Identifying and Deploying Kinetics Internally with SBML

Konstantinos Moutselos, Aristotelis Chatziioannou

Institute of Biological Research & Biotechnology, National Hellenic Research Foundation
kmoutselos@sch.gr, achatzi@eie.gr

Abstract

In order to facilitate the identification and deployment of internally referenced kinetic rate laws in SBML models, a comparison is being made between Gepasi/Copasi@ approaches, and along with the discussion of the limitations of present annotation and identification schemes, an implemented Java tool based on SBMLeditor is presented which uses an approximating identification algorithm and provides analogous facilities in the deployment of rate law expression for rapid SBML model building.

Keywords: Bioinformatics, SBML, reaction kinetics, rate laws, SBMLeditor, pathways simulation

1. Introduction

The use of computer modelling in the study of biochemical pathways is widely appreciated as the means of gaining deeper understanding of the involved processes. The ability to validate and calibrate existing and under construction biochemical models as well as to hypothesize and test ambiguous functions in an inexpensive and time efficient manner, underlines the importance of acquiring proper methods for information collection and processing. In figure 1 a diagram of the information gathering procedure in the construction of a biochemical kinetic pathway model is shown, starting from the definition of the model's static coefficients and then moving forward to its dynamic components, in order to enable the simulation of the model's behaviour over time.

The Systems Biology Markup Language (SBML) [Hucka et al. (2003)] as an Extensible Markup Language (XML) [Bray et al.] specific in describing biochemical models, presents a wide range of application, in parallel with its importance as a uniform means of transporting models in different software packages and simulators. Its gaining acceptance and support from a growing number of relative software tools, makes SBML ideal in achieving the 'write once, use many' notion.

SBML uses the <kineticLaw> tag to define the kinetic of a reaction, and has two ways to declare the equation of the reaction, depending on the level of SBML. As an evolving standard, SBML Level 1 uses the ‘formula’ parameter and an ASCII notation for the representation of the equation, while in Level 2, a portion of the Mathematical Markup Language (MathML) is used for the same purpose. In both cases, a list of parameters accompanies the equation where needed.

As the models are growing, thus rising in complexity by including an ever growing number of reactions according to the “systemic approach”, it is becoming evident that the maintenance and rapid construction of a model is not possible without the use of some sort of kinetic identification and the uncomplicated integration of known rate laws. However, SBML at its current state does not support some standard and uniform way of internal kinetic identification, although SBML Level 2 Version 2 implements the ability to externally reference rate laws, as well as other basic elements of a model with links to online databases.

What can be done towards the direction of the internal kinetic rate law manipulation, which is needed not only prior to version 2 but in latter versions as well, in cases where external referencing is not preferred?

At first we examine the approach taken in two highly used software packages (Gepasi and Copasi) for simulations of biochemical reaction networks, and then we build a prototype tool, which to our best knowledge is the first one implemented to enable the adoption of the Gepasi functionality of the library of typical biochemical reaction mechanisms in the SBML representation of biochemical reaction networks. The aim is to study the kinetic expressions reuse, based solely on a pure SBML model, without resorting to idiomorphic SBML structures whose support is severely limited by the lack of appropriate software implementations which can effectively utilize them.

The aforementioned Gepasi library of predefined kinetics consists of well-known rate law expressions taken from published studies, which are the result of in-vitro experiments. In these experiments variables such as temperature, radiation, pressure etc are ‘hidden’ in the form of constants and modifiers. These predefined rate law expressions are used for the rational design of bottom-up biochemical models.

2. The Gepasi/Copasi approach

Gepasi [Mendes (1997)] accomplishes rate law internal identification using unconditionally two kind of kinetic types: predefined and user-defined kinetic types. An equation can be added and used, only if it has been declared and entered to the relative kinetic type database. The name of a new kinetic is being saved in the ‘.gps’ file as ‘FunctionName’ (eg. FunctionName= Allosteric activation (MWC)) and its

equation as ‘Description’ (e.g. Description=V*S*(1+S/Ks)*(1+Activator/Ka)^2/(Ks*(L+(1+S/Ks)^2*(1+Activator/Ka)^2)), and therefore the corresponding equation can be retrieved from the database the next time a user opens a model containing the relative kinetic type, or if she/he wants to enter the formula in a new reaction. If the kinetic is one of the predefined types, then the name is enough to define the equation, e.g.: KineticType=Mass action (irreversible). The '.gps' file also contains the distinction between pre- or user- defined formulas (e.g.: User-defined=1).

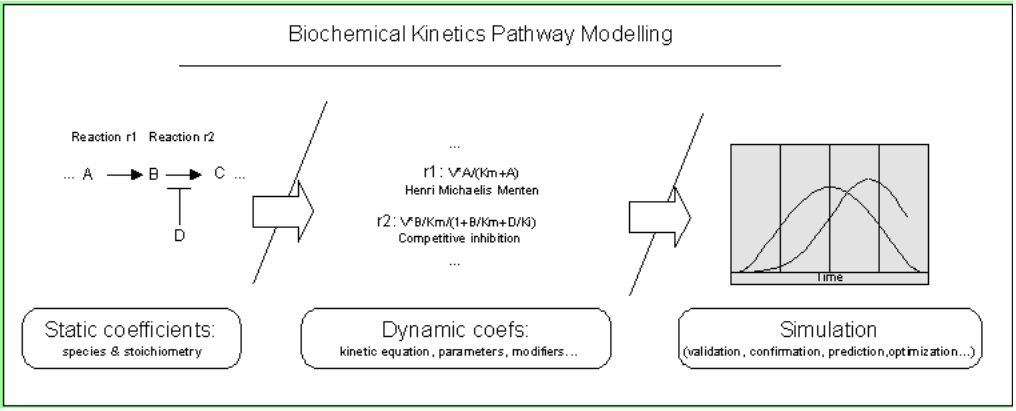


***Figure 1:** A generic diagram of the biochemical kinetics pathway modelling*

This means that for a user in the procedure of building a biochemical model, it is easy to select standard types of biochemical equations or in case of new ones which are not yet listed, to add them in the database once, and then use them thereinafter. In addition, when a user opens a ready-made model, the program recognizes the equation by its name, and so the kinetic type is identified immediately. However, when the model is exported to SBML, this extra information is lost, and the user does not know if the kinetic is one of the recognized and documented kinetics, or if there is some alteration in the specific equation which in this case has to be checked by comparing the relevant terms one by one.

Complex Pathway Simulator (Copasi) [Hoops et al. (2006)] goes a step further. When the model is exported to SBML, the kinetic reaction is saved in a <functionDefinition> block using MathML. So, when the model is imported back into Copasi, the program can identify these 'functionDefinition' blocks as recognizable patterns and consequently retrieve the name of a known kinetic. This means that if the contents of the <functionDefinition> tag are altered in a seemingly trivial way, such as by permuting the members of a multiplication, the program loses its ability to recognize the kinetic, and gives it a new name (such as 'function_4_R1' instead of the initial 'Henri-Michaelis-Menten (irreversible)').

Once more, if the exported SBML model is entered in other software system, the kinetic name of a reaction is irretrievable.

3. Building a ‘pure’ SBML case

In order to gain a better understanding of the intrinsic problems that internal SBML kinetic identification entails, a tool is implemented, which could provide the Gepasi-like functionality in deploying and recognizing kinetics. Even though there are several software packages, which offer a library of predefined rate expressions [Alves et al. (2006)], our study focuses on Gepasi functionality due to its wide acceptance.

As a base of our implementation, the program SBMLeditor [Rodriguez et al. (2006)] was selected, which is an open-source Java software distributed under the GNU General Public License. SBMLeditor is a ‘low level editor of SBML files where users can create and remove all the necessary bits and pieces of SBML in a controlled way, that maintains the validity of the final SBML file’. In addition, it can import and convert models from different Levels/Versions of SBML.

In figure 2 we can see a part of reaction R1of a model in SBMLeditor. In particular, since this is Level 2 SBML model, we have MathML to describe the equation of the reaction. Can a user recognize if this is a known kinetic? Probably not, since MathML is not intended to be edited manually, but to be used as a low-level language for editors and conversion programs [Rowlett (2003)].

If we right-click on the blue ‘kineticLaw’ tag and select ‘Edit’ we have the ‘kineticLaw’ window presented in figure 3, where the user has the option to manually edit the mathml block. What we have implemented: If the button ‘kinetic types’ is pressed, then we have the ‘Kinetic Types’ window, also presented in fig.3, which identifies the reaction as ‘Mixed activation (irr)’ type and presents the equation in simple ASCII. How is this possible and what is its limitation? (i.e. how far this recognition can go?). The presented recognition is based on an algorithm using these criteria:

1) Reversible / non-Reversible reaction
2) Number of parameters
3) The kinetic formula itself

In each step the possible equation set is reduced until the third one where a final identification is attempted using the exact formula pattern.

In the case of an irreversible reaction having four parameters, we have two possible candidates according to the Gepasi database: "Mixed activation (irr)" or "Mixed inhibition (irr)". For the final call (third step), we use Regular Expressions in Java in the form e.g. of the String: “\\b\\w+\\*(\\w+)/(\\w+)/\\(1\\+\\w+/(\\w+)\\+\\1/\\2\\*\\(1\\+\\w+/\\3\\)\\)” for the “Mixed activation (irr)” case. Note that group numbering in regular expressions help towards the certification of the relevant guess.

Of course, this identification approximation cannot include cases where the formula’s terms are altered, even in the most trivial way: permuting the terms of a multiplication, or adding extra parentheses. No further progress seems to be possible in this direction, unless we could somehow include in the regular expression pattern all the commutative ring of each known formula. In addition, it should be feasible to implement this comparison for all the user-defined kinetics.

As far as the part of a uniform representation of known kinetics is concerned, a scheme has been proposed [Halasz (2004)], in an attempt to standardize these equations, alleviating in this way, the burden of rate laws recognition.

In addition to the recognition of existing kinetics in a system, there is also the portion of how easily a system can be augmented when the user wants to add new reactions in the SBML model.

In the tool that has been implemented, as it is shown in fig.3, since the reactants and the products of a new reaction has been defined, the user can choose one of the pre-defined kinetic types, and then the analogous formula/MathML along with its list of parameters and modifiers will be properly inserted. In each case, the list of the presented available pre-defined equations depends on the kind of the reaction (reversible or not), and the number of reactants/products. In the case of reactions of the kind: A→B, 15 predefined kinetics are presented, according to the Gepasi database.

When a new element is either entered, modified or removed from the SBML system, SBMLeditor verifies the consistency of the model, checking for instance if a declared parameter in an equation is properly defined in the listOf parameters. Our added implementation postpones this checking for the final stage of verification, so that the bulk changes accompanying the alteration of the Document Object Model (DOM) [Nicol et al. (2004)] sub tree can be accepted. Furthermore, since SBMLeditor is being constantly updated as well as its underline libraries (such as libsbml, compneur etc), care has been taken as to be possible to easily migrate the added implementation to the latest version of the editor, without hindering the existing class hierarchies which also make use of design patterns. Therefore, for the time being, the tool is based on the SBMLeditor Version 1.0.

4. External references to the rescue?

The ability to annotate model elements with references to terms from controlled vocabularies, has been incorporated into SBML Level 2 Version 2 in two forms: the sboTerm attribute (for simple controlled vocabularies references specifically to terms from the Systems Biology Ontology [Le Novere (2006)]) and the Le Novere-Finney scheme [Le Novere et al. (2005)] for metadata annotations. SBMLeditor provides the ability to use the annotation framework based on the Resource Description Framework (RDF) [Miller et al. (2004)] and dublin core [Baker (2000)] dc:creator and dc:relation elements as well as the sboTerm attribute. Although the controlled vocabularies are an important improvement in the struggle to standardize the referenced terms, the way of presenting and using these vocabularies in relevant tools has not yet reached a consensus. The transition to RDF should provide a unified reference environment, avoiding XML namespace cluttering and Schemas versioning [Wang et al. (2005)].

5. Conclusions and discussion

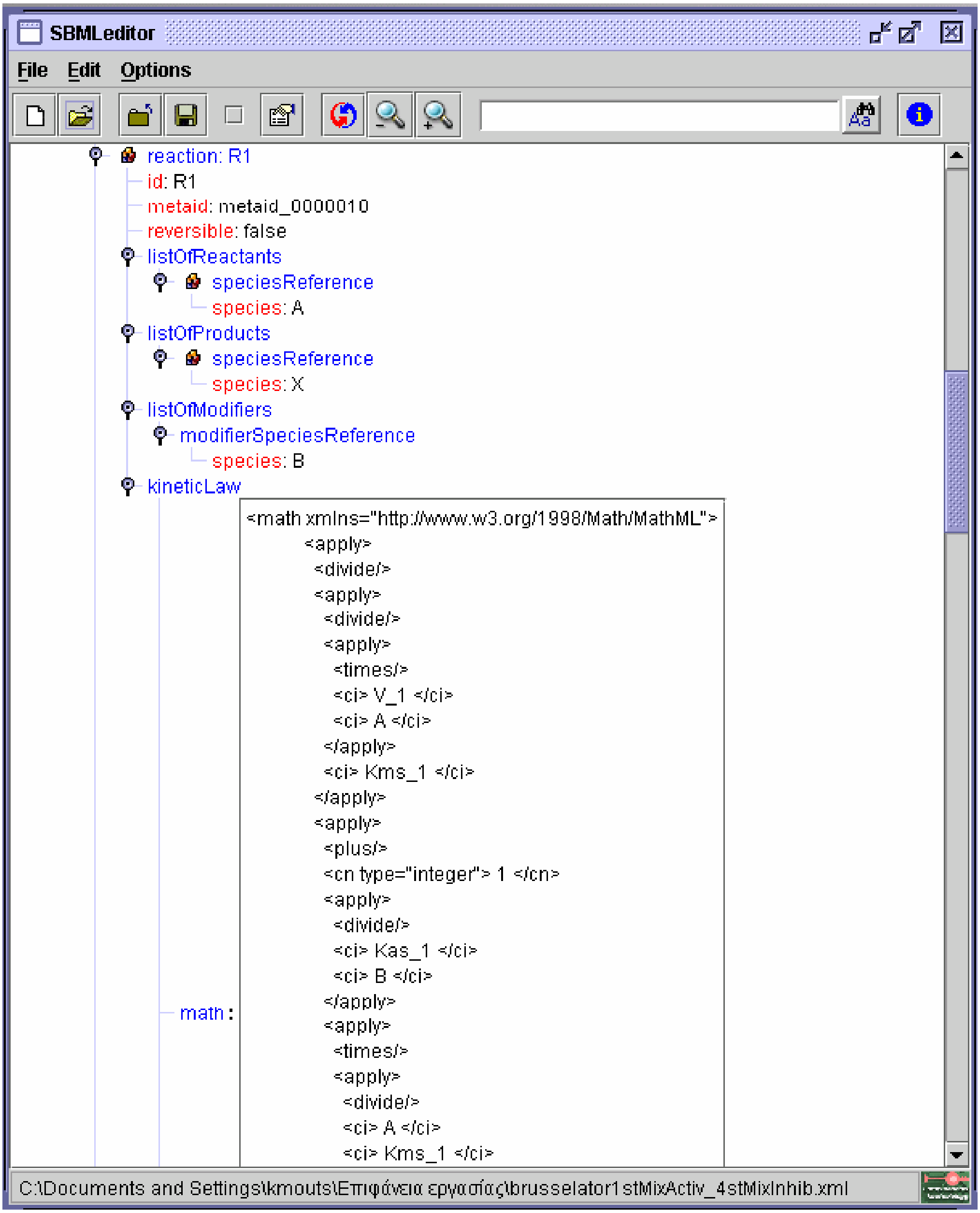
SBML has not a standard and unique method to use for kinetic identification, either internally or externally. This leads to custom implementations for each tool in order to achieve a level of functionality needed in structuring large and complex biological systems. These implementations differ by the way they define, identify and store rate law expressions, thus imposing a usability barrier in the models transportation and in the 'write once, use many' notion. Even though most software tools can import without problems SBML validated models developed in different programs, it is highly probable that the specific annotations used in order to add information value in a model, pass by undecipherable when used by different software. And this of course applies for other SBML elements as well, but in the case of kinetic identification and deployment this problem aggravates, since it involves predefined / user-defined rate expressions, each with its own parameters and modifiers, along with backward compatibility issues due to major changes in SBML Levels such as the formula attribute in the kineticLaw element (which effectively create different standards [Wang et al. (2005)]).

In the implementation presented, the task of kinetic identification is carried out internally (with no external SBML references) based solely on the structure of the rate equation itself. This approach does not use any annotation scheme, but includes a relevant guess in the identification, since even the forms of the known rate law expressions are not yet standardized.

The deployment of predefined kinetics in the implemented tool is conducted with an added menu in the SBMLeditor, offering the needed capability in the developing of large models, similar to the Gepasi functionality.

In each case the tool provides an ASCII representation of the kinetic formula which is especially useful in the SBML Level 2 model, which makes use of the long MathML expressions.

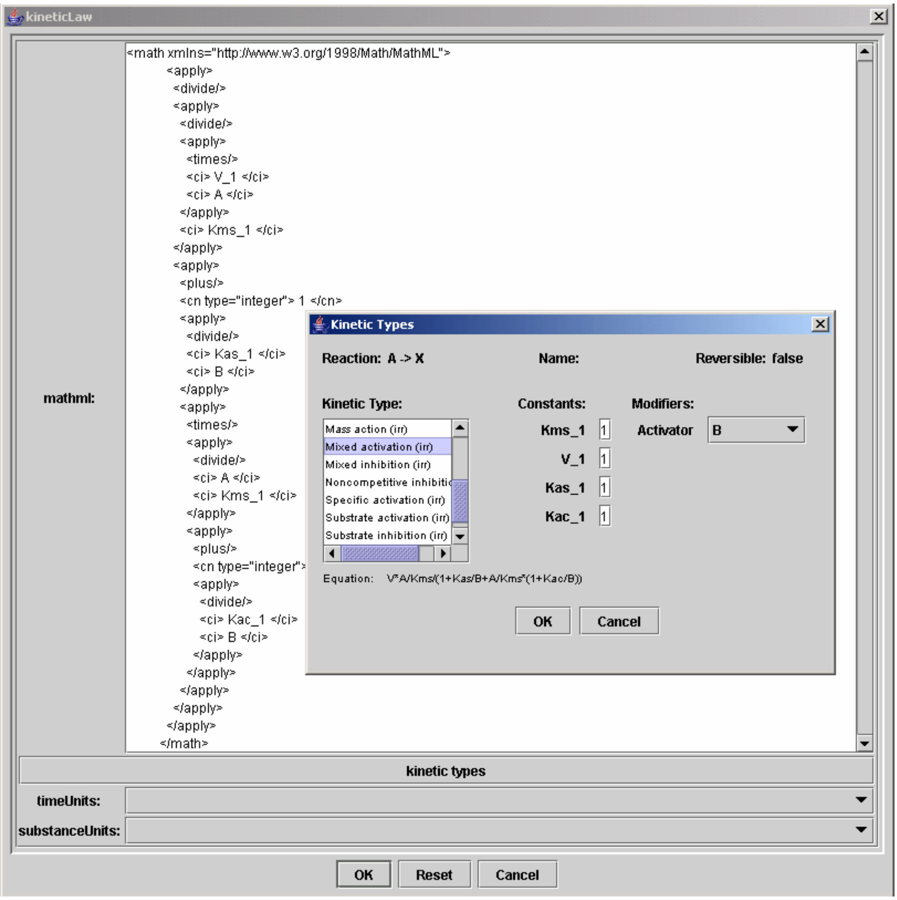
Our intention is to further complement the developed tool with a set of predefined equations for the well-known reversible rate laws and refactor the implemented code in order to facilitate easier development of future enhancements, such as the ability for the user to choose also external referenced rate laws, taken from on-line databases.



```
reaction: R1
  id: R1
  metaid: metaid_0000010
  reversible: false
  listOfReactants
    speciesReference
      species: A
  listOfProducts
    speciesReference
      species: X
  listOfModifiers
    modifierSpeciesReference
      species: B
  kineticLaw
    math : <math xmlns="http://www.w3.org/1998/Math/MathML">
                <apply>
                 <divide/>
                 <apply>
                  <divide/>
                  <apply>
                   <times/>
                   <ci> V_1 </ci>
                   <ci> A </ci>
                  </apply>
                  <ci> Kms_1 </ci>
                 </apply>
                 <apply>
                  <plus/>
                  <cn type="integer"> 1 </cn>
                  <apply>
                   <divide/>
                   <ci> Kas_1 </ci>
                   <ci> B </ci>
                  </apply>
                  <apply>
                   <times/>
                   <apply>
                    <divide/>
                    <ci> A </ci>
                    <ci> Kms_1 </ci>
```

C:\Documents and Settings\kmouts\Επιφάνεια εργασίας\brusselator1stMixActiv_4stMixInhib.xml

***Figure 2:** Displaying the kineticLaw element of the reaction R1 in the SBMLeditor for a Level 2 model*



kineticLaw

mathml:

```
<math xmlns="http://www.w3.org/1998/Math/MathML">
      <apply>
       <divide/>
       <apply>
        <divide/>
        <apply>
         <times/>
         <ci> V_1 </ci>
         <ci> A </ci>
        </apply>
        <ci> Kms_1 </ci>
       </apply>
       <apply>
        <plus/>
        <cn type="integer"> 1 </cn>
        <apply>
         <divide/>
         <ci> Kas_1 </ci>
         <ci> B </ci>
        </apply>
        <apply>
         <times/>
         <apply>
          <divide/>
          <ci> A </ci>
          <ci> Kms_1 </ci>
         </apply>
         <apply>
          <plus/>
          <cn type="integer">
          <apply>
           <divide/>
           <ci> Kac_1 </ci>
           <ci> B </ci>
          </apply>
         </apply>
        </apply>
       </apply>
      </apply>
     </math>
```

Kinetic Types

Reaction: A -> X Name: Reversible: false

Kinetic Type:

- Mass action (irr)
- Mixed activation (irr)
- Mixed inhibition (irr)
- Noncompetitive inhibitic
- Specific activation (irr)
- Substrate activation (irr)
- Substrate inhibition (irr)

Constants:	
Kms_1	1
V_1	1
Kas_1	1
Kac_1	1

Modifiers: Activator B

Equation: V*A/Kms/(1+Kas/B+A/Kms*(1+Kac/B))

timeUnits:

substanceUnits:

Figure 3: *The MathML equation and the implemented Kinetic Types window*

6. References

Alves R, Antunes F, Salvador A. (2006). Tools for kinetic modeling of biochemical networks. Nat Biotechnol 24(6):667-72.

Baker T. (2000). A Grammar of Dublin Core. D-Lib Magazine 6(10).

Bray T, Paoli j, Sperbeg-McQueen C, Maler E, Yergeau F. Extensible Markup Language (XML) 1.0. p W3C Recommendation.

Halasz A. (2004). Proposed Annotation Scheme for Rate Laws in SBML.

Hoops S, Sahle S, Gauges R, Lee C, Pahle J, Simus N, Singhal M, Xu L, Mendes P, Kummer U. (2006). COPASI - a COmplex PAthway SImulator. Bioinformatics.

Hucka M, Finney A, Sauro HM, Bolouri H, Doyle JC, Kitano H, Arkin AP, Bornstein BJ, Bray D, Cornish-Bowden A and others. (2003). The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. Bioinformatics 19(4):524-31.

Le Novere N. (2006). Model storage, exchange and integration. BMC Neurosci 7 Suppl 1:S11.

Le Novere N, Finney A. (2005). A simple scheme for annotating SBML with references to controlled vocabularies and database entries. p 13.

Mendes P. (1997). Biochemistry by numbers: simulation of biochemical pathways with Gepasi 3. Trends Biochem Sci 22(9):361-3.

Miller E, Manola F. (2004). RDF Primer. p W3C Recommendation.

Nicol G, Le Hégaret P, Wood L, Robie J, Champion M, Byrne S, Le Hors A. (2004). Document Object Model (DOM) Level 3 Core Specification.

Rodriguez N, Donizelli M, Le Novere N. (2006). SBMLeditor: An editor for models in the Systems Biology Markup Language (SBML).

Rowlett P. (2003). MathML: the current state of play [MSc dissertation]: Nottingham Trent University. 8 p.

Wang X, Gorlitsky R, Almeida JS. (2005). From XML to RDF: how semantic web technologies will change the design of 'omic' standards. Nat Biotechnol 23(9):1099-103.