

The SBML Semantic Test Suite

Andrew Finney

<http://sbml.org/>

Overview

- **The Problem and a Partial Solution**
- **Overview of the Semantic Test Suite**
- **The Tests**
 - Suite content for each test
 - Current Coverage
 - Roadmap for remaining Level 2 features
- **The Physical Structure of the Suite**
- **The Automation**
 - How to package a simulator to enable automation
 - How to use automation
 - How to create custom test schedules
- **Background**
- **Future Work**
- **Acknowledgements**
- **Conclusions**

The Problem

- **Good News**

- SBML is supported by > 60 applications
- We are well down the road to validating the syntax and logical consistency of SBML

- **Bad News**

- SBML is supported by > 60 applications
- How do we know that these applications interpret SBML in the same way?
- Full Interoperability of software through SBML will not be realized if we don't address this
 - Threatens the viability of SBML

The Semantic Test Suite: A Partial Solution

- **Semantic Test Suite consists of SBML models together with corresponding simulation output**
- **Why?**
 - Most complex form of SBML interpretation is simulation
 - Simulation is a common form of interpretation
 - Simulation is relatively easy to test
 - for example consider issues to do with testing diagrammatic interpretation
 - Apart from Metadata almost all SBML features impact simulation
 - for example diagrammatic interpretation only requires qualitative reaction network
 - Models and Simulation output can be used as useful documentation of the standard even when considering the implementation of other types of analysis
- **The suite is a partial solution**
 - Many applications do not have a simulation capability

Suite Test Approach

- Tests are designed to check interpretation of SBML by simulators
- It is not designed to test:
 - Ability to simulate stiff systems
 - Ability to handle sharp discontinuities
 - Execution Performance
 - Precise accuracy when simulating complex models over 'long' time courses
 - Requires further work and experience
- Test categorization allows simulators to demonstrate strengths in particular areas
- Does expect simulators to complete a test
 - Incomplete test result is a failed test

Suite Logical Structure

- Suite divided into categories of test
- Each category tests a particular feature set
 - Category conceptually similar to ‘Modules’ (?)
- Basic features separated from advanced features
- By default automation test schedule follows test suite physical structure
 - Ordered to test basic features first, followed by advanced features
- Custom schedules can re-categorize and re-order

Default Suite Categories

- **Basic reactions**
 - Simple mass action
 - Boundary conditions
- **Parameter namespace**
 - Global and local (reaction) parameter namespace
- **Complex reactions**
 - Larger reaction networks with complex kinetic law expressions
 - Reactions without products or reactants
 - Reactions with modifiers
- **MathML**
 - Continuous MathML subset
- **Basic rules**
 - Rate and assignment rules

Default Suite Categories (continued)

- **Stoichiometry**
 - Integer, floating and MathML expression stoichiometry
- **Discontinuity, time and delays**
 - Discontinuous MathML
 - `<csymbol>` elements for delay and time
- **Algebraic rules**
- **Function definitions**
 - Tests usage in range of contexts
- **Events**
 - Tests incrementing assignment
- **Compartments**
 - Above categories all operate in a unit size compartment
 - Compartment category tests transport reactions and non-unit sized compartments
 - Under development

Suite Content for Each Test

- **SBML Level 2 Model**
 - SBML Level 1 Model where possible
 - Some Level 1 models missing in beta
- **Simulation Output**
 - Always Species Concentrations
 - Comma Separated Values (CSV)
 - Precise format described later
 - Plots of Simulation data in GIF format
 - Concentrations on normal and log scales
 - Beta contains some poor log plots

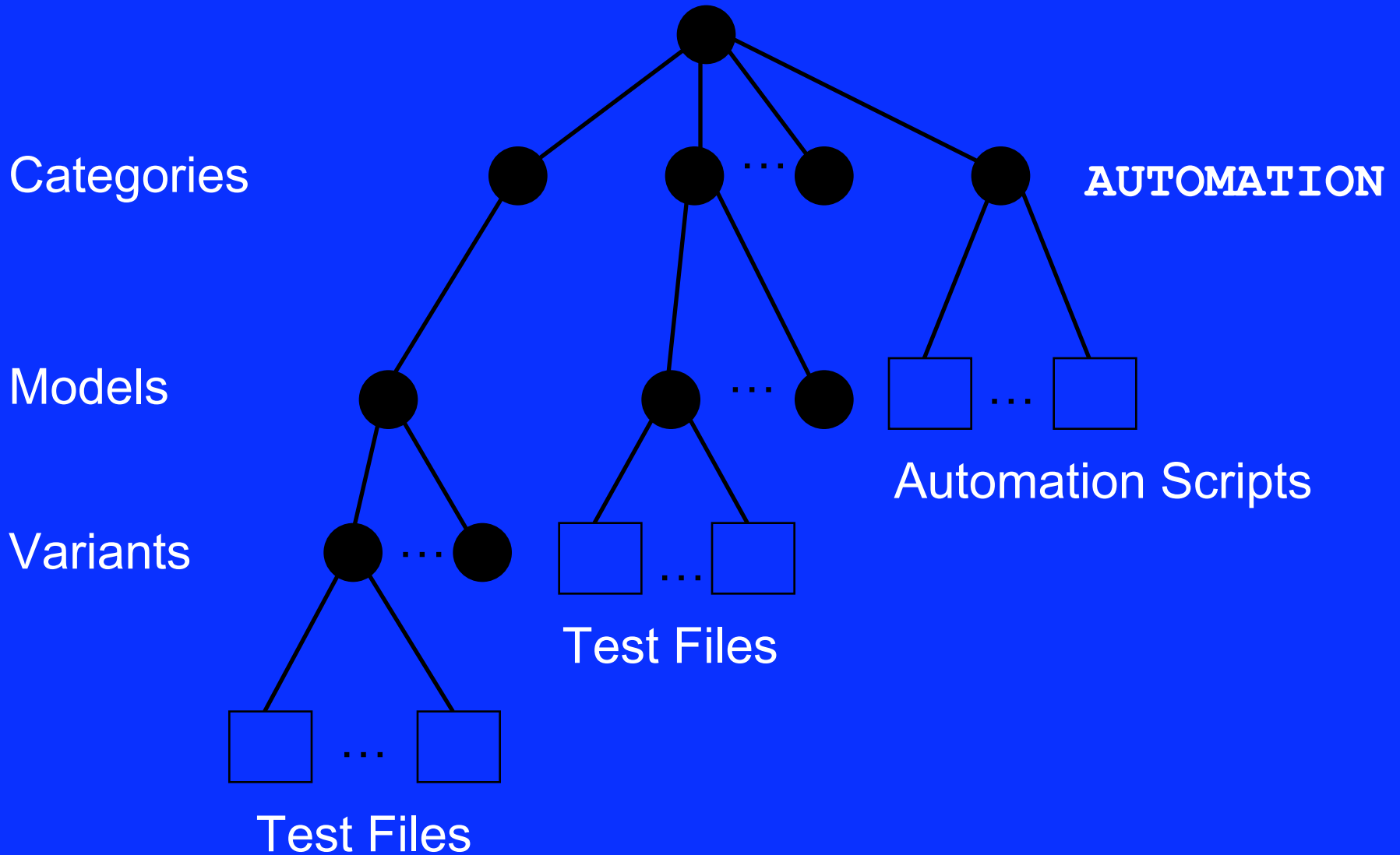
Suite Content for Each Test (continued)

- **‘.Test’ file**
 - contains information to drive test automation
 - format described later
- **Automatically Generated HTML**
- **Automatically Generated Diagram GIF**
- **MathSBML File to drive CSV, GIFs and HTML generation**
 - i.e. simulation script
 - Mathematica
- **Documentation**
 - One online wiki page per test
 - Describes model
 - Giving equations
 - Links to web image of test suite
 - Automatically generated HTML
 - includes ODEs for model
 - Plots, SBML etc
 - Allows web browsing of test suite without download

Suite Physical Structure: Overview

- **The Download**
 - Beta Available at
<http://www.cds.caltech.edu/~afinney/semantic-test-suite.tar.gz>
 - Consists of 3 parts:
 - Automation
 - Automation documentation
 - Tests
- **The Documentation**
 - Constructed on the SBML Wiki at
 - http://sbml.org/wiki/Semantic_Test_Suite
 - Can be developed collaboratively

Download Structure



Automation

- **Goals**
 - Automatic Testing of Simulators against Test Suite
 - Configurable
 - Simple
 - Portable
- **Most goals met by using `awk` and `bash`**
 - Runs on Unix and Cygwin
 - On Windows download Cygwin including `awk` and `bash`

Test Criteria

- For all time steps and all species concentrations
 - test fails fail if
 - $d1 < k.d2$ OR $d2 < k.d1$
 - In beta k fixed at 0.999
 - not difficult to change!
 - In revised version k will be an optional parameter controlled from custom test schedule
- Very simple
- No attempt to compensate for cumulative errors

Simulator Wrappers

- A simulator wrapper is a script that runs your simulator given a simple set of arguments
- Assumes that your simulator can be run and be controlled with command line arguments or via text files

Wrapper arguments

- In order:
 - Path of SBML file
 - Simulation time in seconds
 - Number of simulation time steps
 - Simulation Output
 - Temporary Directory
 - Species Name 1
 - ...
 - Species Name N

Simulation Wrapper Output and 'Correct' Simulation Data Format

- ASCII Comma Separated Values
 - `Excel` loadable
- 1st row is column header for humans and is ignored
- 2nd Row contains data for $t=0$
- Subsequent rows for each time step
- N time steps => N+2 rows
- 1st Column is t
- Remaining columns contain data at the given time step for the species
 - species data ordered as given in the wrapper arguments
- Numbers in `awk` and `Excel` readable floating point number format

Running A Single Test

- Locate a *.test file
- Place AUTOMATION on the PATH
- type
 - test.bsh <wrapper> <*.test file>

Running the Default Test Schedule

- change directory to top of suite
- type
 - `./runtests.bsh -vwrapper=<wrapper>`
 - optional argument `-vhaltOnFailure=true`

Example Schedule

CATEGORY Basic Reactions

TEST basic-reactions/modell/BothBoundary/basic-modell-bothboundary.test

TEST basic-reactions/modell/Boundary/basic-modell-boundary.test

TEST basic-reactions/modell/Boundary2/basic-modell-boundary2.test

TEST basic-reactions/model6/base/basic-model6-base.test

CATEGORY Parameter Namespace

TEST parameterNamespace/global/parameterNamespace-global.test

TEST parameterNamespace/globalAndLocal/parameterNamespace-globalAndLocal.test

TEST parameterNamespace/local/parameterNamespace-local.test

TEST parameterNamespace/speciesAndLocal/parameterNamespace-speciesAndLocal.test

CATEGORY Complex Reactions

TEST complexReactions/FeedbackCompetitive/feedbackCompetitive.test

TEST complexReactions/Hill/Hill.test

Schedule File Format

- Default schedule is
 - `AUTOMATION/testlist.txt`
- Copy and Edit!
- At least one test in each category
- At least one category
- Category declaration precedes test declaration
- Categories don't have to match test suite structure

Running a Custom Test Schedule

- Place `AUTOMATION` on the `PATH`
- `type`
 - `runthroustests.awk -vwrapper="<wrapper>" <test schedule file>`
 - optional argument `-vhaltOnFailure="true"`

Under the Hood

- Tests created using MathSBML
 - helped to improve quality of MathSBML
- *.test files contain data to drive test
- example:

```
TIME 25
```

```
STEPS 50
```

```
SPECIES S1 S2
```

```
URL Boundary_Variant
```

```
REM This model tests the interpretation of  
REM boundary condition attributes.
```

```
REM One species, S1, in this model is a  
REM boundary condition and doesn't change.
```

Future Work

- Parameterize test criteria
 - k set in schedule
- Extend test coverage
- Clean up
- Web Facilities
- Generate comparison GIF plots as part of testing process
- Generate HTML page summarizing result of test schedule
- Resolution of model complexity issue
- Place in CVS
- Averaging output for stochastic simulators

Acknowledgements

- **Bruce Shapiro**
 - MathSBML
- **Additional Comments and Support**
 - Sarah Keating
 - Mike Hucka
 - The Community
- **Funding**
 - National Human Genome Research Institute (USA)
 - National Institute of General Medical Sciences (USA)
- **Additional support is provided by**
 - California Institute of Technology (USA)
 - University of Hertfordshire (UK)
 - Molecular Sciences Institute (USA)

Conclusion

- SBML semantic test suite enables testing of SBML interpretation for simulators
- Useful for tool development and quality control
- Some further work required
- **Please give feedback**
 - Suite will be project in SBML bug tracking database

Any Questions?

SBML Semantic Validation: Testing Software interpretation of SBML

- Is this an important issue?
- How do you test your software?
- Is the SBML semantic test suite a good idea?
- Who plans to use the suite?
- What tests are missing?
- What automation scripts are missing?
- The SBML semantic test suite is limited to simulation
 - Can we find ways to test other types of analysis?
- Can we test the interpretation of qualitative models?
- Can we compare the intermediate products of software interpretation?
 - Examples
 - ODEs, Stoichiometry Matrix
- Can we establish a conformance lab or a review process?