



# **Systems Biology Markup Language (SBML): ChemChains and a Proposal to Formalize the Encoding of Boolean Models**

*Tomáš Helikar*  
([thelika@unmc.edu](mailto:thelika@unmc.edu))

Version	1.20
Date	08/09/2008

## Revision History

Date	Version	Description	Author
08/09/2008	1.00	Initial Release	Helikar
08/10/2008	1.10	Refined representation of network inputs, added element descriptions, added relational tree	Helikar
08/11/2008	1.20	Added ChemChains input file format description	Helikar

## **Our group and the need for mathematical models**

The immense complexity of signal transduction networks lead to the hypothesis that cellular signal transduction networks function as information-processing, decision-making machinery. In order to test this hypothesis our group, the Mathematical Biology research group (MathBio: <http://mathbio.unomaha.edu>), developed a Boolean model representing these networks. This model consists of about 135 nodes and 800 connections (the database of all interaction comprising this model can be accessed at: <http://mathbio.unomaha.edu/Database>) and represents three main signaling pathways: the Receptor Tyrosine Kinase, G-Protein-Coupled Receptor, and Integrin pathways.

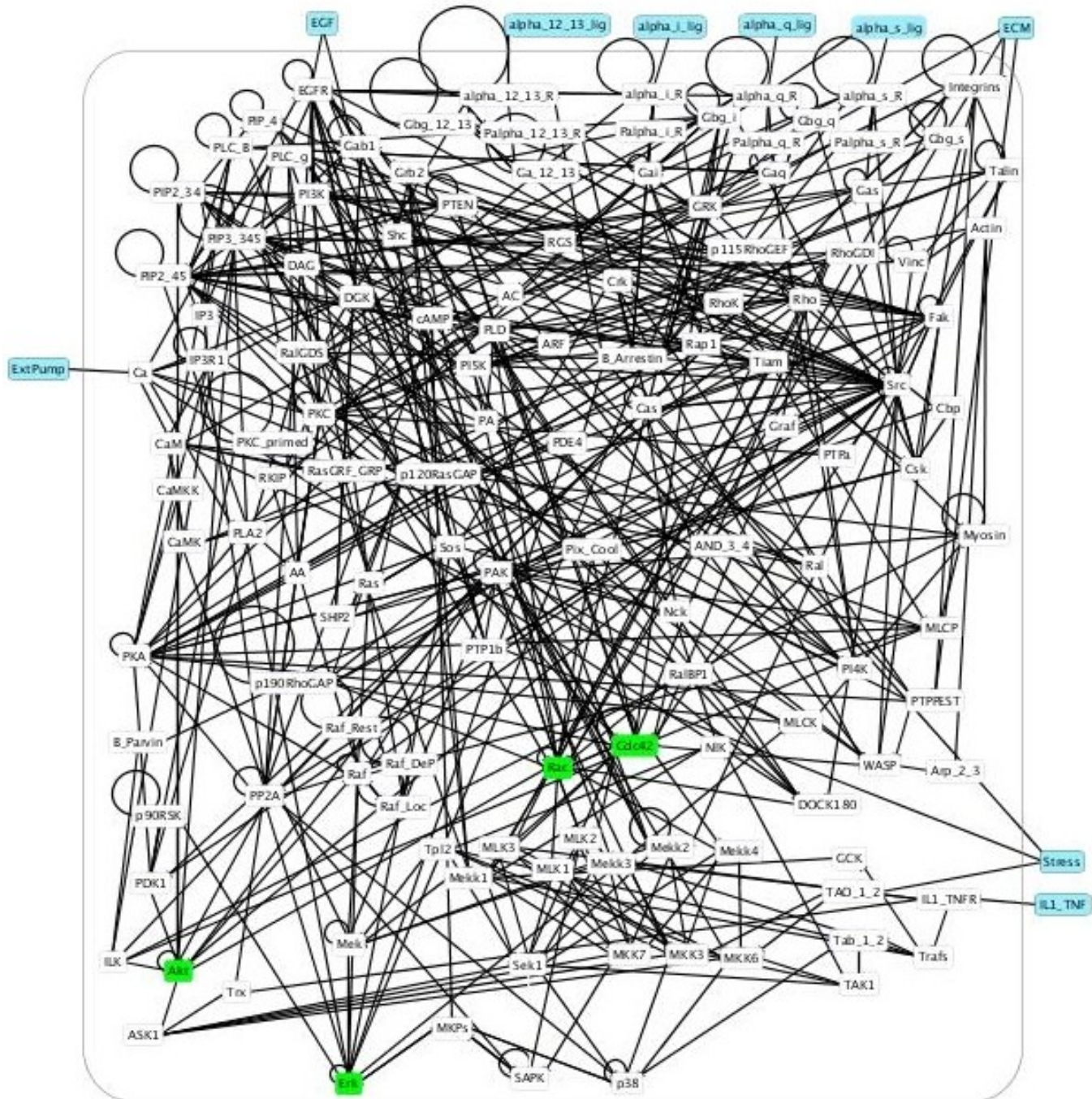


Figure 1: Graphical representation the Boolean signal transduction model developed by MathBio.

To put the model into motion and be able to analyze its dynamics under tens of thousands of biologically-relevant environmental conditions, ChemChains software tool has been developed (described below).

Detailed results of this study can be reviewed in: [T Helikar, J Konvalina, J Heidel and JA Rogers. 2008. Emergent Decision-making in Biological Signal Transduction Networks. *PNAS*. Vol.105, No. 6, pp 1914 - 1918]

## **ChemChains: Boolean models simulation and analysis suite**

Boolean Networks consist of a set of binary nodes (e.g., 0 or 1) and directed edges connecting these nodes. Boolean networks can be further categorized as autonomous or non-autonomous. Autonomous Boolean networks (ABN) consist of nodes whose state is determined by the activation state of upstream nodes connected to a node of interest,  $X$ . The activation mechanism of node  $X$  is determined by its Boolean function, often also represented by a Boolean truth table. Non-autonomous Boolean networks (NABN), in addition to the nodes found in ABNs, also contain so-called external input nodes. In the contrary to the internal nodes, the activation mechanism of the external input nodes is independent of the states of other nodes in the network, and can be represented by any type of a function. In the context of biochemical pathways, internal nodes of Boolean models represent various biological molecules (e.g., proteins, genes, etc.), while external input nodes correspond to biochemical variables controlling the dynamics of a given biological process (e.g., hormones, UV light, etc., ). The edges in the above mentioned models correspond to biochemical interactions between the various molecules.

ChemChains is a command-line based logical network simulation and analysis suite. The most prominent feature of ChemChains is that it allows users to interact with the model in a continuous manner, as they do in the laboratory experiments (e.g., users have the ability to set inputs to a percentage ON, as opposed to only 0 or 1) -see Figure 2. This allows laboratory biologists to utilize ChemChains as a complimentary tool to for their laboratory experiments.

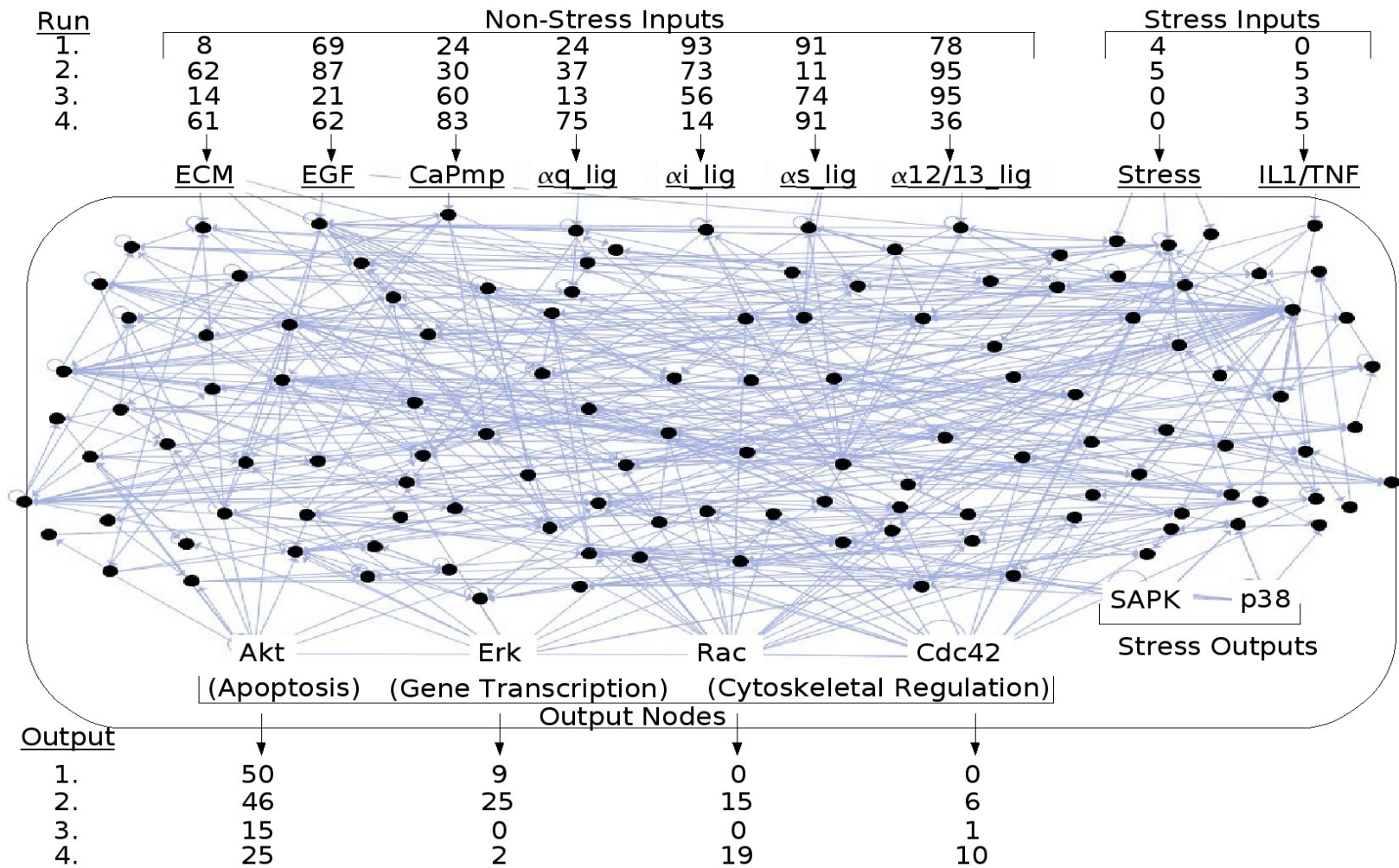


Figure 2: ChemChains input-output data representation.

## ChemChains Input

To simulate logical networks with ChemChains, two input files are required: the network descriptor file and simulation specification file.

### Network Descriptor File:

Network descriptor is a syntax-specific text file that stores the network, and logic for each node.

Consider a sample Boolean network consisting of three internal nodes A, B, C and one external network node IN1, whose graphical representation is captured in Figure 3. Truth tables for nodes A, B, and C are summarized in Figure 4. (For this example, IN1 is fixed as “1” for the entire simulation process.)

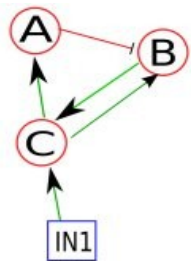


Figure 3: A sample boolean network.

C	A	C	A	B	IN1	B	C
0	0	0	0	0	0	0	0
1	1	0	1	0	0	1	1
		1	0	1	1	0	1
		1	1	0	1	1	1

Figure 4: Truth tables representing nodes A, B, and C in Figure 3.

Once the network and truth tables are established, they can be used to create the network descriptor file. The syntax to create the file follows:

```
Bool:name:initial state[True|False]:input1,input2..inputN:list of states of input nodes for non-initial state(separated by comma)
```

Thus, the network descriptor for our toy network can look like:

```
Bool:A:False:C:T  
Bool:B:True:A,C:FF,FT,TT  
Bool:C:True:IN1,B:FF
```

## ***Simulation Specification file***

ChemChains is a feature-rich logic network simulation software, which offers users many advanced simulation options. These options are specified by users in the specification file, the second of the required input files, that is loaded into the program before each simulation experiment.

The following describes the various options contained within this file:

- **Runtime** - specifies the overall length of each simulation, as well as the number of iterations (transient time) before any analysis is conducted.
- **Inputs** - external inputs (IN1 in the above toy network)
- **Noise** - The noise variable allows the input activity level to randomly bounce around the defined value within a specified range.
- **Mutation** - The mutation parameter enables users to turn ON/OFF any node in the network to perform mutagenesis studies.
- **Delay Nodes** - for asynchronous simulations
- **Sustain Nodes** – for asynchronous simulations
- **Output Nodes**

## ***Representation of Boolean models in SBML***

### *Suggested SBML Representation*

The following is a suggested SBML representation of the above sample boolean network described in Figures 3 and 4 .

#### **Base Declaration:**

```
<sbml>
[.]
<model type="Boolean">
  <listOfCompartments>
    <compartment id="comp1" />
  </listOfCompartments>
  <listOfSpecies>
    <species id="A" name="A" compartment="comp1" initialActivationValue="1" type="bool" />
    <species id="B" name="B" compartment="comp1" initialActivationValue="1" type="bool" />
    <species id="C" name="C" compartment="comp1" initialActivationValue="0" type="bool" />
    <species id="IN1" name="Network Input 1" compartment="comp1" initialActivationValue="0" type="networkInput" />
  </listOfSpecies>
[.]
</model>
</sbml>
```

#### **Boolean Interaction Representation:**

```
<listOfInteractions>
  <interaction>
    <listOfOutputs>
      <output species="A" />
    </listOfOutputs>
    <listOfRegulators>
      <regulator species="C" />
    </listOfRegulators>
    <listOfRegulationRules defaultStateValue="0">
      <regulationRule value="0" />
    </listOfRegulationRules>
  </interaction>
  <interaction>
    <listOfOutputs>
      <output species="B" />
    </listOfOutputs>
  </interaction>
</listOfInteractions>
```

```

</listOfOutputs>
<listOfRegulators>
  <regulator species="C" />
  <regulator species="A" />
</listOfRegulators>
<listOfRegulationRules defaultStateValue="1">
  <regulationRule value="10" />
</listOfRegulationRules>
</interaction>
<interaction>
  <listOfOutputs>
    <output species="C" />
  </listOfOutputs>
  <listOfRegulators>
    <regulator species="IN1" />
    <regulator species="B" />
  </listOfRegulators>
  <listOfRegulationRules defaultStateValue="1">
    <regulationRule value="01" />
    <regulationRule value="10" />
    <regulationRule value="11" />
  </listOfRegulationRules>
</interaction>
<interaction>
  <listOfOutputs>
    <output species="IN1" />
  </listOfOutputs>
  <listOfInputRegulationRules >
    <inputRegulationRule>
      <math xmlns="http://www.c3.org/1998/Math/MathML">
        <cn> 1 </cn>
      </math>
    </inputRegulationRule>
  </listOfInputRegulationRules >
</interaction>
</listOfInteractions>

```

### *More complex scenarios for activity levels of external inputs*

In fact, the activity sequence of IN1 can follow any pattern (e.g., periodic of any size, random, chaotic, etc.), given an appropriate equation. For example, the activity pattern of IN1 may be set to be on a period 2 cycle (e.g., 1,0,1,0...) which can be described by  $F(t) = t \bmod 2$ , where  $t$  represents the current simulation time. This scenario would be represented in SBML as follows:

```
<interaction>
  <listOfOutputs>
    <output species="IN1" />
  </listOfOutputs>
  <listOfInputRegulationRules>
    <inputRegulationRule>
      <math xmlns="http://www.c3.org/1998/Math/MathML">
        <apply>
          <rem/>
          <csymbol encoding="text" definitionURL="http://www.sbml.org/sbml/symbols/time"> t
</csymboli>
          <cn> 2 </cn>
        </apply>
      </math>
    </inputRegulationRule>
  </listOfInputRegulationRules>
</interaction>
```

## Suggested Changes to SBML that will allow the above proposed syntax:

### <species>

Addition of the following attributes to the element <species>:

*initialActivityLevel*: This attribute represents the node's initial state

*type*: The *type* attribute indicates the type of the node. The following are suggested values of the attribute *type*:

“*bool*”: indicating the node is an internal network node

“*networkInput*”: representing an external input node

### <listOfInteractions>

*ListOfInteractions* contains the set of directed edges; it inherits its properties from the *ListOf* element.

### <interaction>

*Interaction* describes a specific activation mechanism of a given node; it inherits its properties from *Sbase*.

### <listOfRegulators>

Container for inputs of a given node. This element inherits its properties from *ListOf*.

### <regulator>

References to the specie representing an input to a given node

<b>regulator</b>
species : sIDREF {use="required"}

### <listOfOutputs>

Container for a node of interest. This element inherits its properties from *ListOf* elements.

### <output>

A node of interest referencing a *species*.

<b>output</b>
species : sIDREF {use="required"}

### <*listOfRegulationRules*>

Container representing a set of activation states where the activity of a given node equals to the value of the *defaultStateValue* attribute. The rationale behind using this approach to describe Boolean functions, as opposed to using Boolean equations as suggested by Duncan and Nicolas in their proposal, is that this approach proved to provide an easy way to manually modify the node's activation mechanism, especially for nodes with a larger number of inputs.

<b>listOfRegulationRules</b>
defaultStateValue : integer {use="required"}

### <*regulationRule*>

The *Value* attribute defines an activation state of regulators of a node of interest, that results in the node's activity state defined by the *defaultStateValue* attribute.

<b>regulationRule</b>
value : string {use="required"}

### <*listOfInputRegulationRules*>

Container representing rules (e.g., mathematical equations) defining the activation level patterns of a given external input node.

### <*inputRegulationRule*>

Holds a mathematical equation representing the activation mechanism of a given external input node.

## *Relational Tree of Proposed Changes*

