RBA.prerba

Biosys - MAIAGE

December 8, 2018

Contents

1 Description of workflow

Figure 1 shows the global workflow of the pipeline. It contains four parts:

1. **RBA.prerba**: Parsing of biological data into RBA compatible XML files. Parts of the process are semi-automated, meaning the user is needed to help solve ambiguous annotations.

2. **RBA.xml**: Parsing/modifying XML files.

3. **RBA.core**: Transforming XML files into matrices and computing optimal resource allocation.

4. **RBA.estim**: Estimation of catalytic constants.

This document focuses on the first part of the package, RBA.prerba.

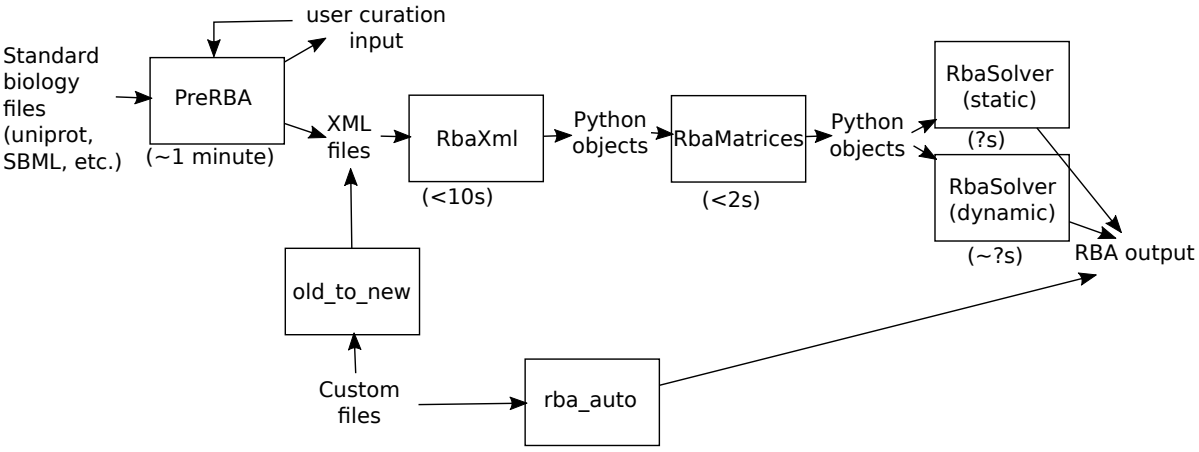

Figure 1: Workflow of pipeline.

1.1 Philosophy of the pipeline

Everything should work from first run on Everything should work on the first run. When a user inputs an approximately standard SBML file (see requirements below), RBA.prerba should generate a full RBA model and RBApy should output results, including optimal growth rate and flux distributions. Because Uniprot annotations are often ambiguous, numerous default parameters will be used for the first run, but the user will be able to parameterize their model progressively.

Helping user to setup main parameters through tsv files Conversion of biological data is very heavy and often ambiguous. When an annotation is ambiguous, the user will be asked for help through tsv (tab-separated values) files. Everything that is input by the user is stored so the user does not have to provide the same information twice.

Helping user to setup fine parameters through XML files After the main parameters have been set, the user will have ready-to-use RBA input files. We cannot handle every single parameter through tsv files. Because RBA input files are written in XML, an advanced user will still be able to control more subtle parameters in a standardized (but programmatical) way.

1.2 Typical expected usage

1. The user provides an SBML file and runs the pipeline. They are happy because everything runs and they get some output value. They are unhappy because this output value is unrealistic.

2. The user spends time going through the helper files and understands why they are here. They provide all the information needed to create a system fully adapted to their organism. They run the whole pipeline to see how growth rate has evolved and generate new and more consistent XML input files.

3. They spend time fine-tuning processes and enzyme catalytic efficiencies by modifying the XML files, finally reaching a biological sound model.

1.3 IMPORTANT WARNING: Every time RBA.prerba is run, the previous XML model is overridden!

It is possible to modify the XML files manually, but we strongly discourage this option. When RBA.prerba is run, the XML model will be overwritten: all manual changes will be lost! There are two ways to work with RBA models:

- (Not recommended) Start by working on the model by modifying tsv files only. tsv files are an integral part of RBA.prerba: they will not be overwritten, so feel free to regenerate the model as often as you want. Once you are sure that you have modified everything you needed in tsv files, generate an XML model using RBA.prerba for the last time. Then you can start to modify the XML files, and *never run RBA.prerba again.* If you realize that you need to rerun RBA.prerba, backup all manual modifications!

- (Recommended) Modify XML files using python scripts with RBA.xml (see tutorial). In that way, every time you need to run RBA.prerba, you generate a new XML model, then run the scripts in order to introduce further changes into the XML.

2 params.in: pipeline parameters

This file contains a handful of parameters that are used to initialize the pipeline. We suggest using a preexisting params.in file and modifying it (e.g. using the sample model in the GitHub repository). The parameters are:

- INPUT_DIR: Directory where input files can be found and where helper files will be stored. Can be asolute path or path relative to where program is RUN (!).
- SBML_FILE: Name of SBML file describing metabolism of target organism. Should be located in INPUT_DIR (or provide a path relative to INPUT_DIR).
- ORGANISM_ID: Taxonomy ID of target organism as established by the NCBI.
- OUTPUT_DIR: Directory where files used to run RBA are stored. Absolute path or path relative to where program is RUN (!).
- EXTERNAL_COMPARTMENTS (optional): list of external compartments. All metabolites in the SBML file that end with a suffix in this list will be considered external metabolites.

In order to get the workflow working, the user has to provide an SBML file describing the metabolism of their organism. A Uniprot file is also needed, but it can be retrieved automatically using the ORGANISM_ID. Several helper files will be generated after the first run of the pipeline. The user needs to adapt these files in order to have a biologically relevant model.

3 SBML: extraction of metabolism and enzyme information

3.1 Requirements

1. All metabolites must end with a suffix indicating their compartment. The suffix is separated from the rest of the name by an underscore. In particular, cytosolic metabolites should end with `_c`.
2. Every reaction should contain information about associated enzyme composition. There are two accepted formats:
   - Using the `<fbc:geneProductAssociation>` tag of the Flux Balance Constraints package for SBML.
   - Using the `<notes>` tag containing a GENE_ASSOCIATION field (COBRA style notes). Gene names should be separated using white spaces `' '` or underscores `'_'`. Association of genes should be described by the keywords `or` and `and`.

3. All enzyme associtations must be described in an `or`s of `and`s form. For example, (G1 AND G2) OR (G1 AND G3) is the correct form, as opposed to G1 AND (G2 OR G3).

3.2 Warnings

- The biomass reaction and reactions used to assemble non-metabolites (*e.g.* proteins, RNAs, etc.) are not used in the RBA model. The solver will usually assign them zero fluxes. They may safely be removed from the system.
- If a gene listed in the gene association cannot be retrieved in Uniprot, it will be replaced by an average protein.
- If a gene association is left empty, the pipeline will assume the reaction is spontaneous. If an empty gene association represents an unknown enzyme, add anything in the gene association field (e.g. UNKNOWN), so that the unknown protein will be replaced by an average protein.

3.3 tsv helpe files associated with SBML file

unknown_proteins.tsv This file lists the genes found in gene associations that could not be matched with a protein in Uniprot. By default they are replaced by an average protein in the cytosol. The user is invited to map unknown genes with valid Uniprot ids.

3.4 Detection of external metabolites

In order to determine medium composition, RBA.prerba uses the following rules to detect external metabolites:

- All metabolites that end with a suffix listed in EXTERNAL_COMPARTMENTS (see pipeline parameters).
- All metabolites that have `boundaryCondition` set to `true` are considered external.
- If all metabolites from a compartment participate only in reactions where they are the unique metabolic species involved ($M \rightarrow$ or $\rightarrow M$), the compartment is considered external.

3.5 Detection of transporters

By default, all enzymes have a constant catalytic activity. In this context, constant means that the catalytic activity does not depend on growth rate.

An enzyme is considered to be a transporter if the following rules apply:

- One of the products has the same prefix as an external metabolite (*e.g.* `M_nad_e` has the same prefix as `M_nad_p`).
- One of the reactants is in the cytosol.

The catalytic activity of a transporter is modified in the following way.

- The main catalytic activity is given by the default catalytic activity applied to all enzymes.
- The main catalytic activity is multiplied by substrate-dependent terms ranging from 0 to 1. More precisely, import activity is given by a Michaelis-Menten function that depends on the concentration of the *external* counterpart of the imported product. (*e.g.* if the imported product is `M_nad_p`, the import activity depends on `M_nad_e`).

Note that the latter choice may lead to non-intuitive behaviors. For example, take a bacterium that is able to transform trehalose into glucose in the *periplasm*. Suppose the external concentration of trehalose is set to 0 but the external concentration of glucose is nonzero. The solver might decide to import the (non-existing) trehalose into the periplasm (because importing into the periplasm does *not* depend on medium concentrations), transform it into glucose that will automatically be assumed to be at medium concentration and then import into the cytoplasm.

4 Uniprot: extraction of protein information

4.1 Requirements

A Uniprot file is needed to cross-reference proteins with SBML data listed in gene associations. The user needs to provide the Uniprot id of its organism (ORGANISM_ID), so a Uniprot file can automatically be retrieved. Alternatively, the user can provide a Uniprot file matching following requirements:

- The file must be located in the INPUT_DIR, with the exact name `uniprot.csv`.
- Required fields are: Entry, Gene names, Protein names, Sequence, Cofactor, Subcellular location [CC], Subunit stucture [CC].

4.2 tsv helper files associated with Uniprot

location.tsv This file contains proteins with ambigous (often empty) location. By default, all these proteins will be located in the compartment `Cytoplasm`. The user is invited to update the location to one of the locations listed in location_map.tsv.

location_map.tsv By default, the compartments in the RBA model will be the compartments found in Uniprot annotations (such as Cytoplasm). This file lists all the unique compartments that have been extracted by RBA.prerba. The user is invited to rename or merge the compartments. For example, if two Uniprot compartments `Cell_Membrane` and `Membrane` are mapped to the same identifier `membrane`, all proteins associated with `Cell_Membrane` and `Membrane` in Uniprot will be pooled into the same RBA compartment `membrane`.

cofactors.tsv This file lists all the protein-cofactor associations that were found in Uniprot. These associations are essential to determine the full composition of proteins and enzymes. We apply automated rules to infer cofactor stoichiometry (see below). The user is invited to check the automated extraction and replace default values. The user may also add a new line at the end of the file in order to create a new association.

subunits.tsv This file lists the stoichiometry of proteins within their preferred enzymatic complex. This information is essential to determine the exact composition of enzymes. We apply automated rules to infer protein stoichiometry (see below). The user is invited to check the automated extraction and replace default values.

4.3 Automated parsing rules

Protein composition in amino acids Protein composition in amino acids is determined from the `Sequence` field by counting letters.

Cofactor stoichiometry From the uniprot `Cofactor` field, we use the following rules to parse protein cofactor information:

- If field is empty, we assume there is no cofactor.
- If there is exactly one occurrence of the keyword `Binds`, we assume stoichiometry is the number that follows `Binds`.
- If there is no stoichiometry information using keyword `Binds`, we assume stoichiometry is 1.
- If there were several names and associated CHEBI identifiers, we assume that only the first cofactor listed is relevant. We give it full stoichiometry and 0 stoichiometry to all other cofactors.

Note that the cofactors are listed in cofactors.tsv in all cases.

Subunit structure From the uniprot `Subunit stucture [CC]` field, we use the following rules to parse the stoichiometry of proteins within their enzymatic complex:

- If field is empty, we assume that the stoichiometry is 1.

- If field contains exactly one occurence of the form "*prefix*mer", we look at the prefix. If prefix is mono or heterodi, we assume stoichiometry is one. If prefix is homodi, homotri, homotera, homopenta, homohexa, hepta, homoocta, homodeca, homododeca, we assume the number of subunits corresponds to the prefix.
- In any other case, field is considered ambiguous and a default stoichiometry of 1 is applied.

Location From the uniprot `Subcellular location [CC]` field, we use the following rules to parse location information.

- If field is non-empty, we use the field as the protein's location.
- If field is empty, the protein is assumed to be in the compartment Cytoplasm.

4.4 Summary of protein information used to build model

protein_summary.tsv contains all the information that was retained during RBA.prerba's last run. This is an output file! Modifying it will not change RBA.prerba's outcome!

5 Generation of default processes

In order to produce a fully functional model, RBA.prerba automatically includes 4 important processes: translation, folding, transcription and RNA degradation.

5.1 FASTA files containing tRNAs and ribosome structure

RBA.prerba needs to know the structure of the ribosome and the sequences of the tRNAs in order to create the default processes. RBA.prerba expects to find this information in the files `ribosome.fasta` and `rnas.fasta` within the INPUT_DIR. In the sample data, we provide two default FASTA files containing the ribosome and the tRNAs of the model bacterium *Escherichia coli*. The user is invited to replace these files with ribosomal proteins, rRNAs and tRNAs from their organism.

5.2 tsv helper files associated with processes

metabolites.tsv This file is used to map all the metabolites that are used in the default processes with species identifiers from the SBML. This mapping will make sure that the processes work as intended. For example, charged tRNAs are normally consumed during translation: these charged tRNAs must be produced by the metabolic network described in the SBML file. If charged tRNAs are not mapped to SMBL identifiers, the translation process will still produce proteins, but no charged tRNA will actually be consumed or produced by the metabolism. Failing to map metabolites will yield extremely unrealistic

growth rates. The user is invited to fill in the SBML identifiers of all metabolites that could not be mapped automatically.

The second purpose of metabolites.tsv is to specify the concentration of metabolites, which will be discussed in the next session.

6 Generation of default targets

Targets are used to place demands on the metabolism: they play an essential role in shaping the growth rate and the distribution of fluxes. If there are no targets to achieve, there is no need to produce enzymes or ribosomes and nothing happens. Typical targets include DNA, RNAs, the cell wall, housekeeping proteins, but also maintaining essential metabolites at high enough concentrations for the metabolism to work properly. Targets are usually specified as concentrations to maintain. When a target, e.g. a metabolite, is added, it must be produced de novo by the metabolism in order to keep the concentration constant in a growing cell. The solution of the RBA model will show evidence that the production pathway for this target is active.

6.1 tsv helper files associated with targets

metabolites.tsv This file is used to map metabolites to SBML identifiers (see above), but also to specify target concentrations for essential metabolites. The user is invited to fill in concentrations wherever possible and add new target concentrations on new lines.

macrocomponents.tsv This file is used to specify all other target concentrations. Because this is the file's only role, it is much simpler than metabolites.tsv. The user is invited to add additional targets by specifying SBML identifiers and target concentrations on new lines.

7 In practice, what do I need to modify?

There are many helper files to modify in RBA.prerba. However, some files are more important than others. In our experience, these are the changes that have the most dramatic effect on the final RBA model:

- metabolites.tsv: mapping metabolites like tRNAs to SBML identifiers is the single most important change to do. Otherwise no tRNAs will be consumed during translation and the growth rate will be strongly overestimated.
- unknown_proteins.tsv: usually not dramatic, but it's important to check what genes have not be mapped to Uniprot and how they were replaced.
- location_map.tsv: renaming/merging Uniprot compartments to match expected compartments.

- SBML file: in some instances, the RBA model was not solvable after mapping metabolites and targets because a metabolite or target could not be synthesized de novo by the metabolism. In these cases, the proper pathway must be added to the SBML model.

Note that the RBA model contains one tsv files called medium.tsv containing the concentrations of external metabolites. By default, RBA.prerba creates a rich medium containing all external metabolites, usually yielding large growth rates. Creating a more realistic medium will help evaluate how good the current model is. Remember that RBA.prerba will override all XML files and medium.tsv every time it needs to be rerun. We advise to create a realistic medium along with a backup version, e.g. curated_medium.tsv. Every time RBA.prerba is run, don't forget to replace medium.tsv with curated_medium.tsv