ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

On modeling a bio-chemical process, from models to simulations

Julien Eperon

Master Thesis

March 2006

Professor
Jean-Yves Le Boudec
LCA/EPFL

Assistant
Irina Baltcheva
LCA/EPFL

Contents

1 Introduction

1.1 Purpose of this project

This project attempts to provide a solution that allows biologists and computer scientist to collaborate in modeling biological processes. The first step of this was to standardize the representation of such processes, which would allow the most complete and detailed description of such model, but which also allows to be strict enough to automate simulation and visualization.

1.2 Biological Processes

This report assumes that the reader knows a little about biology, but it is necessary to specify which aspects we are interested in.

Like any process, biological processes are interactions between different components, the different components being cells or proteins, and the reactions being mainly chemical reactions. In the formalism we used in this approach, we called the different components of the process species, and the interactions between these components reactions.

Where species are described by their names and quantities, reactions are links between species and characterized by the rate at which they occur. In a reaction, a species can be either a reactant, a modifier or a product. A species that is a reactant is consumed by the reaction whereas the product is indeed produced, the modifier is neither consumed nor produced, but is a catalyst and modifies the rate at which a reaction occurs.

Putting all these together builds a model. This kind of model naturally leads a graphical representation, like the one of figure 1.

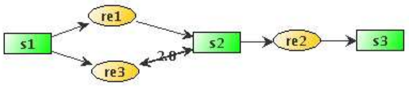


Figure 1: A simple reaction representation

1.3 Systems Biology Markup Language

As such formalism exists for a long time and is widely used across the biologists, it was normal to find an already existing language to describe a biological (but also chemical) process.

SBML, Systems Biology Markup Language, is a free and open language:

“Advances in biotechnology are leading to larger, more complex quantitative models. The systems biology community needs information standards if models are to be shared, evaluated and developed cooperatively. SBML’s widespread adoption offers many benefits, including: (1) enabling the use of multiple tools without rewriting models for each tool, (2) enabling models to be shared and published in a form other researchers can use even in a different software environment, and (3) ensuring the survival of models (and the intellectual effort put into them) beyond the lifetime of the software used to create them." (http://www.sbml.org)

Apart from being an XML based language[1], which is already a good point for building a program around it. The community around the creation of SBML also provides libraries, for various programming languages. for accessing a simple text file written in SBML and also keeps an up-to-date list of programs[5], free or not, using this standard.

SBML is a language of general-purpose. It can describe more than only species and reactions. It can also describe the units used in the model, various additional rules imposed to it and also events occurring within the model, that can be also handled in a simulation.

1.4 Alternatives to SBML

Apart from SBML other languages related to biological systems modeling were developed:

KGML is also a XML based language like the two previous ones. It is a very simple language compared to SBML. It has only 10 kinds of entities, including one that has only the purpose of helping building a graphical representation. This language is also far less used, as it is mainly used for reading the Kyoto Encyclopedia for Genes and Genomes (KEGG, http://www.genome.jp/kegg/). KGML[2] also misses simple fields like stoichiometry which are very convenient (although equivalent fields can be created). For these various reasons we did not choose KGML.

CellML is again a XML based language. It has also in common with SBML version 2 that it supports MathML tags. CellML is adapted to describing the mathematics behind the model in a more general way, while SBML is a format for representing models of biochemical reaction networks which is our interest here. (More info on http://www.cellml.org)

2 The workflow

2.1 Generating the SBML

Although SBML is human readable, it is not intended to be written directly by a computer scientist or a biologist. This is why, we decided to use a new language, that will be used by the biologist to describe the biological process. This language had as a goal, to be close to the way biologist already write their

[1] XML means Extended Markup Language, is a human readable language (i.e. not a binary one) widely used today especially in information systems[10]

[2] KEGG Markup Language

process. With that in mind, it is evident that any XML based language is not suited and the creation of this new language is the best choice. I will give here a brief introduction to it, as this is important to understand to full flow from the writing of a model description to the simulation of it.

This new language developed by Michel Sede is called TSed[8]. It can be written in a simple text file, with the standard text editor of your choice. It is divided in two blocks, in the first one we declare parameters and functions, and in the second one we declare reactions. The parameters are generally the rates at which the following reactions will occurs, and functions can be declared when a simple parameter is not sufficient for describing the rate of a reactions (for example, when a reaction occurs only when a species quantity reaches a given level).

```
declare:

    pi = 3.14159;
    a = 1.0;
    b = 2*pi*a;

    def sum(x,y) {
        x+y
    };

end declare;
```

The reactions are then written in a very simple manner very close to how biologists would do on paper, with reactants on the left side with or without their stoichiometry, then a simple arrow or a bidirectional arrow, then the products with or with out their stoichiometry, and finally a vertical bar followed by the rate of the reaction.

```
body:

    BIRTH -> A | pi;
    B -> DEATH | b;
    A + B -> DEATH | sum(a,b);

end body;
```

Using the TSed program, we can transform the text file written in TSed into a SBML file. The transformation is pretty straightforward, as TSed parameters become SBML global parameters, TSed functions become SBML function definitions, etc. The only non-trivial step is the transformation of the "rate constant" into a SBML kinetic law. Because in SBML a reaction is described by its kinetic law and not its "rate constant". In TSed a reaction has a "rate constant", the kinetic of this reaction is proportional to the concentration of the species raised to the power of its stoichiometry.

A simple reaction in TSed, where the rate constant is k1:

```
P + 2 * A -> PA + A | k1;
```

The kinetic law in SBML would be:

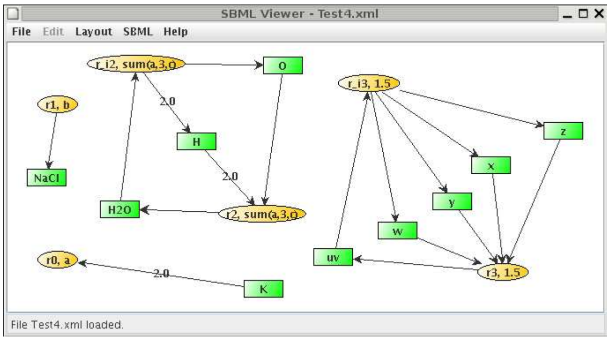
$k1 \cdot P \cdot A^2$

For any details concerning the TSed language, please refer to Michel Sede's report [8].

2.2 Visualizing SBML

Another important step to simplify the verification and the comprehension of an existing model is the visualization of it. In a model where there are species and reactions, we can symbolize species by nodes of a graph and reactions by arrows, but as reactions can have multiple products and multiple reactants, we decided to use to different types of nodes, one for reactions and one for the species and arrows to show the link and the role between then.

This leads to the creation by Laurent Francioli of a graphical interface, visible on figure 2, that allows the biologist (or the computer scientist) to visualize the model of the biological process, and help him validating it.



File Test4.xml loaded.

Figure 2: SView allows a graphical visualization of the SBML file

In the full application developed during this project, this graphical interface plays a central role, as most functions can be called from it. For a full description of the capabilities of this graphical interface, please refer to Laurent Francioli's report [3].

2.3 From SBML to Simulation

Now that we have built the SBML file from a TSed file, or obtained it by another SBML compatible program, we want to simulate it.

SBML can represent multiple kind of model and the biologist might be interested in multiple aspects of it that can studied with the help of simulation. The main problem in which we will be interested here is the simulation of a model given by its reactions and its initial values, although other kinds of problems such as reachability of a certain state of the system, equilibrium or fitting of parameters to a given set of measures can also be solved based on what we will be discussing here.

Given a SBML file, the tools developed during this project will generate two files intended to be used for a Matlab simulation.

The first file, that should be kept untouched, provides a Matlab representation of an ordinary differential equations system of the model given in the SBML file. This means that in this file, we will find all parameters and species, but the reactions will be translated to their corresponding term in the differential equation of the species that they affect.

As each reaction is given in the SBML file with its kinetic law, the transformation of set of species and reactions to an ODEs system is straightforward. These operations have been widely described in literature, you can refer to [1] or [9] for more details on them.

For example, the transformation of a species A into a species B, in TSed:

```
A -> B | a;
```

would lead to the following mathematical form:

$$\frac{dA}{dt} = -a \cdot A \qquad (1)$$

$$\frac{dB}{dt} = +a \cdot A \qquad (2)$$

The corresponding line in a Matlab file:

```
xdot(1) = - a * A;
xdot(2) = + a * A;
```

The second provides a simple Matlab script that allows the user to make a simulation using Matlab's ODE toolbox of the ODEs system provided in the first file.

A various set of tools allows the user to regenerate the two files, with different initial values. The goal is to allow the user to use the ODEs system, with the already existing Matlab tools to observe it and eventually compare it with real data.

Another simple analysis, is the symbolic resolution of the ODEs system. Matlab providing a simple command called "dsolve" to do such a task, the program can generate the entry that the user must provide.

A simple entry for "dsolve", same as the solution of the previous example:

```
DA = - a * A
```

The output of dsolve will be:

```
C1 * exp( - a * t)
```

2.4 The software

The software is intended to be easily integrated with the existing tools of the biologist and the computer scientist. Thus, the entry point is the TSed language which can be edited with any text editor[3].

The next step will be to convert it to SBML, which can be done from a normal command line, or from Matlab. As the user might want to visualize his model directly, the graphical interface, SView[3], also allows to import a TSed file and then save the graph as an SBML file. SView it self can be called from a normal command line or directly from Matlab with the command `SView`.

Finally, generating the Matlab file can also be done from a command line, Matlab or SView.

For all details of these procedures, please refer to the manual provided as appendix.

2.4.1 sbml2latex

As the previously generated ODEs can only be viewed from within the corresponding Matlab file, it was interesting to develop another output that would allow the user to have a complete summary of its model in a simple readable file, that would group the information given in the TSed file and the information generated for Matlab.

Thus, it was decided to provide a way to generate a LaTeXfile that would contain all these information, that the user could integrate to his own publications or reports.

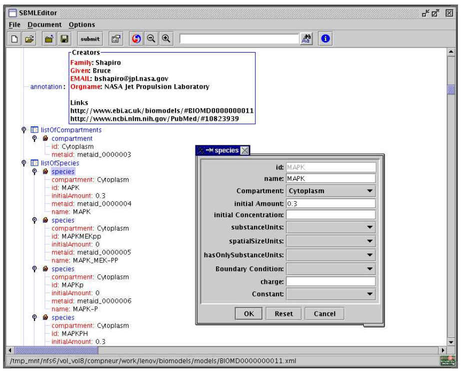
This tool is called `sbml2latex`, all additional details concerning this tools are in the provided manual.

2.5 Other existing tools

SBMLeditor is a good complement to our software. It allows the user to view a SBML, but not in a graph way, simply in a "colored text" way, see figure 3. It can be used when the biologist (or the computer scientist) want to go more in details of a SBML model, it is also valuable when the model was not generated by a TSed file.[6]

CellDesigner is a very complete solution, written in Java, it integrates a graphical view (figure 4) and a simulation part. It is very near the goal that we fixed, but unfortunately it is not a free software, which was one of our prerequisite. It was also unable to open SBML files that it did not generate. Finally, its possibilities were far too wide, and we were looking for simplicity.[2]

[3] For example: `notepad` under windows and `emacs` under Linux, even the default editor provided by Matlab.



```
annotation: Creators
            Family: Shapiro
            Given: Bruce
            EMAIL: bshapiro@jpl.nasa.gov
            Orgname: NASA Jet Propulsion Laboratory

            Links
            http://www.ebi.ac.uk/biomodels/#BIOMD0000000011
            http://www.ncbi.nlm.nih.gov/PubMed/#10823939
listOfCompartments
  compartment
    id: Cytoplasm
    metaid: metaid_0000003
listOfSpecies
  species
    compartment: Cytoplasm
    id: MAPK
    initialAmount: 0.3
    metaid: metaid_0000004
    name: MAPK
  species
    compartment: Cytoplasm
    id: MAPKMEKpp
    initialAmount: 0
    metaid: metaid_0000005
    name: MAPK_MEK-PP
  species
    compartment: Cytoplasm
    id: MAPKp
    initialAmount: 0
    metaid: metaid_0000006
    name: MAPK-P
  species
    compartment: Cytoplasm
    id: MAPKPH
    initialAmount: 0.3
```

id: MAPK
name: MAPK
Compartment: Cytoplasm
initial Amount: 0.3
initial Concentration:
substanceUnits:
spatialSizeUnits:
hasOnlySubstanceUnits:
Boundary Condition:
charge:
Constant:

/tmp_mnt/nfs6/vol_vol8/compneur/work/lenov/biomodels/models/BIOMD0000000011.xml

Figure 3: SBMLeditor allows a low level edition of the SBML file

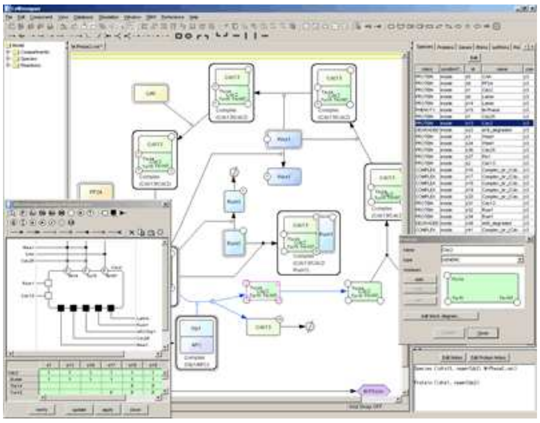


Figure 4: CellDesigner also allows a graphical visualization of the SBML file

System Biology Workbench is a software framework that allows heterogeneous application components-written in diverse programming. It uses SBML as a model exchange format, and try to address the problem of unifying the tools.[7]

2.6 A full example

In order to verify the usability of our tool, we tried it on a real size example. This model describes to dynamics of the pheromone pathways in haploid yeast cells, for details refer to the source article[4].

This model is already of a real size, it has 47 reactions so 47 parameters between 36 species. This leads to 36 differential equations.

The first part was to write this model in a TSed file, which did not state any difficulties because the conventions of this model and the one adopted in TSed were already very close.

The second second step was the visualization. The article was also providing a scheme of the reactions, so that we were able to compare those too:

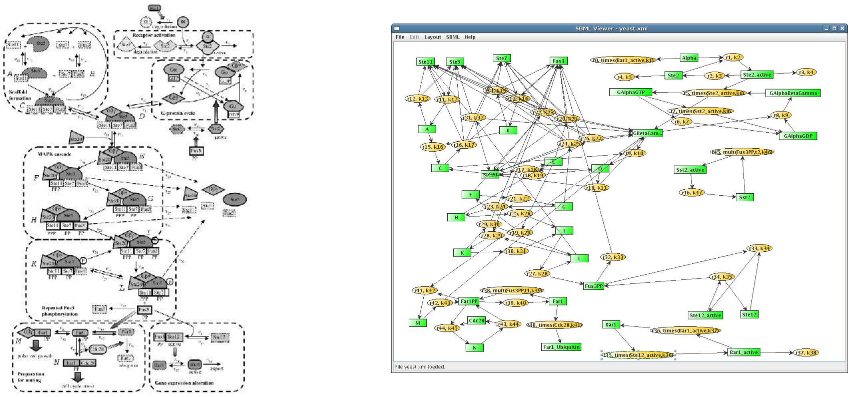


Figure 5: Comparison between the two graph representations

The last step was to simulation and compare the result obtained with the one of the article, we could immediately see that the curves were of the same shape, and comparing the numerical results, approved that the simulation was working correctly.

3 Conclusion

Although their can be some work more on ergonomy, this project shows the simplicity expected combined with the powerfulness of a mathematical software like Matlab to handle the different issues of simulations.

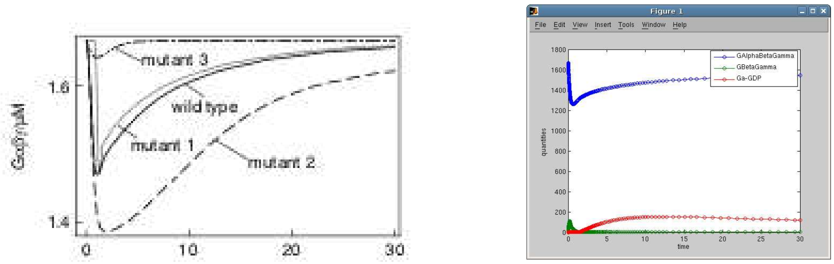


Figure 6: Comparison between the simulation results

This project's difficulties were mostly related to interfacing already existing partial solutions to a problem and regroup them in the same coherent entity. This goes from learning a new descriptive language and implement a translation of this language into another language of mathematical nature. Another difficulty related to this problem already appeared at early stage of this project, when we had to explore already existing solutions and determine their relevantness and usability.

3.1 The delivered product

The final product consist in a set of four tools, three of them are for conversion (ie from TSed to SBML, SBML to Matlab, SBML to Latex) and the last one is for the graphical vizualisation. The conversion tools can either be called from a command line, or from Matlab (requires a few more step in the installation process), or from the graphical interface. The final program can run on a Linux or a Windows platform, the two flavors are packaged and documented.

3.2 Some numbers

About 8600 of core code, separated in 1000 for the LaTeXpart, 1600 for the Matlab part and 6000 common between those two parts. These lines corresponds to the "sbml2matlab" and "sbml2latex" programs, which are also embedded into the SView program in the final version. In addition to these comes 5000 lines of tests, unit tests and full regression tests.

From the two semester projects from Michel Sede and Laurent Francioli, we can add 2800 lines for the TSed program and 2800 lines for the SView graphical interface.

The full suite of tools makes a total of 19200 lines of Java code (including comments).

3.3 Acquired skills and self-assessment

The goal of this project unlike other more theoretical projects had as a goal to have a working solution. In this perspective, the program was heavily tested with more than 200 tests, including about 60 full regression tests (i.e. a input and a output that have to match, not only a single function test).

This project has also to be portable, but part of its dependencies (the SBML library) was written in C so that some function definitions differs from one system two another, but fortunately this library was already compatible with both system. So that writing the Java code had to be done carefully with adding non-portable C calls. Difficulties were encountered when it comes to deallocating some SBML components, but finally a solution was found.

Regarding the self-assessment of the strategy adopted during this project, I would add that probably the first months of this project were too much concentrated in evaluating the solutions proposed by other and comparing them to our needs, so that out real needs were fixed too late on the project schedule.

4 Future, known issues and weak points

The part of SBML that we use in this project is really small, we could benefit of the use of units and events. Another point stated during the project was the use of compartments which would allow an easier visualization of bigger models, but also integrates a way to split the model itself into multiple parts.

The last point to mention is the fact that this kind of project should also wait for the feedback of biologist to continue evolving.

References

[1] Bower and Bolouri. Computational modeling of genetic and biochemical networks, 2000.

[2] CellDesigner. http://www.celldesigner.org. CellDesigner is a structured diagram editor for drawing gene-regulatory and biochemical networks.

[3] Laurent Francioli. Modeling the immune system, laboratory for computer communications and applications, epfl, 2006.

[4] Bente Kofahl and Edda Klipp. Modeling the dynamics of the yeast pheromone pathway. *Yeast*, (21):831–850, 2004.

[5] SBML. http://www.sbml.org. The Systems Biology Markup Language (SBML) is a computer-readable format for representing models of biochemical reaction networks. SBML is applicable to metabolic networks, cell-signaling pathways, regulatory networks, and many others.

[6] SBMLeditor. http://www.ebi.ac.uk/compneur-srv/sbmleditor.html. The SBML editor try to answer this need by provided a very simple, low level editor of SBML files.

[7] SBW. http://www.sys-bio.org. The Systems Biology Workbench (SBW), is a software framework that allows heterogeneous application components-written in diverse programming languages.

[8] Michel Sede. Modeling the immune system, laboratory for computer communications and applications, epfl, 2006.

[9] Darren J. Wilkinson. Statistical bioinformatics, 2004.

[10] XML. http://www.w3.org/xml/. Extensible Markup Language (XML) is a simple, very flexible text format derived from SGML (ISO 8879). Originally designed to meet the challenges of large-scale electronic publishing, XML is also playing an increasingly important role in the exchange of a wide variety of data on the Web and elsewhere.