Overview

HTTPS https://bitbucket.org/armish/gsoc14

Last updated	2015-06-23	1 Branch	13 Tags
Language	Java		
Access level	Read	0 Forks	5 Watchers

About

This is the repository that contains code relevant to the Google Summer of Code 2014 project titled "Idea 15: Work on the Pathway Database Converters for the Expansion of Pathway Commons". This project is being conducted under the mentorship of National Resouce for Network Biology (NRNB) with the help of Google's GSOC'14 program. Feel free to contact the author, B. Arman Aksoy if you have any questions about the project and the code.

Pathway Commons

Pathway Commons (PC) is a network biology resource and acts as a convenient point of access to biological pathway information collected from public pathway databases, which you can search, visualize and download. The PC framework allows aggregating and normalizing data from multiple biological pathway databases by utilizing BioPAX, a standard language that aims to enable integration, exchange, visualization and analysis of biological pathway data.

PC currently includes data from Reactome, NCI PID, HumanCyc, PhosphoSitePlus and PANTHER—data resources that already export their data in BioPAX format. It further imports data from HPRD by taking advantage of a tool that converts data from PSI-MI to BioPAX format.

Although there are many other BioPAX-supporting data resources, PC currently lacks biological pathway data about drug activity, transcription factor mediated events and detailed metabolism reactions. Data for such biological processes already exist and publicly available from various resources, but inclusion of these databases into PC requires converters that will convert these data sets to BioPAX.

All PC data is freely available, under the license terms of each contributing database. This allows PC to combine and re-distribute databases that utilize different databases and when necessary (e.g. if the data provider does not originally allow re-distribution of the data) permission from the data provider is granted before importing it into PC.

Project Goals

Pathway Commons aggregates biological pathway information from several pathway databases; the data are stored primarily in the format known as BioPAX. The PC database currently includes data from resources that already provide data in BioPAX format, such as Reactome and HumanCyc. The aim of this project is to extend Pathway Commons framework by implementing importers/converters for other data resources that do not provide their data in BioPAX but are of high interest to biologists.

Goal 1: Recon 2 Converter

Proposal

Although there already exists an SBML-to-BioPAX converter, the produced BioPAX does not validate via the official BioPAX Validator and contain semantic errors, hindering its

Recent activity


CTD2BioPAX: avoid '+' sign in URIs
Issue #8 created in armish/gsoc14
rodche · 2016-04-10


CTD2BioPAX: don't create all-in-one e…
Issue #7 created in armish/gsoc14
rodche · 2016-04-10


6912e6e
Commit commented on in armish/gsoc14
rodche · 2015-06-23


1 commit
Pushed to armish/gsoc14
e4a1f83 goal2: ctd now creates RelXref in…
B. Arman Aksoy · 2015-03-13


1 commit
Pushed to armish/gsoc14
b02edbd goal2: we don't create uxrefs for n…
B. Arman Aksoy · 2015-03-13


Goal 3: Acetylation events are not nec…
Issue #5 updated in armish/gsoc14
B. Arman Aksoy · 2015-03-13


1 commit
Pushed to armish/gsoc14
65841b5 goal3: fixes issue ~~#5~~.
B. Arman Aksoy · 2015-03-13


Goal 4: MirTarBase problematic xrefs ids
Issue #4 updated in armish/gsoc14
B. Arman Aksoy · 2015-03-13


1 commit
Pushed to armish/gsoc14
2ff3f51 goal4: fixes issue ~~#4~~.
B. Arman Aksoy · 2015-03-13


Goal 4: MirTarBase problematic xrefs ids
Issue #4 commented on in armish/gsoc14
B. Arman Aksoy · 2015-03-13


Goal 2 (CTD): Some pathways are mis…
Issue #6 updated in armish/gsoc14

import into PC. For this part of the project, I will fix the SBML-to-BioPAX converter and make sure that it produces a valid BioPAX file with proper external identification information.

- **Home page**: http://humanmetabolism.org (also see Thiele *et al.*, 2013)
- **Type**: Human metabolism
- **Format**: SBML (Systems Biology Markup Language)
- **License**: N/A (Public)

Implementation details

The existing converter was originally written for converting SBML 2 models into BioPAX and obviously was extended later to support BioPAX L3 as well. This being said, the converter was not making good use of all Paxtools utilities that can make the code much simpler and cleaner. I first tried to modify the existing code, but stuck with library conflicts and was not able to resolve the problems. See the initial changesets starting from tag `base1` till `milestone1.1`.

To keep things much simpler, I created a new project from the scratch under `Goal1-SBML2BioPAX/sbml2biopax`. This project depends on two libraries: Paxtools and JSBML. I implemented the converter so that this project can be used a library by other projects as well. The main class of this project, `SBML2BioPAXMain`, serves as an example to show how to use this API:

```
// ...
SBMLDocument sbmlDocument = SBMLReader.read(new File(sbmlFile));
SBML2BioPAXConverter sbml2BioPAXConverter = new SBML2BioPAXConverter();
Model bpModel = sbml2BioPAXConverter.convert(sbmlDocument);
// where bpModel is the BioPAX model
// ...
```

During implementation, I tried to seperate utility methods and main flow as much as possible, so that we have all main conversion logic in the `SBML2BioPAXConverter` class and all utility methods in the `SBML2BioPAXUtilities`.

The logic of the conversion is as follows:

1. Load SBML document
2. Get the parent model in the document
3. Convert SBML::model to BioPAX::Pathway
4. Iterate over all reactions within SBML::model
    1. Convert SBML::reaction to BioPAX::Conversion
    2. Convert all SBML::modifiers to this reaction into BioPAX::Control reactions
    3. Convert all SBML::reactants to BioPAX::leftParticipants
    4. Convert all SBML::products to BioPAX::rightParticipants
    5. If SBML::reaction::isReversible, make BioPAX::Conversion reversible as well
    6. Add all reactions to the parent pathway
5. Fix outstanding issues with the model and complete it by adding missing components

One key thing with this conversion is that, often, external knowledge is required to decide which particular BioPAX class to create. For example, an SBML::species can be a BioPAX::Complex, Protein, SmallMolecule and *etc.* Or you can have SBML::reactions as BioPAX::BiochemicalReaction or BioPAX::Transport. To make these distinctions, this implementation uses SBO Terms used in Recon 2 model. The good news is that SBO terms serve as a nice reference; and the bad news is that not all SBML models have these terms/annotations associated with SBML entities.

Due to these issues, the current implementation is coupled to the Recon 2 model. Although it is possible to convert any other SBML model into BioPAX, the semantics might suffer depending on the annotation details in that particular model.

Usage

After checking out the repository, change your working directory to the Goal1-SBML2BioPAX/sbml2biopax:

```
$ cd Goal1-SBML2BioPAX/sbml2biopax
```

To compile the code and create an executable JAR file, run ant:

```
$ ant
```

You can then run the converter as follows:

```
$ java -jar out/jar/sbml2biopax/sbml2biopax.jar
> Usage: SBML2BioPAX input.sbml output.owl
```

To test the application, you can download the Recon 2 model either from the corresponding BioModel page or from this project's download page: goal1_input_recon2.sbml.gz. The following commands, for example, convert this file into BioPAX:

```
$ wget https://bitbucket.org/armish/gsoc14/downloads/goal1_input_recon2.sbml.gz
$ gunzip goal1_input_recon2.sbml.gz
$ java -jar out/jar/sbml2biopax/sbml2biopax.jar goal1_input_recon2.sbml goal1_outpu
```

For sample output, you can check goal1_output20140529.owl.gz.

Validation results

The validation report for the converted model is pretty good and include only a single type of `error` due to the lack of annotations to some entities in the SBML model. The HTML report can be accessed from the `Downloads` section: goal1_sbml2biopax_validationResults_20140529.zip. The outstanding error with the report is related to `EntityReference` instances that don't have any `UnificationXref`s associated with them. This is not an artifact of the conversion, but rather a result of the lack of annotations in the Recon 2 model, where some of the `SmallMolecule` species do not have any annotations to them, hence don't have any `UnificationXref`s.

Goal 2: Comparative Toxicogenomics Database (CTD) Converter

Proposal

Unlike many other drug-target databases, this data resource has a controlled vocabulary that can be mapped to BioPAX, for example:

“nutlin 3 results in increased expression of BAX”

Therefore implementation of a converter first requires a manual mapping from CTD terms to BioPAX ontology. Once the mapping is done, then the actual conversion requires parsing and integrating multiple CSV files that are distributed by the provider.

- **Home page**: http://ctdbase.org/
- **Type**: Drug activity
- **Format**: XML/CSV
- **License**: Free for academic use

Implementation details

The converter is structured as a maven project, where the only major dependencies are *Paxtools* and *JAXB* libraries. The project can be compiled into an executable JAR file that can be used as a command line utility (described in the next section).

For the conversion, the utility uses three different input files:

1. Chemical-Gene Interactions (XML)
2. Gene Vocabulary (CSV)
3. Chemical Vocabulary (CSV)

all of which can be downloaded from the CTD Downloads page. User can provide any of these files as input and get a BioPAX file as the result of the conversion. If user provides more than one input, then the converted models are merged and a single BioPAX file is provided as output.

The gene/chemical vocabulary converters produce BioPAX file with only `EntityReference`

s in them. Each entity reference in this converted models includes all the external refernces provided within the vocabulary file. From the chemical vocabulary, `SmallMoleculeReference`s are produced; and from the gene vocabulary, various types of references are produced for corresponding CTD gene forms: `ProteinReference`, `DnaReference`, `RnaReference`, `DnaRegionReference` and `RnaRegionReference`.

The interactions file contains all detailed interactions between chemicals and genes, but no background information on the chemical/gene entities. Therefore it is necessary to convert all these files and merge these models into one in order to get a properly annotated BioPAX model. The converter exactly does that by making sure that the entity references from the vocabulary files match with the ones produced from the interactions file. This allows filling in the gaps and annotations of the entities in the final converted model.

The CTD data sets have nested interactions that are captured by their structured XML file and their XML schema: CTD_chem_gene_ixns_structured.xml.gz and CTD_chem_gene_ixns_structured.xsd. The converter takes advantage of `JAXB` library to handle this structured data set. The automatically generated Java classes that correspond to this schema can be found under src/main/java/org/ctdbase/model. The simple flow that show how the conversion happens is available as the main executable class: CTD2BioPAXConverterMain.java.

Usage

Check out the latest code and change your directory to Goal2-CTD2BioPAX/ctd2biopax:

```
$ cd Goal2-CTD2BioPAX/ctd2biopax
```

and do a clean mvn install:

```
$ mvn clean install assembly:single
```

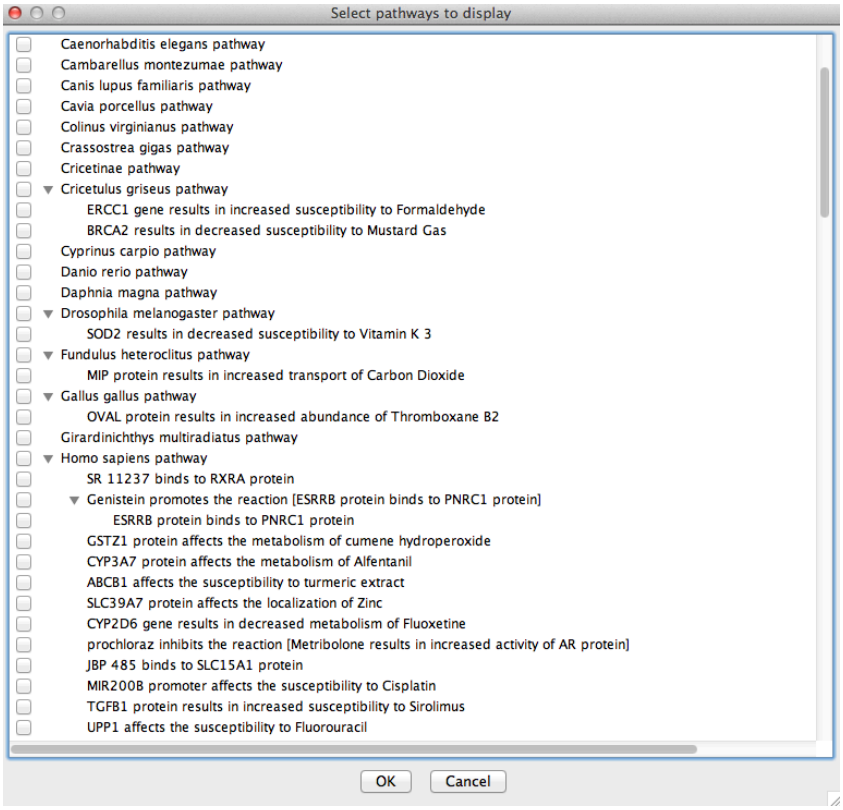
This will create a single executable JAR file under the `target/` directory, with the following file name: `ctd2biopax-{version}-single.jar`. You can also download this file under the downloads, e.g. ctd2biopax-1.0-SNAPSHOT-single.jar. Once you have the single JAR file, you can try to run without any command line options to see the help text:

```
$ java -jar ctd2biopax-1.0-SNAPSHOT-single.jar
usage: CTD2BioPAXConverterMain
 -c,--chemical <arg>      CTD chemical vocabulary (CSV) [optional]
 -g,--gene <arg>          CTD gene vocabulary (CSV) [optional]
 -o,--output <arg>        Output (BioPAX file) [required]
 -r,--remove-tangling     Remove tangling entities for clean-up [optional]
 -t,--taxonomy <arg>      Taxonomy (e.g. '9606' for human) [optional]
 -x,--interaction <arg>   structured chemical-gene interaction file (XML)
                          [optional]
```

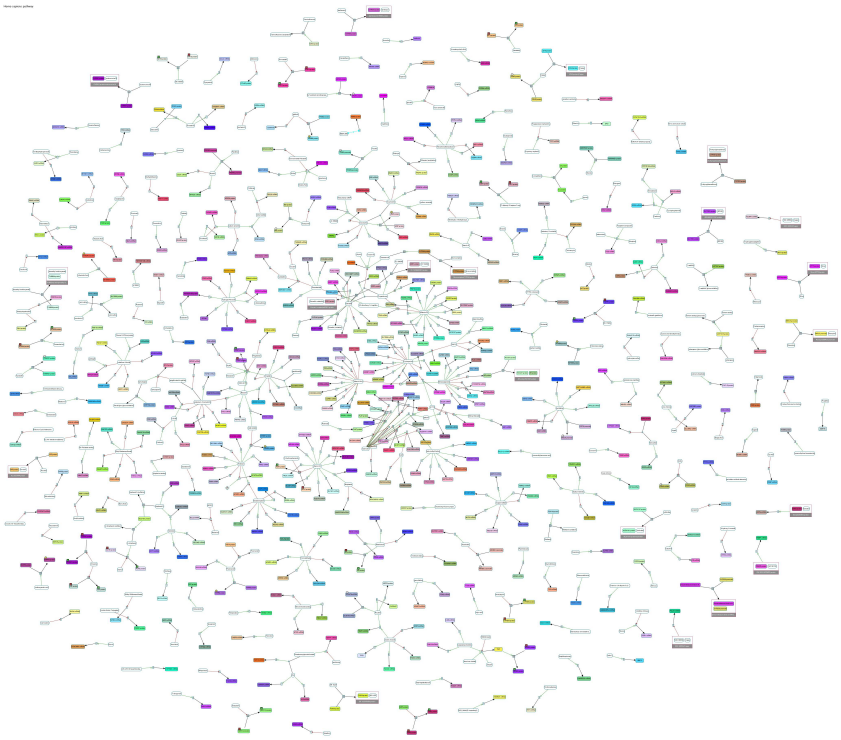
All input files (chemicals/genes/interactions) can be downloaded from the CTD Downloads page. If you want to test the converter though, you can download smallish examples for all these files from the downloads page: goal2_ctd_smallSampleInputFiles-20140702.zip. To convert these sample files into a single BioPAX file, run the following command:

```
$ java -jar ctd2biopax-1.0-SNAPSHOT-single.jar -x ctd_small.xml -c CTD_chemicals_sm
```

which will create the `ctd.owl` file for you. You can find a sample converted BioPAX file from the following link: goal2_ctd_smallSampleConverted-20140703.owl.gz. Once you have the file, you can then visualize this small sample file with ChiBE, which will list all available pathways in the model first.



and you can, for example, load the `Homo sapiens` pathway and this is what you will get:



The tool will also print log information to the console, for example: goal2_ctd_smallSampleConversion.log.gz.

If you'd like, you can download the fullly converted CTD data and its log from the following links: goal2_ctd_fullConverted.owl.bz2 and goal2_ctd-fullConversion.log.gz.

Validation results

Since the fully converted CTD model is huge (> 4 Gb), I only validated the small sample data set, which is representative of the full one: goal2_ctd_validationResults_20140703.zip. In the validation reports, we have a single

type of `ERROR` that reports the lack of external references for some of the `EntityReference`s. These are mostly due to lack of information in the sample chemical/gene vocabularies and are not valid for the full CTD data set -- which has all the necessary background information on all entities.

Original CTD model assumes all entities are forms of `Gene`s, hence provides unification xrefs to the *NCBI Gene* database for all entities. This creates a problem in the converted BioPAX file, where we add gene xrefs to the protein entities for some of the CTD reactions. This also causes some of the unification xrefs to be shared across entities (e.g. `DNA` and `Protein`). The options to get rid of these problems will be discussed with the mentors/curators.

These issues that are related to the data source are all entered to our Issue Tracker.

Goal 3: DrugBank converter

Proposal

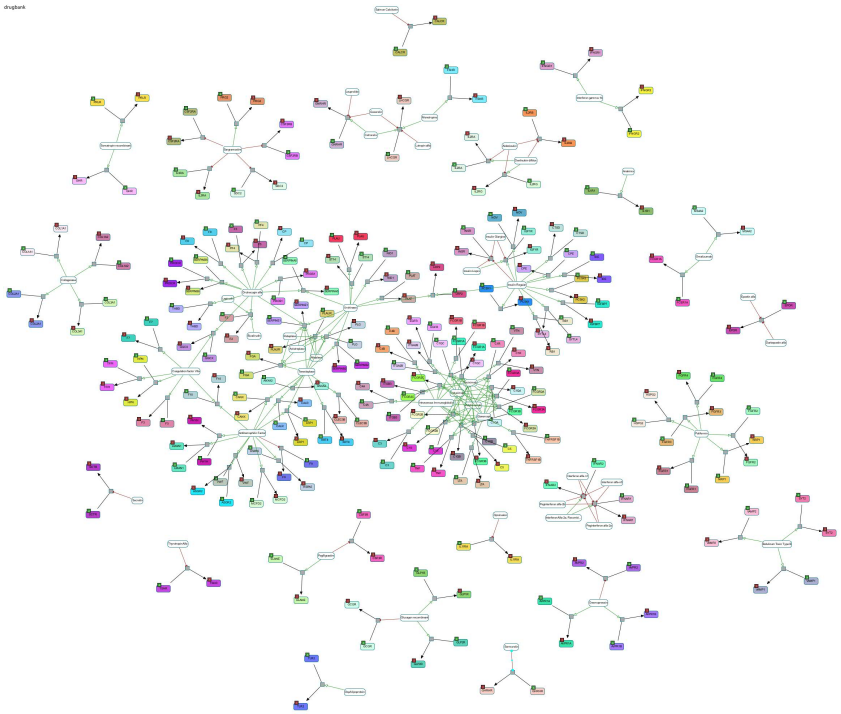
Most of the drug-target interactions in DrugBank do not contain detailed process information—i.e. how the drug actually potentiates/inhibits the protein. Although their XML data export is easy to parse, conversion of this to BioPAX will require discussion with Pathway Commons and deciding on the best way to represent these interactions.

- **Home page**: http://www.drugbank.ca
- **Type**: Drug activity
- **Format**: XML (DrugBank XML Schema)
- **License**: Free for academic use

Implementation details

DrugBank XML file contains sufficient information to create `SmallMolecule` and `Protein` instances with proper external identification to them. There are multiple types of relationships between `SmallMolecule`s and `Protein`s presented in this database: Transporters, Targets, Enzymes. For the purpose of this project, we are interested in capturing drug-target relationships, hence we parse only this information and convert it to BioPAX `BiochemicalReaction`s. These binary relationships do not contain structured mechanism information, e.g. how a drug inhibits or potentiates a target. We have decided to encode this information in BioPAX by using `SequenceModificationFeature`s with `active` and `inactive` terms and associating these features with target proteins. Therefore the final BioPAX model contains `BiochemicalReaction`s where drugs regulate the reaction and the participant protein either gets activated. The type of regulation is based on the controlled vocabulary adopted by DrugBank. If the interaction type is one of the following, then we represent that as a negative regulation, meaning that drug inhibits the inactivation of the protein (double negative): substrate, agonist, inducer, potentiator, stimulator, cofactor or ligand. Otherwise, the drug positively regulates the inactivation reaction.

The following screenshot, for example, shows positive (green edges) and negative (red edges) regulations by some drugs:



For some drugs, the XML file also contains information about how the drug is metabolized by various enzymes, but we currently do not capture this in the final model. The reason we are not doing this is partly due to incomplete knowledge (especially regarding to the intermediate chemicals) and partly due to the fact that SMPDB knowledgebase already has these in BioPAX.

Usage

Check out the latest code and change your directory to Goal3-DrugBank2BioPAX/drugbank2biopax:

```
$ cd Goal3-DrugBank2BioPAX/drugbank2biopax
```

and do a clean mvn install:

```
$ mvn clean install assembly:single
```

This will create a single executable JAR file under the `target/` directory, with the following file name: `drugbank2biopax-{version}-single.jar`. You can also download this file under the downloads, e.g. goal3_drugbank2biopax-1.0-SNAPSHOT-single.jar. Once you have the single JAR file, you can try to run without any command line options to see the help text:

```
$ java -jar goal3_drugbank2biopax-1.0-SNAPSHOT-single.jar
usage: DrugBank2BioPAXConverterMain
 -d,--drugs <arg>    structured DrugBank data (XML) [required]
 -o,--output <arg>   Output (BioPAX file) [required]
```

The only input file required for this conversion is the XML file that is distributed by DrugBank. You can also download this file from the project page: goal3_drugbank_20140730.xml.zip. Once downloaded and unzipped, you can then convert the model as follows:

```
$ java -jar goal3_drugbank2biopax-1.0-SNAPSHOT-single.jar -d goal3_drugbank_2014073
```

The fully converted BioPAX model is available under Downloads: goal3_drugbank_20140730.owl.gz.

Validation results

The full validation report is available under the Downloads: goal3_drugbank_validationResults_20140730.zip. The converted model does not have

errors, but it produces warnings for few known cases.

The first noticable problem has to do with the `active` and `inactive` terms of the modicification features. These terms are not registered in Miriam, but they have been used in BioPAX models, especially in NCI-PID. Ideally instead of these terms, we encode the activation reaction with all mechanistic details; but since this information is not available, we have to ignore these warnings for the time being.

Another problem that validator captures has to do with the external links. During conversion, we create `RelationshipXref`s for `SmallMolecule`s as they are provided by DrugBank. The database names for some of these links are not registered in Miriam, hence we get `unknown.db` warnings in the validation report. One way to deal with this is to manually fix the database names to match them with the existing Miriam identifiers, but ideally we will work with DrugBank to get these names standardized.

Goal 4: MiRTarBase converter

Proposal

This data resource is manually curated and it contains validated miRNA-target interactions. These interactions can easily be converted to BioPAX format.

- **Home page**: http://mirtarbase.mbc.nctu.edu.tw
- **Type**: microRNA-target interactions
- **Format**: XLS/TSV
- **License**: Free for academic use

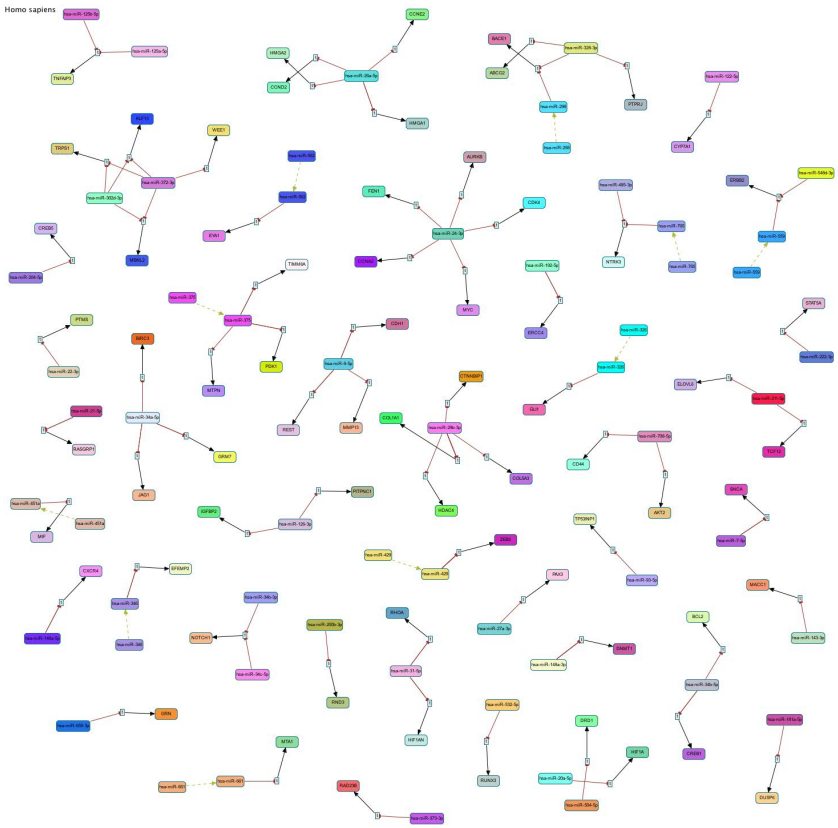
Implementation details

The database provides downloadable miRNA-target relationships in Excel format. For each interaction, we have knowledge about the miRNA, the target gene, the organism and the coresponding publication that describes the interaction. Each miRNA has a unique MiRTarBase ID, but these IDs are registered in the Miriam database. This creates a problem for us when creating miRNA references with `UnificationXref`s to them. To overcome this problem, we also import miRBase aliases which provides official unique IDs for given miRNA names.

We encode miRNA-target relationships with `TemplateReaction`s through which the corresponding gene product is produced and the reaction is inhibitied by the corresponding miRNA. miRNAs are represented by `Rna` objects, where they have `RelationshipXref`s to MiRTarBase and `UnificationXref`s to miRBase. If a miRNA name is associated with multiple unique miRNAs, then we capture this information via adding different-named miRNAs as a `MemberPhysicalEntity` to the original miRBase reference.

We encapsulate each interaction in an organism-specific pathway so that users can only work with pathways that are of interest to them, for example human miRNA-targets.

The image below shows a partial `Homo sapiens` pathway:



Usage

Check out the latest code and change your directory to Goal4-miRTarBase/mirtarbase2biopax:

```
$ cd Goal4-miRTarBase/mirtarbase2biopax
```

and do a clean mvn install:

```
$ mvn clean install assembly:single
```

This will create a single executable JAR file under the `target/` directory, with the following file name: `mirtarbase2biopax-{version}-single.jar`. You can also download this file under the downloads, e.g. goal4_mirtarbase2biopax-1.0-SNAPSHOT-single.jar. Once you have the single JAR file, you can try to run without any command line options to see the help text:

```
$ java -jar goal4_mirtarbase2biopax-1.0-SNAPSHOT-single.jar
usage: MiRTarBase2BioPAXConverterMain
-m,--mirbase-aliases <arg>      miRNA aliases from mirBase (txt)
                                [optional]
-o,--output <arg>               Output file (BioPAX) [required]
-r,--remove-tangling            Removed tangling Rna objects [optional]
-t,--mirtarbase-targets <arg>   miRTarBase curated targets (XLS)
                                [optional]
```

For a full conversion, two input files are required:

1. MiRTarBase Excel file (either full or partial, e.g. human)
2. miRBase aliases (download)

both of which can be downloaded from the repository.

Once downloaded and gunzipped, these can be coverted into BioPAX via the following command:

```
$ java -jar goal4_mirtarbase2biopax-1.0-SNAPSHOT-single.jar -m goal4_mirbase_aliase
```

The `-r` switch is optional, but helps reduce the size of the final model. When provided, this makes sure the final model does not included `Rna` and `RnaReference` objects that do

not participate in a reaction.

You can download the miRNA-target relationships as a BioPAX file either for all organism (goal4_output_all_mirna-20140731.owl.gz) or only human (goal4_output_human_mirna-20140731.owl.gz)

Validation results

The fully converted model is too big to be validated via the web, but the validation results for the human interactions are available under the Downloads: goal4_mirtarbase_validationResults_20140731.zip.

We don't have any validation errors, but since we are identifying `Protein`s with `Entrez Gene ID`s and there can be multiple proteins associated with a single gene. This is not the best practice and this is why we have *denied xrefs* in the report. To fix this, we can bring in new background files, for example Gene -> Uniprot mappings, into the conversion tool, but since Pathway Commons already has these utilities in place, I think we can leave this mapping issue to the PC integration pipeline for now.

Goal 5: MSigDB converter

Proposal

This type of data sets are available from different resources, such as TRANSFAC, JASPAR and ENCODE projects; but converting the data provided by these databases will require additional processing—e.g. mapping ChIP-Seq peaks to near-by genes or finding genes by binding motifs.

The easiest way, however, to import this data type is to use MSigDB, where transcription factors and their predicted targets are present in GSEA format provided by TRANSFAC. These associations can be converted to BioPAX by transcription events regulated by corresponding transcription factors.

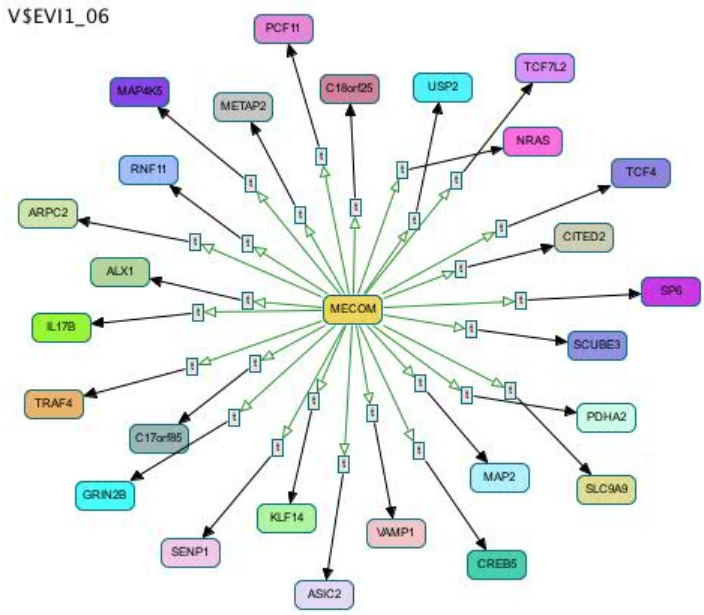
- **Home page**: http://www.broadinstitute.org/gsea/msigdb/collections.jsp
- **Type**: Transcription factor - target
- **Format**: XML (MSigDB format)
- **License**: Free for academic use (also see MSigDB license)

Implementation details

MSigDB provides data in various formats, the most commonly used being a tab-delimited format, GMT or GSEA, to describe a gene set. It also provides downloadable files in XML format where full description of gene sets can be obtained in a single file. This converter, with the help of the Java-based GSEA software as a library, parses the XML to obtain information about gene sets that describe transcription factors and their targets as gene sets. For this, we are specifically extracting information from the `C3` category, specifically the `TFT` subcategory of it.

The genes in the gene sets are identified either via their symbols or Entrez Gene IDs. Although unstructured, the name of the transcription factor associated with each gene set is present in the description of the gene set. This converter has a built-in HGNC utility that helps better map the gene identifiers to full BioPAX `RnaReference`s. For each TFT gene set, it creates a separate pathway identified by the unique name, and within this pathway, the transcription factor positively regulates the transcription of targets listed in the corresponding gene set. This information is captured via `TemplateReaction`s where target `Rna`s are being produced and the reaction itself is being regulated by the transcription factor via `TemplateReactionRegulation`.

Below is a screenshot that shows a sample gene set converted into a `Pathway`:



Usage

After checking out the repository, change your working directory to the Goal5-MSigDB/msigdb2biopax:

```
$ cd Goal5-MSigDB/msigdb2biopax
```

To compile the code and create an executable JAR file, run ant:

```
$ ant
```

You can then run the converter as follows:

```
$ java -jar out/jar/msigdb2biopax/msigdb2biopax.jar
Usage: MSigDB2BioPAXConverterMain /path/to/msigdb_v4.0.xml /path/to/output.owl
```

or directly download the executable JAR: goal5_msigdb2biopax-20140802.jar.

For the conversion, you need to download the MSigDB database as an XML file: msigdb_v4.0.xml. Once downloaded, you can convert this into BioPAX as follows:

```
$ java -jar out/jar/msigdb2biopax/msigdb2biopax.jar msigdb_v4.0.xml msigdb_v4.0.owl
```

You can download the converted model from the Downloads: goal5_msigdb_c3_tft-20140802.owl.gz.

Validation results

The validation report for the fully converted model is available here: goal5_validationResults-20130802.zip. The BioPAX model does not have any major validation warnings or errors, hence is pretty clean.