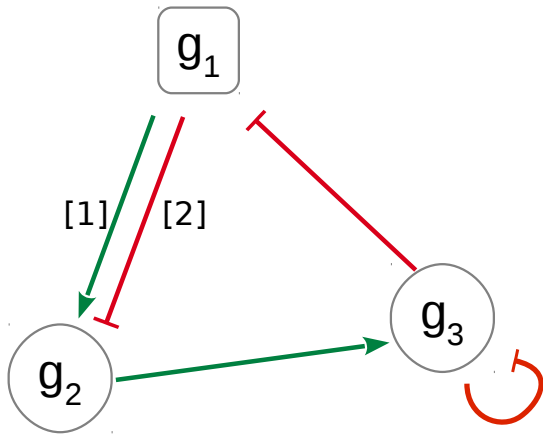


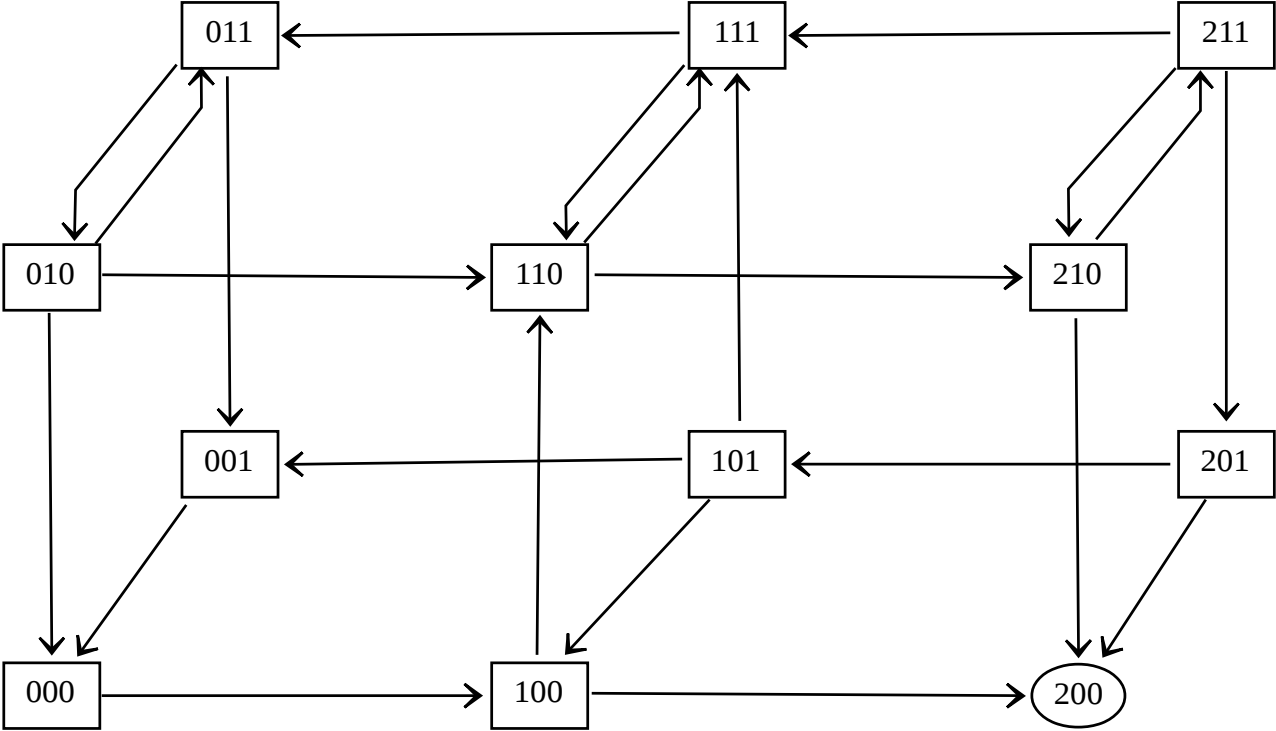
# GINsim overview and new features

# Logical modeling



- Nodes: genes, proteins...  
Activity level (Boolean, MV)
- Edges: interactions  
Threshold for MV source
- Logical rules  
 $\text{next}(G3) = G2$  and not  $G3$

# State Transition Graph



# GINsim

The screenshot shows the GINsim software interface for a file named 'phageLambda4'. The main window displays a state transition graph with four nodes: CI, Cro, N, and CII. Red arrows represent inhibitory interactions, and green arrows represent activating interactions. A red box highlights the 'E' button in the toolbar, and a blue box highlights the 'Modelling Attributes' panel.

**Modelling Attributes**

Id	Cro
Name	
Input	<input type="checkbox"/>
Max	3

1

Value	Active Interactions
3	(basal value)
2	Cro:3

2

CI:2 [2,max] ; negative  
Cro:3 [3,max] ; negative

Logical parameters or functions

State transition graphs: (a)synchronous, with priorities

A Naldi et al. Bio Systems (2009)

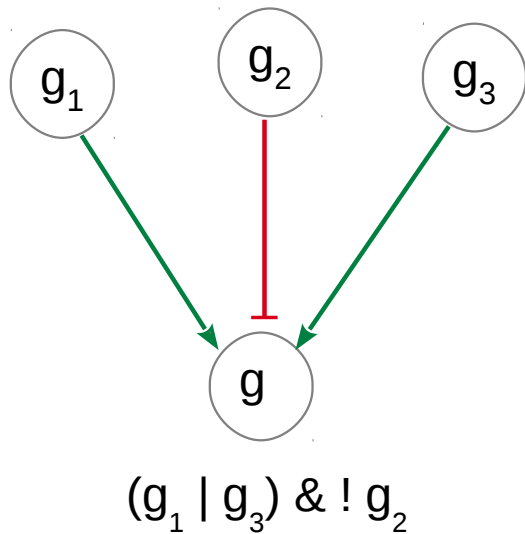
# Features overview

- Graphical User Interface
- Edit the Regulatory Graph
- Simulation: compute the State Transition Graph
- Analysis tools
- Exports

# Construct a Regulatory Graph

- Add Components and Interactions
  - Manual layout and visual settings
- Annotations
  - Needs improvements, rely more on SBML standards
- Real model: logical parameters
- Perturbations

# Reminder on Logical Parameters



g1	g2	g3	g
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

Logical parameters
$\emptyset$ (basal value)
<b>g3</b>
g2
g2,g3
<b>g1</b>
<b>g1,g3</b>
g1,g2
g1,g2,g3

logical parameter = List of active interactions

1 parameter = 1 line in the truth table

# Simulation

- Updating mode
  - (a)synchronous
  - Priority classes
- Selected Initial State(s)
- Apply perturbation
- Regular or “compact” result (Hierarchical graph)



# Perturbations

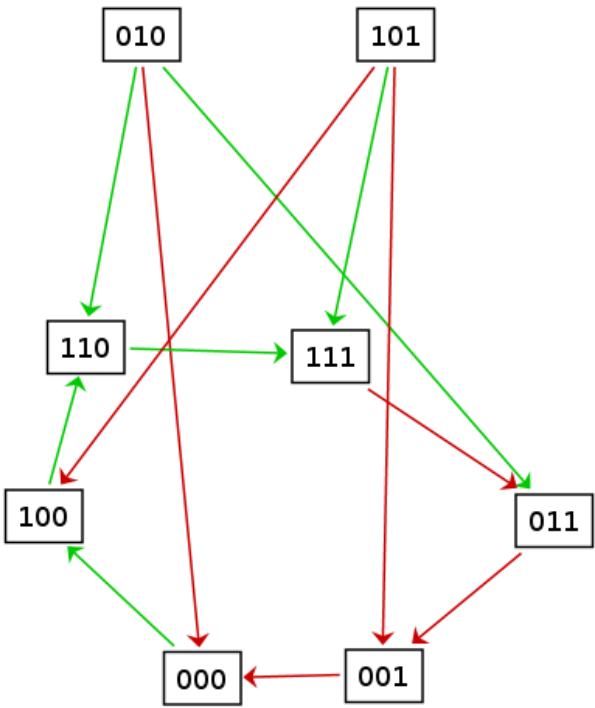
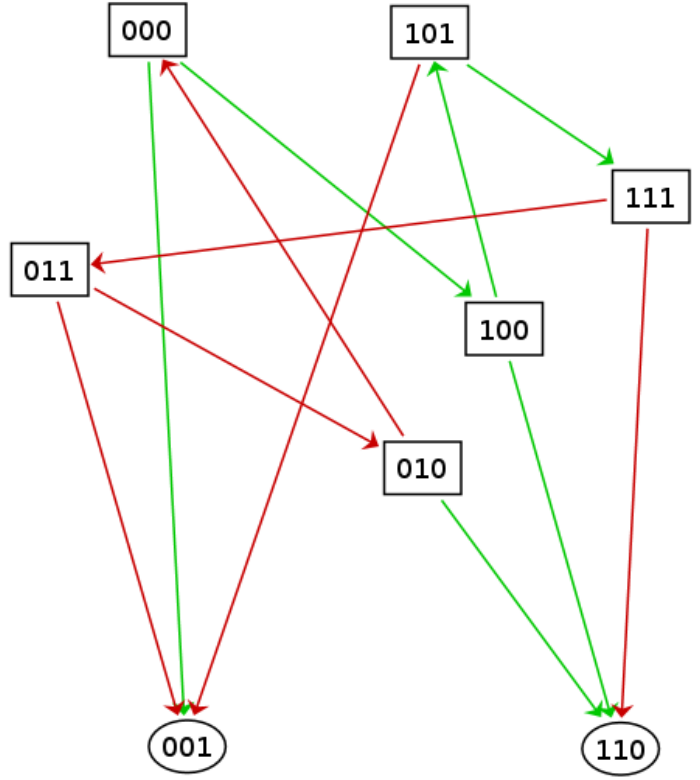
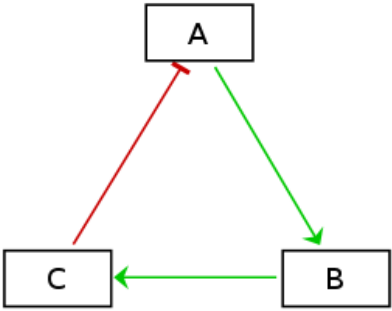
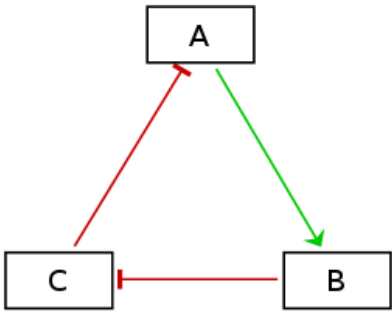
Anything which changes the model :)

- Fix the value of a component (KO, ectopic expression)
- Restrict a component in a range (multivalued case)
- Remove an interaction  
(block a component only for one of its targets)
- Multiple perturbations

# Circuit Analysis

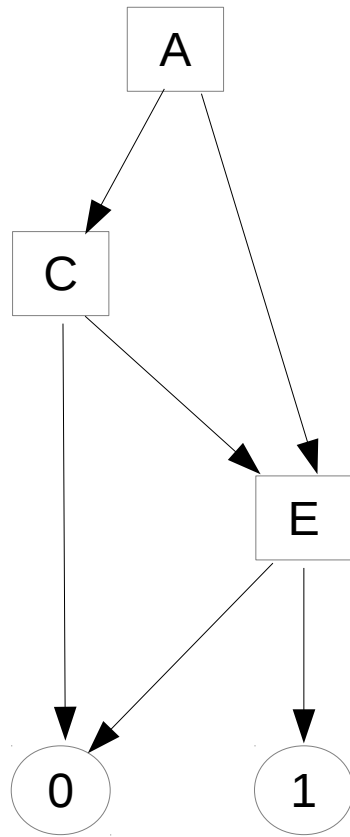
Positive: multiple attractors

Negative: oscillations



# Decision Diagrams (MDD)

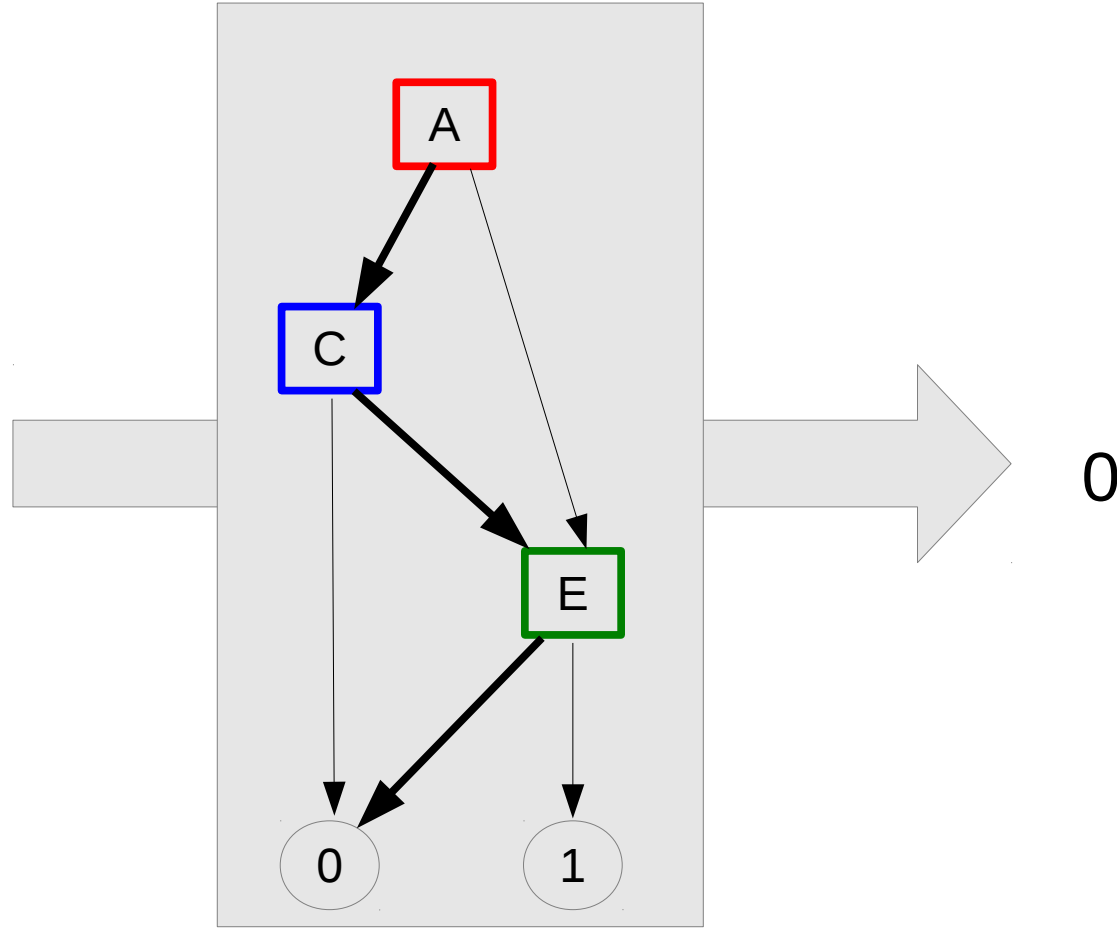
(A or C) and E



- A,B,C,D,E,F: ordered
- Fast (at most 6 tests, here 3)
- Predictable
- Picking order is hard

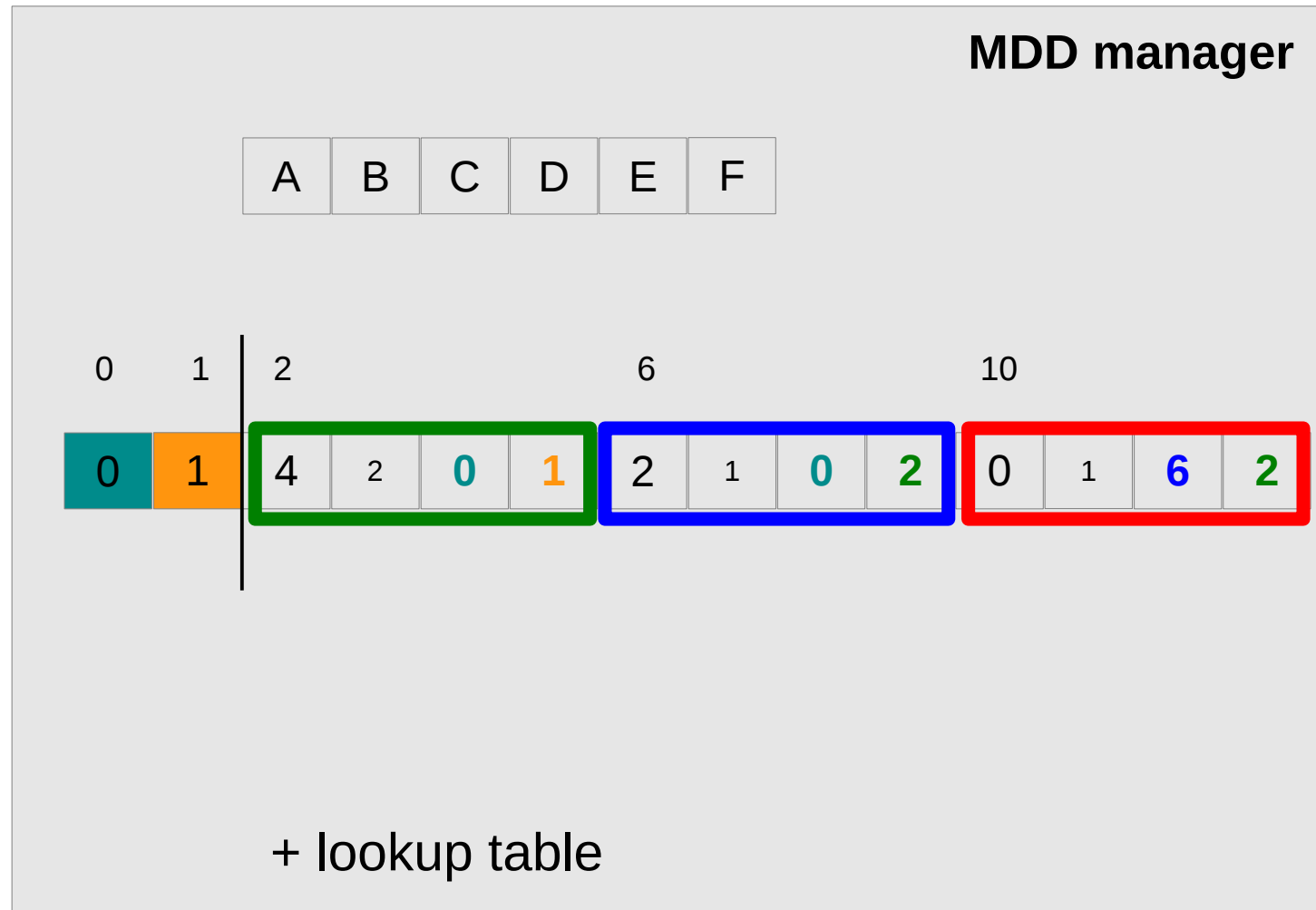
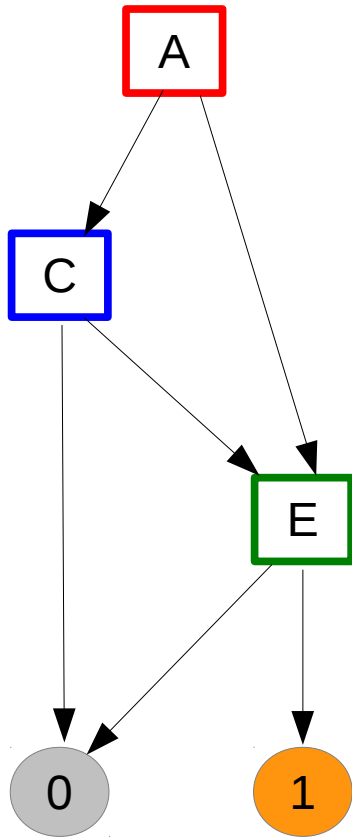
# MDD query

A	0
B	1
C	1
D	1
E	0
F	0



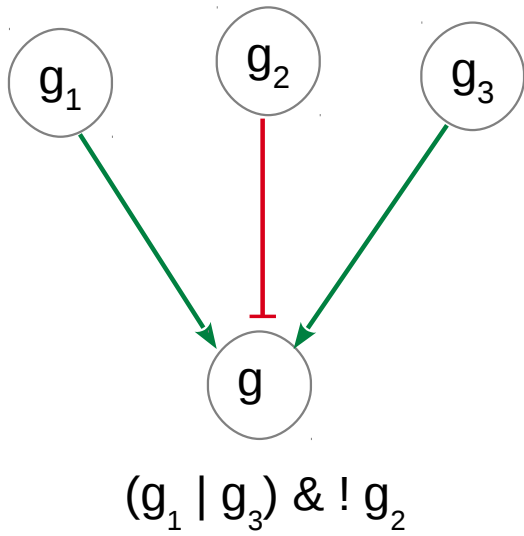
# Decision Diagrams (MDD)

(A or C) and E

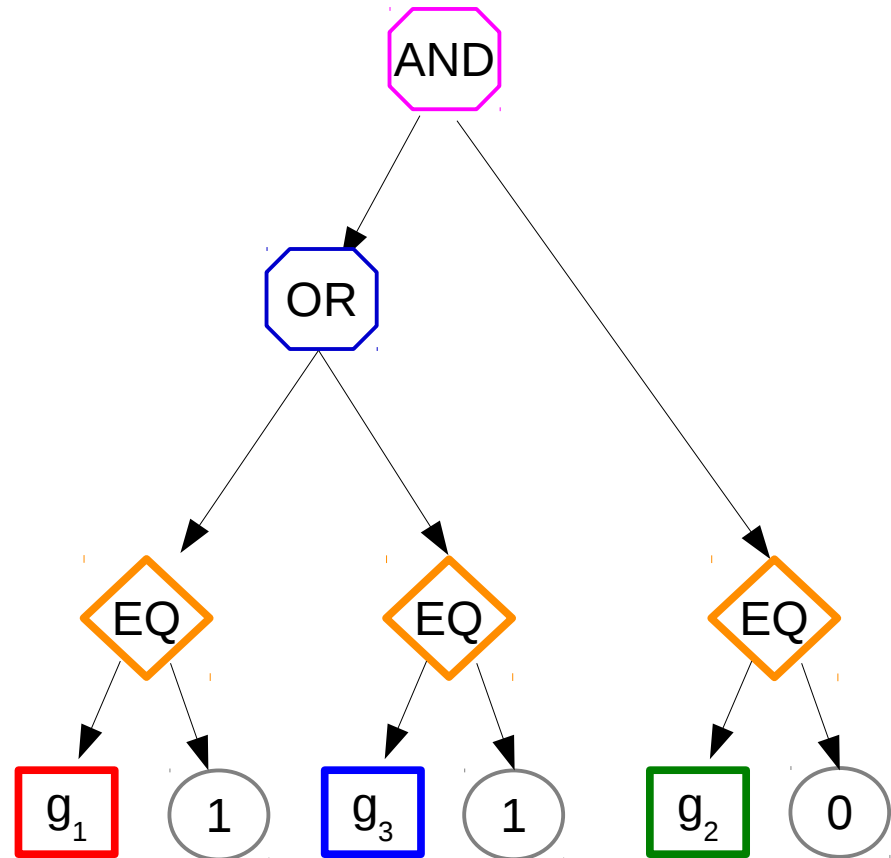


Multi-valued: some variables have more than two children

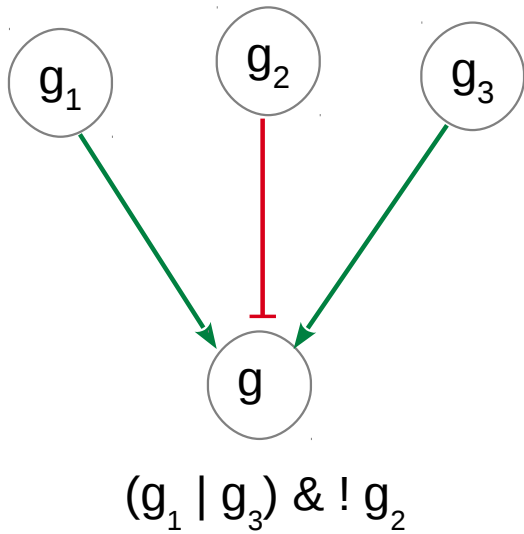
# Functions in SBML-qual (and other)



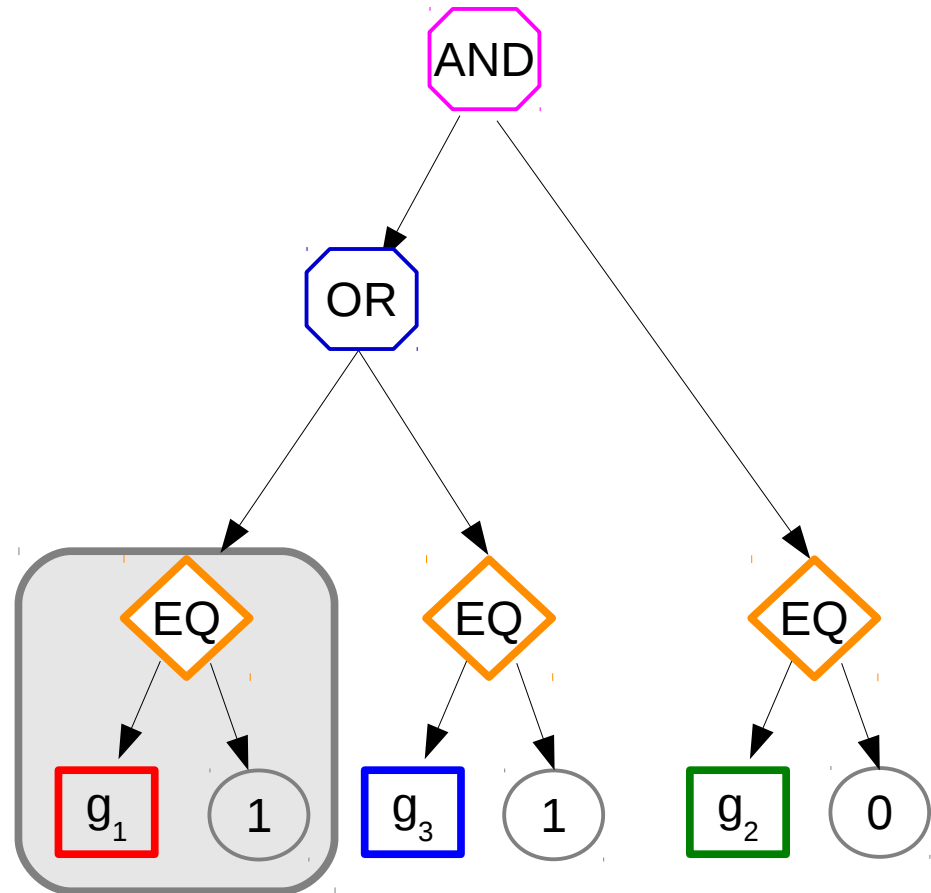
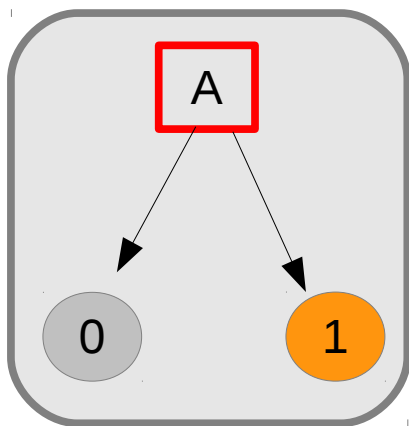
$(g_1=1 | g_3=1) \& g_2=0$



# Functions in SBML-qual (and other)

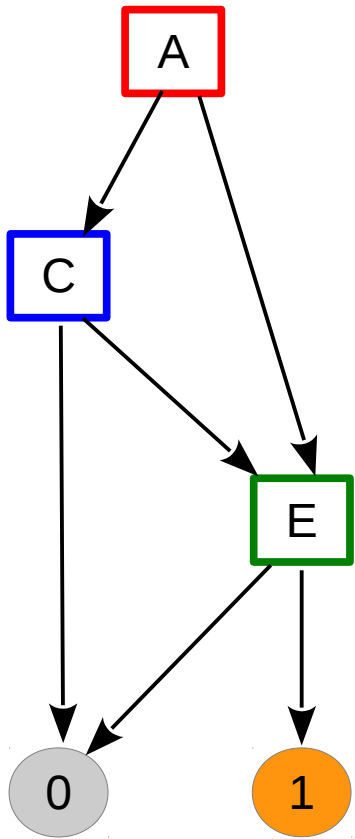


$$(g_1=1 | g_3=1) \& g_2=0$$

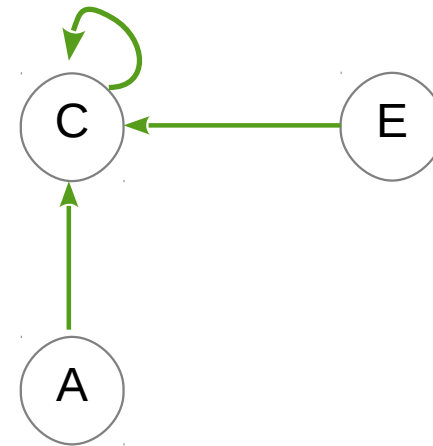


MDD manipulation for “OR”, “AND” operators

# Identification of stable states



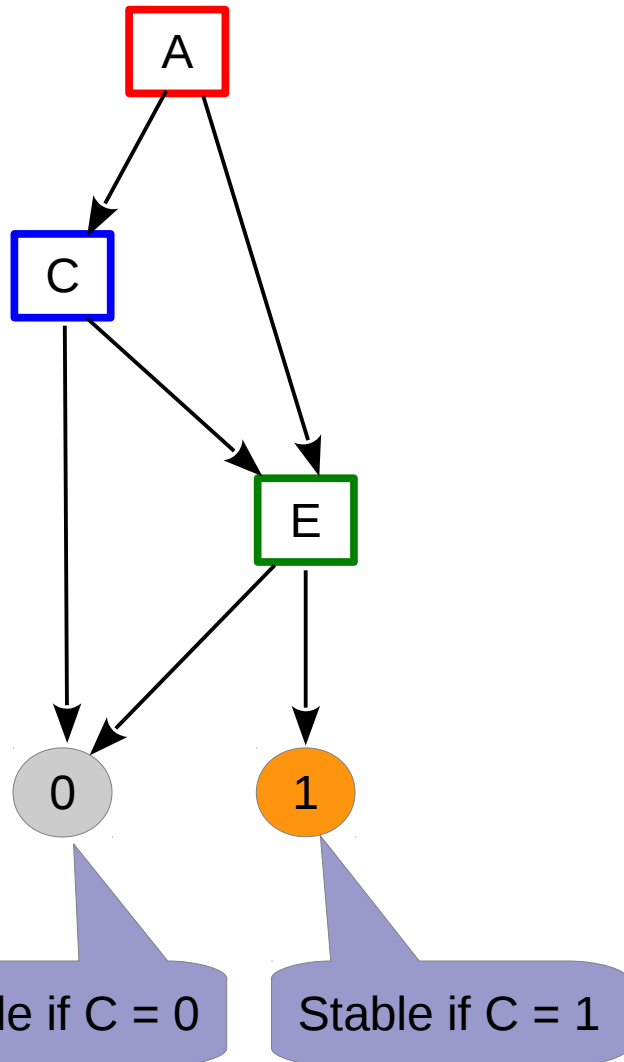
Function for C



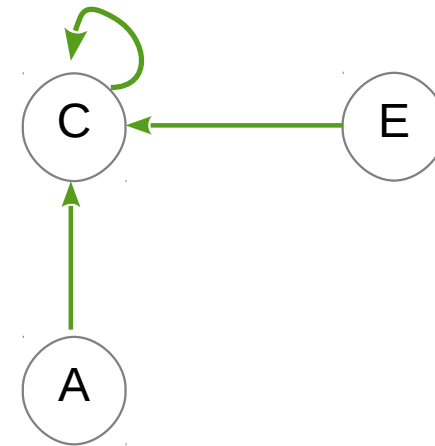
**(A | C) & E**



# Identification of stable states

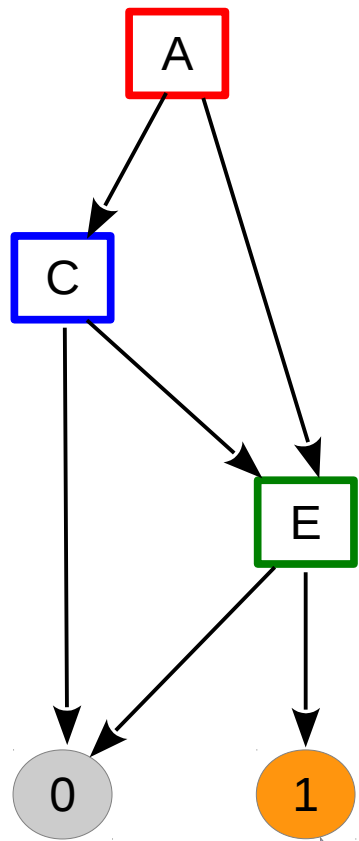


Function for C



**(A | C) & E**

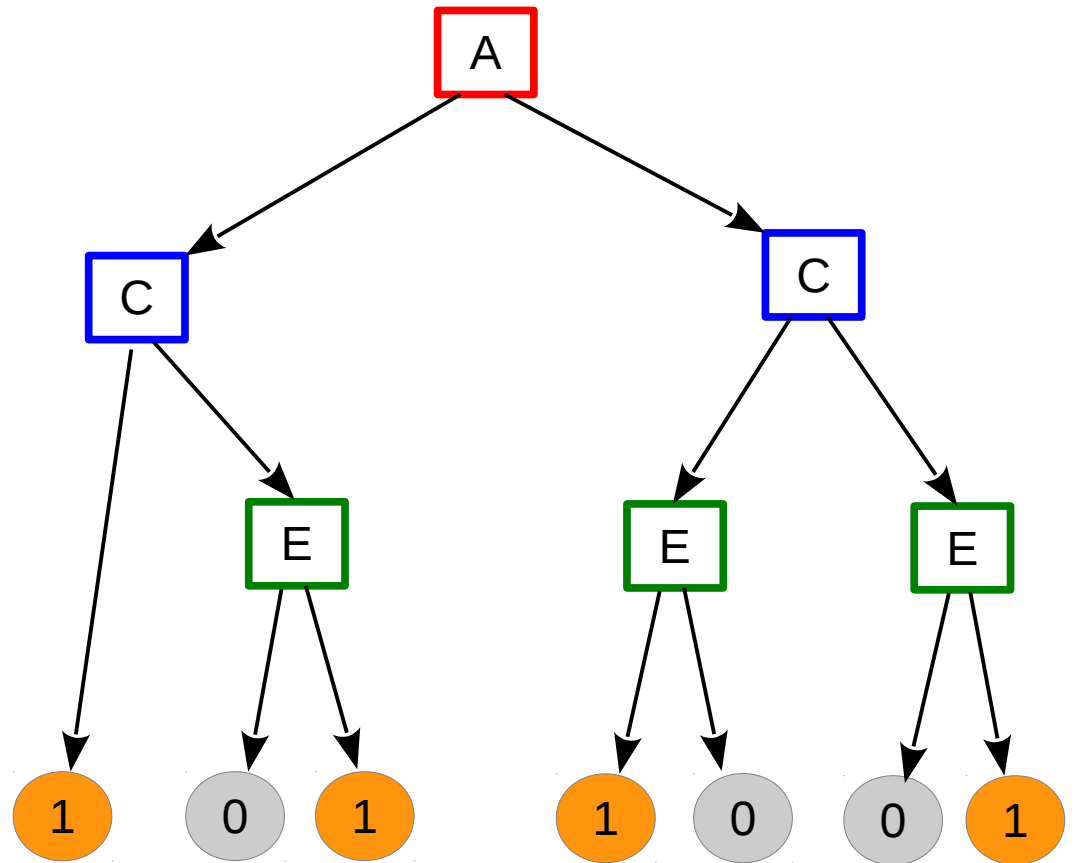
# Identification of stable states



Stable if  $C = 0$

Stable if  $C = 1$

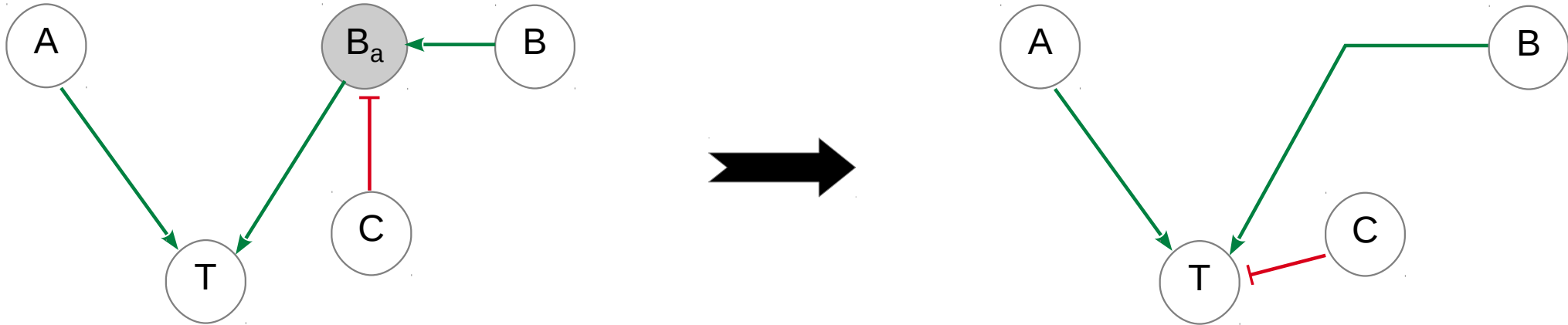
$(A \mid C) \ \& \ E$



Stability condition for C

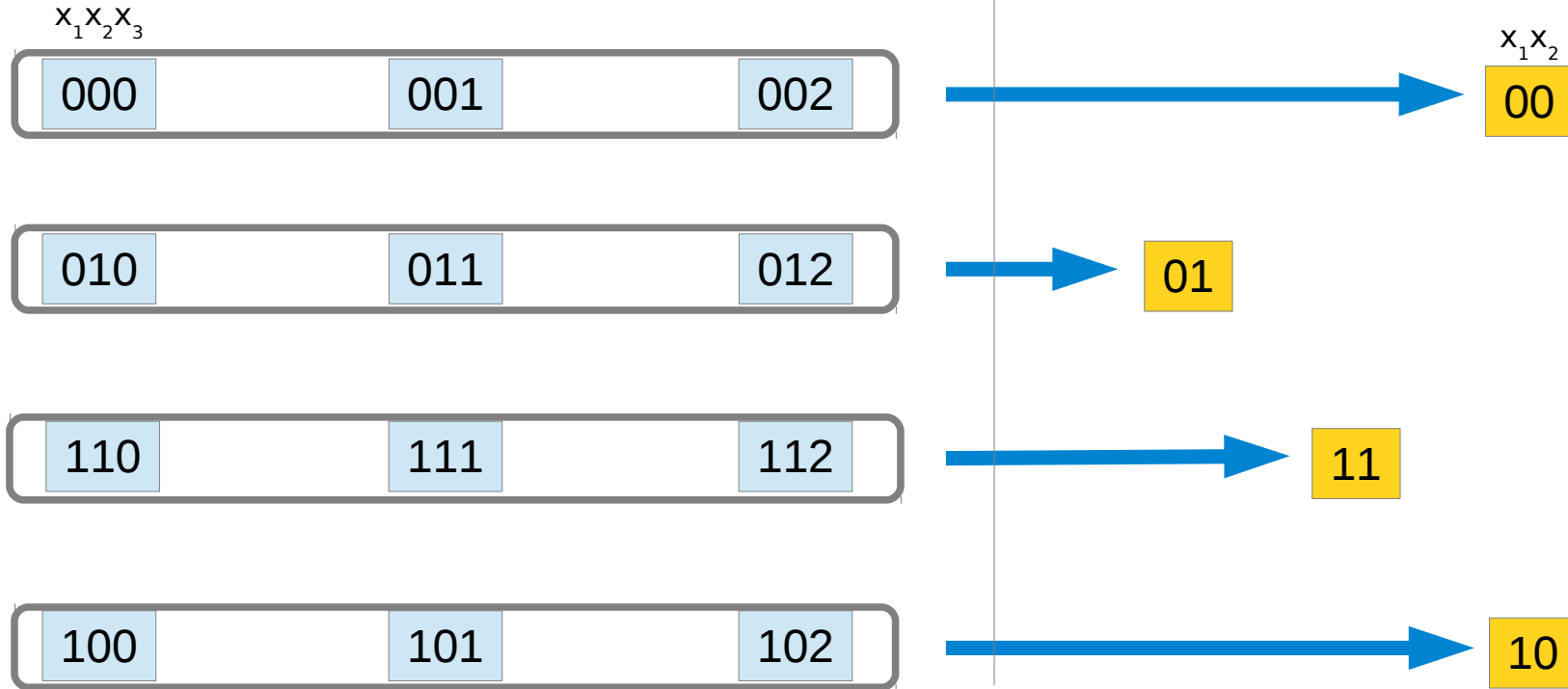
$(\neg C \ \& \ \neg(A \ \& \ E)) \ \mid \ (C \ \& \ E)$

# Model reduction

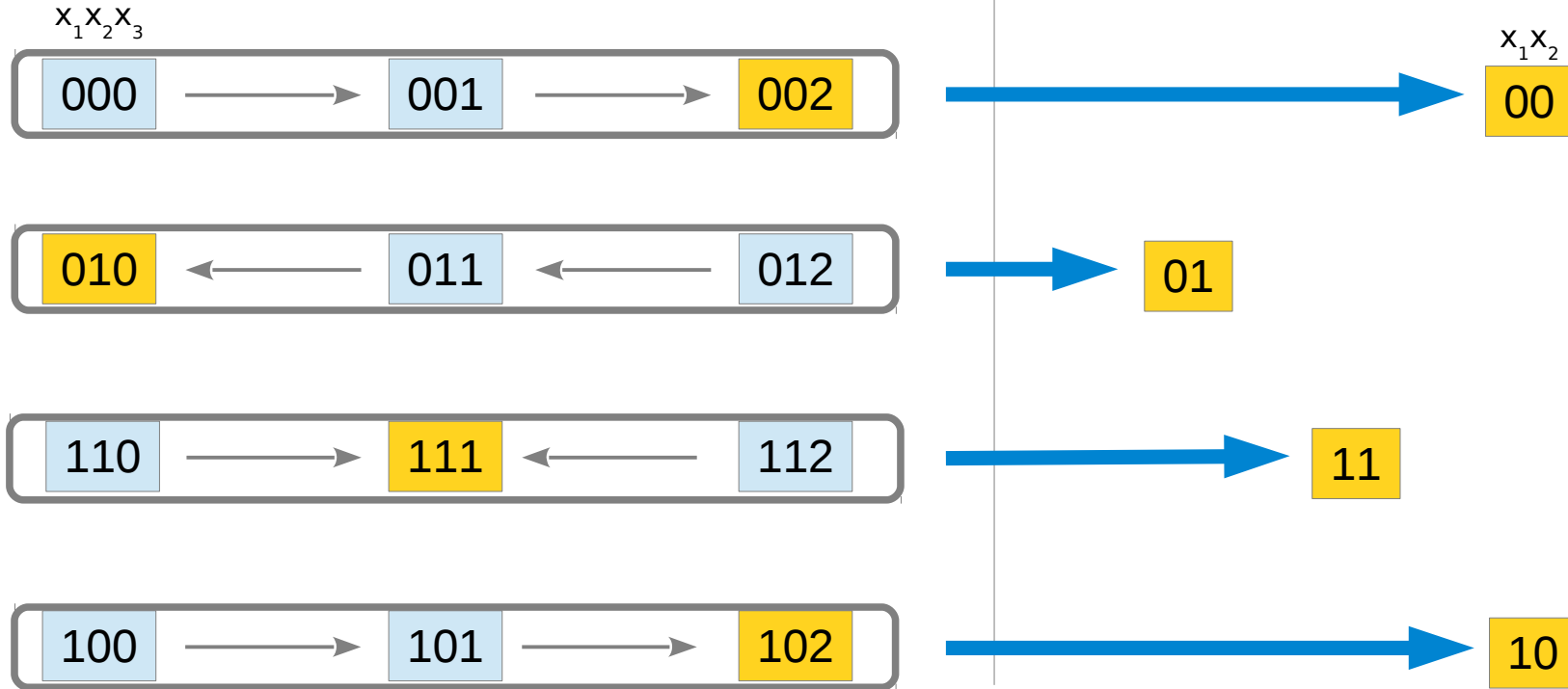


- Construct an explicit model
  - Algorithm to “hide” some components
    - Transfer the role of regulators into targets
    - Preserve dynamical properties
- Self-regulated components are protected

# Impact on dynamics



# Impact on dynamics

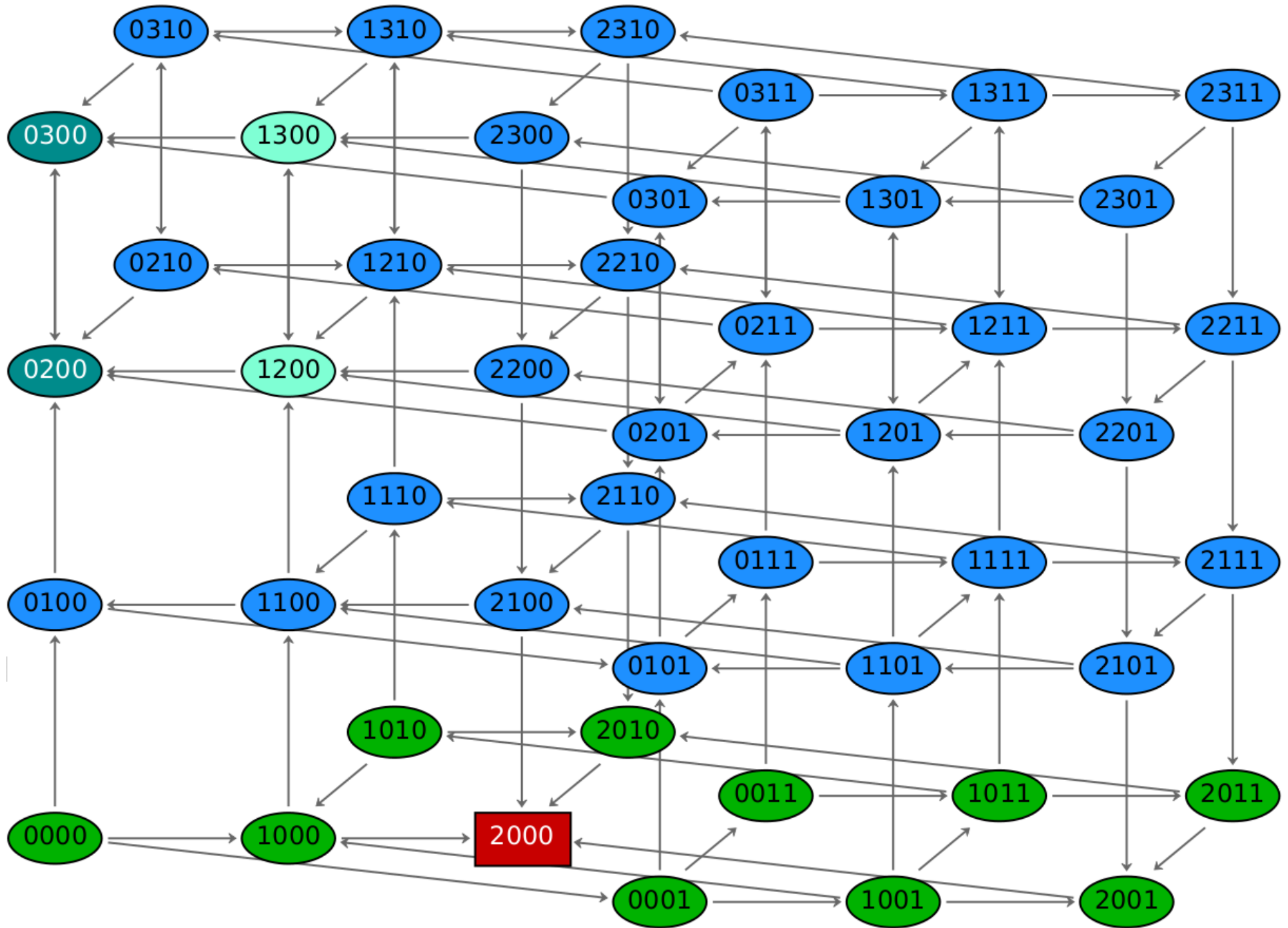




# Properties of the reduction

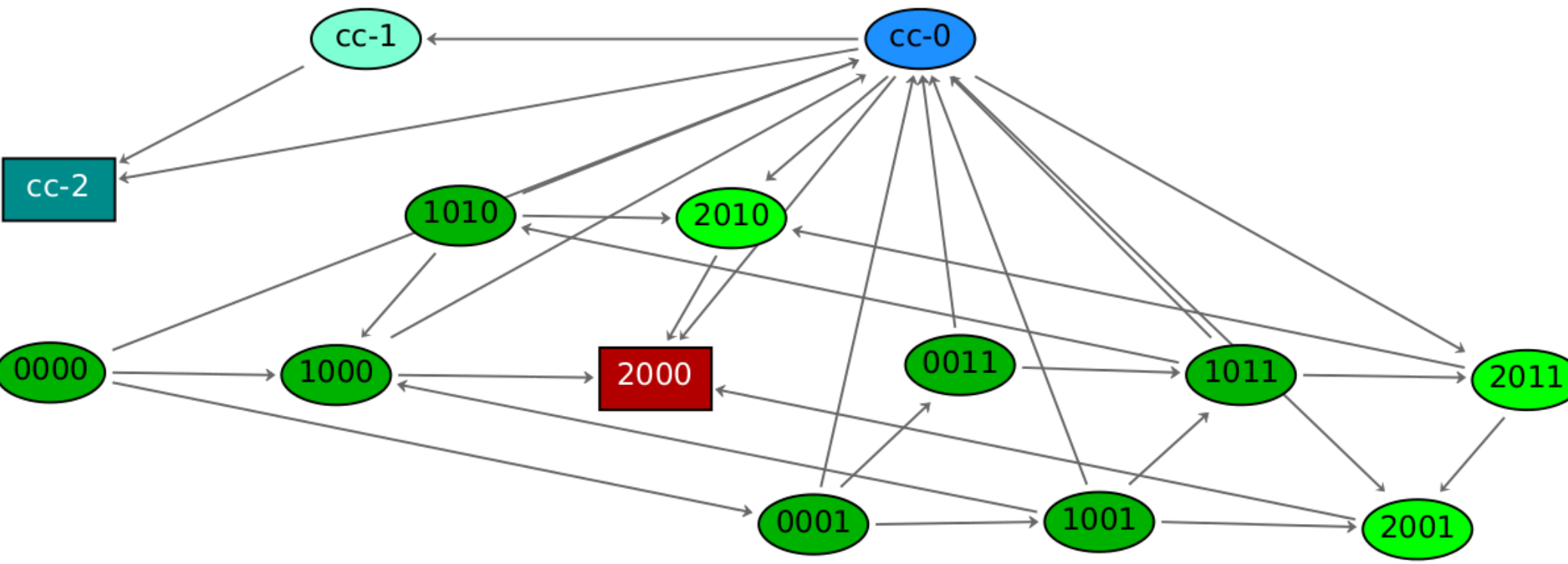
- Dynamics similar to priority classes (hidden = fast)
- Representative states
- Stable states are identical
- Reachability can be lost, not made-up
- Complex attractors are “preserved”
  - Can be split, transient cycles can become terminal
- Some “safe” reductions: no transition lost

# Hierarchical Graph

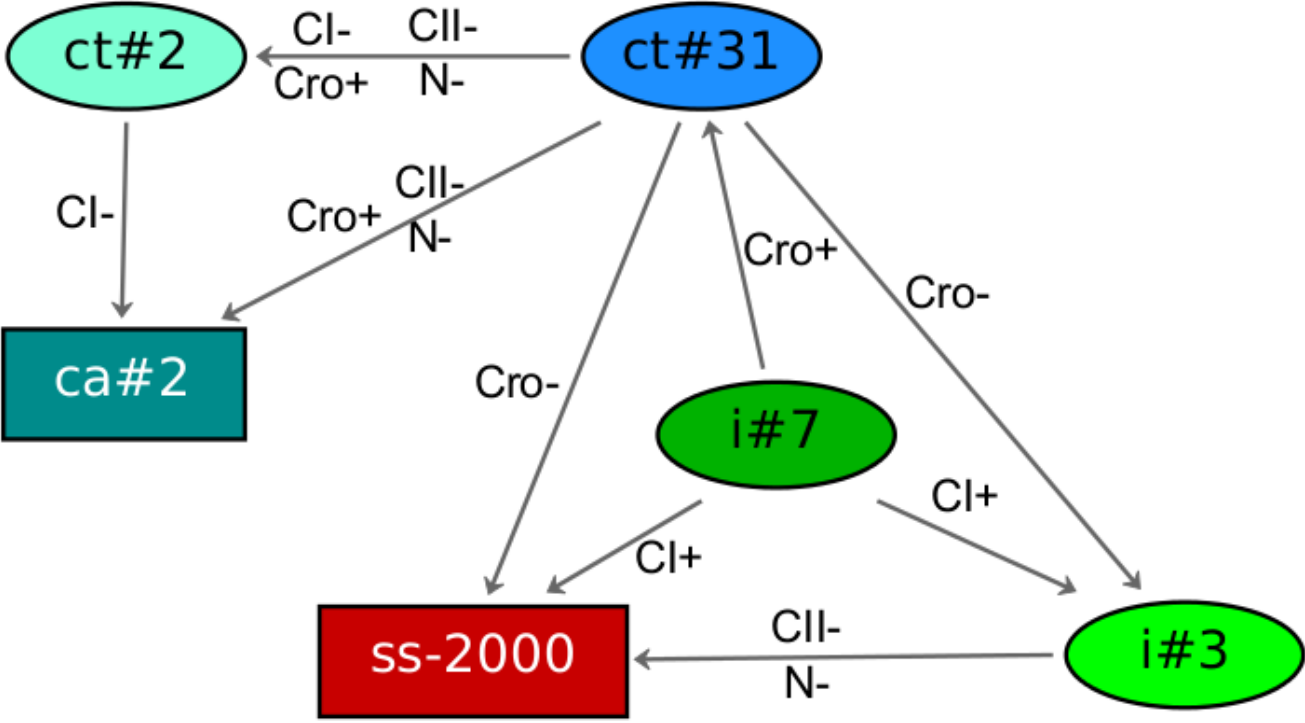




# Hierarchical Graph



# Hierarchical Graph



# Model Composition

The screenshot displays a software interface for model composition. The main window shows a graph with three nodes: Delta, Notch, and Delta\_ext. Delta is connected to Notch by a red line, and Notch is connected to Delta\_ext by a green line. A 'Specify Composition parameters' dialog is open, showing 7 instances (M1-M7) and their connections. The dialog also shows the integration function for Delta\_ext as AND(Delta).

**Specify Composition parameters**

Number of Instances: 7

Specify Neighbouring Modules:

	M1	M2	M3	M4	M5	M6	M7
M1	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
M2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
M3	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
M4	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
M5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
M6	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
M7	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Symmetric topology  
 Non-symmetric topology

Specify Integration Function for Inputs:

Delta\_ext AND

Delta  
Notch

Delta\_ext : AND(Delta)

Reduce mapped input components

Close Compose instances

# Model Composition

Specify Composition parameters

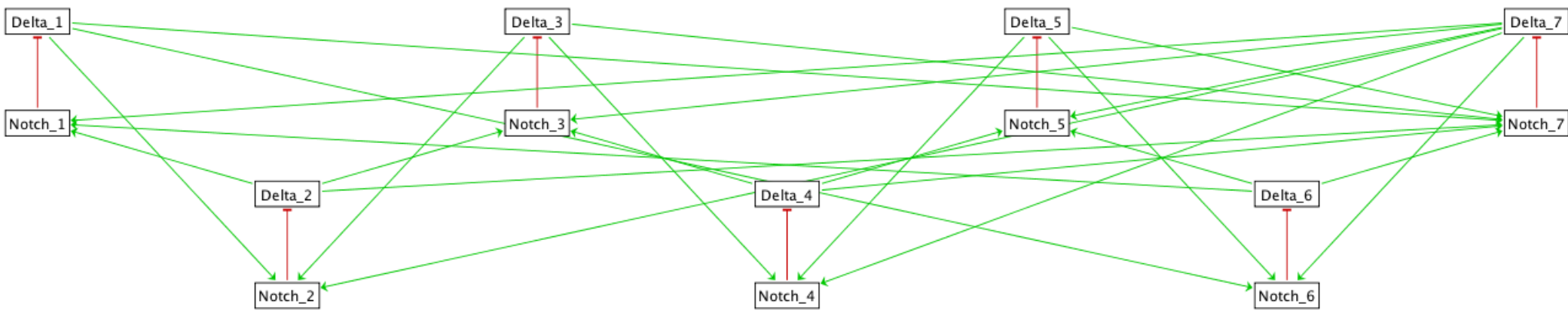
Number of Instances: 7

Specify Neighbouring Modules:

	M1	M2	M3	M4	M5	M6	M7
M1	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
M2	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
M3	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
M4	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
M5	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
M6	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
M7	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

Specify Integration Function for Inputs:

Delta\_ext AND  Delta  Notch



# Exports

- Image of the graph (SVG, PNG)
- Documentation
- Petri net
- Model checker
- SBML qual
- Others: GNA, boolsim

# Upcoming in GINsim 3.0

- Large architecture change
  - Backend: Core, Services
  - GUI (core and services)
  - Much improved scripting (using Jython)
- Backend relies on LogicalModel lib
  - Better MDD implementation
- New view canvas and styles
- Imports
  - SBML qual, Truth table, boolsim

# Future plans

- Improve annotation support
- Function editor
  - Automatic functions
  - Improve hand-crafted functions
- Allow external extensions

# People



TAGC (INSERM)  
Marseille, France



# People



IBENS  
Paris, FR



IGC & INESC  
Lisbon, PT



CIG - UNIL  
Lausanne, CH