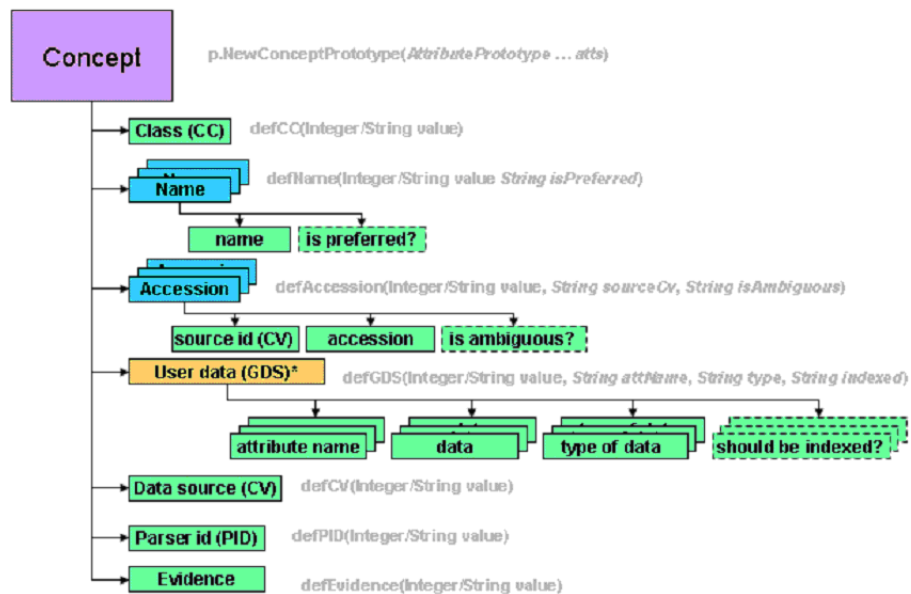
```
"MERGE_ON_GDS"
```

```
.
```

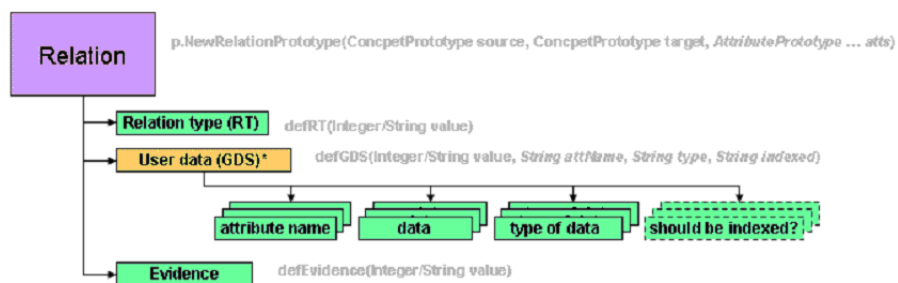
Accessions supported by Ondex are listed in Appendix B.

2.2.3 Relations

Figure 2.21(b) above shows the data structure of an Ondex relation and its possible attributes. Relation types supported by Ondex are listed in Appendix C.



(a) Data structure of an Ondex concept



(b) Data structure of an Ondex relation

Figure 2.20: Ondex data structure

2.3 Accession based mapping examples

This section illustrates how accession based mapping works with examples.

2.3.1 No parameters

Name	Concept Class	Data Source	Accession
P1	Protein	UNIPROTKB	UNIPROTKB:ACC1
P2	Protein	TAIR	UNIPROTKB:ACC1

P1 and P2 will map with no parameters needed because this mapping method maps concepts of the same concept class (Protein) and of different data source (UNIPROTKB and TAIR).

2.3.2 EquivalentCC parameter (equivalent concept class)

Name	Concept Class	Data Source	Accession
P1	Protein	UNIPROTKB	UNIPROTKB:ACC1
G1	Gene	TAIR	UNIPROTKB:ACC1

P1 and G1 will map if parameter EquivalentCC is set to "Protein, Gene" so that the mapping method recognises these two concept classes as equivalent.

2.3.3 EquivalentCV parameter (equivalent accession type)

Name	Concept Class	Data Source	Accession
P1	Protein	UNIPROTKB	UNIPROTKB:ACC1
P2	Protein	TAIR	TAIR:ACC1

P1 and P2 will map if parameter EquivalentCV is set to "UNIPROTKB, TAIR" so that the mapping method recognises these two accession type as equivalent.

2.3.4 WithinCVMapping parameter (within data source)

Name	Concept Class	Data Source	Accession
P1	Protein	UNIPROTKB	UNIPROTKB:ACC1
P2	Protein	UNIPROTKB	UNIPROTKB:ACC1

P1 and P2 will map if parameter WithinCVMapping is set to true so that the methods maps concepts of the same concept class and same data source.

2.3.5 IgnoreAmbiguity parameter (use ambiguous accessions)

Name	Concept Class	Data Source	Accession	Ambiguous
P1	Protein	UNIPROTKB	UNIPROTKB:ACC1	false
P2	Protein	TAIR	UNIPROTKB:ACC1	true

P1 and P2 will map if the parameter IgnoreAmbiguity is set to true so that the mapping method ignores the P2's ambiguous flag.

2.3.6 CCrestriction parameter (concept class restriction)

Name	Concept Class	Data Source	Accession
P1	Protein	UNIPROTKB	UNIPROTKB:ACC1
P2	Protein	TAIR	UNIPROTKB:ACC1
G1	Gene	UNIPROTKB	UNIPROTKB:ACC1
G2	Gene	TAIR	UNIPROTKB:ACC1

If no parameters are specified, two relations will be created: one between the two genes and one between the two proteins. If the parameter CCrestriction is set to Protein, only the two proteins will map.

2.3.7 CVrestriction parameter (accession type restriction)

Name	Concept Class	Data Source	Accession
P1	Protein	UNIPROTKB	UNIPROTKB:ACC1
P2	Protein	UNIPROTKB	TAIR:ACC1
P3	Protein	TAIR	UNIPROTKB:ACC1
P4	Protein	TAIR	TAIR:ACC1

If no parameters are specified, two relations will be created: one between P1 and P3, and one between P2 and P4. If the parameter CVrestriction is set to TAIR, only P2 and P4 will map.

2.3.8 WithinCVMapping parameter (within data source) and EquivalentCV parameter (equivalent accession type)

Name	Concept Class	Data Source	Accession
P1	Protein	UNIPROTKB	UNIPROTKB:ACC1
P2	Protein	UNIPROTKB	TAIR:ACC1

P1 and P2 will map if WithinCVMMapping is set to true and EquivalentCV is set to "UNIPROTKB,TAIR".

2.4 Exercise

Open file Tutorial_files/Data_integration/Exercise/exercise.tab in a text editor.

2.4.1 Scripting Console

- Edit Data_integration/Exercise/exercise_scripting.txt to import the 6 concepts in exercise.tab in Ondex using the scripting console. As Section 2.2 showed, the scripting console maps equivalent concepts by default and collapses them. For the purpose of the following exercise, exercise_scripting.txt contains a p.setProcessingOptions() command which unables this feature. Please use "UNIPROTKB" whenever defining "defAccession".
- Once the scripting console has generated six concepts, save your graph.

2.4.2 Integrator

- Open the integrator and compose a workflow to import the graph you have just saved, map concepts based on their accessions (with no parameters for now) and export.
- Before opening your resulting graph, guess which concepts have mapped and are now linked by new relations.
- Come back to your workflow and experiment with parameters aiming to map:
 - P2 and P4
 - P3 and G2
 - P1 and G1

Do other concepts map too? Why?

Chapter 3

GUI Reference

3.1 Installation

3.1.1 Requirements

Hardware: Ondex requires at least

- 2GHz CPU
- 1GB RAM (2GB recommended)
- 300 MB of free HDD space

Operating system: Ondex is a Java 6 application and therefore compatible with

- Windows XP / Vista / 7
- Linux/UNIX/Solaris
- Mac OS 64 bit (as this version supports Java 6) however never tested

Software: Ondex needs Java version 6 update 12 or later versions. Make sure you have an adequate Java Runtime Environment installed on your system. If not, you can download it from <http://java.sun.com>.

Also make sure your PATH variable contains your java executable. To test whether this is the case, see section “Testing the requirements” below.

3.1.2 Installer

There is an Ondex installer for Windows users. In the Ondex setup, the page “Choose Components” lets the user decide which features of Ondex they wish to install. Several options are possible:

1. Ondex front-end plug-ins

2. Ondex front-end plug-ins (including experimental)
3. All Ondex front-end and Integrator plug-ins
4. All Ondex front-end and Integrator plug-ins (including experimental)
5. Custom

The first option will allow users to use data visualisation tools which are stable. The second option will allow users to use data visualisation tools which are stable as well as experimental. The third option will allow users to use data visualisation and integration tools which are stable. The fourth option will allow users to use data visualisation and integration tools which are stable as well as experimental. The fifth option will allow users to customize what their setup and manually select what they wish to install. (For this fifth option, the plug-ins selected by default depend on what option out of the first four was last selected.)

The installer will create shortcuts (which users can then access from the Start menu) unless stated otherwise during the setup. Running the exe file for Ondex (using this shortcut or by double-clicking on it) will check that Java Runtime Environment version 1.6 or higher is installed. If it is not, it will automatically download it (Java Runtime Environment version 1.6 update 17).

Users of other operating systems may download the zip file and execute the runme.sh file to start Ondex. In Linux, a “chmod 755 runme.sh” might be needed to make the script executable from your user’s account.

3.1.3 Testing the requirements

WINDOWS: Click Start -> Run enter “cmd” and press enter.

LINUX/GNOME: Right-click on your desktop and choose “Open Terminal” from the appearing context menu.

LINUX/KDE: Choose “Konsole” from your Quick launcher.

Type “java -version” in the appearing console window and press enter. If your system is configured correctly you will see a message that contains your current Java version. Make sure it is higher than version 1.6. If you don’t see a message, then you have either not installed Java yet, or you have not set your PATH variable.

3.1.4 How to set the PATH variable

WINDOWS: Right-click on “My computer” and choose “Properties” from the appearing context menu. Switch to the tab “Advanced” in the appearing window. Click the button “Environment Variables” on the bottom left of the window. You will see a list of variables with their assigned values. If the PATH variable already occurs in it, select it and click “Edit”. Append a semicolon to the end of the value field and then enter the path to your Java binary directory. This is usually something like “C:\Program Files\Java\jdk1.6.0_17\bin”. If the

Path contains any white spaces, make sure you put it in double quotes. If the variable PATH didn't exist yet on your system, create a new one and call it "PATH". Assign the binary path as it's value. However you don't need the semicolon in this case.

LINUX: Go to your home directory and open the file ".bashrc" in your favourite editor. Append the line `PATH=$PATH:<javadir>` where `<javadir>` is your path to the java binary directory, something like `/usr/java/latest/bin/` to find out what it actually is you can execute the command "type java" in a console window.

3.1.5 To the attention of sysadmins

A "system-wide install" of Ondex is discouraged because putting the Ondex directory tree in root space makes it hard for users to manage input, output, scripts and databases. Installing Ondex in users' spaces or creating an Ondex user is a good alternative.

Databases will mostly have to be obtained by the sysadmin/users as flat-files on a need-to-use basis. The expected location of those files is under `OndexDir/data/importdata/` (where `OndexDir` is the directory where Ondex was installed).

For a dataset which an average user is expecting to employ, it is essential to use a machine with enough memory. It is also essential to increase the memory setting in the command line or runme file to run Ondex with more memory. Ondex's GUI can currently manage about 250,000 concepts and relations. Limitations depend on available memory and CPU.

3.1.6 Using Ondex behind a firewall

In order to use Ondex when working behind a firewall, please modify the last line of `runme.bat` as follows:

```
java -DproxySet=true -DproxyHost=your.proxy.com
-DproxyPort=yourport -Dhttp.proxyUser=yourproxyuser
-Dhttp.proxyPassword=yourproxypassword -Xmx%MEMORY%
-Dondex.dir=%DATA% -Dovtk.dir=%OVTK_DATA% -Dplugin.scan.lib=false
-jar ovtk2.jar
```

3.2 Loading Networks

It is possible to open biological networks into Ondex by:

- Importing network files.
- Creating an empty network and manually adding concepts and relations.

3.2.1 Loading Existing Biological Networks into Ondex

Ondex can read files written in the following formats:

- oxl (Ondex XML, biological networks created using Ondex)
- nwb (NetWork Bench)
- Pajek (<http://pajek.imfm.si/doku.php?id=pajek>)
- SBML, in development
- BioPax, in development

3.2.2 Creating an empty network and manually adding concepts and relations

It is also possible to create new, empty networks and to manually add concepts and relations. To create an empty network, go to File -> New. Click on the icon for “Add a concept or relation” or go to Edit -> Concept/Relation -> Add New.

If you wish to add a concept, click on “Add a new Concept”. You will need to fill in all the fields highlighted in orange as they are compulsory. This will require creating Controlled Vocabularies, etc.

If you wish to add a relation, click on “Add a new Relation”. You will need to select concepts beforehand (shift + left mouse button to create a select area) so the drop-down lists offer these concepts as origin/target of the relation.

3.2.3 Saving Files

You can save your Ondex files and export visualisations.

- To save a file, go to File -> Save to File. The panel on the right offers options
 - Save invisible concepts/relations: if hidden items have not been deleted from the graph yet, you can untick this option to delete them before saving.
 - Save appearance: will save colours, shapes, sizes and layout of concepts and relations.
- To export images from the visualization window, go to File -> Export as
 - Save current view: will save the visualisation exactly as it is showing
 - Fit entire graph to view: will scale the graph to fit the visualisation first before saving

3.3 Metagraph

The metagraph view has 2 buttons at the bottom of its window:

- Metadata Legend: will pop up a window which displays a modifiable colour/shape legend as well as the number of concepts and relations in the network for concept classes, data sources and relation types
- Main Network: will open the main network visualization window if it was minimized

The metagraph has its own menu system:

- File
 - Export: will export the metagraph as an image (see Section 3.2.3 for more information)
- Edit
 - Show All/Selection: Set items on the metagraph to visible - All or a selection (select using left click, press shift to select several items)
 - Hide All/Selection: Set items on the metagraph to invisible - All or a selection (select using left click, press shift to select several items)
- Appearance
 - Labels: show the labels of concepts/reasons on the metagraph
 - Config: change the relation thickness, concept size and font size showing on the metagraph
 - Scale to Fit: after resizing the window, will scale the metagraph to fit it
 - Refresh Layout: will draw the metagraph again should users have moved anything and will end with a “Scale to Fit”

3.4 Manipulating Networks

There are several ways of manipulating networks in Ondex:

- Zooming Ondex provides two mechanisms for zooming:
 - Using the mouse - move the mouse scroll forwards and backwards (the graph moves away from you and towards you respectively)
 - Using the two zoom buttons on the toolbar to zoom in and out of the network (or Appearance -> Zoom)
 - Using a combination of the mouse and the zoom buttons, you can zoom in on a group of concepts - press shift to select a few concepts with the left mouse button, zoom in on the selected group

- Selecting an area to look at from the overview When studying large graphs, using the View -> Satellite View to move and navigate across the network can be very useful.
- Manually rearrange a network Select the concepts of interest and move them around. See last exercise (and help points) in Section 1.1.
- Automatically rearrange a network There are a variety of layout algorithms which can be found in the Appearance -> Layouts menu. The Gem layout is the most popular one.
- Rotating and Sheering your network Finally you can rotate and scale your network within Ondex. In the transforming mode (See Section 3.5 for icon or, in the menus, Edit -> Mouse Mode), shift + mouse will rotate and control + mouse will sheer your graph.

3.5 Buttons in toolbar

The toolbar is composed of several icons followed by a search bar which is explained in further details in Section 1.2.2. It is possible to undock this toolbar by a simple “drag and drop”. When it is undocked, it is possible to re-dock it by clicking on its far left side and again dragging and dropping to its original position.



Figure 3.1: Ondex icon bar



Figure 3.2: The “load from Webservice” icon on the left, the “open” icon in the middle and the “save” icon on the right. All available from the File menu.

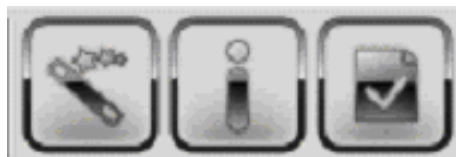


Figure 3.3: The “add a concept or relation” on the left, the “item information” icon in the middle and the “edit selected concept or relation” on the right. Available in the menus from Edit -> Concept/Relation -> Add New, View -> Item Info, Edit -> Concept/Relation -> Edit Selected.



Figure 3.4: The “delete selected concept or relation” icon on the left and the “copy whole network as new” icon on the right. Available in the menus from Edit -> Concept/Relation -> Delete Selected and Edit -> Clone Network.



Figure 3.5: The “refresh layout”, “zoom in”, “zoom out” and “center network” icons from left to right. Available in the menus from Appearance -> Refresh Layout, Appearance -> Zoom, Appearance -> Center Network.



Figure 3.6: The left-hand side icon sets the mouse to “transforming” mode (to move within the network) whereas the white hand icon sets the mouse to “picking” mode (to select concepts, drag and drop them). When a graph is not dense, the two modes interchange depending on the position of the mouse (positioned over blank or over concept or relation). When a graph is dense, manually selecting a mode can be useful. The black hand sets the mouse to “annotating” mode for annotation purposes. All available from Edit -> Mouse Mode.

3.6 Search bar in toolbar

Ondex includes a Search feature, which enables you to quickly find concepts and relations.



Figure 3.7: Search bar

If you select (single click) one item in the search results, you will need to single click on the icon “Zoom In” in order for Ondex to zoom in on this particular concept. You can then use the mouse scroll to get to the concept. If you select several items (using the Control key) in the search results, Ondex will automatically zoom in to show all those items as close as possible.

Configuring an Ondex Search

- At the end of the toolbar, there are 2 options that can be configured for each search. If you tick “Regular Expression”, you may enter a java regular expression. An example would be: `p\\d{1,4}`. It will match a “p” followed by 1 to 4 digits. For more information on regular expressions in java, please visit <http://java.sun.com/docs/books/tutorial/essential/regex/>. If you tick “Case Sensitive”, the search will be sensitive to lower/upper case of each character.
- Under the search box, there are 3 drop-down lists which allow users to restrict their search (by Concept Class, Data Source or Tag).

3.7 Menu system

3.7.1 File

The File menu contains basic file functionality.

- New: creates a new network
- Open: opens an Ondex file
- Load from Webservice: opens a graph that is available from our Webservice
- Upload to Webservice: loads a graph to our Webservice
- Save graph as: saves a file
- Save image as: saves the visualization window as an image
- Import: imports files in Network WorkBench format and Pajek
- Export: exports a graph to Prolog
- Print: prints the visualization window
- Exit: exits Ondex

3.7.2 Edit

- Undo: to undo last change (only available on right-click actions and filters)
- Undo All: to undo all changes (same restriction)
- Redo: to redo last change (only available on right-click actions and filters)
- Redo All: to redo all changes (same restriction)
- Revert Visibility to Last Save: to go back to the way the graph was the last time it was saved
- Legend: to modify colours/shapes of concepts/relations
- Labels: to change concept label font, composition (to allow for various names to be displayed next to each concept on the graph) or to change relation label font
- Delete Hidden Items: to remove invisible items from the network
- Concept/Relation -> Add New, Edit Selected, Delete Selected
- Mouse Mode -> Transforming, Picking, Annotating (see end of Section 3.5)

- Clone Network: to copy whole network as new (useful to do comparisons)
- Tabular Graph Editor: to edit concepts and relations using tables (rather than the graph)
- Settings: Settings for Web services and notifications for users (as well as login information for developers only)

3.7.3 View

- Metagraph: to display a metagraph containing all the types of concepts/relations contained in the main network
- Item Info: to display information on selected items (concepts/relations) of the network
- List: to display a list of all concepts or relations (also allows users to choose which labels to display)
- Satellite View: to open a movable overview of the network
- Windows: to view a list of all the windows opened in Ondex (also allows users to display, minimize, maximize or close all windows)

3.7.4 Appearance

- Labels: to shows the names of concepts/relations/both on the main network
- Layouts: a collection of algorithms to organise networks visually (described below)
- Colour Concepts:
 - by Data Source: colours concepts based on their data source
 - by Concept Class: colours concepts based on their concept class
 - by Evidence Type: colours concepts based on their evidence type
- Colour Relations:
 - by Relation Type: colours relations based on their relation type
 - by Evidence Type: colours relations based on their evidence type
- Shape Relations:
 - using Quad Curves
 - using Cubic Curves
 - using Bent Lines
 - using Straight Lines

- Smooth Relations: anti-aliasing of relations in the network
- Update Display: to update the visualization window with recent modifications
- Load Appearance: to load some or all items of appearance that had been previously saved
- Save Appearance: to save layout, concept colours, concept shapes, concept sizes, relation colours and relation widths
- Center Network: centers the network within the visualization window
- Refresh Layout: to get back to the latest layout that was drawn (e.g. useful after moving concepts and relations around)
- Zoom: zoom in or out

Layouts:

- Circular: for a circular arrangement of concept classes
- GEM Algorithm: for the GEM layout algorithm¹:
- More
 - Connectivity: for a mixture of the circular and hierarchical layouts
 - Flip: for a flip around of selected concepts (mirror effect)
 - Force Directed: for a force-directed algorithm based on attributes values
 - Hierarchical: for a hierarchy of concept classes
 - Kamada-Kawai: for the Kamada-Kawai layout algorithm²
 - Genomic: for a view of the data where chromosomes are represented as batons
 - Radial Tree: for a layout determined by the selection of a focus concept
 - Relation Type Specific: for a force-directed algorithm based on relation types
 - Static: for the latest saved layout
 - Sugiyama: for the Sugiyama layout algorithm³

¹ Frick A., Ludwig A. and Mehldau H. (1994). “A Fast Adaptive Layout Algorithm for Undirected Graphs (Extended Abstract and System Demonstration)”. Lecture Notes In Computer Science; Vol. 894. Proceedings of the DIMACS International Workshop on Graph Drawing, 388–403.

²Kamada T. and Kawai S. (1989). “An algorithm for drawing general undirected graphs.” Information Processing Letters, 31, 7–15.

³Sugiyama, K. and Misue, K. “Visualization of structural information: Automatic drawing of compound digraphs”, 1991.

- Tree: for a directed rooted tree
- Layout Options: for options on the parameters of some algorithms (some algorithms do not have options, others do but are not yet supported)
- Refresh layout on resize: tick for automatic relayouting of the network when the visualization window’s size is modified

3.7.5 Tools

- Integrator: user interface to allow users to create their own data integration pipeline (see Section 2.1)
- Filters: a collection of algorithms to make some concepts/relations invisible (described below)
- Annotators: a collection of algorithms to visualise the data held on concepts/relations (described below)
- Selecting Concepts/Relations: this submenu allows users to select all concepts/relations on the network, or invert their original selection
- Console: a scripting console (see Section 2.2)
- Statistics: displays statistics about the opened network

Filters:

- Neighbourhood: to see a particular number of neighbours only to a particular concept
- Tag: narrows down the graph to all concepts which were annotated with the same tag
- Unconnected: filters out all unconnected concepts
- More
 - Attribute Value Matcher: filters out concepts or relations based on a given attribute and value
 - Concept Class: filters out some particular class of concepts
 - Data Source: filters out concepts from a specified data source
 - Defluff: filters out single-linked chains that otherwise end in unconnected concepts
 - Evidence Type: filters out concepts from a specified evidence type
 - Missing Relation Type: filters out concepts which are missing a specified type of relation

- Genomic: shows only genes of the specified regions and their neighbourhood
- Relation Type: filters out some particular type of relations
- Shortest Paths
 1. All Pairs Shortest Path: computes the shortest paths between all possible pairs of concepts and removes all relations that are not part of these shortest paths
 2. Shortest Path: computes the Dijkstra shortest path algorithm from a particular concept
 3. Single Shortest Path: computes the shortest path between two given concepts
- SubGraph: filters out all concept classes and relation types that are not selected by the user in this sort of metagraph (from a concept selected by the user to start with)
- Threshold: selects a threshold for one of the concepts attributes' value

Annotators:

- Colour Concepts by General Attribute: to colour concepts based on one of their attribute's value
- Scale Concepts by General Attribute: to resize concepts based on one of their attribute's value
- Shape Concepts by General Attribute: to shape concepts based on one of their attribute's value
- Scale/Colour Concepts by Numerical Value: to resize and colour concepts based on one of their attribute's numerical value
- Scale/Colour Relations by Numerical Value: to resize and colour relations based on one of their attribute's numerical value
- More
 - BD Centrality: to resize concepts based on their degree centrality (or hub-likeness) and change their colour depending on their betweenness centrality (or how influential they are) within the network
 - Betweenness Centrality: to resize concepts based on how influential they are within the network⁴
 - Colour Relations by Tag Intersection: to colour relations green if the two concepts they link have more than one tag in common, red otherwise

⁴ Brandes U. (2001). "A faster algorithm for betweenness centrality." *Journal of Mathematical Sociology*, Vol. 25, 163-177.

- Graph Algorithms: to select one of five algorithms (all cliques, connected components, cut concepts, minimum equivalent, strongly connected). Results are added as attributes (General Data Store) to the concepts
- Scale Concepts/Relations by Type: to choose the size of concepts/relations for each concept class and relation type
- Virtual Knock-Out: to remove one concept at a time virtually and calculating the structure changes in the network
- Weighting Relation Attribute: to add an attribute to relations which corresponds to a weighted calculation of the existing attributes (only works on networks where all relations have attributes)

3.7.6 Help

- About: a brief message about the project
- Documentation: a documentation of Ondex with screenshots
- Tutorial: this tutorial is available in html format through this menu
- Welcome message: the welcome message which shows when starting Ondex

3.8 Integrator

The integrator is Ondex's pipeline manager. It is available from the Tools menu and looks like this

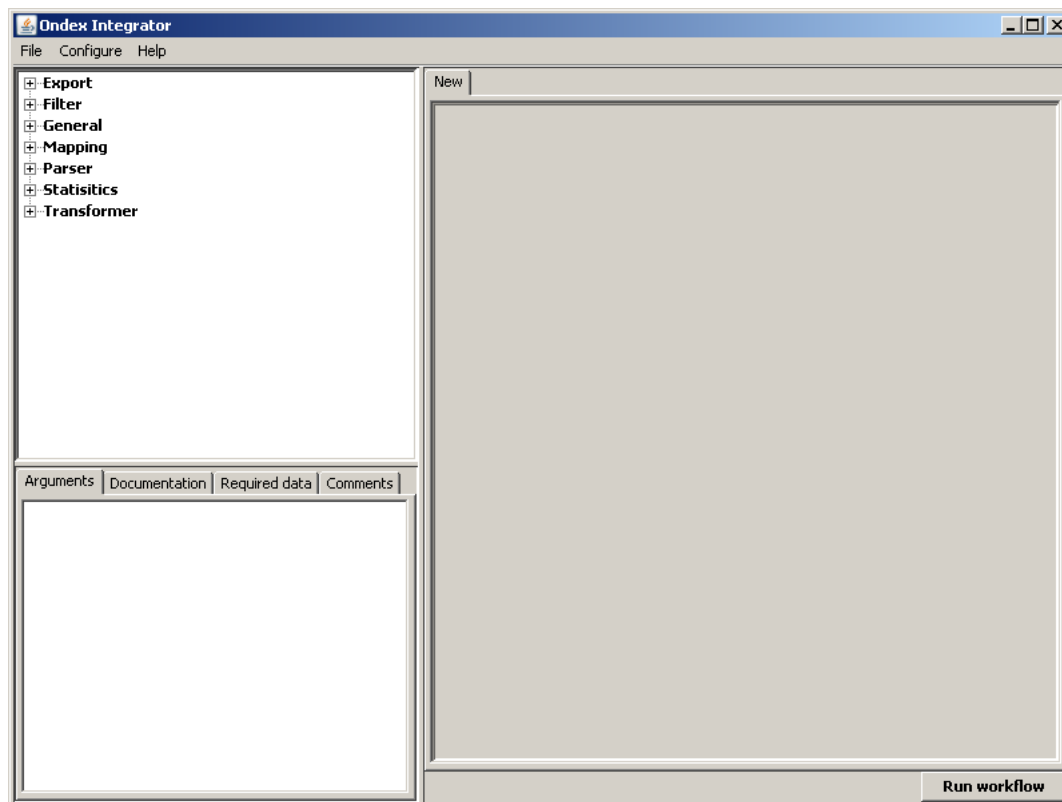


Figure 3.8: Integrator

There are 3 main areas to the integrator (which are all adjustable using the mouse):

- Top left: listing panel. This window lists all the plugins available to add to your pipeline.
- Bottom left: doc panel. This window contains 4 tabs and shows documentation for the selected item (single click) in the listing panel above.
- Right: pipeline panel. Selected plugins (double click) will appear in this window for you to set parameters.

Other useful points:

- File -> Validate allows users to check their pipeline before running it.
- Configure -> Show stable plugins only can be ticked/unticked to hide/show experimental plugins as well.
- There is a “Run workflow” button at the bottom-right corner. Saving your workflow before running is safer (File -> Save as). Running the workflow

will highlight the name of the plugins in yellow (while running), green (when finished). A error report pops up when error(s) are found. A “>>” button is placed next to each error to show users where each of them is in the pipeline. The name of the workflow becomes

- orange while it is running
- red if running has failed
- back to black if it completed successfully

Information on the plugins themselves can be found on <http://ondex.rothamsted.bbsrc.ac.uk/ondex-plugins/list>.

3.9 Console

The console is Ondex’s scripting interface. It is available from the Tools menu and looks like this

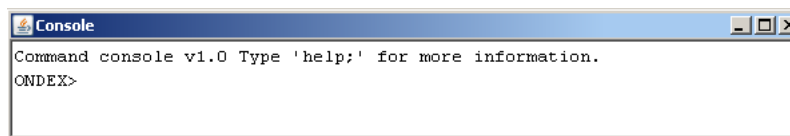


Figure 3.9: Console

Section 2.2 explains how to import tab-delimited file using it.

3.10 Statistics

The statistics module is available from the Tools menu. You see a window that is divided into 3 main areas as shown in Figure 3.10.

- Top left: listing panel. It contains 4 lists with different graph elements that can be used as variables or filters.
- Top right: selection panel. It contains the variable field (with two buttons that load it with the currently selected element or unload it, respectively). The filters list is below the variable field (same buttons to load/unload). Select from the listing panel and enter in value(s) of your choice. They will be handled as conditions linked with an “or” operation.
- Bottom: display panel. It shows the number of concepts that were found that meet the selected conditions and the respective number of relations. In case the variable is a numerical value, it also features the mean and the standard deviation values, and a histogram.

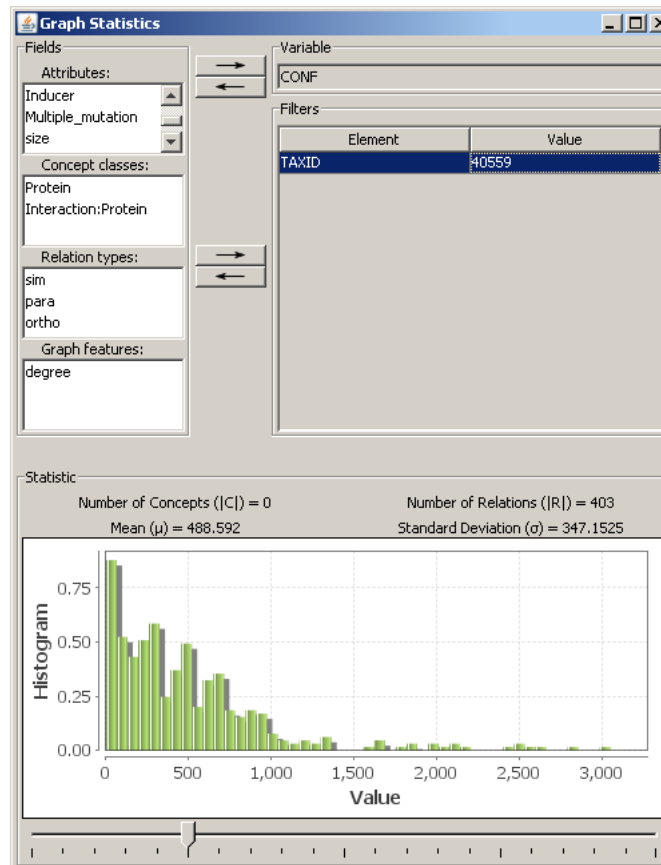


Figure 3.10: Statistics module

Chapter 4

How do I compare genomes in Ondex?

Application case: Combining comparative genomics with data integration for the prediction of potential pathogenicity genes

We integrated PHI-base (<http://www.phi-base.org/>), a database containing expertly curated molecular and biological information on genes proven to affect the outcome of pathogen-host interactions. Additionally, we loaded the genome sequence of Botrytis Cinerea (http://www.broad.mit.edu/annotation/genome/botrytis_cinerea/). See section 2.1 for more information.

Running our implementation of the InParanoid algorithm based on BLAST mappings of the genomic data gives us new biological insights. Indeed PHI-base contains a lot of annotation which can be displayed on the network using the “Annotators” menu. This type of visualization in Ondex allows us to gain new hypotheses on the Botrytis genome.

After loading the data file botrytis_phibase.xml.gz from the directory Tutorial_files/Application_cases, a metagraph is displayed. The metagraph (see Figure 4.1) shows red circles are genes from Botrytis and blue triangles are genes from PHI-base (see explanation for “Interaction:Protein” in section 2.1).

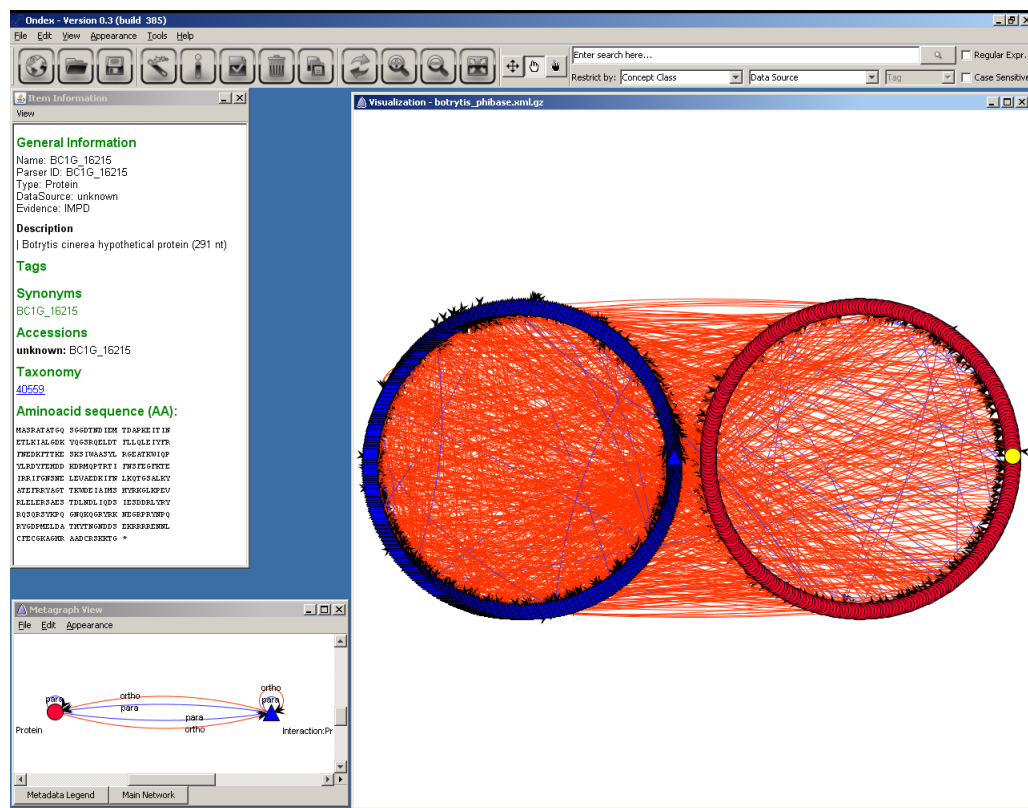


Figure 4.1: Circular layout, metagraph and item information window

They are linked by ortholog and paralog relations indicating whether genes were separated by the event of speciation (ortholog) or genetic duplication (paralog). Orthologs retain the same function in the course of evolution, whereas paralogs evolve new functions, even if these are related to the original one. Concepts imported from PHI-base contain a lot of annotation. In order to make this annotation visible, we are going to use the Annotators menu. First of all, let us use the Colour Concepts by General Attribute annotator (see Figure 4.2).

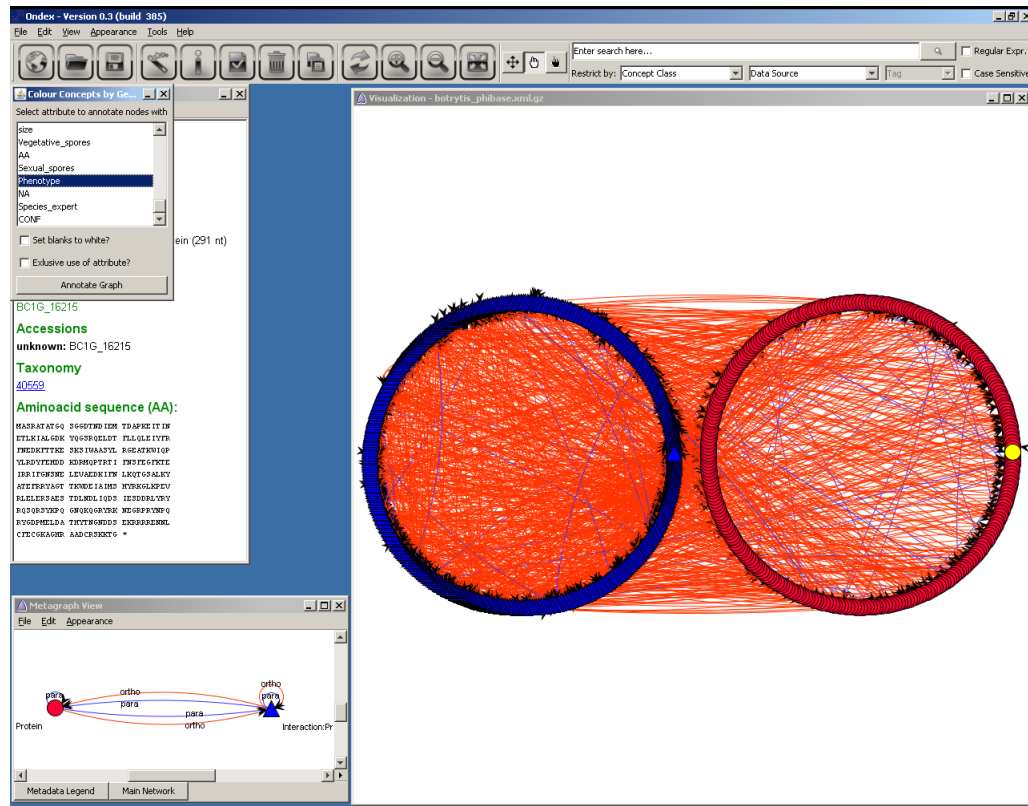


Figure 4.2: Colour Concepts by General Attribute annotator

As a lot of phenotypic information is encoded in PHI-base, we select the first attribute: “Phenotype”. Then click on “Annotate Graph”. We get a colour legend and colour annotation on the triangles (PHI-base concepts) in the graph as shown in Figure 4.3.

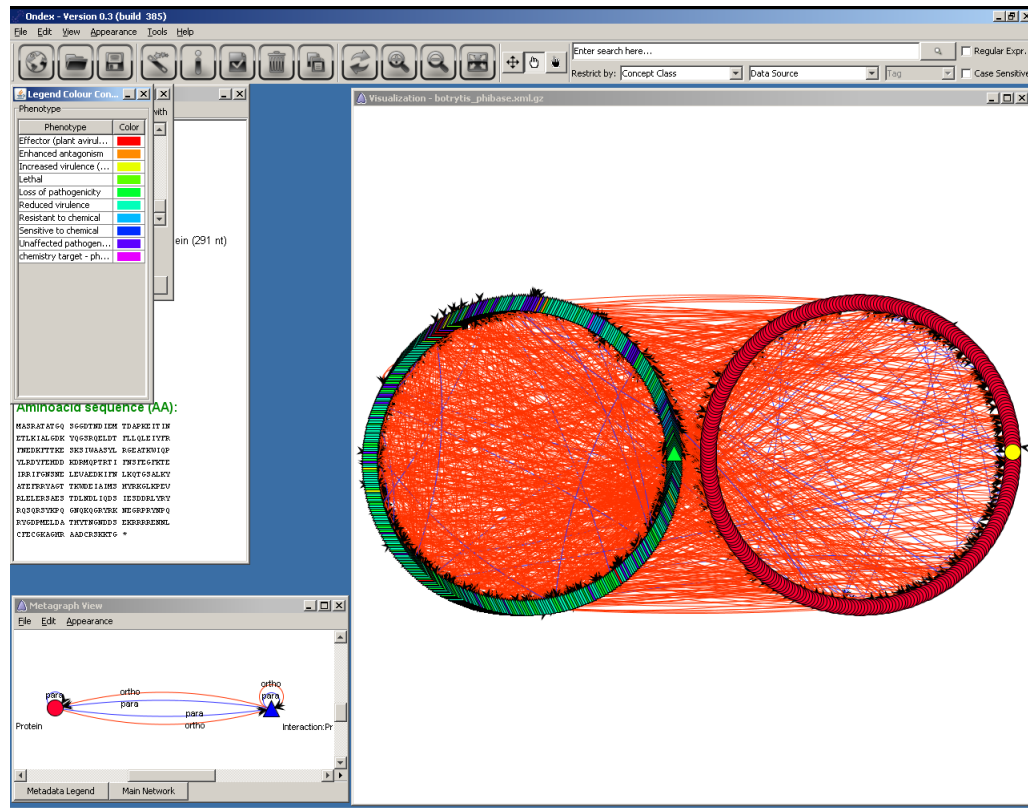


Figure 4.3: Results of the Colour Concepts by General Attribute annotator

We use Appearance –> Layouts –> Gem in order to visualize clusters (see Figure 4.4).

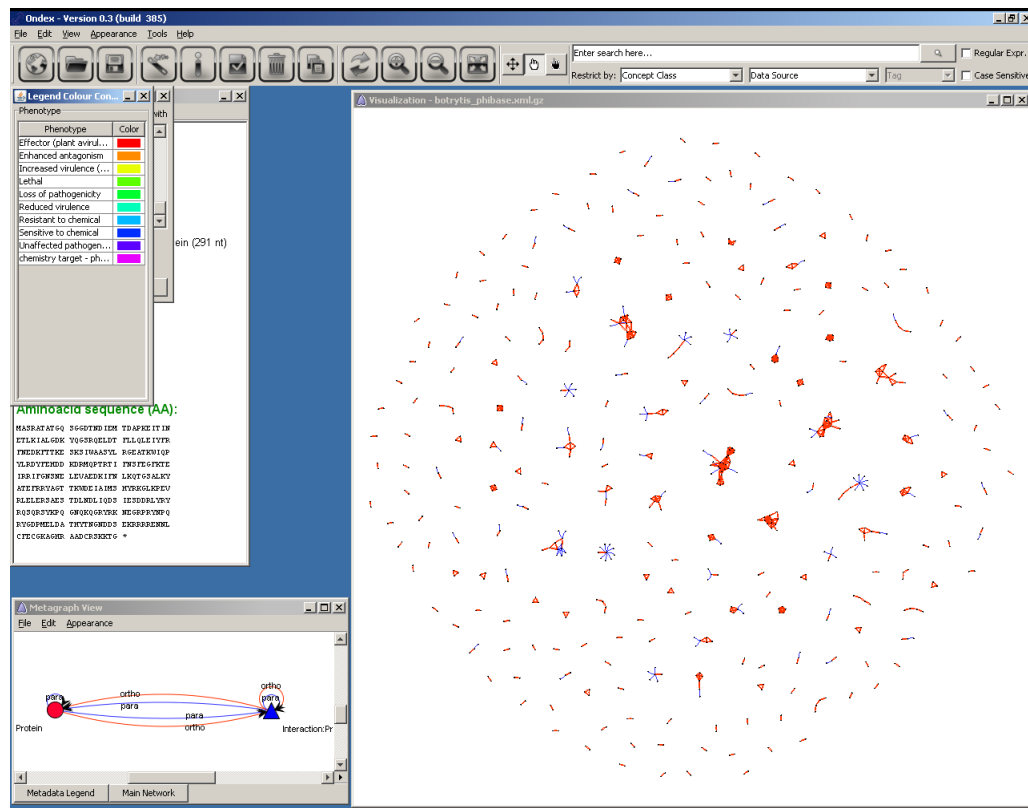


Figure 4.4: After applying the GEM layout

We may now zoom in on a particular cluster as shown in Figure 4.5.

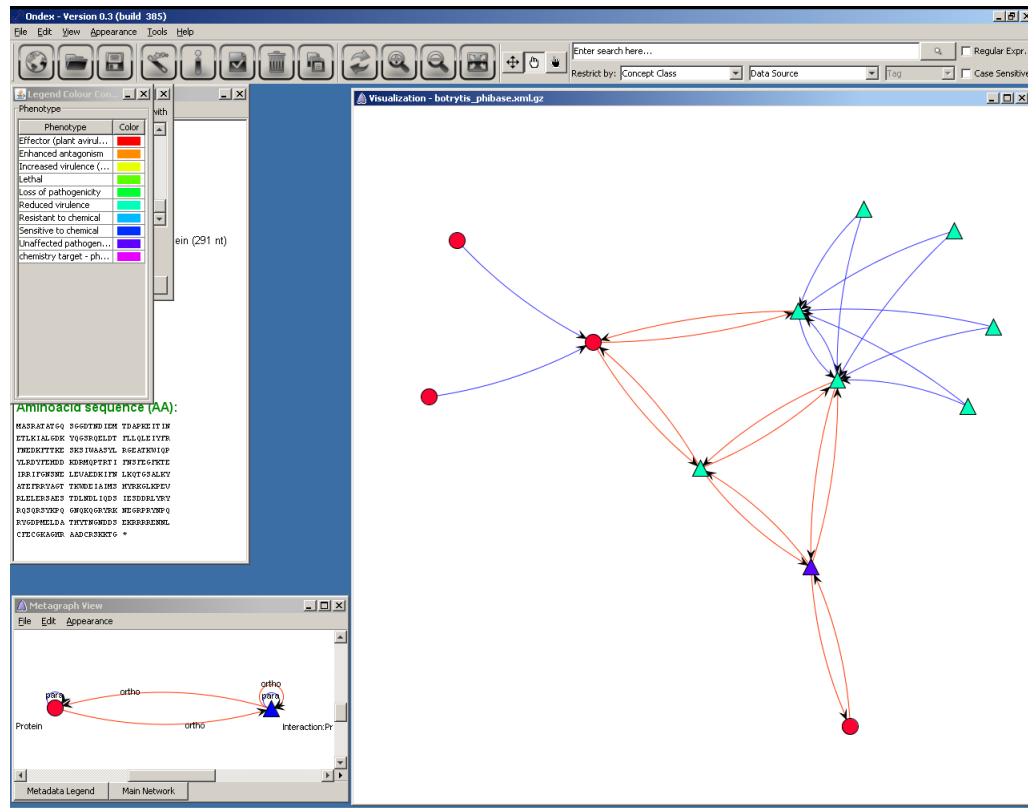


Figure 4.5: Zooming in on a cluster

Another Annotator we may use is the Shape Concepts by General Attribute Annotator as PHI-base also contains information on the pathogen species. We select the attribute “TAXID” (Taxonomy Identifier) in the list as shown in Figure 4.6.

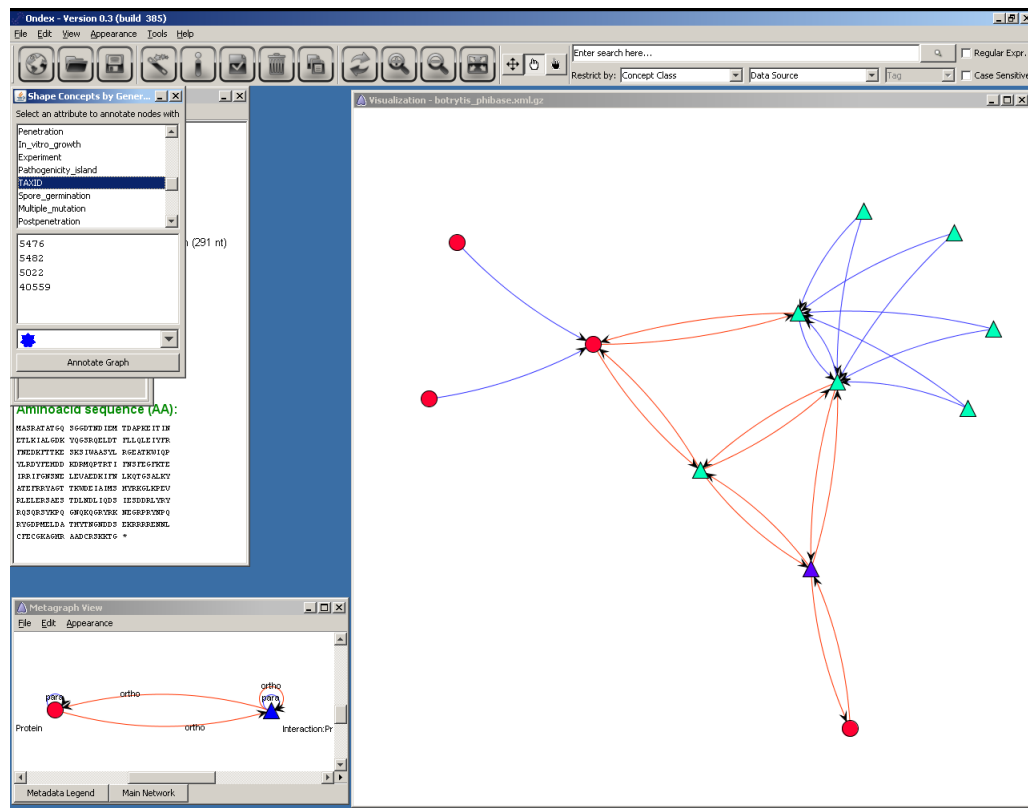


Figure 4.6: Selecting an attribute

We can look up the concepts' TAXID by editing their concept properties as shown in Figure 4.7.

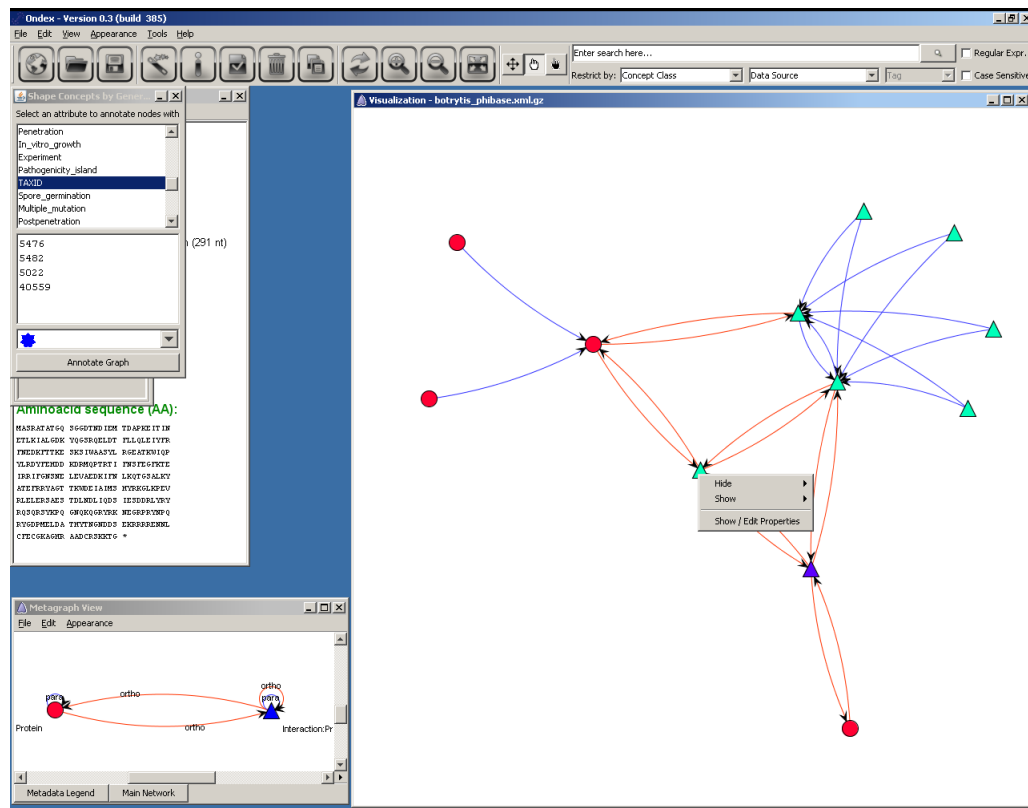


Figure 4.7: Concept properties

By clicking on “View/Edit Concept General Data Store”, we get Figure 4.8.

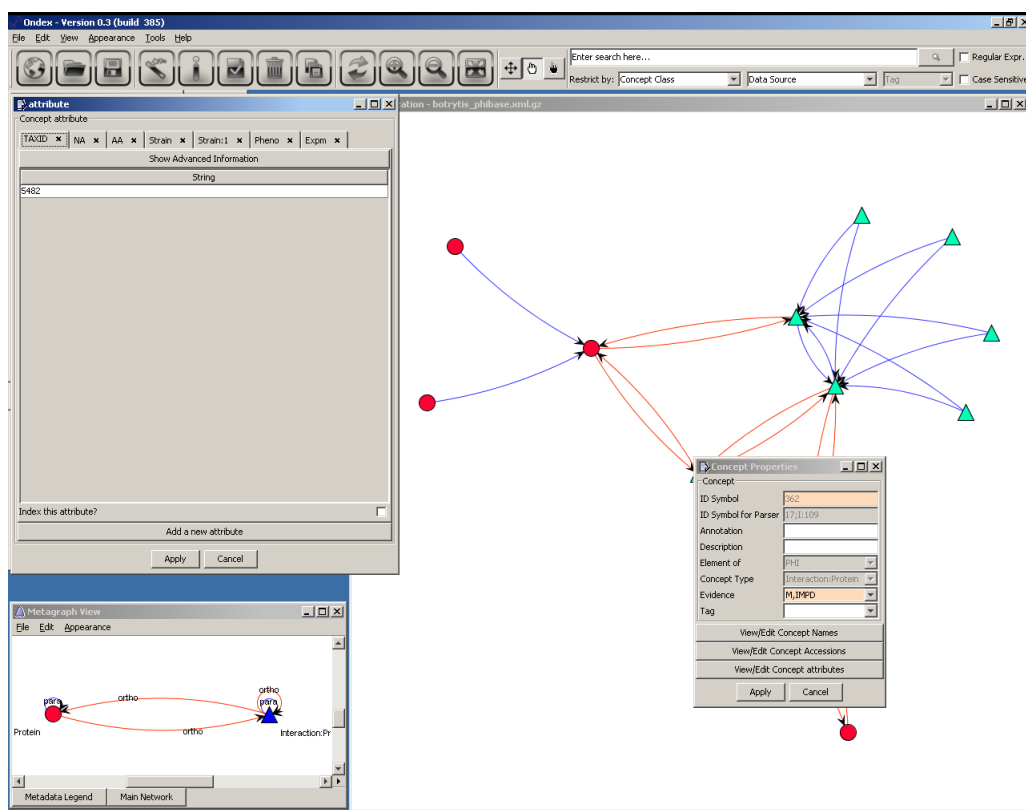


Figure 4.8: Concept attributes

The taxonomy ID is under the TAXID tab. When there are a lot of tabs, you may have to click on a right hand side arrow to get to it. Once you know what TAXID a concept holds, you may enter it in the box below and choose a shape you would like to associate to it (see Figure 4.9).

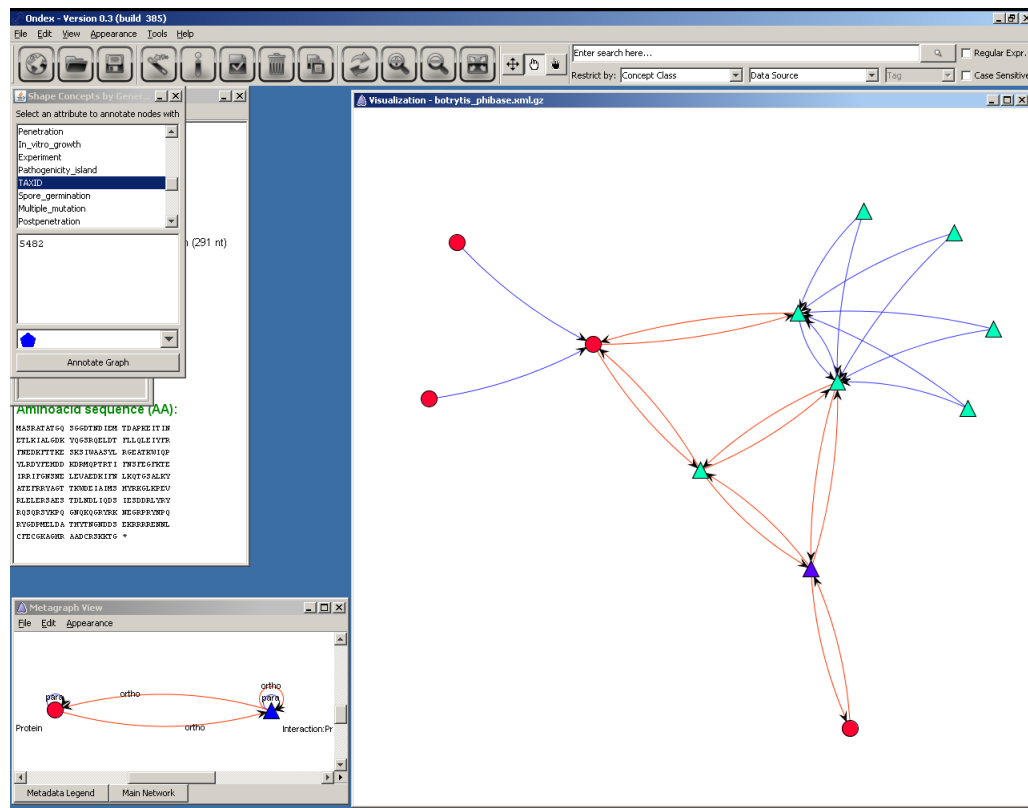


Figure 4.9: Applying the Shape Concepts by General Attribute annotator

The results are displayed once you click on the graph as shown on Figure 4.10.

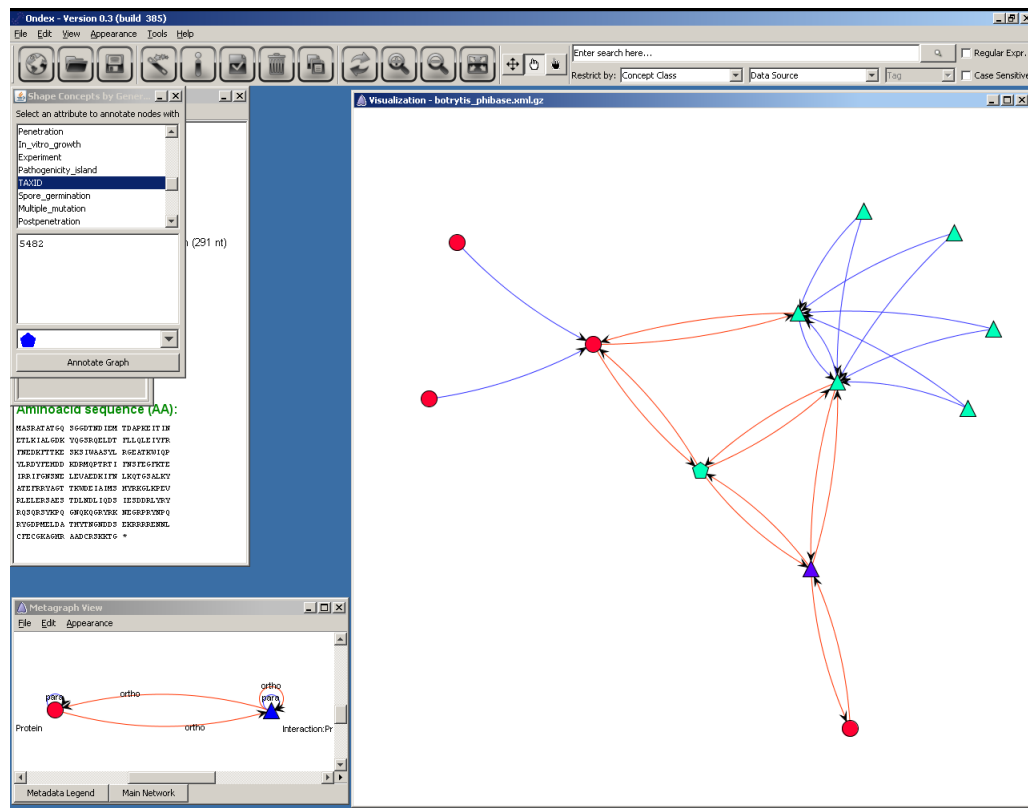


Figure 4.10: After applying the Concepts by General Attribute annotator

Let us change the shapes of all the PHI-base concepts in this cluster by using the same annotator several times (see Figures 4.11 and 4.12). (Note: Entering more than one TAXID in the blank box would make the same shape be associated to all of those TAXIDs.)

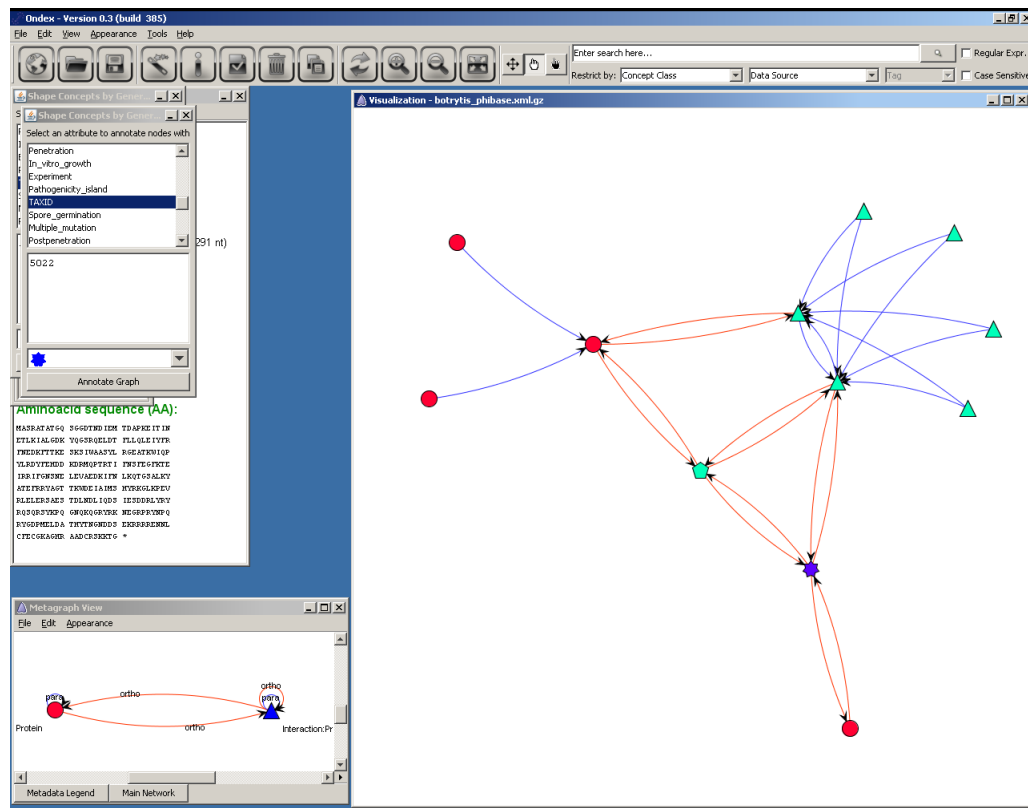


Figure 4.11: Applying the Concepts by General Attribute annotator again

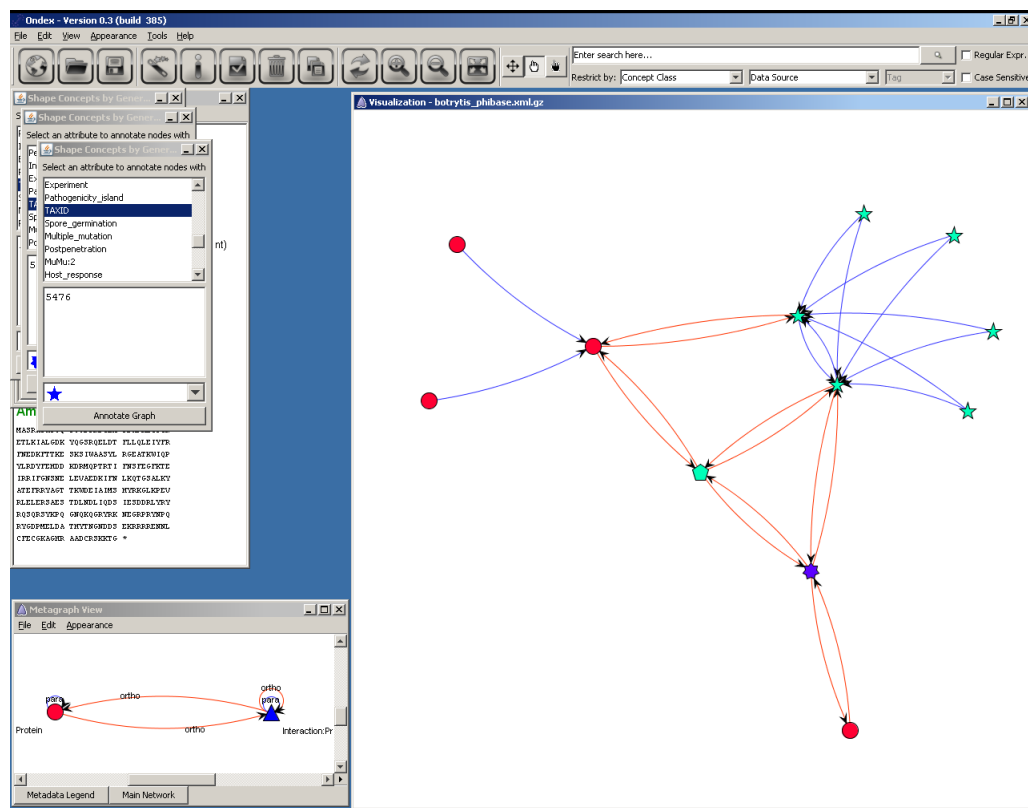


Figure 4.12: After applying the Concepts by General Attribute annotator a second time

In this example we have shown how data integration combined with visualization of annotation from a manually curated database can yield new hypotheses for a genome.

Chapter 5

How do I analyse micro-array data using Ondex?

Application case: Analysing Experimental Data in the context of Integrated Biological Networks

We integrated AraCyc (<http://www.arabidopsis.org/biocyc/index.jsp>), a database containing pathway information for the plant *Arabidopsis thaliana*, with data from the former DRASTIC-INSIGHT database for information on plant gene expression. Additionally, we loaded microarray expression data onto the concepts of the network using attributes. (A workflow named `pathways_expdata_integrator_workflow.xml` is available under `Tutorial_files/Application_cases` for this data integration pipeline). The expression data has been analysed for statistical significance and normalized. We show that an integrative approach to the exploration of microarray expression data can leverage the understanding in the context of pathways and might yield new biological insights.

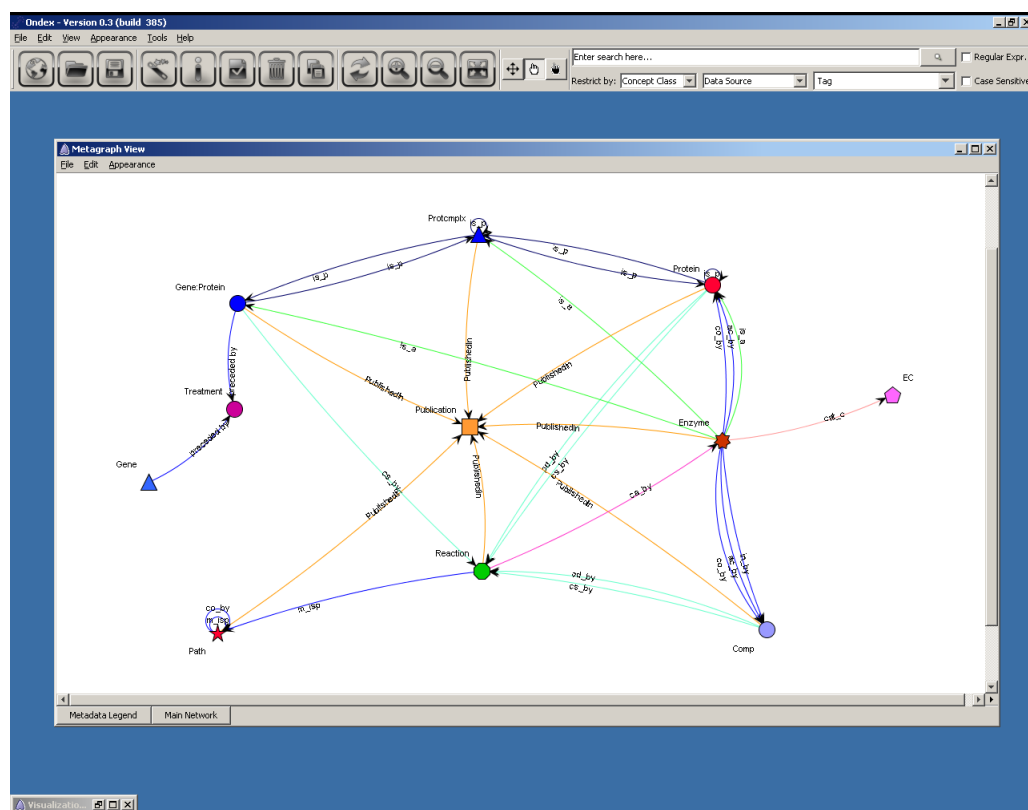
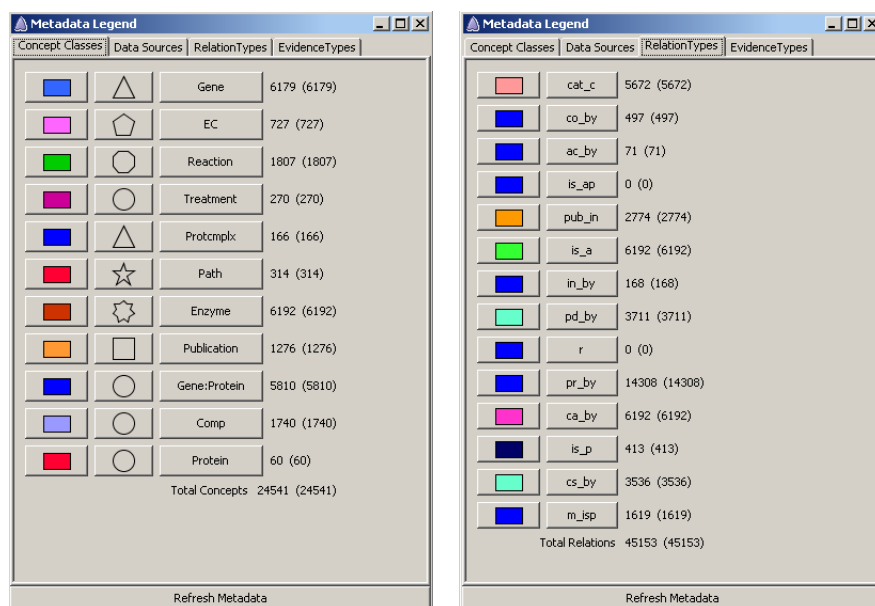


Figure 5.1: Metagraph

After loading the data file Tutorial_files/Application_cases/pathways_expdata.xml.gz, a metagraph is displayed in Figure 5.1. The different concept classes are:

- Genes from the DRASTIC database, which did not map to any AraCyc genes
- Treatments (e.g. drought, ABA) from the DRASTIC database under which a gene showed a significant change in expression
- Merged entity consisting of Genes from DRASTIC and Proteins from AraCyc together with microarray expression data to achieve a more compact pathway representation
- Protein complexes described in AraCyc
- Proteins in AraCyc, for which there is no expression data available
- Enzymes catalysing reactions from AraCyc
- Enzyme classifications for enzymes from AraCyc

- Chemical compounds involved with reactions and enzymes in AraCyc
- Reactions belonging to an AraCyc pathway
- AraCyc pathways which can be part of a hierarchy
- Publications assigned to entries in AraCyc



(a) Metadata - Concept Classes tab

(b) Metadata - Relation Types tab

Figure 5.2: Metadata - legend and numbers

The whole network contains 24541 concepts and 45153 relations as shown in Figure 5.2. Using context information we can display only the relevant sub network instead of having to work with the whole network.

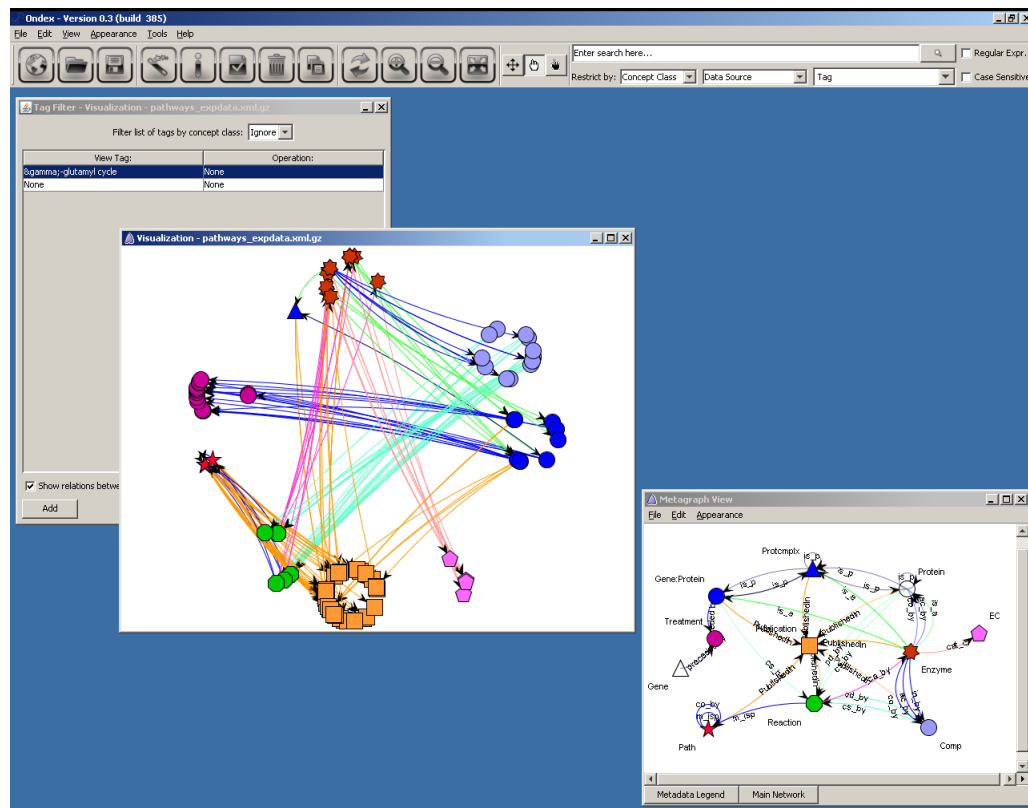


Figure 5.4: Tag filter results

The Gem layout (Appearance → Layouts → Gem) can easily be applied to this smaller network, which produces a more pleasant representation of the network (see Figure 5.5).

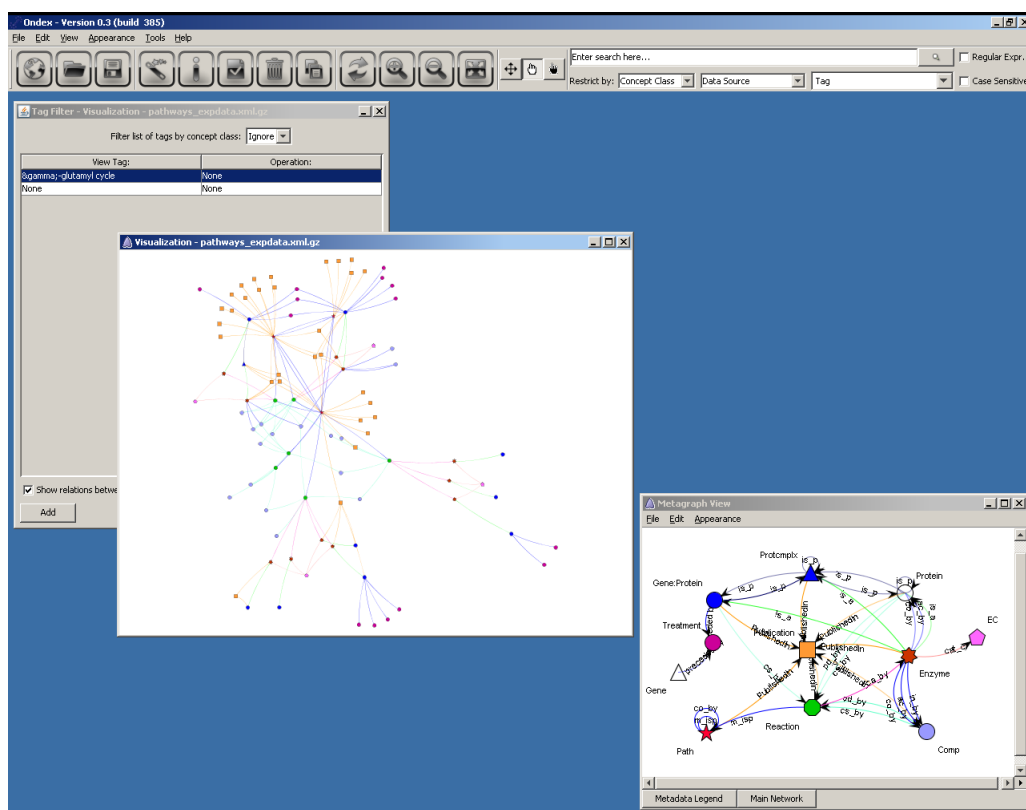


Figure 5.5: After applying the GEM layout

Now additional features like displaying concept and relation labels (Appearance —> Labels) and anti-aliased painting (Appearance —> Smooth Relations) can be turned on to enrich the current visualisation. Using the Mouse wheel you can zoom into the network view. Try to locate a group of membrane associated aminopeptidases (see Figure 5.6).

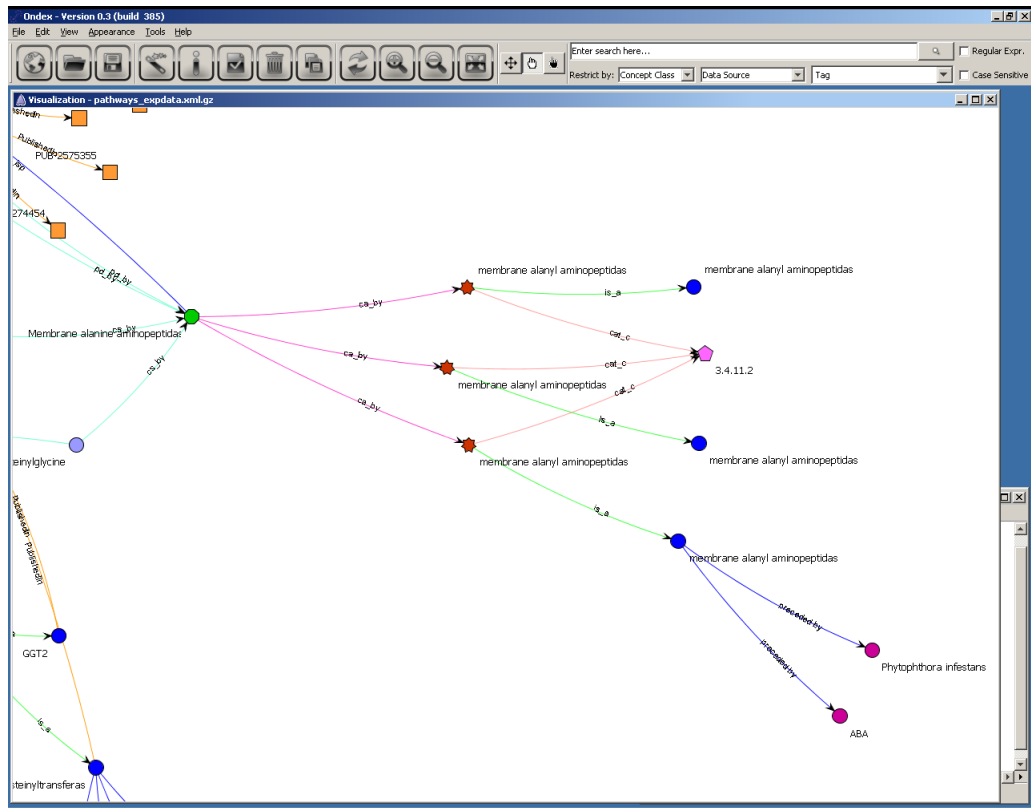


Figure 5.6: After zooming in

Here only one Gene (blue circle) has information associated from the DRAS-TIC databases (purple circle). On all of these three genes additional information can be displayed by right-clicking on a concept (see Figure 5.7).

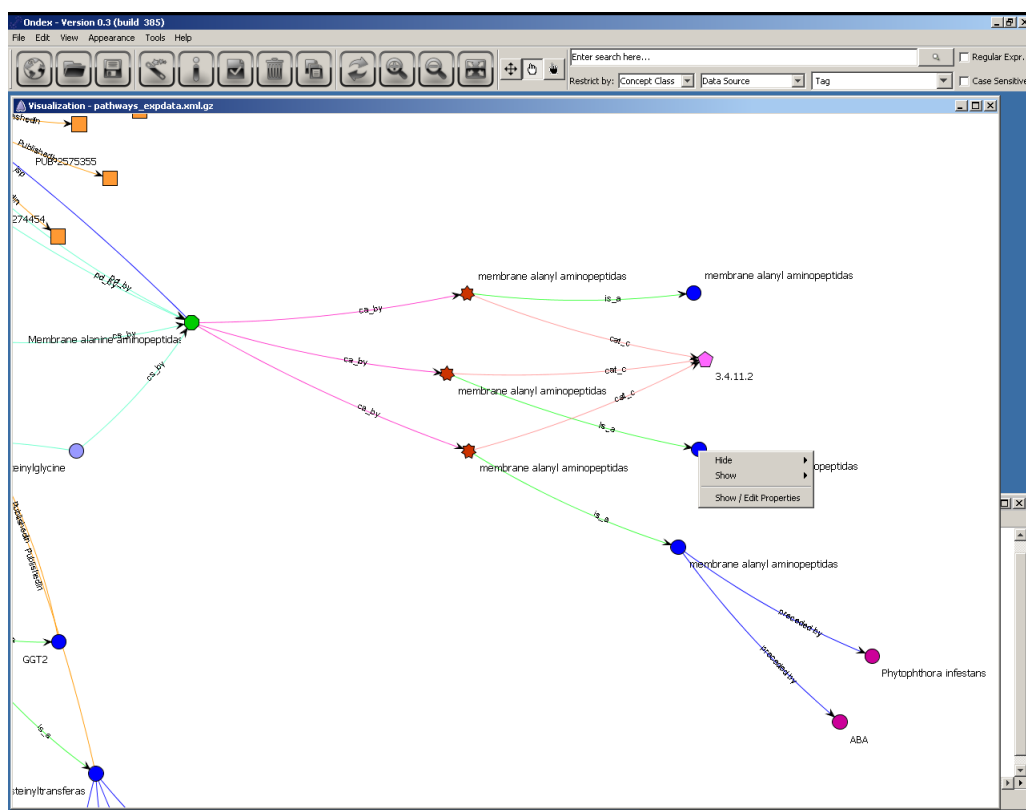


Figure 5.7: Concept properties

By clicking on Edit Concept Properties you are able to inspect all the properties assigned to this concept. Selecting View/Edit Concept Attributes displays a tab based representation of all values of the attributes (see Figure 5.8).

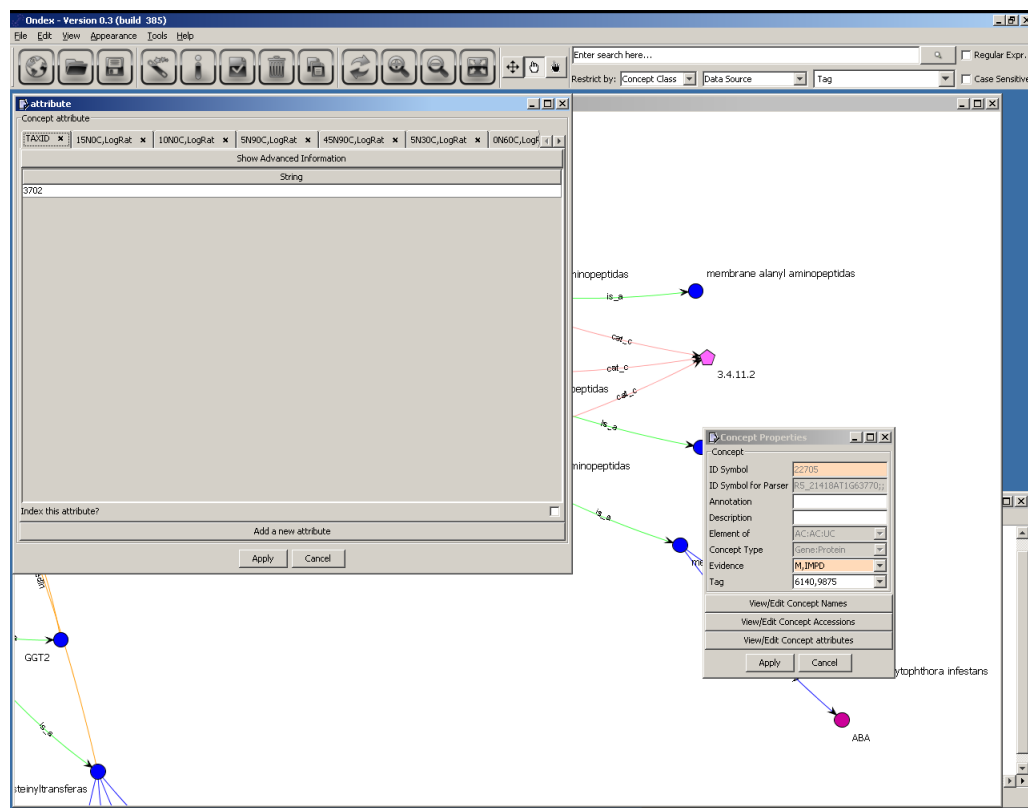


Figure 5.8: Concept attributes

In this case the values of the attributes are the microarray expression data listed according to the treatment. Now we can use the Scale Concepts by Numerical Value annotator (Tools –> Annotators –> Scale Concepts by Numerical Value) to actually map the values of the attributes to the visualisation of the corresponding concepts (see Figure 5.9).

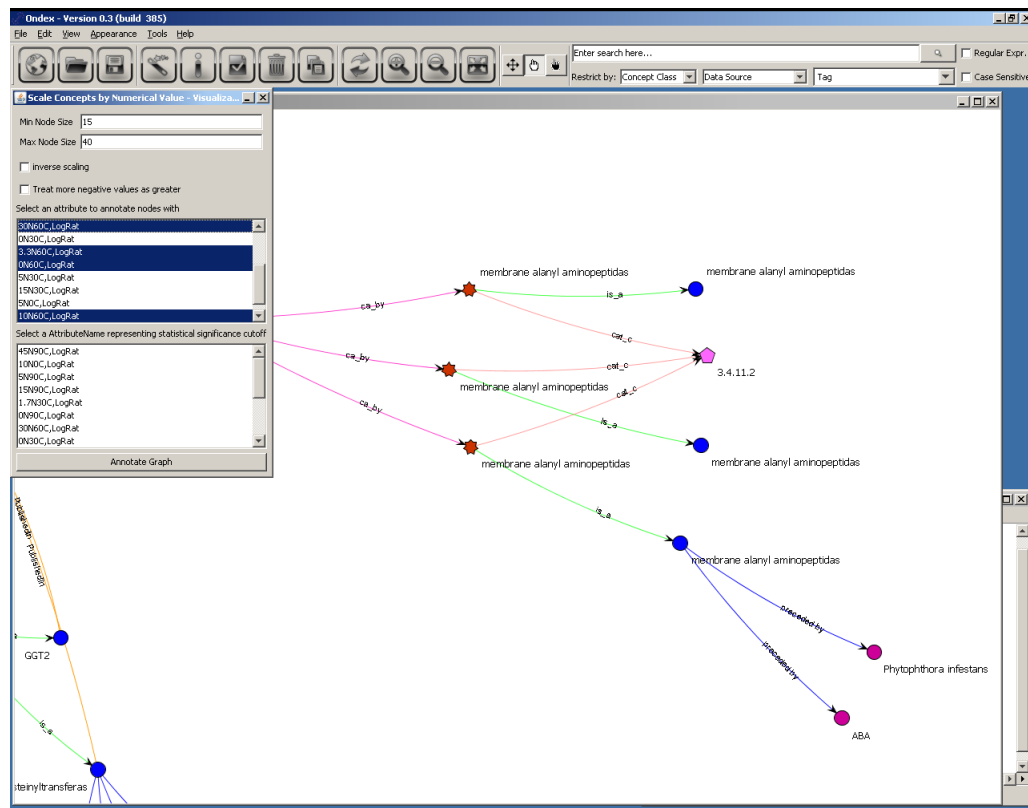


Figure 5.9: Scale Concept by Numerical Value annotator

The annotator supports multiple value selection. Here all treatments of the form xN60C are selected. Annotate Graph performs the changes to the visualisation. Figure 5.10 shows the results.

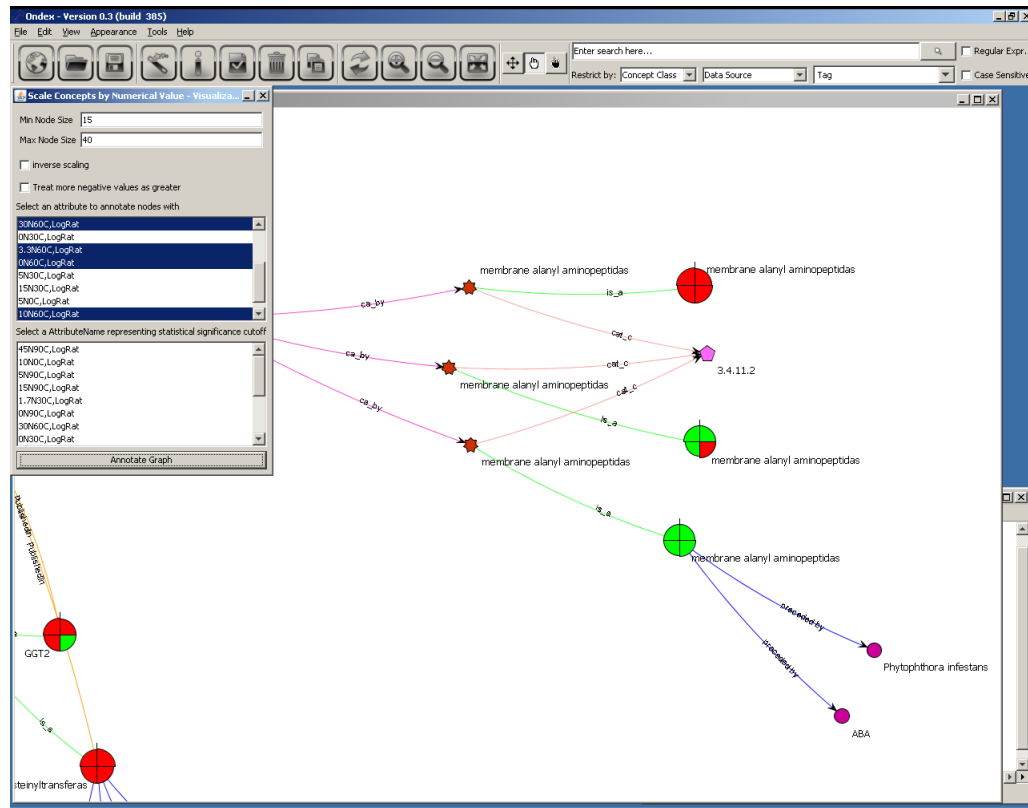


Figure 5.10: Results of the Scale Concept by Value annotator

The visualisation for the concepts now changed to pie charts. The pie chart is divided into the number of treatments that were selected in the annotator. It displays information from the top clockwise in the order in which they appear in the annotator (it is possible to drag and drop the order around in the annotator). Visual inspection of this network now reveals that the gene at the bottom shows an irregular expression pattern, whereas the other two Genes are oppositely regulated (red for up-regulation and green for down-regulation).

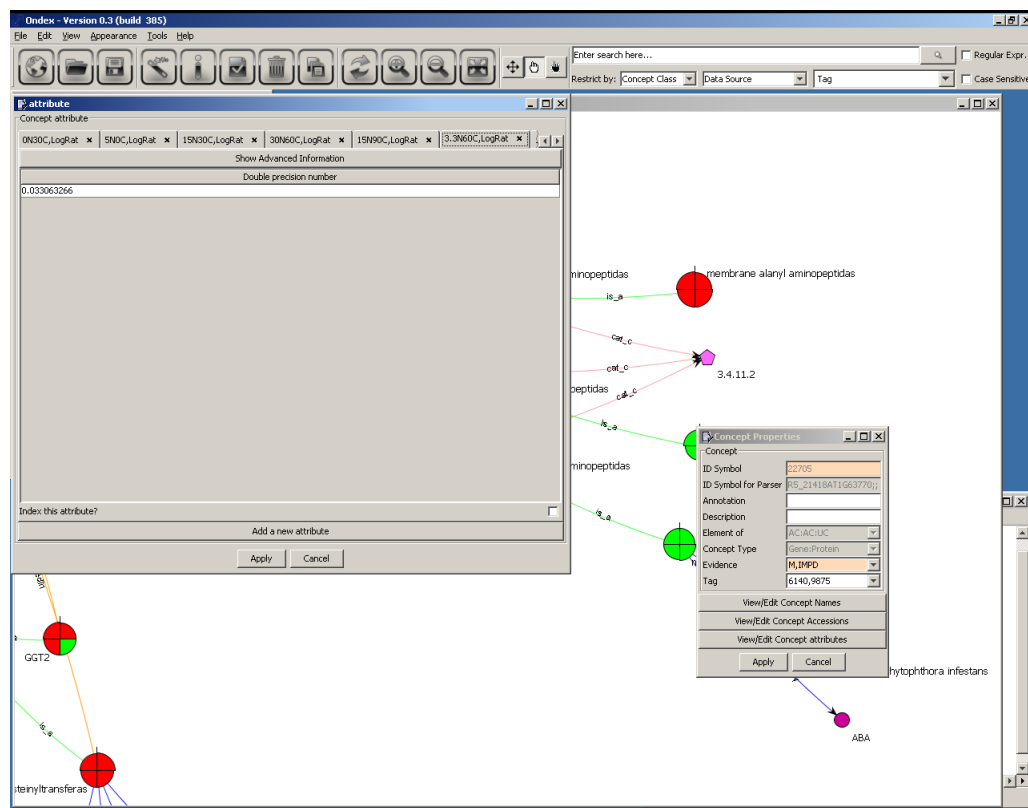


Figure 5.11: Further analysis

Further inspection (see Figure 5.11) of the attribute's value supports the hypothesis that this measurement might be wrong and the Gene is behaving like the other down-regulated Gene. In addition we can annotate one of these three Genes with information from the DRASTIC database showing that this particular Gene has a significant change in expression under the associated conditions.

It is also possible to have a second visualisation window showing the same graph with different annotations. To do so, use the 8th icon ("Copy whole network as new") or use the Edit menu -> Clone Network. A question will pop up "Do you want to link views?". If you answer no, the whole graph will load up using the circular layout. If you answer yes, the concepts/relations visible in your current visualisation window will appear in the second one (at the same positions), as shown in Figure 5.12.

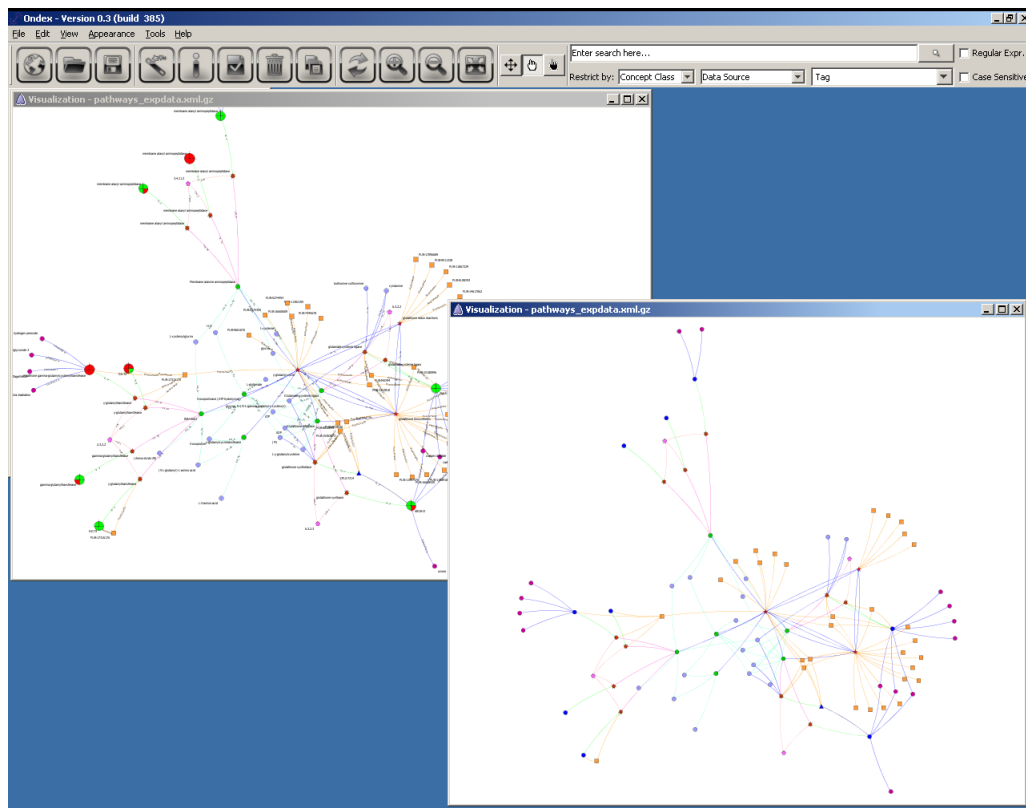


Figure 5.12: Copy whole graph

Figure 5.13 shows how moving concepts happen in both windows from then on and that different annotations can be displayed.

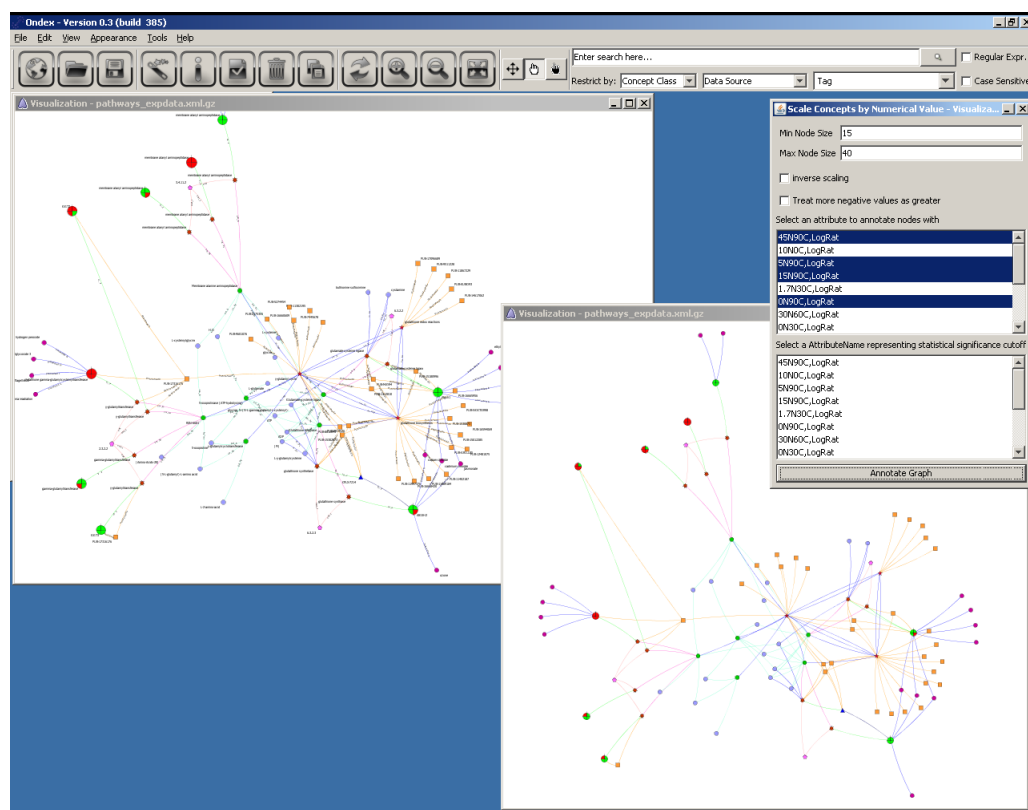


Figure 5.13: Layout is cloned in both windows

Summary

In this example, we showed how visualisation of expression patterns can leverage the understanding in terms of comparing conditions and influenced pathways. We identified one particular Gene with an irregular expression pattern. Furthermore, data integration helped to enrich pathways from AraCyc with information about influential treatments from DRASTIC.

Chapter 6

How do I look at QTLs in Oindex?

Application case: Candidate genes for biomass traits

As there is a need to understand plant architecture related to yield (*e.g.*, branching process), we wish to identify genes controlling biomass production in willow (a second generation bioenergy crop). Even well-defined QTL may encompass many potential candidate genes (perhaps hundreds), it is therefore difficult to objectively choose underlying candidate(s) that drive the phenotype. We are developing means to support systematic analysis of QTL regions and to prioritise genes for experimental analyses.

The willow genome is not yet sequenced. Poplar is the first tree with fully sequenced genome. It has 19 chromosomes, 45555 predicted genes (4 times larger than Arabidopsis's genome). Not much is known yet about the function of poplar genes.

We are using comparative genomics to compare poplar to Arabidopsis to find linked references, expression patterns, pathways, plant hormones and ontologies in order to link genes between the two. The genome annotation pipeline is lead by integrating data together from UniProtKB, TAIR, Gramene, GO, GOA, TraitOnt, PlantOnt, Medline, AraCyc and Pfam. (A workflow named `biomass_integrator_workflow.xml` is available under `Tutorial_files/Application_cases` for this data integration pipeline). The methods used for this integration are based on sequence similarity, domain analysis and text mining as shown in Figure 6.1. This allows us to achieve automatic function prediction with different levels of evidence.

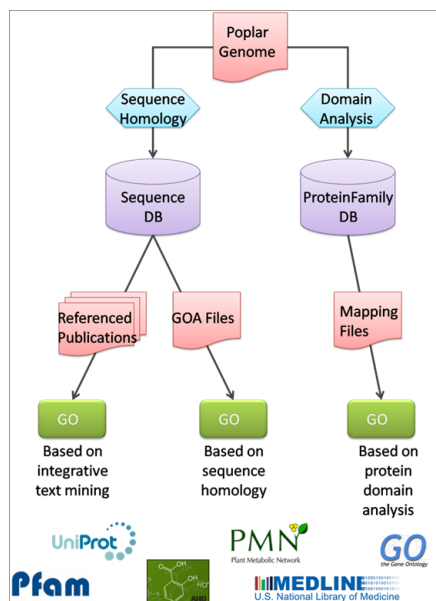


Figure 6.1: Genome annotation pipeline for poplar

This data integration in Ondex resulted in the metagraph shown in Figure 6.2. (The data is available under Tutorial_files/Application_cases, the graph is named biomass_knowledge_base.xml.gz). The left side of the metagraph shows the genes and the QTLs enriched with positional information while the right side shows the proteins annotated with GO, EC, KEGG and publications based on comparative genomics and protein family analysis.

Looking at the main network, a layout is available in Ondex which displays chromosomes, genes and QTLs where chromosomal regions and QTLs can be selected (see Figures 6.3 and 6.4).

In many domains, we can observe various phenotypes without being able to link them to genes which are responsible for them. The aim of this application case is to identify genes controlling biomass production in willow (a second generation bioenergy crop). We are developing software which can support biologists and users to analyse QTLs (genomic regions that help us narrow down potential candidate genes related to a phenotype).

We currently have two sources of data:

- QTL data is available for willow, however the willow genome is not yet sequenced
- the poplar genome sequence is available and this species is very similar to willow, however very little is known about the function of poplar genes

A QTL can contain up to hundreds of genes. We therefore look to identify candidate genes that correspond to the observed phenotype.

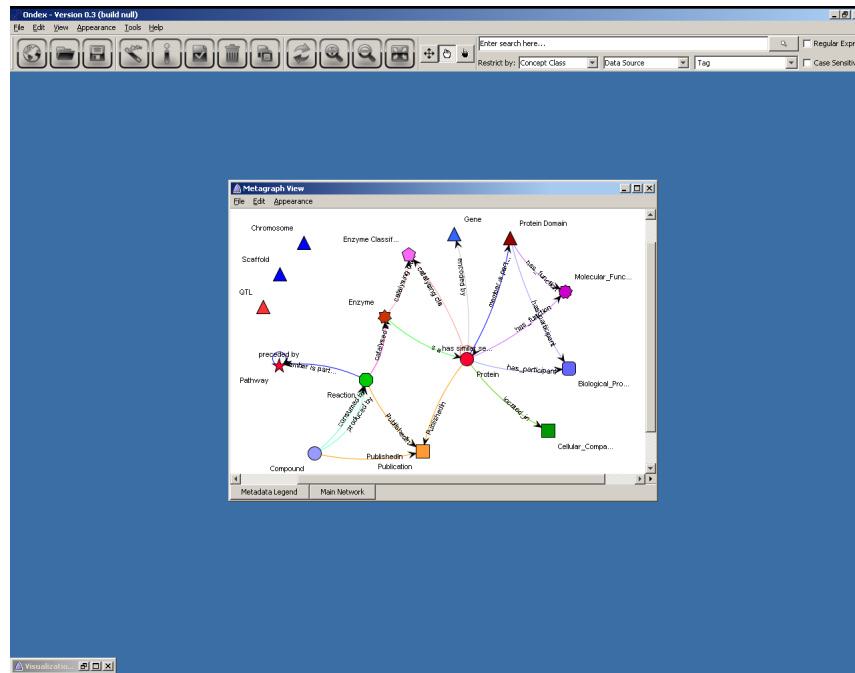


Figure 6.2: Integrated poplar network

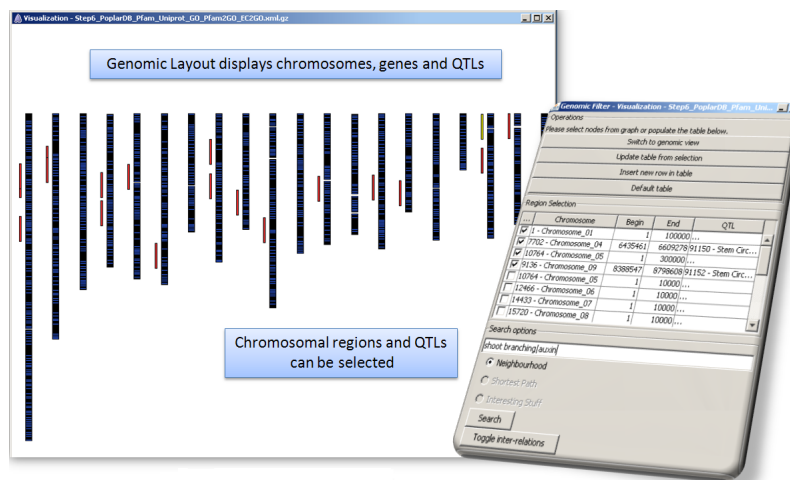


Figure 6.3: Genomic view in OncoX

In order to find similar sequences in other species that hold information on gene function, we have undertaken a comparative genomics approach. This allows us to gather data from free text (where information then needs to be

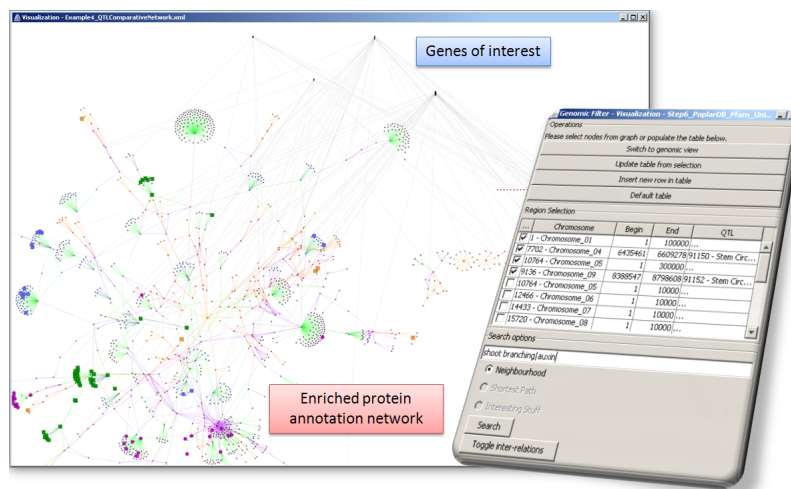


Figure 6.4: Network view in Index

automatically extracted), expression patterns, pathways, ontologies, *etc.*

The genomic view in Index allows users to visualise chromosomes, genes and QTLs (in red). The genomic filter allows users to select the regions they wish to study in more detail. Users may also enter a few keywords in a search box. Once they click on the search button, users are presented with the genes contained in the selected QTLs of interest (blue rectangles) and these are linked to their corresponding proteins which, in turn, are linked to the orthologs found by the comparative analysis. The orthologs are themselves linked to the information imported during the data integration. The keywords previously entered in the search box are then used to highlight some concepts in the graph which contain them in their information. This feature is very useful as it allows users to directly focus on proteins that are relevant to their research work. On the right-hand side of the visualisation window we can see some poplar proteins which are not linked to any extra information because no orthologs were found.

Users can zoom into particular areas of the graph, they can also select a “hot” candidate and look at its neighbourhood (using the neighbourhood filter). In Figure 6.5, three orthologs of a particular poplar protein are all connected to the same GO term (“auxin mediated signalling pathway”, auxin being one of the plant hormones that regulate growth and architecture). The right side of the Figure shows the Multiple Sequence Alignment of these four proteins.

Figure 6.6 shows that, although LAX is not linked to any specific GO term, it is linked to a publication whose title and abstract contain information on shoot branching. This figure also displays the various methods used for our comparative genomics approach (BLAST for protein sequences, HMMer for protein families), their e-values and bitscores. The width of the relation linking LAX to the yellow poplar protein (217086) is bigger as this relation holds a higher value for its bitscore attribute.

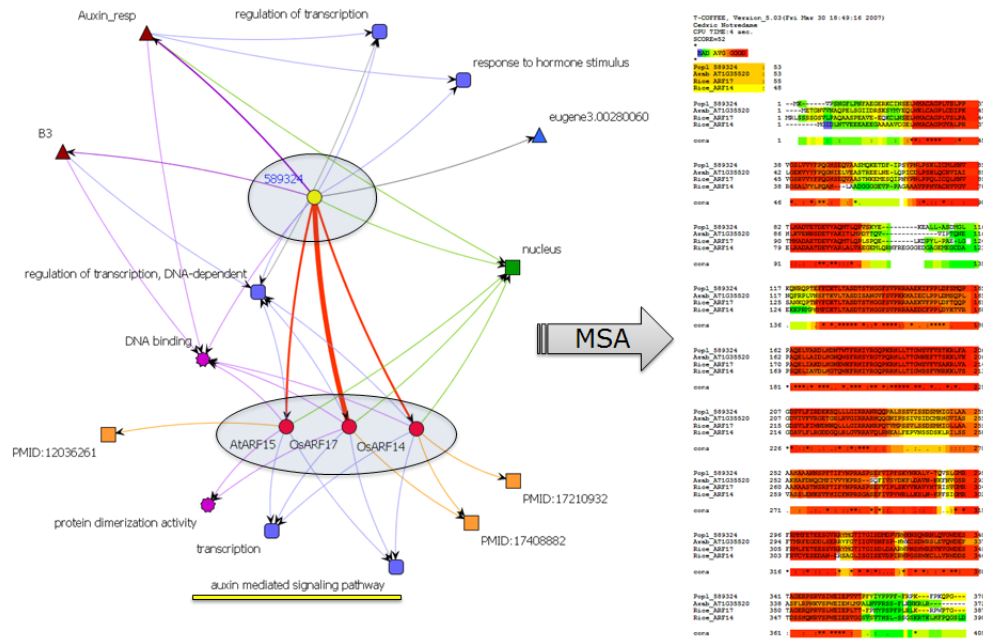


Figure 6.5: Three orthologs of a poplar protein connected to the same GO term

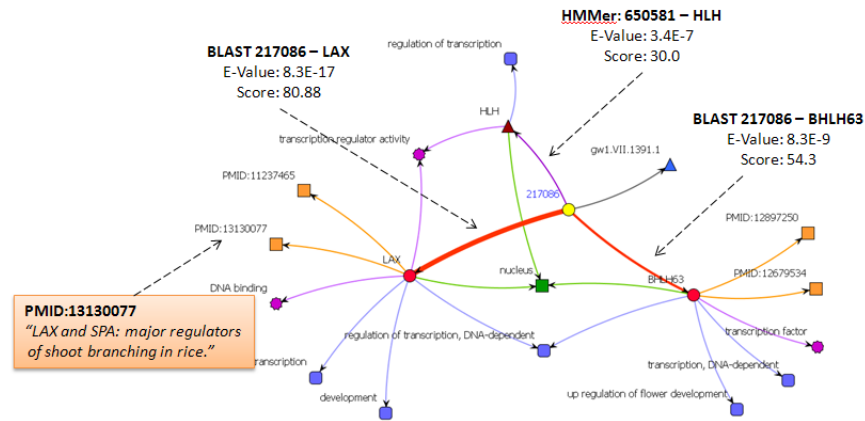


Figure 6.6: LAX linked to a relevant publication

Here are examples of combinations of steps that can be used to analyse the data:

- Load Tutorial_files/Application_cases/biomass_knowledge.base.xml.gz (this is a large file, this should take 2 minutes if you do not modify the default amount of memory given to Oindex)

- Launch the Genomics filter (Tools -> Filters -> More -> Genomics)
- Click on the first button (“Switch to Genomic View”), the visualisation window should open using the genomics layout (similar to Figure 6.3)
- In genomics filter, untick the first 4 chromosomes (so that the filtering will be relatively quick)
- On the last line of the table (only selected chromosome), select the last QTL in the drop-down list (the column begin and end will get filled in automatically)
- Enter “auxin” in search options
- Select “Neighbourhood” and click on “Search” (results should be similar to Figure 6.4)
- Search for protein 589324 (“589324” in search box and restrict to concept class “Protein” so searching is faster)
- In search results, select 589324 and use in-built neighbourhood filter to filter at depth 1
- Appearance -> Layouts -> Gem
- Appearance -> Labels -> Concepts
- Appearance -> Smooth Relations to use anti-aliased painting
- Use right-click functionalities to study neighbourhood of the 3 ortholog proteins (results on left-hand side of Figure 6.5)

Chapter 7

How do I study protein-protein interactions in Ondex?

Application case: Exploring interactions in Arabidopsis

Three protein-protein interaction (PPI) networks were merged in this application case.

- 4625 PPI from IntAct (data derived from literature curation or direct user submissions)
- 1143 PPI from TAIR (The Arabidopsis Information Resource which contains genome sequence, gene structure, gene product information, metabolism, gene expression, DNA and seed stocks, genome maps, genetic and physical markers, publications)
- 1223 PPI from BioGrid (General Repository for Interaction Datasets which contains collections of protein and genetic interactions from major model organism species derived from high-throughput studies and conventional focused studies)

These three databases were mapped using TAIR accessions after which 3 sources of evidence were added in order to facilitate the identification of functionally related groups of proteins. They were based on:

- co-expression
- sequence similarity
- co-occurrence in scientific literature

The co-expression evidence came from ATTED II (Arabidopsis thaliana trans-factor and cis-element prediction database), publicly available microarray data collected by AtGenExpress (multinational effort to uncover Arabidopsis transcriptome). The database:

- provides co-regulated gene relationships in Arabidopsis to estimate gene functions
- gives the Pearson correlation coefficients of co-expressed genes in Arabidopsis calculated from available microarray data

The data from ATTED II is mapped using TAIR accessions again.

The sequence similarity evidence was obtained by running NCBI PSI-BLAST in order to identify similarities between our reference set of proteins (and against the Arabidopsis subset of UniProt). The co-occurrence of protein names was gathered by using the integrated Lucene-based mapping method on 25,900 Medline abstracts related to Arabidopsis thaliana. At this stage, concepts in the network are connected if there is evidence from interaction, co-expression, sequence similarity or co-occurrence. Extra information was added as attributes to concepts/relations on

- Network statistics
 - Betweenness centrality (how influential a concept is) [BWC]
 - Degree centrality (hub likeness) [DC]
- Markov Clustering (identifies strongly connected groups of proteins in the network)

Load Tutorial_files -> Application_cases -> ppi_aba_cluster.xml.gz in Ondex (this is a single cluster of the whole file available as ppi_full_network.xml.gz). The Scale/Colour Relations by Numerical Value annotator offers users the possibility to use the rainbow scale (see online help by pressing F1) which means the minimum values are represented in purple while the maximum ones are in red. Figure 7.1 shows the sources of evidence associated with the relations where the strength of the relations is indicated by this colour coding.

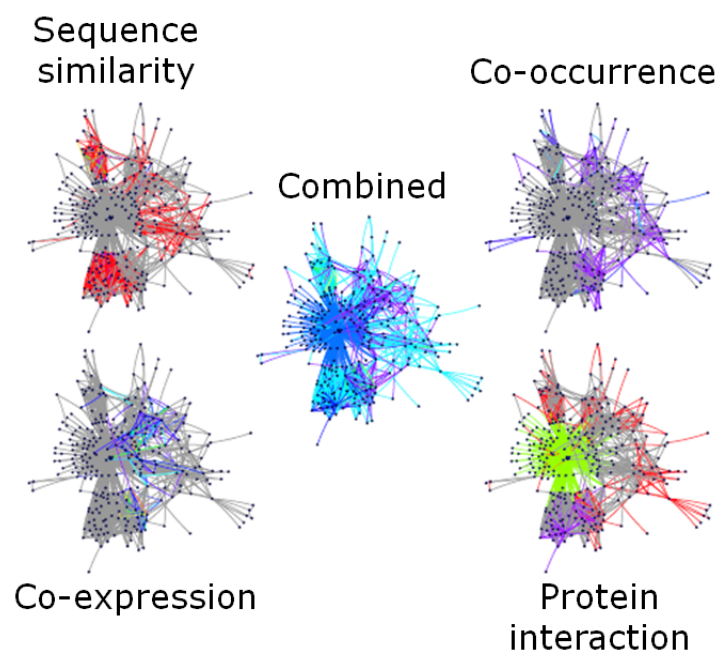


Figure 7.1: Four sources of evidence on relations (+ all of them combined) using the rainbow scale representation

Here are examples of combinations of steps, annotators and filters that can be used to analyse the data:

- Appearance -> Layouts -> Gem
- Appearance -> Smooth Relations to use anti-aliased painting (see Figure 7.2)

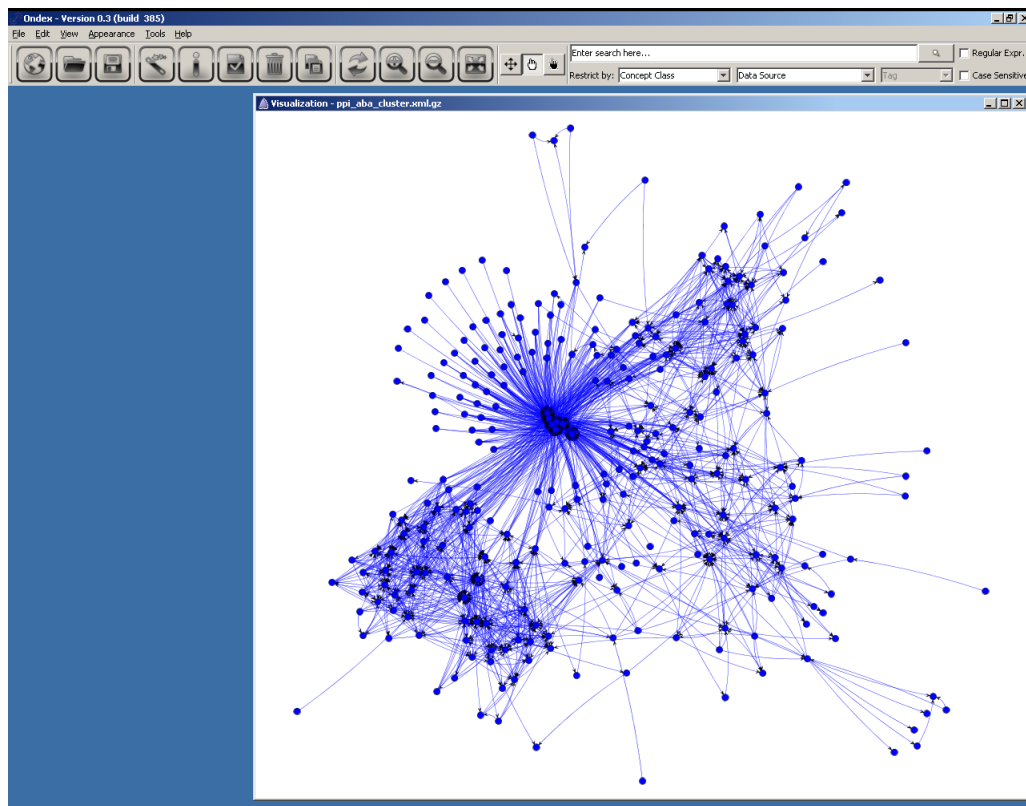


Figure 7.2: Absciscic acid cluster before annotation

- Tools -> Annotators -> Scale/Colour Relations by Numerical Value
- Enter relation size of min 4 and max 4
- Tick “colours relations”, “no attribute colour” and change colour to grey
- Tick “Colour on rainbow scale”
- Select “INTERACTION_WEIGHT” in list of attributes
- Click on “Annotate Graph” (see results in Figure 7.3)

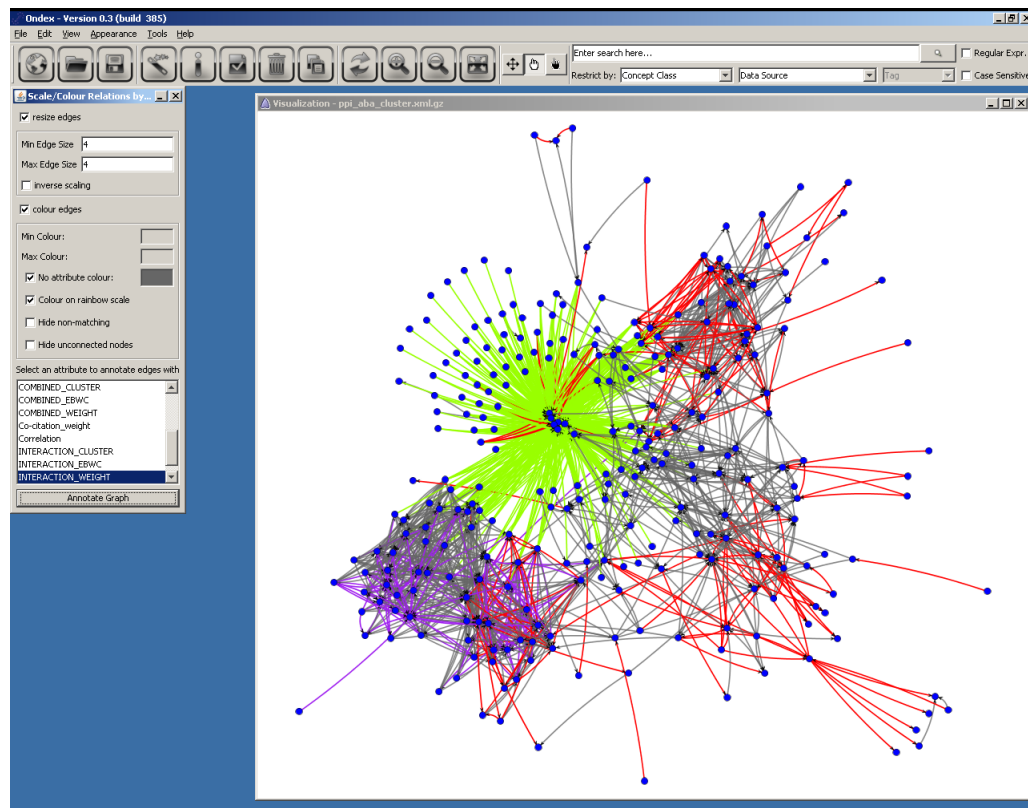


Figure 7.3: Absciscic acid cluster after annotating relations

- Tools -> Annotators -> Scale/Colour Concepts by Numerical Value
- Enter concept size of min 10 and max 100
- Tick “No attribute size” and enter 5
- Tick “Colour concepts”
- Tick “No attribute colour” and select to grey
- Tick “Colour on rainbow scale”
- Select “COMBINED_DC” and click on “Annotate Graph” (see results in Figure 7.4)

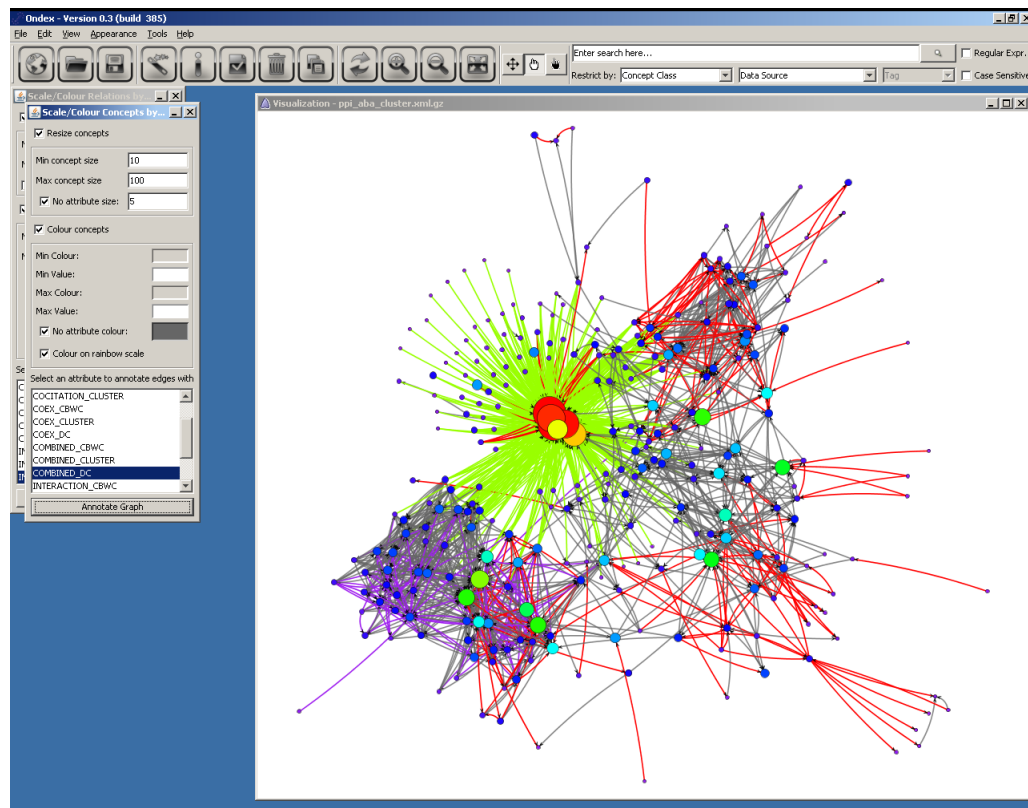


Figure 7.4: Absciscic acid cluster after annotating concepts

- Tools -> Filters -> More -> Theshold
- Go to the “Attributes on Relations” tab
- Select the “INTERACTION_EBWC” (edge betweenness centrality) as attribute, a distribution should appear below (see Figure 7.5)

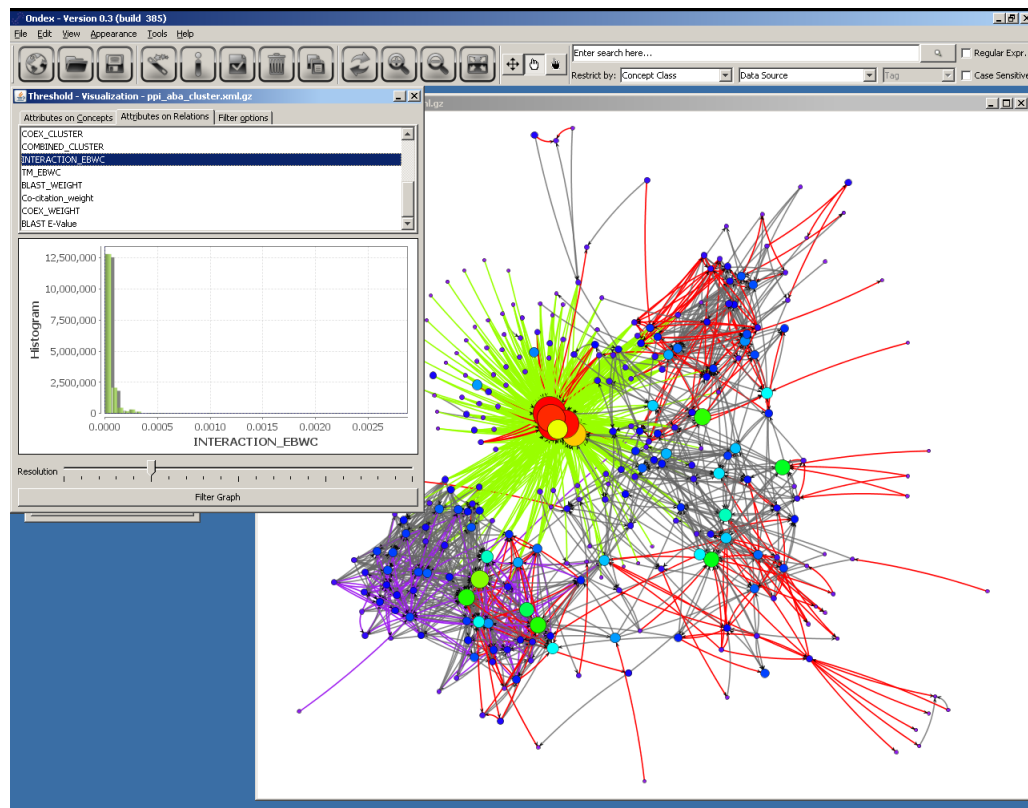


Figure 7.5: Before applying the Threshold filter

- Left-click to draw a rectangle to zoom in on the tail part of the distribution
- Click anywhere in the distribution to select a threshold
- Click on “Filter Graph” (see results in Figure 7.6, some concepts/relations should disappear)

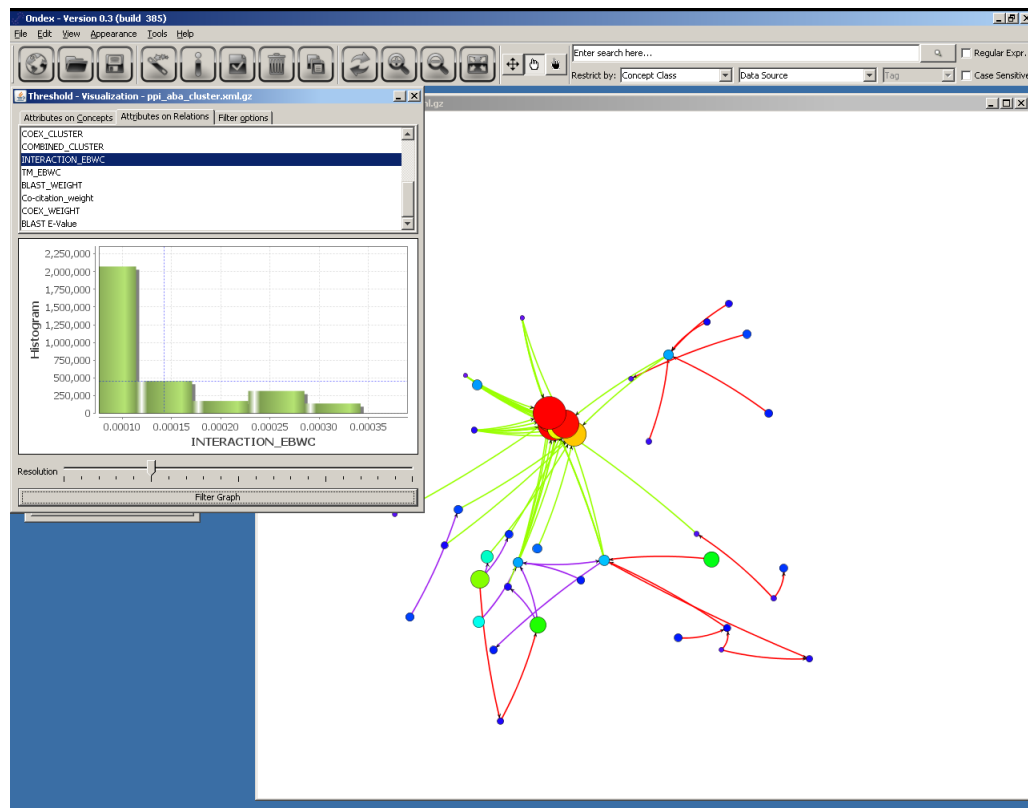


Figure 7.6: After zooming in on the distribution and applying the Threshold filter

- Appearance -> Labels -> Concepts
- Zoom in and move concepts to look at the resulting PPI network (use shift and left-click to select a group of concepts to move)

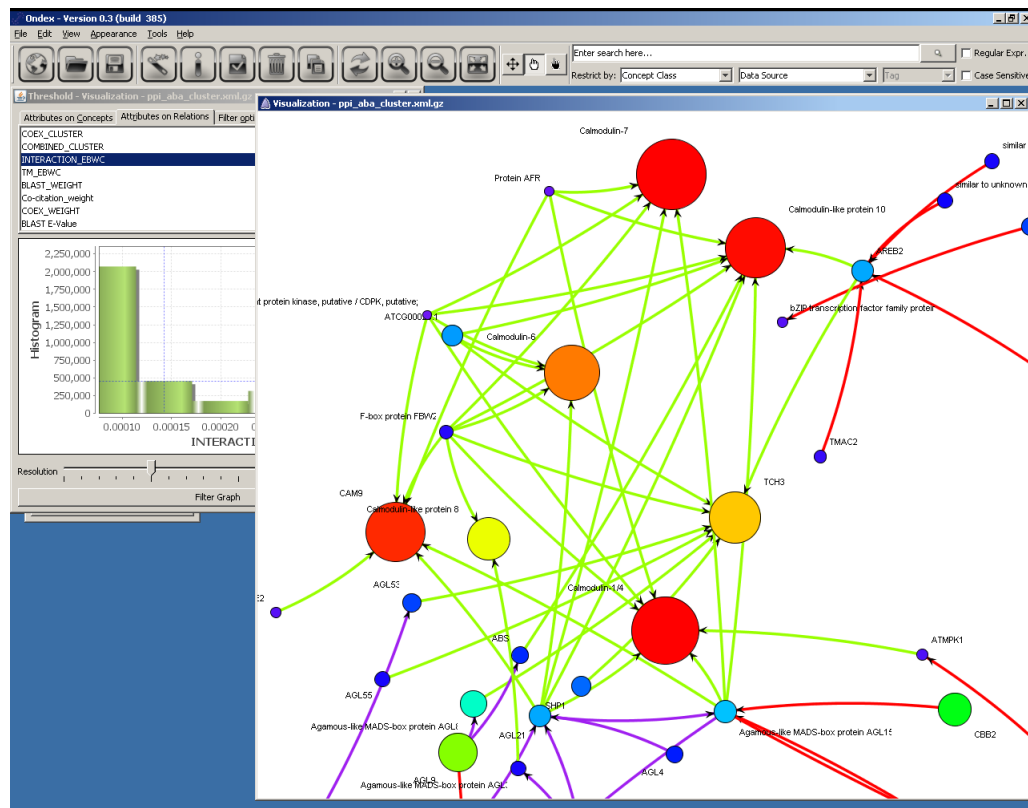


Figure 7.7: Before applying the Threshold filter

Experiment with the data and Ondex's tools to explore these Arabidopsis protein protein interactions.

Appendix A

FAQs

Check links. Add new questions (from sheet)

- How do I install Ondex? See Section 3.1.
- How do I know what an icon does?
Place your mouse over it and a tooltip will pop up. Alternatively, read Section 3.5.
- How do I know what an item of the menu does?
Place your mouse over the corresponding question mark and a tooltip will pop up. Click on the question mark to open the Ondex documentation (also available by pressing F1) which contains examples along with screenshots.
- Can I search for a specific element in the network and get Ondex to zoom on it?
Yes, see Section 1.2.2.
- Is there an import wizard for tab-delimited text?
Yes. When there is not a parser available to upload data into Ondex, this general tab-delimited parser can help. You will need to tell Ondex which column corresponds to what kind of data. The simplest way to use it at the moment is through the scripting interface offered in the “Console” (Tools -> Console). See Section 2.2 for more information.
- What is the largest graph I can load?
For the GUI to respond within reasonable time, you can load up to 100,000 elements (concepts + relations).
- I keep getting a java heap error. What can I do?
You can change the amount of memory you give Ondex when launching it. In order to do so, change the number after the Xmx option in the command line contained in the runme script (.exe for Windows, .sh for

Linux). Be careful not to set this number any higher than your computer's capabilities.

- Where can I find data files?
If you have downloaded the tutorial zip file, you can find data files in the tutorial directory's subfolders. Other data files are available on www.ondex.org/doc.html.
- Where can I submit a bug or a feature request?
<http://ondex.rothamsted.bbsrc.ac.uk> and click on "Report bugs" or "Request new features" (registration/login is required).
- How can I define new concept attributes and colour concepts based on those?
Right-click on a concept, select "Edit concept properties", "View/Edit Concept Attributes", "Add a new Attribute" (at the bottom of the window), a "New" tab is created. The "Advanced Information" is showing and the required fields (in orange) need to be filled in. At this stage, users can either: enter a name of their choice in "ID Symbol" and "java.lang.String" in Class for a text attribute ("java.lang.Integer" for a number, "java.lang.Double" for a double precision number). Click on the "Select Attribute Name" and select an attribute name that has been used in the past. Note that if you have previously entered an attribute name, you will be able to select it in this list. To colour concepts to show the added annotation, use the "Colour by Value" annotator (in the "Annotators" menu) selecting the new attribute.
- How can I avoid waiting for a big graph to load?
You can apply filters before loading the main network. This will make the graph smaller which means it will get painted quicker. See F1 help for a documentation on filters or Section 1.2 for information on various ways of filtering out data before opening the main visualisation window.

Appendix B

List of accessions

This is a list of all the accession types currently defined in Ondex.

- KNAPSACK
Full name: KNApSAcK
Description: A Comprehensive Species-Metabolite Relationship Database
- GDB
Full name: GDB
Description: Johns Hopkins University Genome Data Bank
- ENSEMBL
Full name: ENSEMBL
Description: ENSEMBL
- EC
Full name: EC
Description: Enzyme Nomenclature Committee
- UMBBD
Full name: University of Minnesota Biocatalysis/Biodegradation Database
Description: University of Minnesota Biocatalysis/Biodegradation Database
- BROAD
Full name: Broad Institute
Description: <http://www.broad.mit.edu>
- UWASH
Full name: University of Washington Multimegabase Sequencing Center
Description: Washington University School of Medicine in St. Louis <http://www.genome.washington.edu/uwgc/>

- FB
Full name: FB
Description: A Database of the Drosophila Genome
- EO
Full name: Environment Ontology
Description: A set of standardized controlled vocabularies to describe various types of treatments given to a individual plant / a population or a cultured tissue and/or cell type sample to evaluate the response on its exposure. It also includes the study types, where the terms can be used to identify the growth study facility. Each growth facility such as field study, growth chamber, green house etc is a environment on its own it may also involve instances of biotic and abiotic environments as supplemental treatments used in these studies. http://www.gramene.org/plant_ontology/ontology_browse.html
- SPENZYME
Full name: SwissPros Enzyme nomenclature database
Description: ENZYME is a repository of information relative to the nomenclature of enzymes. It is primarily based on the recommendations of the Nomenclature Committee of the International Union of Biochemistry and Molecular Biology (IUBMB) and it describes each type of characterized enzyme for which an EC (Enzyme Commission) number has been provided <http://www.expasy.ch/enzyme/>
- NC_NP
Full name: NC_NP
Description: NCBI protein sequence from REFSEQ (National Center for Biotechnology Information)
- NC_NM
Full name: NC_NM
Description: NCBI DNA/RNA sequece from REFSEQ (National Center for Biotechnology Information)
- ECOCYC
Full name: ECOCYC
Description: EcoCyc
- ATG
Full name: ATG
Description: TIGR: The Institute for Genomic Research
- PROSITE
Full name: PROSITE
Description: PROSITE consists of documentation entries describing protein domains, families and functional sites as well as associated patterns and profiles to identify them. <http://www.expasy.ch/prosite>

- GR
Full name: Gramene
Description: A Resource for Comparative Grass Genomics
- GO
Full name: GO
Description: Gene Ontology
- TAIRC
Full name: TAIRC
Description: The Arabidopsis Information Resource clone IDs
- SMART
Full name: Simple Modular Architecture Research Tool
Description: <http://smart.embl-heidelberg.de/>
- PATHODB
Full name: BIOBASE PATHODB
Description: A database on pathological forms of transcription factors and binding sites
- AC
Full name: AC
Description: AraCyc
- KAZUSA
Full name: KAZUSA DNA Research Institute
Description: Japanese Genetics Institute <http://www.kazusa.or.jp>
- GLYCODB
Full name: GlycomeDB
Description: A carbohydrate structure metadatabase <http://www.glycome-db.org/>
- HOMSTRAD
Full name: HOMologous STRucture Alignment Database
Description: HOMSTRAD (HOMologous STRucture Alignment Database) is a curated database of structure-based alignments for homologous protein families. <http://tardis.nibio.go.jp/homstrad>
- SUGERCANE_GI
Full name: SUGERCANE Gene Identifier
Description: The SUGERCANE gene identifier e.g. (PTSo00012.1)
- FBbt
Full name: FBbt
Description: Fungal Anatomy Ontology Project (http://www.yeastgenome.org/fungi/fungal_anatomy_ontology/index.html)

- RSNP
Full name: RSNP
Description: SNPs on regulatory gene regions: <http://wwwmgs.bionet.nsc.ru/mgs/systems/rsnp>
- CL
Full name: CL
Description: Cell Ontology
- TRANSCOMPEL
Full name: TRANSCOMPEL
Description: BIOBASE TRANSCOMPEL Database
- LIFEdb
Full name: LIFEdb
Description: database for the integration and dissemination of functional data
- MC
Full name: MetaCyc
Description: MetaCyc
- MESHQ
Full name: MESHQ
Description: Medical Subject Headings Qualifiers
- MGI
Full name: Mouse Genome Informatics
Description: Provided by mouse genome informatics
- GOAEBI
Full name: GOAEBI
Description: The GOA project at the EBI
- MIRBASE
Full name: miRBase
Description: <http://microrna.sanger.ac.uk>
- unknown
Full name: unknown data source
Description: concepts imported from unspecific data source
- PO
Full name: Plant Ontology
Description: Ontologie that describe plant structures and growth and developmental stages, providing a semantic framework for meaningful cross-species queries across databases. <http://www.plantontology.org/>

- NEWT
Full name: NEWT
Description: Taxonomy database maintained by the UniProt group
- CAS
Full name: CAS
Description: Chemical Abstracts Service <http://www.cas.org/>
- GENEDB
Full name: GeneDB
Description: The GeneDB project is a core part of the Sanger Institute Pathogen Sequencing Unit's (PSU) activities <http://www.genedb.org/>
- RESID
Full name: The RESID Database of Protein Modifications
Description: The RESID Database of Protein Modifications is a comprehensive collection of annotations and structures for protein modifications including amino-terminal, carboxyl-terminal and peptide chain cross-link post-translational modifications.
- TAIRGOSLIM
Full name: TAIR GOSLIM
Description: GOSLIM from the TAIR database
- CCSD
Full name: CCSD
Description: Complex Carbohydrate Structure Database
- VO
Full name: VO
Description: Virtual Ontology
- BBD
Full name: BBD
Description: Microbial Biocatalytic reactions and Biodegradation pathways database
- ProDom
Full name: ProDom
Description: ProDom is a comprehensive set of protein domain families automatically generated from the SWISS-PROT and TrEMBL sequence databases
- GOSLIM
Full name: GOSLIM
Description: GOSLIM
- CAMJE
Full name: CAMJE
Description: Campylobacter jejuni proteome

- UC
Full name: UC
Description: unclassified
- INTACT
Full name: IntAct
Description: A protein interaction data. All interactions are derived from literature curation or direct user submissions.
- TX
Full name: TX
Description: NCBI Taxonomy
- CATMA
Full name: CATMA
Description: Complete Arabidopsis Transcriptome MicroArray <http://www.catma.org/>
- TIGR
Full name: TIGR
Description: The Institute for Genomic Research
- PROID
Full name: PROID
Description: The protein identifier shared by DDBJ/EMBL-bank/GenBank nucleotide (CAA71991)
- REAC
Full name: Reactome
Description: A curated knowledgebase of human biological pathways with orthologs in other species @see www.reactome.org
- ARAB_REAC
Full name: Arabidopsis Reactome
Description: A curated knowledgebase of biological pathways (see <http://www.arabidopsisreactome.orgderivedfromintegrationofKEGGandAraCyc>)
- WN
Full name: WN
Description: WordNet
- WB
Full name: WormBase
Description: @see www.wormbase.org
- GeneRIF
Full name: Gene Reference Into Function
Description: GeneRIF provides a simple mechanism to allow scientists

to add to the functional annotation of genes described in Entrez Gene (NCBI).

- TAIR
Full name: TAIR
Description: The Arabidopsis Information Resource
- HSSP
Full name: HSSP homology-derived secondary structure of proteins
Description: The HSSP database is a database of homology-derived secondary structure of proteins. <http://swift.cmbi.kun.nl/gv/hssp/>
- PRFS
Full name: PRFS
Description: Protein Research Foundation
- BRENDA
Full name: BRENDA
Description: Brenda Database
- TF
Full name: TF
Description: BIOBASE Transfac Database
- Sorghum-JGI
Full name: Joint Genome Institute Sorghum genome
Description: <http://genome.jgi-psf.org/Sorbi1/Sorbi1.home.html>
- PFAM
Full name: PFAM
Description: Pfam is a large collection of multiple sequence alignments and hidden Markov models covering many common protein domains and families. <http://pfam.sanger.ac.uk/>
- TO
Full name: Trait Ontology
Description: Trait Ontology
- GRASSIUS
Full name: GRASSIUS
Description: GRASSIUS: a platform for comparative regulatory genomics across the grasses.
- TP
Full name: TP
Description: BIOBASE Transpath Database
- TC
Full name: TC
Description: TC

- SO
Full name: Sequence Ontology
Description: Sequence Ontology
- OMIM
Full name: OMIM
Description: Mendelian Inheritance in Man
- SMILES
Full name: Simplified Molecular Input Line Entry Specification
Description: An unambiguous chemical descriptor
- REACTION
Full name: KEGG REACTION Database
Description: It is a curated database of chemical reactions, mostly enzymatic reactions, that form the KEGG metabolic pathways.
- SA
Full name: SA
Description: Databases at the Sanger Institute
- VEGA
Full name: VEGA
Description: The Vertebrate Genome Annotation (VEGA) database
- BKL
Full name: BIOBASE KNOWLEDGE LIBRARY
Description: BIOBASE Knowledge Library
- NITE
Full name: National Institute for Technology and Evaluation
Description: <http://www.nite.go.jp/index-e.html>
- UM-P
Full name: UM-P
Description: UM-BBD_pathwayID
- SORGUM_GI
Full name: SORGUM Gene Identifier
Description: The SORGUM gene identifier e.g. (PTSb00227.1)
- UM-E
Full name: UM-E
Description: UM-BBD_enzymeID
- maizeGDB
Full name: maizeGDB
Description: MaizeGDB is the community database for biological information about the crop plant *Zea mays* ssp. *mays*. Genetic, genomic, sequence, gene product, functional characterization, literature reference,

and person/organization contact information are among the datatypes accessible through this site. (<http://www.maizegdb.org/>)

- PRINTS
Full name: PRINTS
Description: A compendium of protein fingerprints. A fingerprint is a group of conserved motifs used to characterise a protein family; its diagnostic power is refined by iterative scanning of a SWISS-PROT/TrEMBL composite. <http://www.bioinf.manchester.ac.uk/dbbrowser/PRINTS>
- ATREGNET
Full name: ATREGNET
Description: ATREGNET database
- ENCODE
Full name: Encyclopedia Of DNA Elements
Description: Functional elements in the human genome sequence
- DOI
Full name: DOI
Description: Digital Object Identifier
- Chlamydomonas_reinhardtii-JGI
Full name: Joint Genome Institute Chlamydomonas genome
Description: <http://genome.jgi-psf.org/Chlre3/Chlre3.home.html>
- MAIZE_GI
Full name: MAIZE Gene Identifier
Description: The MAIZE gene identifier e.g. (AC198603.3)
- FLYBASE
Full name: FlyBase
Description: @see flybase.bio.indiana.edu
- UNIGENE
Full name: UNIGENE
Description: An Organized View of the Transcriptome <http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=unigene>
- HGNC
Full name: HUGO Gene Nomenclature Committee
Description: HUGO symbols
- Oryzabase
Full name: Oryzabase
Description: A rice genome database hosted at <http://www.shigen.nig.ac.jp/rice/oryzabase/top/top.jsp>

- DRA
Full name: DRA
Description: Drastic Insight Database
- TRANSPRO
Full name: TRANSPRO
Description: BIOBASE TRANSPRO Database of promotor regions. TRANSPRO is a collection of human, mouse, and rat promoter sequences
- Physcomitrella_patens-JGI
Full name: Joint Genome Institute Moss genome
Description: http://genome.jgi-psf.org/Phypa1_1/Phypa1_1.home.html
- SCOP
Full name: Structural Classification of Proteins
Description: The SCOP database, created by manual inspection and abetted by a battery of automated methods, aims to provide a detailed and comprehensive description of the structural and evolutionary relationships between all proteins whose structure is known
- chemPDB
Full name: chemPDB
Description: a chemPDB accession
- FAO
Full name: FAO
Description: FAO
- DPTF
Full name: DB of Poplar TFs
Description: Database of Poplar Transcription Factors <http://dptf.cbi.pku.edu.cn>
- MENDEL
Full name: Mendel plant gene nomenclature for Commission on Plant Gene Nomenclature (CPGN)
Description: Mendel, a database of nomenclature for sequenced plant genes
- MEROPS
Full name: MEROPS
Description: The MEROPS database is an information resource for peptidases (also termed proteases, proteinases and proteolytic enzymes) and the proteins that inhibit them. <http://merops.sanger.ac.uk/>
- 3DMET
Full name: A three-dimensional-structure database of natural metabolites
Description: 3DMET is a database collecting three-dimensional structures of natural metabolites <http://www.3dmet.dna.affrc.go.jp/>

- PIR
Full name: PIR
Description: Protein Identification Resource
- UNIPARC
Full name: UNIPARCc
Description: A non-redundant archive of protein sequences extracted from public databases
- CAZY
Full name: CAZy
Description: Carbohydrate-Active enZymes
- UNIMOD
Full name: Protein Modifications for Mass Spec
Description: Protein Modifications for Mass Spec
- PSI-MIE
Full name: Proteomics Standards Initiative - Molecular Interactions the Example database
Description: Proteomics Standards Initiative - Molecular Interactions the Example database
- PSI-MIF
Full name: Proteomics Standards Initiative - Molecular Interactions Format
Description: XML file containing molecular interactions information according to the official PSI-MI schema definition 2.5
- PSI-MIO
Full name: Proteomics Standards Initiative - Molecular Interactions Ontology
Description: Proteomics Standards Initiative - Molecular Interactions Ontology
- PHI
Full name: PHI-base
Description: Pathogen - Host Interaction database
- HUGE
Full name: HUGE
Description: A Database of Human Unidentified Gene-Encoded Large Proteins Analyzed @see www.kazusa.or.jp/huge/
- F_GEN
Full name: F_GEN
Description: fungal genesymbol defined by a regular expression

- HINVDB
Full name: H-Invitational Database
Description: H-Invitational Database (H-InvDB) is an integrated database of human genes and transcripts.
- RGD
Full name: Rat Genome Database
Description: Rat Genome Database
- Ostreococcus_tauri-JGI
Full name: Ostreococcus tauri Genome Project
Description: <http://genome.jgi-psf.org/Ostta4/Ostta4.home.html>
- Cyanidioschyzon_merolae-GP
Full name: Cyanidioschyzon merolae Genome Project
Description: <http://merolae.biol.s.u-tokyo.ac.jp/>
- IPRO
Full name: InterPro
Description: A database of protein families, domains and functional sites in which identifiable features found in known proteins can be applied to unknown protein sequences. <http://www.ebi.ac.uk/interpro>
- MSDchem
Full name: Macromolecular Structure Database Group: MSDchem
Description: Consistent and enriched library of ligands, small molecules and monomers that are referred as residues and hetgroups in any PDB entry.
- DDANA
Full name: DDANA
Description: DDANAT
- JGI
Full name: Joint Genome Institute
Description: <http://www.jgi.doe.gov>
- MPATH
Full name: MPATH
Description: Mouse Pathology Ontology
- NCI
Full name: NCI Chemical Catalog
Description: <http://cactus.nci.nih.gov/>
- EMBL
Full name: EMBL
Description: EMBL-EBI International Nucleotide Sequence Data Library

- Poplar-JGI
Full name: Joint Genome Institute Poplar genome
Description: http://genome.jgi-psf.org/Poptr1_1/Poptr1_1.home.html
- PSI-MOD
Full name: The protein modification ontology (PSI-MOD)
Description: Proteomics Standards Initiative - Molecular Interactions
- KEGG
Full name: KEGG
Description: Kegg Database
- SMARTDB
Full name: The scaffold matrix transaction DataBase
Description: <http://smartdb.bioinf.med.uni-goettingen.de>
- O-GLYCBASE
Full name: O-GlycBase
Description: O-GLYCBASE is a revised database of O- and C-glycosylated proteins. <http://www.cbs.dtu.dk/databases/OGLYCBASE/>
- PDB
Full name: PDB
Description: Protein Data Bank
- AFFYMETRIX
Full name: AFFYMETRIX
Description: Microarray providers
- WIKI
Full name: Wikipedia
Description: Wikipedia, the free encyclopedia that anyone can edit <http://wikipedia.org>
- AFCS
Full name: AfCS
Description: Alliance for Cellular Signaling
- GrainGenes
Full name: GrainGenes
Description: A database for Triticeae and Avena <http://wheat.pw.usda.gov/GG2/index.shtml>
- EPD
Full name: The Eukaryotic Promoter Database
Description: <http://www.epd.isb-sib.ch>

- ICD9
Full name: ICD9
Description: ICD9
- PUBCHEM
Full name: PubChem
Description: National Lib Of Med
- BIOGRID
Full name: BioGRID
Description: General Repository for Interaction Datasets <http://thebiogrid.org>
- dbEST
Full name: dbEST
Description: dbEST (Nature Genetics 4:332-3;1993) is a division of GenBank that contains sequence data and other information on "single-pass" cDNA sequences, or Expressed Sequence Tags, from a number of organisms.
- HDIS
Full name: HDIS
Description: Human Disease Ontology
- GENB
Full name: GENBANK
Description: GenBank Nucleic Acid Sequence Database
- SGD
Full name: SGD
Description: Saccharomyces Genome Database
- NLM
Full name: NLM
Description: (US) National Lib Of Med (PubMed)
- EMBLC
Full name: EMBLC
Description: EMBL-EBI International Nucleotide Sequence Data Library clone IDs
- MIPS
Full name: MIPS
Description: Munich Information center for Protein Sequences
- LOC
Full name: LOC
Description: NCBI locus link (123456)

- NC_GE
Full name: NCBI GeneID
Description: These GeneID values are specific for the ENTREZGENE database at the National Center for Biotechnology Information (THIS IS ABSOLUTELY NOT THE SAME AS: NC_GI)
- COG
Full name: COG
Description: Clusters of Orthologous Groups of proteins
- NC_GI
Full name: NC_GI
Description: National Center for Biotechnology Information
- NC_GL
Full name: NC_GL
Description: NCGL gene identifier
- MESH
Full name: MESH
Description: Medical Subject Headings Descriptors
- GOA
Full name: GO-Annotation
Description: GO Annotation
- CHEBI
Full name: Chemical Entities of Biological Interest (ChEBI)
Description: Chemical Entities of Biological Interest (ChEBI)
- UNIPROTKB
Full name: UniProtKB
Description: UniProtKB/TrEMBL is a computer-annotated protein sequence database complementing the UniProtKB/Swiss-Prot Protein Knowledgebase.
- TRRD
Full name: Structural and functional organization of transcription regulatory regions of eukaryotic genes
Description: <http://wwwmgs.bionet.nsc.ru/mgs/gnw/trrd>
- PlnTFDB
Full name: Plant Transcription factor database
Description: PlnTFDB (2.0) is a public database arising from efforts to identify and catalogue all Plant genes involved in transcriptional control.
<http://plntfdb.bio.uni-potsdam.de/v2.0/>
- ZFIN
Full name: The Zebrafish Information Network
Description: <http://zfin.org>

- PRODOM
Full name: PRODOM
Description: ProDom is a comprehensive set of protein domain families automatically generated from the SWISS-PROT and TrEMBL sequence databases. <http://prodom.prabi.fr/prodom/current/html/home.php>
- CSD
Full name: CSD
Description: Complex Carbohydrate Research Center
- CGSW
Full name: Catalogue of Gene Symbols for Wheat
Description: A Catalogue as distributed on the MacGene CD at the 10th International Wheat Genetics Symposium. <http://wheat.pw.usda.gov/ggpages/wgc/2003/>
- IRGSP
Full name: The International Rice Genome Sequencing Project
Description: The International Rice Genome Sequencing Project (IRGSP), a consortium of publicly funded laboratories, was established in 1997 to obtain a high quality, map-based sequence of the rice genome using the cultivar Nipponbare of *Oryza sativa* ssp. japonica. It is currently comprised of ten members: Japan, the United States of America, China, Taiwan, Korea, India, Thailand, France, Brazil, and the United Kingdom. The IRGSP adopts the clone-by-clone shotgun sequencing strategy so that each sequenced clone can be associated with a specific position on the genetic map and adheres to the policy of immediate release of the sequence data to the public domain. In December 2004, the IRGSP completed the sequencing of the rice genome. The high-quality and map-based sequence of the entire genome is now available in public databases. <http://rgp.dna.affrc.go.jp/IRGSP/>
- IPI
Full name: International Protein Index
Description: International Protein Index (see <http://www.ebi.ac.uk/IPI>)
- CYORF
Full name: Cyanobacteria Gene Annotation Database
Description: Cyanobacteria Gene Annotation Database <http://cyano.genome.jp/>
- DATF
Full name: The Database of Arabidopsis Transcription Factors
Description: <http://datf.cbi.pku.edu.cn>
- CRN
Full name: CRN

Description: Database of Corynebacterial Transcription Factors and Regulatory Networks

- MINT
Full name: Molecular INTERaction database
Description: Molecular INTERaction database

Appendix C

List of relation types

This is a list of all the relation types currently defined in Oindex.

- `co_by`
Full name: `co_by`
Description: cofactored by
- `pr_by`
Full name: preceded by
Description: A is the direct outcome of B
- `is_a`
Full name: is a
Description: For continuants: C is_a C' if and only if: given any c that instantiates C at a time t, c instantiates C' at t. For processes: P is_a P' if and only if: that given any p that instantiates P, then p instantiates P'.
- `is_p`
Full name: is part of
Description: OBO relation type
- `is_ap`
Full name: `is_ap`
Description: is a part of
- `rp_by`
Full name: `rp_by`
Description: replaced by
- `ca_by`
Full name: catalysed by
Description: catalysed by
- `in_by`
Full name: `in_by`
Description: inhibition

- pr_by
Full name: preceded by
Description: A is the direct outcome of B
- reg
Full name: reg
Description: factor regulates expression
- pr_by
Full name: preceded by
Description: A is the direct outcome of B
- ex_by
Full name: ex_by
Description: expressed by
- c_mod
Full name: c_mod
Description: none
- not_located_in
Full name: not_located_in
Description: A concept that has been shown not to be located in the specified component.
- en_by
Full name: encoded by
Description: a protein is encoded by a gene
- pub_in
Full name: PublishedIn
Description: The place where this concept was published
- m_isp
Full name: member is part of
Description: member is part of (example: reactions is part of a pathway and it is a member of that pathway)
- is_p
Full name: is part of
Description: OBO relation type
- direct_correlation_with
Full name: direct_correlation_with
Description: Entities that are directly correlated (+ve coefficient)
- correlated_with
Full name: correlated_with
Description: Entities with correlation

- `has_chem_f_p`
 Full name: has chemical functional parent
 Description: Used to denote the relationship between two molecular entities (or classes of entities), one of which possesses one or more characteristic groups from which the other can be derived by functional modification. This relationship is especially useful to demonstrate the relationships between a number of functionalised entities and a common less-functionalised parent.
- `chem_rel`
 Full name: chem_rel
 Description: relations between chemicals
- `correlated_with`
 Full name: correlated_with
 Description: Entities with correlation
- `colocalized_with`
 Full name: colocalized_with
 Description: colocalized with
- `c_org`
 Full name: c_org
 Description: none
- `proper_part_of`
 Full name: proper_part_of
 Description: As for `part_of`, with the additional constraint that subject and object are distinct
- `clusters_with`
 Full name: clusters_with
 Description: Entities for which the pattern of correlation constitutes a cluster
- `correlated_with`
 Full name: correlated_with
 Description: Entities with correlation
- `translated_from`
 Full name: translated_from
 Description: a protein is translated from a mRNA (other keywords: encode, translation)
- `inst`
 Full name: inst
 Description: none

- `is_a`
 Full name: `is_a`
 Description: For continuants: `C is_a C'` if and only if: given any `c` that instantiates `C` at a time `t`, `c` instantiates `C'` at `t`. For processes: `P is_a P'` if and only if: that given any `p` that instantiates `P`, then `p` instantiates `P'`.
- `KI`
 Full name: `KI`
 Description: none
- `glycosylated_by`
 Full name: `glycosylated by`
 Description: `glycosylated by`
- `h_chem_p_hybride`
 Full name: `has chemical parent hybride`
 Description: Used to denote the relationship between an entity and its parent hydride (defined by IUPAC as "an unbranched acyclic or cyclic structure or an acyclic/cyclic structure having a semisystematic or trivial name to which only hydrogen atoms are attached").
- `chem_rel`
 Full name: `chem_rel`
 Description: relations between chemicals
- `part_of`
 Full name: `part_of`
 Description: For continuants: `C part_of C'` if and only if: given any `c` that instantiates `C` at a time `t`, there is some `c'` such that `c'` instantiates `C'` at time `t`, and `c *part_of* c'` at `t`. For processes: `P part_of P'` if and only if: given any `p` that instantiates `P` at a time `t`, there is some `p'` such that `p'` instantiates `P'` at time `t`, and `p *part_of* p'` at `t`. (Here `*part_of*` is the instance-level part-relation.)
- `c_reg`
 Full name: `c_reg`
 Description: compound regulates
- `c_op`
 Full name: `c_op`
 Description: none
- `loc`
 Full name: `loc`
 Description: OBO relation type
- `pt_in`
 Full name: `pt_in`
 Description: OBO relation type

- `ub_by`
Full name: `ub_by`
Description: Ubiquinated by
- `cont`
Full name: `cont`
Description: OBO relation type
- `is.tautomer_of`
Full name: `is.tautomer_of`
Description: A cyclic relationship used to show the interrelationship between two tautomers, where the differences between the structures are significant enough to warrant their separate inclusion in ChEBI.
- `chem_rel`
Full name: `chem_rel`
Description: relations between chemicals
- `equ`
Full name: `equ`
Description: mappinglistbased or vmatchbased equivalent proteins of the same specie
- `lgoop`
Full name: `lgoop`
Description: first gene of the operon
- `err`
Full name: Error
Description: Error in RelationType
- `st_fr`
Full name: `st_fr`
Description: state change (entry2 has a state change from entry1)
- `is.chem_sub_gr_from`
Full name: Is chemical substituent group from
Description: Indicates the relationship between a substituent group (or atom) and its parent molecular entity, from which it is formed by loss of one or more protons or simple groups.
- `chem_rel`
Full name: `chem_rel`
Description: relations between chemicals
- `me_by`
Full name: `me_by`
Description: methylated by

- ac_by
Full name: activated by
Description: activated by
- pr_by
Full name: preceded by
Description: A is the direct outcome of B
- si_to
Full name: si_to
Description: situated to
- transformation_of
Full name: transformation_of
Description: Relation between two classes, in which instances retain their identity yet change their classification by virtue of some kind of transformation. Formally: C transformation_of C' if and only if given any c and any t, if c instantiates C at time t, then for some t', c instantiates C' at t' and t' earlier t, and there is no t2 such that c instantiates C at t2 and c instantiates C' at t2.
- phys_int
Full name: physical interaction
Description: Two compounds have an unspecified physical interaction or aggregate together
- is_conjugate
Full name: is_conjugate
Description: Cyclic relationships which are used mainly between acids and their conjugate bases. When creating a new relationship, only one of these needs to be entered, as the system will create the reverse relationship.
- chem_rel
Full name: chem_rel
Description: relations between chemicals
- fn_of
Full name: fn_of
Description: OBO relation type
- acetylated_by
Full name: acetylated_by
Description: acetylated_by
- cs_by
Full name: consumed by
Description: consumed by

- `elongates`
Full name: Chemical Elongation
Description: Catalysis of chain elongation
- `chem_rel`
Full name: `chem_rel`
Description: relations between chemicals
- `sim`
Full name: `sim`
Description: NONE
- `enz_int`
Full name: enzymatic interaction
Description: Two compounds have an unspecified enzymatic interaction
- `has_participant`
Full name: `has_participant`
Description: `P has_participant C` if and only if: given any process `p` that instantiates `P` there is some continuant `c`, and some time `t`, such that: `c` instantiates `C` at `t` and `c` participates in `p` at `t`
- `p_isp`
Full name: `p_isp`
Description: OBO relation type
- `is_p`
Full name: is part of
Description: OBO relation type
- `located_in`
Full name: `located_in`
Description: `C located_in C'` if and only if: given any `c` that instantiates `C` at a time `t`, there is some `c'` such that: `c'` instantiates `C'` at time `t` and `c` `*located_in*` `c'`. (Here `*located_in*` is the instance-level location relation.)
- `hs_ch`
Full name: `hasChemical`
Description: has Ec or CAS ID
- `pd_by`
Full name: produced by
Description: produced by
- `instance_of`
Full name: `instance_of`
Description: A relation between an instance and a class. For components: a primitive relation between a component instance and a class which it instantiates at a specific time. For processes: a primitive relation, between

a process instance and a class which it instantiates, holding independently of time

- ipara
Full name: ipara
Description: blastbased inparalog proteins of same species
- di_fr
Full name: di_fr
Description: dissociated from
- adj
Full name: adj
Description: OBO relation type
- has_function
Full name: has_function
Description: Is used to describe that a concept has a specified function. For instance, gene has a molecular function (TAIR, GO).
- rg_by
Full name: regulated by
Description: regulated by
- transcribed_from
Full name: transcribed_from
Description: a mRNA is transcribed from a gene (other keywords: encode, transcription)
- act
Full name: act
Description: none
- id_by
Full name: id_by
Description: indirect (same as used for GErel)
- derives_from
Full name: derives_from
Description: Derivation on the instance level (*derives_from*) holds between distinct material continuants when one succeeds the other across a temporal divide in such a way that at least a biologically significant portion of the matter of the earlier continuant is inherited by the later. We say that one class C derives_from class C' if instances of C are connected to instances of C' via some chain of instance-level derivation relations. Example: osteocyte derives_from osteoblast. Formally: C derives_immediately_from C' if and only if: given any c and any t, if c instantiates C at time t, then there is some c' and some t', such that c' instantiates C' at t' and t' earlier-than t and c *derives_from* c'. C

derives_from C' if and only if: there is an chain of immediate derivation relations connecting C to C'.

- it_wi
Full name: it_wi
Description: interacts with
- anto
Full name: anto
Description: none
- is_enantiomer_of
Full name: is_enantiomer_of
Description: A cyclic relationship used when two entities are enantiomers of each other. An entity may have this relationship with only one other entity.
- chem_rel
Full name: chem_rel
Description: relations between chemicals
- cat_c
Full name: catalysing class
Description: part of catalysing class
- preceded_by
Full name: preceded_by
Description: on P preceded_by P' if and only if: given any process p that instantiates P at a time t, there is some process p' such that p' instantiates P' at time t', and t' is earlier than t.
- adjacent_to
Full name: adjacent_to
Description: n C adjacent to C' if and only if: given any instance c that instantiates C at a time t, there is some c' such that: c' instantiates C' at time t and c and c' are in spatial proximity
- intersects
Full name: has intersection with
Description: The meaning of one concept has an intersection with the meaning of another concept
- unkwn
Full name: unkwn
Description: unknown relation type
- is_hypermym_of
Full name: i_isa
Description: Instance Hypernym

- para
Full name: para
Description: blastbased paralog (homolog) proteins or genes of the same specie
- ph_by
Full name: ph_by
Description: phosphorylated by
- is_ap
Full name: is_ap
Description: is a part of
- not_participant
Full name: not_participant
Description: A concept that is known to not be a participant in the specified process.
- contained_in
Full name: contained_in
Description: on C contained_in C' if and only if: given any instance c that instantiates C at a time t, there is some c' such that: c' instantiates C' at time t and c located_in c' at t, and it is not the case that c *overlaps* c' at t. (c' is a conduit or cavity.)
- re_by
Full name: repressed by
Description: repressed by
- inv_in
Full name: involved_in
Description: Involvement in something e.g. disease
- inverse_correlation_with
Full name: inverse_correlated_with
Description: Entities that are inversely correlated (-ve coefficient)
- correlated_with
Full name: correlated_with
Description: Entities with correlation
- im_of
Full name: im_of
Description: intermediate of
- pr_by
Full name: preceded by
Description: A is the direct outcome of B

- de_by
Full name: de_by
Description: dephosphorylated by
- dm_by
Full name: dm_by
Description: demethylated by
- der
Full name: der
Description: none
- dev
Full name: dev
Description: OBO relation type
- s_isp
Full name: s_isp
Description: none
- is_p
Full name: is part of
Description: OBO relation type
- integral_part_of
Full name: integral_part_of
Description: C integral_part_of C' if and only if: C part_of C' AND C' has_part C
- hvfs
Full name: hvfs
Description: none
- h_s_s
Full name: has similar sequence
Description: blastbased homolog proteins of different species
- sensu
Full name: sensu
Description: GO concept names contain a sensu part which we break up into separate concepts and relations
- h_pwm
Full name: h_pwm
Description: position weight matrix (transfac)
- has_agent
Full name: has_agent
Description: As for has_participant, but with the additional condition that the component instance is causally active in the relevant process

- not_function
Full name: not_function
Description: Is used to describe that a concept has been shown not to have a specified function. For instance, a gene doesn't have a specified molecular function (TAIR, GO).
- sh_im
Full name: sh_im
Description: none
- ortho
Full name: ortho
Description: blastbased ortholog proteins of different species
- dr_fr
Full name: dr_fr
Description: OBO relation type
- bi_to
Full name: bi_to
Description: binds to
- att
Full name: att
Description: none

Appendix D

Importing data in Ondex

Various “parsers” allow users to import data and convert it to OXL format (Ondex eXchange Language). This format of data can then be used by Ondex mapping methods, transformers, etc. or can simply be open in the Ondex Visualisation Toolkit.

D.1 Parsers

D.1.1 OXL import

Parser for OXL files.

- Importfile
Specify Ondex XML file.
- IgnoreGDS
Do not parse GDS attributes specified here.

D.1.2 Aracyc

Download the Aracyc flatfile database from http://www.plantcyc.org/downloads/license_agreement.faces (<ftp://ftp.arabidopsis.org/home/tair/home/tair/tmp/private/plantcyc/aracyc.tar.gz>). Extract contents of the data directory inside the tar to the root directory (this includes .col and .dat files).

- Input folder
Folder with data to import.

D.1.3 Atregnet

Parses AtRegNet (*Arabidopsis thaliana* Regulatory Networks) focussing on protein protein interactions. This requires AtRegNet_(some date).zip. Getting a

hold of this file requires registering with AGRIS (<http://arabidopsis.med.ohio-state.edu/downloads.html>). This zip file then needs to be decompressed to give a .tbl (the corresponding .txt file is not read by the parser) which must be placed within the data directory in the Ondex folder hierarchy (specified by -Dondex.dir).

- Input File
Relative path from data directory to file with data to import.

D.1.4 Atregnet2

General parsing of AtRegNet (*Arabidopsis thaliana* Regulatory Networks). This requires AtRegNet_(some date).zip, AtcisDB.zip, and AtTFDB.zip. Getting a hold of this file requires registering with AGRIS (<http://arabidopsis.med.ohio-state.edu/downloads.html>).

The following zip files need to have the following files unpacked to the same folder. Specifically, the following files are required:

- reg_net_(some date).tbl from AtRegNet_(some date).zip
- GeneInfo.tbl from AtcisDB.zip
- families_data.tbl from from AtTFDB.zip
- families_pep.tbl from AtTFDB.zip

The file reg_net_(some date).tbl will need to be renamed to reg_net.tbl

- Input File
Relative path from data directory to file with data to import.

D.1.5 Biogrid

Parses the BioGRID (General Repository for Interaction Datasets). Download from <http://www.thebiogrid.org/downloads.php>.

- Input File
File with data to import.

D.1.6 BioCyc

The BioCyc (<http://biocyc.org/intro.shtml>) parser is a generic parser. Beside the import data directory it requires two arguments, one for CV (the Ondex term for data source, *e.g.*, SolCyc) and one for Taxid (the NCBI organism identifier, *e.g.*, 3702 for *A.thaliana*).

The BioCyc parser only parses the very basic set of features from a BioCyc database. That is why there are more specific parser like AraCyc, which parse more accessions and adapt to the notions of the curators used for not clearly structured data.

BioCyc databases are usually organism specific, *i.e.* you would be downloading the flat files per organism anyway, *e.g.*, one for SolCyc, one for LycoCyc and so on however they might be called. Therefore to import multiple organism you simply call the parser multiple times, at each time pointing to the respective directory of the organism specific BioCyc database.

The parser takes a biopax.owl file (not a .gzip). It also needs a translation file that specifies the translation of the Ondex representation that should be placed in the same directory as the data being parsed. This file is included in the core data directory.

D.1.7 Brenda

Parser for the BRENDA database (comprehensive enzyme information system). Download from <http://www.brenda-enzymes.info/> (download link in the left hand side menu).

- Species
Species taxonomy identifier from NCBI preferred.
- Input folder
Folder with data to import.

D.1.8 Coex

Parser for coexpression databases. Download data from <http://coexpresdb.jp/> or <http://atted.jp/>.

- RelationType
Specify which existing relations to add correlation score to.
- Rank
Rank cut-off.
- Filter_file
Accessions of stable ges to be excluded, one accession per line.
- Threshold
Correlation strength cut-off.
- Accession
Accession used by the coexpression database to identify entries.
- Random_nodes
Number of random nodes in the network.
- Random_edges
Number of random edges in the network.
- Input folder
Folder with data to import.

D.1.9 Correlationtab

A relation centric approach to importing concepts and relations. Concepts are defined as a header list. Relations are defined in matrices where the values are added as a GDS attribute on the relation. Where no attributes are specified, no relations are created. Multiple values may be assigned by the use of additional matrices. For example,

- Table headers: gene names
- Matrix 1: correlation coefficients corresponding to the genes
- Matrix 2: p values

NB: A restriction of this parser is that `AttributeName` objects parsed in must be instantiatable with a single string as the constructor.

- `HeaderValues`
Header Values representing concepts in a list form (concept values will be added as unambiguous accessions).
- `InclusionList`
Genes from the matrices to include (all other genes will be ignored).
- `HeaderConceptClass`
CC to assign to concepts.
- `CV`
CV to assign to these concepts and relations.
- `AccessionCV`
CV for the accession identifying the headers.
- `EvidenceType`
EvidenceType to assign to these concepts and relations.
- `HeaderAttributeName`
An `AttributeName` and value to create as a GDS on each concept.
- `CorrelationAttributeName`
Specify a correlation `AttributeName`.
- `CorrelationFile`
Specify a `CorrelationFile`.
- `PValueCutOff`
The highest P-value allowed.
- `MinCorrelation`
The minimum correlation allowed, defined as values greater than the absolute correlation value (parsed when the `MinCorrelation` > `abs(correlation)`).

- Input folder
Folder with data to import.

D.1.10 Customtab

Customisable tab-delimited file parser.

- Input File
File with data to import..

D.1.11 Drastic

Drastic (Database Resource for the Analysis of Signal Transduction In Cells) parser for the file: “Drastic Data.csv” from <http://www.scri.sari.ac.uk/TiPP/PPS/DRASTIC/mpage/downloads.asp>.

- Input folder
Folder with data to import.
- Input File
File with data to import.

D.1.12 EC

EC nomenclature parser for the files: “enzclass.txt”, “enzyme.dat” from ftp://ftp.expasy.org/databases/enzyme/release_with_updates/.

- Deleted
This specifies whether the enzymes with a deleted flag will be parsed from the EC database.
- Input folder
Folder with data to import.

D.1.13 Fasta

- FastaFiles
The fasta files to be parsed and imported into Ondex
- FastaFileType
This setting must be used in order to parse the fasta header as best as possible. The fasta file type “simple” is the most generic type. Other types are “AFFYMETRIX”, “NCBI”, “gramene” and “custom”.
- TaxId
Specify a taxonomy identifier (species).
- CC
The type of sequences (*e.g.*, target, gene, protein).

- CV
The source of the sequences (*e.g.*, AFFY, TAIR, dbEST, unknown).
- SeqType
The type (attribute name) of the sequences (*e.g.*, “NA”, “AA”).
- Separator
Regular expressions to split header in simple FASTA parser.
- AccessionRegex
Regular expressions to be used as an additional accession.
- Input folder
Folder with data to import.
- Input File
File with data to import.

D.1.14 Genericobo

OBO Ontology parser.

- OboType
This setting must be used in order to parse the OBO file as best as possible. The fasta file type “generic” is the most generic type. Other types are “GO”, “PO” and “ChEBI”.
- Obsoletes
This specifies whether OBO-concepts with the obsolete flag should be parsed.
- Input folder
Folder with data to import.

D.1.15 Generif

Parser for the GeneRIF file. Download from ftp://ftp.ncbi.nih.gov/gene/GeneRIF/generifs_basic.gz Creates relations from Genes to Publications and assigns the GeneRIFs to them.

- TaxId
Will parse genes with comma separated taxonomy identifiers specified here.
- Input folder
Folder with data to import.

D.1.16 Go

GO Ontology parser for the file: “gene_ontology_edit.obo” in OBO v1.2 format from http://www.geneontology.org/ontology/gene_ontology_edit.obo (latest tested version: 16/01/07).

- **Obsoletes**
This specifies whether OBO-concepts with the obsolete flag should be parsed.
- **Input folder**
Folder with data to import.

D.1.17 Goa

The GOA (Gene Ontology Annotation) parser. Info: <http://www.ebi.ac.uk/GOA/> and <http://www.geneontology.org/GO.current.annotations.shtml>. Download: <ftp://ftp.ebi.ac.uk/pub/databases/GO/goa/>.

- **Species**
Specify the species for which the GOA files have to be parsed. When all are selected, all species will be parsed.
- **Expanded**
If set to true, will create a Gene to PubMed to GO relation (2 binary relations) instead of a Gene to GO relation with a PubMed identifier qualifier (1 ternary relation).
- **Input folder**
Folder with data to import.

D.1.18 Gramene

Gramene parser. Info: a resource for comparative grass genomics, <http://www.gramene.org/>. ftp: ftp://ftp.gramene.org/pub/gramene/CURRENT_RELEASE/data/database_dump/mysql-dumps/ (downloading during daytime is slow at night the speed is a lot better)

- **Input folder**
Folder with data to import.

D.1.19 Kegg52 (experimental package)

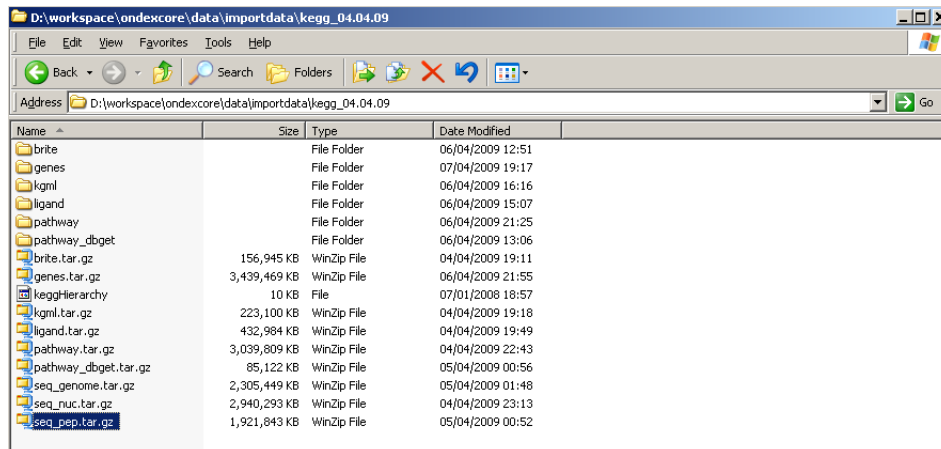
Parser for the KEGG database - not in working order. Fully utilizes ternary relations and context lists.

- **Species**
This option provides a list of species to import pathways from KEGG for. Use “all” to include all available species.

- ParseSequences
This option specifies whether, for every enzyme/protein/gene, the sequence will be parsed and added as a GDS value. Please note that parsing all the sequences can take a considerable amount of time.
- Input folder
Folder with data to import.

D.1.20 Kegg52 (stable)

Parser for the KEGG database (<http://www.genome.jp/kegg/download/>). Download all the gzip files for the current release of KEGG (brite, genes, kgml, ligand, pathway, pathway_dbget), plus the hierarchy file (keggHierarchy 10KB file).



Name	Size	Type	Date Modified
brite		File Folder	06/04/2009 12:51
genes		File Folder	07/04/2009 19:17
kgml		File Folder	06/04/2009 16:16
ligand		File Folder	06/04/2009 15:07
pathway		File Folder	06/04/2009 21:25
pathway_dbget		File Folder	06/04/2009 13:06
brite.tar.gz	156,945 KB	WinZip File	04/04/2009 19:11
genes.tar.gz	3,439,469 KB	WinZip File	06/04/2009 21:55
keggHierarchy	10 KB	File	07/01/2008 18:57
kgml.tar.gz	223,100 KB	WinZip File	04/04/2009 19:18
ligand.tar.gz	432,984 KB	WinZip File	04/04/2009 19:49
pathway.tar.gz	3,039,809 KB	WinZip File	04/04/2009 22:43
pathway_dbget.tar.gz	85,122 KB	WinZip File	05/04/2009 00:56
seq_genome.tar.gz	2,305,449 KB	WinZip File	05/04/2009 01:48
seq_nuc.tar.gz	2,940,293 KB	WinZip File	04/04/2009 23:13
seq_pep.tar.gz	1,921,843 KB	WinZip File	05/04/2009 00:52

Figure D.1: KEGG directory

- Species
Specify the species to be loaded from the KEGG database.
- ParseSequences
Boolean flag. KEGG species code.
- ImportOrthologFillers
Import ortholog pathway fillers.
- Input folder
Folder with data to import.

D.1.21 Kegg53 (stable)

Parser for the KEGG database (<http://www.genome.jp/kegg/download/>) version 53.

- InputDir
Absolute path to input directory.
- Species
Specify the species code (NCBI taxID) of species to import pathways for.
- ParseAllSequencesForSpecies
Specifies to parse all genes for a species regardless of membership of pathway.
- SpeciesOrthologs
Parse orthologs to specified species.
- OnlyReferenced
Import only pathway entries which are referenced in relation or reaction section.

D.1.22 Medline

Parser for MEDLINE (<http://medline.cos.com/>).

- Summary http://www.nlm.nih.gov/bsd/licensee/2009_stats/baseline_med_filecount.html
- Download baseline: <ftp://ftp.nlm.nih.gov/nlmdata/.medleasebaseline/gz/>
- Download updates: ftp://ftp.nlm.nih.gov/nlmdata/.medlease/gz/*.gz
- reads files: PubMed export XML files, PMID files
- webservice: Efetch to get publications

All arguments are optional:

- Prefix
The Prefix letters that start the medline file in the form of "medline09n" where "09" is the year
- Compression
The Compression file type currently on gunzip denoted by "gz" is permitted
- PMIDInputList
List of PUBMED IDs separated by semicolon

- **XmlFiles**
Medline XML files to include.
- **PubMedFile**
Absolute path to an XML file which contains the results of a PubMed search, *e.g.* search in PubMed for “Arabidopsis”, click on “Send to: ” at the top right-hand side of the PubMed page, “Choose destination” and “File” to save the results.
- **ImportOnlyCitedPublications**
Import only publications that are cited in the Ondex graph.
- **LowerXmlBoundary**
The value of this argument is only taken into account if nothing is specified for PMIDInputList, XmlFiles, PubMedFile and ImportOnlyCitedPublications. In that case it will only parse Medline XML files which filename contains a number greater than this boundary.
- **UpperXmlBoundary**
The value of this argument is only taken into account if nothing is specified for PMIDInputList, XmlFiles, PubMedFile and ImportOnlyCitedPublications. In that case it will only parse Medline XML files which filename contains a number lower than this boundary.
- **Input folder**
Folder with data to import.
- **UseEfetchWebService**
If set to true, uses NCBI’s efetch to get publications. If false (default) parses local files.

D.1.23 Nodelist

Integrates a tab-delimited file that contains a list of nodes.

- **Id column**
Specifies the number of the column that contains the node identifier.
- **Name column**
Specifies the number of the column that contains the name.
- **Accession column**
Specifies the number of the column that contains the accession data.
- **Source of accession**
Specifies the source database that the accession stands for.
- **Concept class**
Specifies the concept class that all nodes will be defined as.

- Data source name
Specifies the name of the data source.
- Input File
File with data to import.

D.1.24 Nwb

Parser for the Network Workbench format.

- Input File
File with data to import.

D.1.25 Omim

Parser for the Online Mendelian Inheritance in Man (a database of human genes and genetic disorders). Download from <http://www.ncbi.nlm.nih.gov/Omim/omimfaq.html#download>.

- Input folder
Folder with data to import.

D.1.26 Pdb

Protein Data Bank Parser. Download data from <ftp://ftp.wwpdb.org/pub/pdb/data/structures/all/pdb/>. The Parser searches in a folder for all “.enz-files” and parses them.

- DataDir
Specifies the data directory. Can be used to define an absolute path, if the data is stored within the ondex data directory use the datadir parameter.

D.1.27 Pfam

Parser for the pfam database, <http://pfam.sanger.ac.uk/>.

- SearchForPfam
When set to true, it just searches for references.
- Input File
File with data to import.

D.1.28 Plntfdb

PlnTFDB (2.0) is a public database <http://plntfdb.bio.uni-potsdam.de/v2.0/> arising from efforts to identify and catalogue all Plant genes involved in transcriptional control. Download data from <http://plntfdb.bio.uni-potsdam.de/v3.0/downloads.php>. The data from that site should be added to a directory (only the transcription factor list and protein sequence files are required).

- Input folder
Relative path to folder with data to import.

D.1.29 Poplar

Parser for Poplar sequences and functional annotations. Files can be downloaded from: http://genome.jgi-psf.org/Poptr1_1/Poptr1_1.download.ftp.html.

- Input folder
Folder with data to import.

D.1.30 Psimi

PSI-MI (Proteomics Standards Initiative - Molecular Interactions) parser - not in working order.

- Input folder
Folder with data to import.

D.1.31 Psimi2

PSI-MI (Proteomics Standards Initiative - Molecular Interactions) parser.

- Input folder
Folder with data to import.

D.1.32 Sbml

Systems Biology Markup Language (SBML) parser - not in working order.

- Input File
File with data to import.

D.1.33 Sbml2

Parser for Systems Biology Markup Language (SBML), using libSBML, targeting SBML level 2. Install the 3.4 Xerces version of libSBML and modify `runme.bat` or `runme.sh` adding a parameter `-Djava.library.path="Path to SBML JAVA libraries"`.

- Input folder
Folder with data to import.

D.1.34 Simplegraph

Imports tab-separated relation lists.

- CC
Concept class of concepts created by this parser
- RTS
Realtion type set of realtions created by this parser
- CV
CV to use
- Input File
File with data to import.

D.1.35 Tab

Parses relations from tabular files.

- Skip
How many rows to skip at begin of document.
- FromCol
Index of concept parser id for from concept.
- ToCol
Index of concept parser id for to concept.
- FromNameCol
ConceptName index for use with from concept.
- ToNameCol
ConceptName index for use with to concept.
- FromPhenoCol
GDS Pheno index for use with from concept
- ToPhenoCol
GDS Pheno index for use with to concept.
- ConfCol
Index of confidence score of relation.
- TaxId
Which taxonomy id should be assigned to the sequences.
- CC
The type of the concepts (e.g. target, gene, protein)
- CV
The source of the concepts (e.g. TAIR, KEGG, unknown)

- **RelationType**
The relation type to create for every line in the tabular file.
- **Threshold**
Import threshold for confidence value.
- **FromTaxId**
TaxID index for use with from concept.
- **ToTaxId**
TaxID index for use with to concept.
- **Input File**
File with data to import.

D.1.36 Tableparser

- **Sheet**
If parsing data from an Excel spreadsheet this is a required argument.
- **Col_separator**
If parsing from a delimited file this is a required argument (Java regex).
- **FirstRow**
First row (Optional).
- **LastRow**
Last row (Optional for tab delimited files, strongly recommended for MS-EXCEL files).
- **Attribute**
Convert value in the column into the specified attribute (GDS, name or accession), the format is
concept_id, column number, type,[type specific options]
where the first three arguments are required and the rest are optional. Concept_id can be anything but is used to group the attributes to appropriate concepts. Types can be GDS, NAME or ACC. Examples are:
c1,1,NAME
c1,2,ACC,TAIR
c2,3,GDS,p-value,NUMBER
c2,4,GDS,description,TEXT
c3,5,GDS,count,INTEGER
- **Concept_class**
Tuple of concept_id and corresponding concept class it should have *e.g.*:
c1,Protein
- **Input File**
File with data to import.

D.1.37 TAIR9

The hierarchy below describes both the structure of the required TAIR9 folder and subfolders, and where to download their content from (urls starting with ftp).

- TAIR9
 - `ftp://ftp.arabidopsis.org/home/tair/Genes/Locus_history/locushistory_2009` (change this filename to locushistory.txt)
 - `ftp://ftp.arabidopsis.org/home/tair/User_Requests/` download LocusPublished.date.txt (where date changes depending on latest version)
 - `ftp://ftp.arabidopsis.org/home/tair/Genes/TAIR9_genome_release/TAIR9_NCBI_REFSEQ_mapping_PROT`
 - `ftp://ftp.arabidopsis.org/home/tair/Genes/TAIR9_genome_release/TAIR9_NCBI_REFSEQ_mapping_RNA`
 - TAIR9_blastsets
 - * `ftp://ftp.arabidopsis.org/home/tair/Sequences/blast_datasets/TAIR9_blastsets/TAIR9_cdna_20090619`
 - * `ftp://ftp.arabidopsis.org/home/tair/Sequences/blast_datasets/TAIR9_blastsets/TAIR9_pep_20090619`
 - Proteins
 - * Domains
 - `ftp://ftp.arabidopsis.org/home/tair/Sequences/blast_datasets/TAIR9_blastsets/TAIR9_cdna_20090619`
 - * Id.conversions
 - `ftp://ftp.arabidopsis.org/home/tair/Proteins/Id_conversions/Uniprot2AGI.20091219`

The two parameters for this parser are as follows:

- Annotation
 - If set to true, GO annotations, protein domain information and UniProt accessions (rather than just the FASTA sequence files) will also be parsed in.
- Input folder
 - Folder with data to import.

D.1.38 Taxonomy

Parser for the NCBI Taxonomy database. Creates a is_a hierarchy of terms. Uses names.dmp and nodes.dmp.

- Root
At which root node in taxonomy tree should the parser start. Any valid taxonomy id.
- Input folder
Folder with data to import.

D.1.39 Tigrricefasta

FASTA file parser

- Importfiles
Importfiles
- Input folder
Folder with data to import.

D.1.40 Timeseries

- TAXID
NCBI TaxID
- ConceptClass
ConceptClass to map probes/targets to
- IgnoreLSD
Ignore the Least Significant Difference restrictions
- IsRatio
Data is a ratio to t0
- Input File
File with data to import.

D.1.41 Transfac

Parses the Transfac database from BioBase (<http://www.gene-regulation.com/pub/databases.html>).

- Input folder
Folder with data to import.

D.1.42 Transpath

Parses the Transpath database from BioBase (<http://www.gene-regulation.com/pub/databases.html>).

- ParseMolecules
ParseMolecules

- ParsePathways
ParsePathways
- Input folder
Folder with data to import.

D.1.43 Uniprot

Parser for the UniProt Knowledgebase database (<http://www.uniprot.org/downloads>, download in XML format). Implements this <http://www.devx.com/Java/Article/30298/0/page/2> way of parsing a xml file.

- TaxId
will parse proteins with given taxonomy id
- DbRefAcc
if set to true, will search for accessions which already belong to a concept in the given graph
- Accessions
a list of comma-separated accession numbers (any kind of accession, does not need to be uniprot accession), which should be searched and parsed in the given uniprot database
- AccessionsFile
path to the file containing a list of accession numbers (one accession per line), which should be searched and parsed in the given uniprot database
- ContextInformation
if set to true, will attach context information to the concepts
- HideLargeScaleRef
if set to true, will hide large scale references (publications)
- GoFile
Absolute path to the GO OBO file, which is indexed and used to create the right relation type between proteins and GO annotations
- Input folder
Folder with data to import.

D.1.44 Wordnet

Works only with most recent Windows version of WordNet 2.1. Download from <http://wordnet.princeton.edu/wordnet/download/#win> Put data.adv, data.verb, data.adj, data.noun into data/importdata/wordnet or folder to be parsed.

- Input folder
Folder with data to import.

Appendix E

Mapping data in Ondex

E.1 Mapping

Mapping methods allow users to create “equ” relations (or whatever you wish to name them) between concepts identified as equivalent based on a specified item, whether accession, name, sequence, attribute value, *etc.*

E.1.1 Concept accession-based mapping (Memory-efficient)

Accession based mapping maps all concepts of the same concept class (*CC*) and of different data source or controlled vocabulary (*CV*) unless stated otherwise. See examples in Section 2.3.

- **EquivalentCC** (optional)
EquivalentCC means equivalent concept class. If you want your accession based mapping to map concepts across CCs rather than within CCs, this is the option you need to fill in. This option should contain a pair of CCs separated by a comma (*e.g.*, “Target,Protein”).
- **EquivalentCV** (optional)
EquivalentCV means equivalent accession type. This option is on the CV of the accession of the concept (not the CV of the concept). The usage of this setting is to explicitly tell that two accessions with different accession types actually contain the same information. This option should contain a pair of CVs separated by a comma (*e.g.*, “TIGR,TAIR”).
- **IgnoreAmbiguity** (optional)
IgnoreAmbiguity means use ambiguous accessions. When true, this allows ambiguous concept accessions to be mapped.
- **RelationType** (optional)
This mapping method will create relations between concepts that map.

This option specifies the relation type of the relation to be created. By default, the relation type “equals” (equ) is used.

- **ConceptClassRestriction** (optional)
ConceptClassRestriction means concept class restriction. This is the option to use if you wish to restrict the mappings to concepts within one or more specified CC(s). For example, if a network contains genes and proteins but you only want to map proteins between themselves (rather than proteins between themselves and genes between themselves), you would use this option specifying “Protein”.
- **CVRestriction** (optional)
CVRestriction means accession type restriction. This restriction is on the CV of the accession of the concept (not the CV of the concept).
- **GDSEqualsConstraint** (optional)
GDSEqualsConstraint means attribute restriction. This will limit the mapping method to only map concepts with a GDS (General Data Store) or attribute value matching the attribute name specified by this parameter.
- **WithinCVMapping** (optional)
WithinCVMapping means within data source. Instead of mapping across all CVs, the mapping will be performed within CV(s) specified here.

E.1.2 BLAST based

This mapping uses the Basic Local Alignment Search Tool (BLAST).

- **EquivalentCC** (optional)
If you want your BLAST based mapping to map concepts across CCs rather than within CCs, this is the option you need to fill in. This option should contain a pair of CCs separated by a comma (*e.g.*, “Target,Protein”).
- **PathToBlast** (required)
Specify the path to your BLAST executable.
- **Evalue** (optional)
E-value cutoff BLAST argument (http://www.swbic.org/origin/program/Blast/BLAST_tutorial.html).
- **SeqGDS** (optional)
Specifies the GDS (General Data Store) or attribute containing the sequence data.
- **SeqType** (optional)
Specifies what sequence type is contained in the GDS (*e.g.*, NA, AA).
- **WithinSameCV** (optional)
This option allows mapping within the same CV.

E.1.3 Cross-species

The cross-species mapping method calculates the best orthologs for each species present in the graph for each of the query concepts. The best ortholog in this instance is defined as the largest coverage of the shortest sequence. An example usage is when annotating a microarray to find the closest orthologs for each sequence in each database.

- QueryTAXID (optional)
TaxID for which to perform the cross species mapping.
- Overlap (optional)
The minimum overlap to tolerate. The default value is 0.25.
- QueryConceptClass (optional)
Filter by a target Query CC (default is all).
- EValue (required)
Cutoff for the BLAST E-Value (http://www.swbic.org/origin/proc_man/Blast/BLAST_tutorial.html). This option reduces the search space for identifying orthologs. It does not mean the algorithm searches for the best E-value rather than the best alignment.
- TargetSequenceType (required)
Sequence type must inherit from either Amino Acid (AA) or Nucleic Acid (NA).
- ComparisonType (optional)
Comparison type that defines the better alignment (bitscore, covera).
- TargetConceptClass (optional)
Filter by a target CC.
- Cutoff (optional)
The minimum length of sequence to allow as a valid alignment.
- EntryPointIsCV (optional)
Indicates whether CV (database) should be considered a unique entry point for blasting. This forces the algorithm to identify the best orthologs for each species and database (CV). *E.g.*, it is possible to find the best orthologs for each species and each database (KEGG, AraCyc, TransFac).
- ProgramDir (required)
Specify the directory where the sequence alignment program is located.
- EntryPointIsTAXID (optional)
Indicates whether species should be considered a unique entry point for blasting.

- QuerySequenceType (required)
Sequence type must inherit from either Amino Acid (AA) or Nucleic Acid (NA).
- AlignmentThreshold (optional)
Indicates the number (integer parameter) of alignments that should be taken per entry point, based on the top ranking coverage/alignment length of the longer/shorter(default) sequence in the match. The default value is 1.
- SequenceAlignmentProgram (required)
The Alignment program to use (*e.g.*, blast, patternhunter, fsablast). The default value is decypher.

E.1.4 GDS equality

Creates relation where all specified GDS are equal.

- WithinCVMMapping (optional)
Map within CVs.
- RelationType (required)
RelationType to create when conditions are met.
- AttributeNames (required)
AttributeNames that must be present and equal for concept to map.

E.1.5 Inparanoid

Implements the INPARANOID algorithm¹ as a mapping method for the Oindex system.

- PathToBlast (required)
Path to BLAST executable.
- Evaluate (optional)
Evaluate cutoff BLAST argument.
- SeqGDS (optional)
Specifies the GDS attribute containing the sequence data. The default is AA.
- SeqType (optional)
Specifies what sequence type is contained in the GDS (*e.g.*, “NA, AA”).
- Cutoff (optional)
Bit-score cutoff (default 30).

¹Automatic clustering of orthologs and in-paralogs from pairwise species comparisons. Rmm M., Storm C.E., Sonnhammer E.L. Journal of molecular biology, 314(5):1041-52, PubMed ID: 11743721

- Overlap (optional)
Sequence overlap of match length compared to longest sequences (default 0.5).

E.1.6 Name based

Implements a ConceptName based mapping which is always case insensitive whatever options are ticked.

- GDSEqualsConstraint (optional)
This will limit the mapping method to only map concepts when the GDS Value with the attribute name specified by this parameter is the same.
- EquivalentCC (optional)
This option should contain a pair of CCs seperated by a comma (*e.g.*, “Thing,EC”). The usage of this setting is to allow the mapping method to cross the CC boundary in some special cases and thus be able to map for example similar GO and EC concepts to each other.
- ExactSynonyms (optional)
Matches preferred concept names or synonyms (shown in green in the item information window) only.
- NameThreshold (required)
By default two names/synonyms of two concepts must match for them to be mapped. This is set so as to avoid mapping ambiguous names. However the threshold can be set to any number, 1 if concepts have a single name, 3 or more for ambiguous data.
- ConceptClassRestriction (optional)
Instead of mapping all concepts of the same CC for all CCs, the mapping will be restricted to all concepts of the CC(s) specified here.
- CVRestriction (optional)
Instead of mapping across all CVs, the mapping will be restricted to the CV(s) specified here.
- WithinCVMapping (optional)
Instead of mapping across all CVs, the mapping will be performed within the CV specified here.
- ExactNameMapping (optional)
Forces exact matching.

E.1.7 Ortholog prediction

Predicts orthologs between species based on a best bidirectional hits.

- ProgramDir (required)
The directory where the sequence alignment program is located.
- Cutoff (optional)
The minimum length of sequence to allow as a valid alignment.
- ScoreCutoff (required)
Score above which to register similar sequences.
- TaxId (optional)
Taxonomy identifiers to include in the paralog prediction.
- SequenceAlignmentProgram (required)
The Alignment program to use (*e.g.*, blast/patternhunter/fsablast).
- EValue (required)
Cutoff for the BLAST E-Value.
- Overlap (optional)
The minimum overlap to tolerate
- SequenceType (required)
Sequence type must inherit from either Amino Acid (AA) or Nucleic Acid (NA).

E.1.8 Sequence2pfam

The sequence to Pfam mapping method maps a protein with an attached amino acid sequence to pfam protein-family entries.

- ProgramDir (required)
Specify blast/hmmer directory.
- PfamPath (required)
Location of the Pfam database.
- TmpDir (required)
Temporary directory.
- Evalute (required)
E-value cutoff argument.
- Method (required)
BLAST (default), Hmmer or Decypher.
- BitScore (optional)
Bitscore cutoff argument. (NB: will only work with decypher).
- IgnorePfamAccessions (optional)
If true, the mapping method tries to identify the protein family even if there are already protein family annotations added to the protein.

- HMMThresholds (optional)
The HMM THRESHOLDS (can be null) specifies the use of one or more of the threshold values specified in a hidden Markov model as a minimum criteria that search results must meet to be presented in the output file. Pfam and other curated databases of hidden Markov models may contain within them annotation lines specifying per-model threshold values for scoring alignments that use the model (*i.e.*, GA, NC or TC).

E.1.9 StructAlign

Implements the StructAlign mapping which maps two concepts if they have the same structure of neighbours at depth one.

- EquivalentCC (optional)
This option should contain a pair of CCs seperated by a comma (*e.g.*, “Thing,EC”). The usage of this setting is to allow the mapping method to cross the CC boundary in some special cases and thus be able to map for example similar GO and EC concepts to each other.
- GDSEqualsConstraint (optional)
This will limit the mapping method to only map concepts when the GDS Value with the attribute name specified by this parameter is the same.
- ExactSynonyms (optional)
Force matching of exact synonyms (preferred concept names) only.
- Depth (required)
Depth of graph traversal to look for other matches.
- ConceptClassRestriction (optional)
Instead of mapping all concepts of the same CC for all CCs, the mapping will be restricted to all concepts of the CC(s) specified here.
- CVRestriction (optional)
Instead of mapping across all CVs, the mapping will be restricted to the CV(s) specified here.

E.1.10 Tmbased

The text-mining based mapping method links concepts of any given Concept Class to concepts of type Publication if one or more of their names occur(s) in either the title or the abstract of the publication (both saved as GDS values). The Lucene environment is used to search for concept names (terminologies, controlled vocabularies) in title and abstracts (GDS). Text-mining based mapping can be used to create relations of type “is_r” between concepts of class *e.g.* Gene Ontology (GO), EC, Gene, Protein, Taxonomy, *etc* to concepts of class Publication. Evidence sentences supporting the mapping are attached to the relation as well as a text-mining score (TF-IDF²). This score is actually boosted

²<http://nlp.stanford.edu/IR-book/html/htmledition/tf-idf-weighting-1.html>

if a word is found in the title rather than in the abstract of the article.

- **ConceptClass** (required)
The Concept Class that will be mapped to concepts of type Publication. Multiple instances of this parameter are allowed.
- **OnlyPreferredNames** (optional)
Set to true to consider preferred concept names of concepts only.
- **Search** (required)
Choose one instance between the following: Exact (default value), Fuzzy, Proximity, And. Exact only looks for exact occurrences of the words in the title and the abstract of the publication. Fuzzy looks for exact occurrences as well as similar words to the query and it allows any order of words. Proximity looks for words within the given distance (10 words by default). And operates in the same way as Exact except it accepts any order of words.
- **Filter** (optional)
In many cases, a publication is mapped to many concepts of different concept classes. This optional parameter allows users to eliminate some of the low level mappings. Choose one instance between the following: LowScore, MaxSpecificity, BestHits. It is also possible to combine filters (comma separated). Note the order of the filters then has an importance. LowScore removes all hits with a TF-IDF score lower than 0.3. MaxSpecificity only keeps the most specific hit if several hits exist within an ontology. BestHits only keeps the best six hits between a publication and a concept class.