Model-checking logical models of large regulatory networks

Pedro T. Monteiro[1,3] Aurélien Naldi[2] Claudine Chaouiya[1]

[1]Instituto Gulbenkian de Ciência - Oeiras, PT

[2]Center for Integrative Genomics, UNIL - Lausanne, CH

[3]INESC-ID - Lisbon, PT

3rd CoLoMoTo meeting @ Lausanne, CH
April 18, 2014

FUNDAÇÃO CALOUSTE GULBENKIAN
Instituto Gulbenkian de Ciência

UNIL | Université de Lausanne

Outline

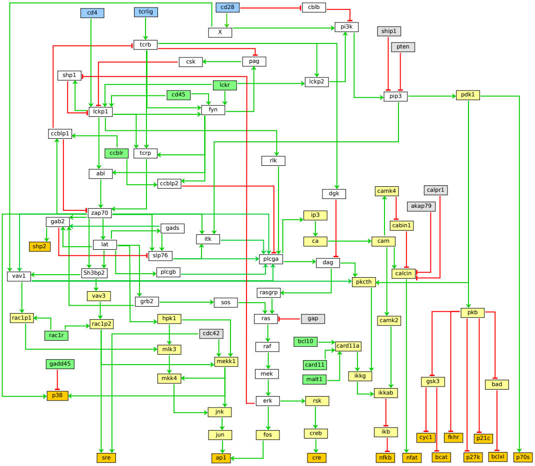
1 *Introduction*

2 *Reduction/NuSMV enconding*

3 *Application*

4 *Conclusions and Prospects*

General motivation: study large biological networks



(Saez-Rodriguez *et al.*, *PLoS Comput. Biol.* 2007)

- Signalling pathways, regulatory modules
- Lack of quantitative data
- ON/OFF mechanisms, thresholds

$\Rightarrow$ Discrete modelling

Discrete modelling: logical formalism (Thomas and d'Ari, *Biological Feedback* 1989)

Discrete modelling: logical formalism (Thomas and d'Ari, *Biological Feedback* 1989)

Logical regulatory graph (LRG) $\mathcal{R} = (\mathcal{G}, K)$

- $\mathcal{G} = \{g_i\}_{i=0,\dots,n}$ is a set of regulatory components
- $Max : \mathcal{G} \to \mathbb{N}^*$ associates a maximum level M_i to each component g_i
- $\mathcal{S} = \prod_{g_i \in \mathcal{G}} D_i$: is the state space, where $D_i = \{0, \dots, Max(g_i)\}$
- $\forall g_i : K_i : \mathcal{S} \to D_i$ is the regulatory function specifying the behaviour of g_i

Discrete modelling: logical formalism (Thomas and d'Ari, *Biological Feedback* 1989)

Logical regulatory graph (LRG) $\mathcal{R}$= ($\mathcal{G}$, K)

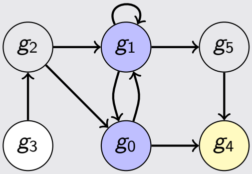
- $\mathcal{G} = \{g_i\}_{i=0,\ldots,n}$ is a set of regulatory components
- $Max : \mathcal{G} \rightarrow \mathbb{N}^*$ associates a maximum level M_i to each component g_i
- $\mathcal{S} = \prod_{g_i \in \mathcal{G}} D_i$: is the state space, where $D_i = \{0, \ldots, Max(g_i)\}$
- $\forall g_i : K_i : \mathcal{S} \rightarrow D_i$ is the regulatory function specifying the behaviour of g_i

State transition graph (STG)

The dynamic behaviour of an LRG, is represented by an STG where:

- nodes are states in $\mathcal{S}$
- and arcs $(v, w) \in \mathcal{S}^2$ denote transitions between states

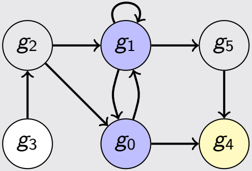
Toy example (Boolean)



$$
\begin{array}{ll}
K_0(v) = 1 & \textit{if } v_1 = 1 \vee v_2 = 1 \\
K_1(v) = 1 & \textit{if } v_0 = 1 \vee v_1 = 1 \vee v_2 = 1 \\
K_2(v) = 1 & \textit{if } v_3 = 1 \\
K_3(v) = \textit{input} & \textit{fixed or unconstrained} \\
K_4(v) = 1 & \textit{if } v_0 = 1 \vee v_5 = 1 \\
K_5(v) = 1 & \textit{if } v_1 = 1
\end{array}
$$

Logical formalism

Toy example (Boolean)



$$
\begin{aligned}
K_0(v) &= 1 && \textit{if } v_1 = 1 \vee v_2 = 1\\
K_1(v) &= 1 && \textit{if } v_0 = 1 \vee v_1 = 1 \vee v_2 = 1\\
K_2(v) &= 1 && \textit{if } v_3 = 1\\
K_3(v) &= \textit{input} && \textit{fixed or unconstrained}\\
K_4(v) &= 1 && \textit{if } v_0 = 1 \vee v_5 = 1\\
K_5(v) &= 1 && \textit{if } v_1 = 1
\end{aligned}
$$

Interesting properties

- What are the attractors of the system? (stable states, complex attractors)
- Are these attractors reachable from initial conditions?
- Are these attractors maintained under input variations?
- ...

Formal verification and model checking

1st objective: automate model verification

Confront model predictions with biological observations

Formal verification and model checking

1st objective: automate model verification

Confront model predictions with biological observations

Approach: use of formal verification techniques

Formal verification based on temporal logic and model checking provides a powerful technology to query models of interaction networks.

(Chabrier-Rivier *et al.*, *Theor Comput Sci* 2004) (Batt *et al.*, *Bioinformatics* 2005) (Monteiro *et al.*, *Bioinformatics* 2008)

Formal verification and model checking

1st objective: automate model verification

Confront model predictions with biological observations

Approach: use of formal verification techniques

Formal verification based on temporal logic and model checking provides a powerful technology to query models of interaction networks.

(Chabrier-Rivier *et al.*, *Theor Comput Sci* 2004) (Batt *et al.*, *Bioinformatics* 2005) (Monteiro *et al.*, *Bioinformatics* 2008)

Model checking

Fully automated exhaustive exploration of the state space of the model.

- Transform models into a Kripke structure $K = (S, AP, L, TR)$, where:
  - S are the states (Kripke, *Acta Phil. Fennica* 1963)
  - $TR \subseteq SxS$ the transition relation between states
  - $L : S \rightarrow 2^{AP}$ a state labeling function, with a set of atomic propositions true in that state (values of variables, signs of derivatives, ...)
- Specify dynamical properties as statements in temporal logic that are interpreted on state transition graph.

(Emerson and Clarke, *ICALP* 1980) (Queille and Sifakis, *Intl. Symp. Program.* 1982)

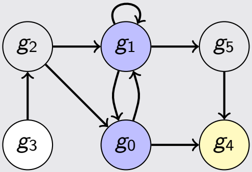
2nd objective: ease dynamical analysis

Deal with the combinatorial explosion!
$\rightarrow$ define methods to **safely reduce** the state space

2nd objective: ease dynamical analysis

Deal with the combinatorial explosion!
→ define methods to **safely reduce** the state space

Toy model characteristics



Type of components:

- 1 input component
- 1 pseudo-input component
- 2 core components
- 1 pseudo-output component
- 1 output component

Complete state transition graph has $2^6 = 64$ states

Outline

1 Introduction

2 Reduction/NuSMV enconding

3 Application

4 Conclusions and Prospects

Reduction?

Remove components: reduce complexity, control the dynamical impact

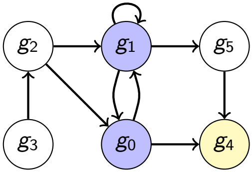
Reduction of output components

Reduction?

Remove components: reduce complexity, control the dynamical impact

Output reduction

- No computation
- No impact
- Retrieve values

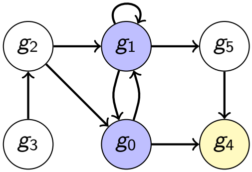


Reduction?

Remove components: reduce complexity, control the dynamical impact

Output reduction

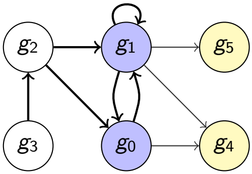
- No computation
- No impact
- Retrieve values



Extend to pseudo-outputs

- No impact
- Harder retrieval

⇒ Rewire the model: pseudo-outputs become outputs



Reduction of (pseudo-)outputs components

Implementation in GINsim

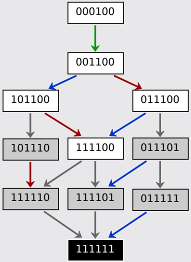
- Lossless reduction of (pseudo-)outputs
- Preservation of attractors and their reachability

Reduction of (pseudo-)outputs components

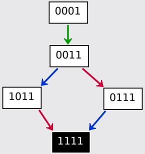
Implementation in GINsim

- Lossless reduction of (pseudo-)outputs
- Preservation of attractors and their reachability

Complete STG



STG with output reduction



- Generation of the STG without (pseudo-)output components
- Computation of all (pseudo-)output values on demand

Reduction of (pseudo-)outputs components

Implementation in NuSMV export

Effective representation for logical models:

- Symbolic model representation
- Combine different updating policies
- (Pseudo-)Outputs are:
  - Not part of the state description
    → reduction of the state-space
  - Defined as macros
    → computation of all (pseudo-)output values on demand

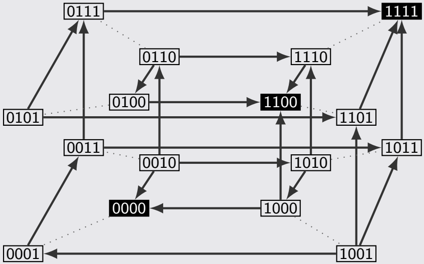
```
MODULE main
VAR
 properVar1 : { 0, 1 };
 ...
 properVari : { 0, 1 };

ASSIGN
next(properVar1) :=
  case
    logicalRule1 : 1;
    ...
    TRUE : 0;
  esac;
  ...
```

```
DEFINE
outputVar :=
  case
  logicalRule1 : 1;
    ...
  logicalRulej : 2;
    TRUE : 0;
  esac;
  ...
```

Reduction of inputs components

$$
\begin{array}{ll}
\mathcal{K}_0(v) = 1 & \textit{if } v_1 = 1 \vee v_2 = 1 \\
\mathcal{K}_1(v) = 1 & \textit{if } v_0 = 1 \vee v_1 = 1 \vee v_2 = 1 \\
\mathcal{K}_2(v) = 1 & \textit{if } v_3 = 1 \\
\mathcal{K}_3(v) = \textit{input} & \textit{fixed or unconstrained}
\end{array}
$$

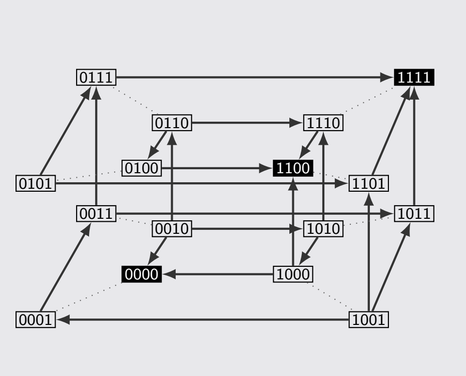
Complete STG



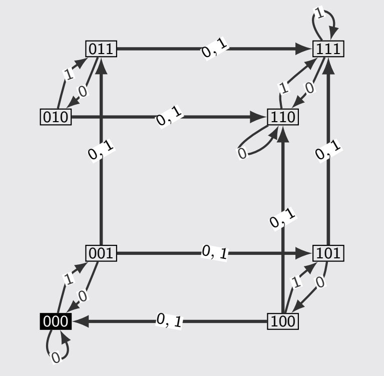
Reduction of inputs components

$$\begin{array}{ll} \mathcal{K}_0(v) = 1 & \text{if } v_1 = 1 \vee v_2 = 1 \\ \mathcal{K}_1(v) = 1 & \text{if } v_0 = 1 \vee v_1 = 1 \vee v_2 = 1 \\ \mathcal{K}_2(v) = 1 & \text{if } v_3 = 1 \\ \mathcal{K}_3(v) = \textit{input} & \textit{fixed or unconstrained} \end{array}$$

Complete STG



STG with input reduction



Reduction of inputs components

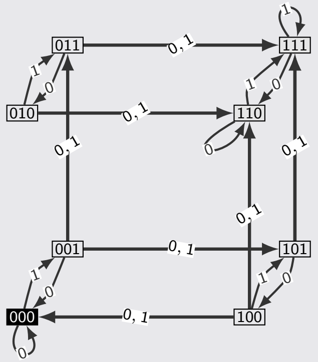
$$\begin{array}{ll} \mathcal{K}_0(v) = 1 & \textit{if } v_1 = 1 \lor v_2 = 1 \\ \mathcal{K}_1(v) = 1 & \textit{if } v_0 = 1 \lor v_1 = 1 \lor v_2 = 1 \\ \mathcal{K}_2(v) = 1 & \textit{if } v_3 = 1 \\ \mathcal{K}_3(v) = \textit{input} & \textit{fixed or unconstrained} \end{array}$$

Kripke transition system

```
IVAR
 G3 : { 0, 1 };

VAR
 G0 : { 0, 1 };
 G1 : { 0, 1 };
 G2 : { 0, 1 };
```

(Müller-Olm *et al.*, *SAS* 1999)

STG with input reduction



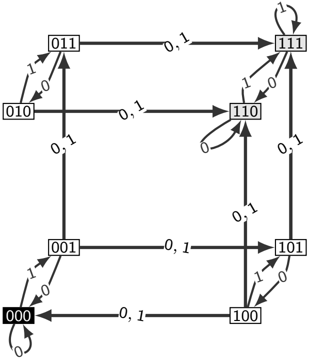
Reduction of inputs components: stable patterns

Types of stable states

- Strong stable state
- Weak stable state

Types of stable core ensembles

- Strong stable core ensemble
- Weak stable core ensemble

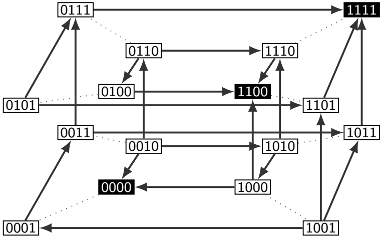


$$\mathcal{K}_0(v) = 1 \quad \textit{if } v_1 = 1 \vee v_2 = 1$$
$$\mathcal{K}_1(v) = 1 \quad \textit{if } v_0 = 1 \vee v_1 = 1 \vee v_2 = 1$$
$$\mathcal{K}_2(v) = 1 \quad \textit{if } v_3 = 1$$
$$\mathcal{K}_3(v) = \textit{input} \quad \textit{fixed or unconstrained}$$

Question: What's the impact of different switches of input conditions on the reachability of the biological attractors? and the system's behaviour?

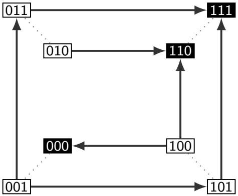
Pseudo-inputs components not subject to reduction

Reduction of pseudo-inputs can cause reachability problems!



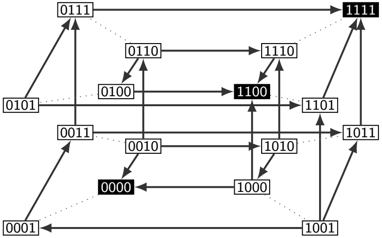
Pseudo-input reduction

$\Longrightarrow$

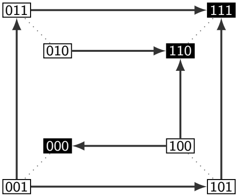


Pseudo-inputs components not subject to reduction

Reduction of pseudo-inputs can cause reachability problems!



Pseudo-input reduction $\Longrightarrow$



Lost transitions

- 000 $\not\rightarrow$ 010
- 000 $\not\rightarrow$ 100
- 001 $\not\leftarrow$ 101

Reduction of input/ouput components: implementation

NuSMV export

Approach: Use a Kripke Transition System, representing information both:

- on states (core + pseudo-input components)
- on transitions (input components)

Advantages:

- Implicit representation of the model
- Reduction of (pseudo-)outputs by defining them as macros
- Projection of input components over transitions

In GINsim, input components remain (constant) part of state characterization

Checking reachability properties with NuSMV

Considered temporal logics

- Computation Tree Logic (CTL)
  Verifying properties with all unconstrained inputs
- Action Restricted CTL (AR-CTL) (Pecheur and Raimondi, *MoChArt* 2006)
  Verifying properties with some (or all) fixed inputs

Checking reachability properties with NuSMV

Considered temporal logics

- Computation Tree Logic (CTL)
  Verifying properties with all unconstrained inputs
- Action Restricted CTL (AR-CTL) (Pecheur and Raimondi, *MoChArt* 2006)
  Verifying properties with some (or all) fixed inputs

Example of reachability properties

With unconstrained inputs (CTL):

- `INIT s000;`
  `EF(s110);`

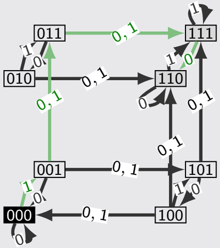
Considered temporal logics

- Computation Tree Logic (CTL)
  Verifying properties with all unconstrained inputs
- Action Restricted CTL (AR-CTL) (Pecheur and Raimondi, *MoChArt* 2006)
  Verifying properties with some (or all) fixed inputs

Example of reachability properties

With unconstrained inputs (CTL):

- `INIT s000;`
  `EF(s110);`



Checking reachability properties with NuSMV

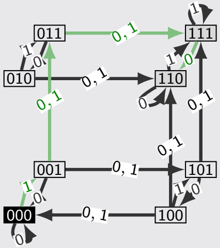
Considered temporal logics

- Computation Tree Logic (CTL)
  Verifying properties with all unconstrained inputs
- Action Restricted CTL (AR-CTL) (Pecheur and Raimondi, *MoChArt* 2006)
  Verifying properties with some (or all) fixed inputs

Example of reachability properties

With unconstrained inputs (CTL):

- `INIT s000;`
  `EF(s110);`



With fixed inputs (AR-CTL):

- `INIT s000;`
  EAF(g_3=1)(s110);

Checking reachability properties with NuSMV

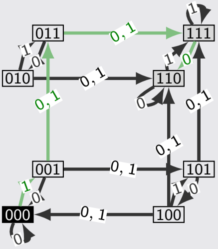
Considered temporal logics

- Computation Tree Logic (CTL)
  Verifying properties with all unconstrained inputs
- Action Restricted CTL (AR-CTL) (Pecheur and Raimondi, *MoChArt* 2006)
  Verifying properties with some (or all) fixed inputs

Example of reachability properties

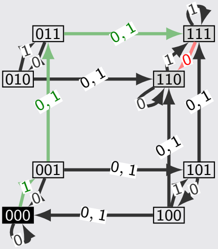
With unconstrained inputs (CTL):

- `INIT s000;`
  `EF(s110);`



With fixed inputs (AR-CTL):

- `INIT s000;`
  EAF(g_3=1)(s110);



Outline

1 Introduction

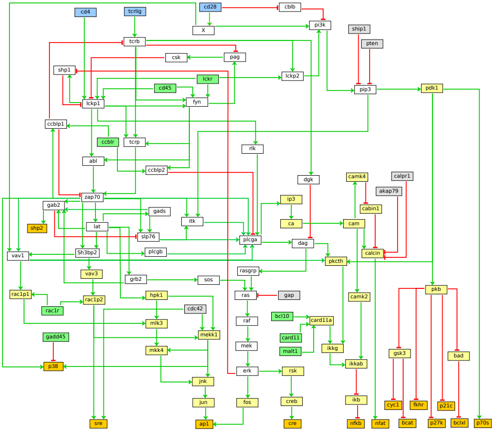
2 Reduction/NuSMV enconding

3 Application

4 Conclusions and Prospects

TCR activation model

T-cell activation through TCR is a key part of the specific immune response



(Saez-Rodriguez, *PLoS Comp. Biol.* 2007)

LRG with 91 components

3 inputs, 14 fixed-inputs

35 core components

28 pseudo-outputs, 14 outputs

STG before reductions

$2^3 = 8$ disconnected STGs

$2^{88} = 3 \times 10^{26}$ states for each

Size: 2.5×10^{27} states

STG after reductions

1 single compacted STG!

Size: $2^{35} = 3.4 \times 10^{10}$ states

TCR activation model: structure of the dynamics

Approach

Impose combinations of fixed inputs (AR-CTL), to test reachability properties:

- From the initial state towards the attractors
- Between all the attractors

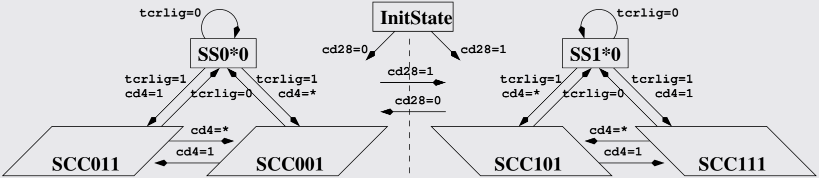
TCR activation model: structure of the dynamics

Approach

Impose combinations of fixed inputs (AR-CTL), to test reachability properties:

- From the initial state towards the attractors
- Between all the attractors

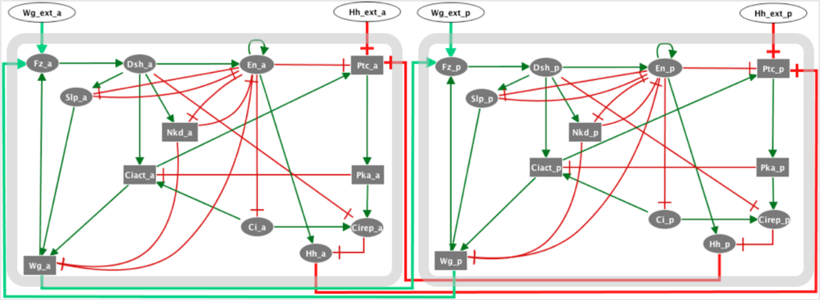
Necessary input conditions to switch between attractors



- cd28: creates a separation on the state space
- tcrlig=0: system evolves towards a stable state
- tcrlig=1: system evolves towards a complex attractor
- cd4: augments the size of the complex attractor

Segment-polarity model in Drosophila: 4 input variables

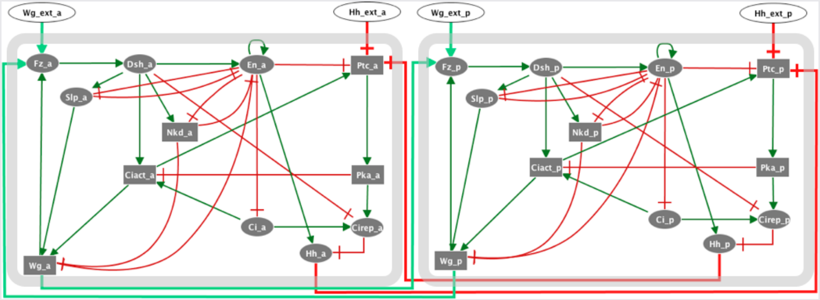
Inputs accounting for other neighbouring cells



(Sánchez *et al*, *Intl J. Dev. Biol* 2008)

Segment-polarity model in Drosophila: 4 input variables

Inputs accounting for other neighbouring cells

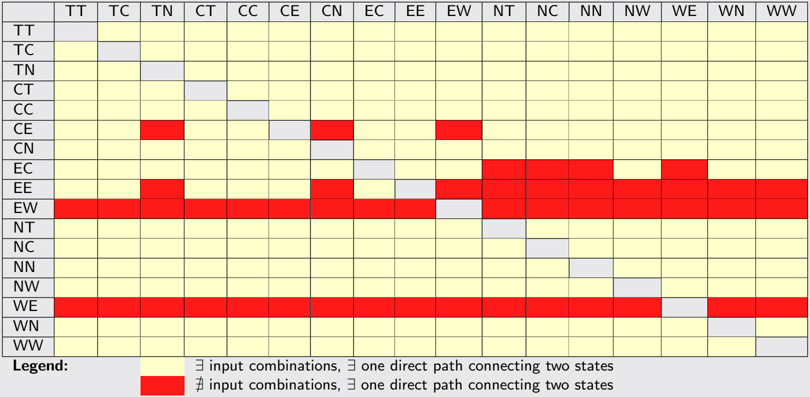


(Sánchez *et al*, *Intl J. Dev. Biol* 2008)

External inputs														
Wg	Hh	Wg	Fz	Dsh	Slp	Nkd	En	Hh	Ci	Ciact	Cirep	Pka	Ptc	Letter code
0	0	0	0	0	0	1	0	0	1	0	1	2	1	**T** (trivial)
0	1	0	0	0	0	1	0	0	1	1	0	0	0	**C** (CiCiact)
0	1	2	1	1	1	2	0	0	1	2	0	0	0	**W** (Wg)
1	0	0	1	1	0	0	1	1	0	0	0	0	0	**E** (En)
1	0	0	1	1	1	2	0	0	1	1	0	2	2	**N** (Nkd)
1	1	2	1	1	1	2	0	0	1	2	0	0	0	**W**
1	1	0	1	1	0	0	1	1	0	0	0	0	0	**E**

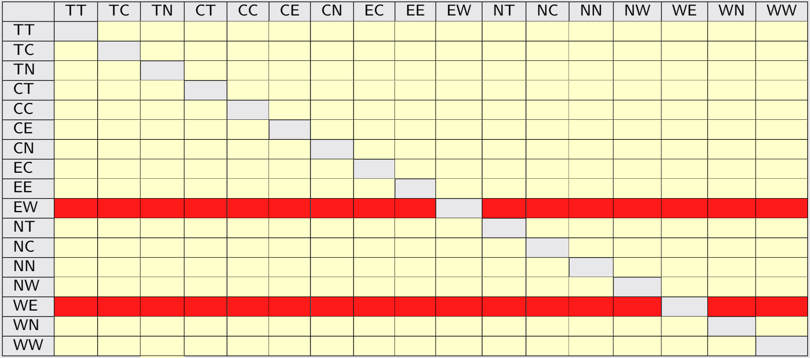
Segment-polarity model in Drosophila: 4 input variables

Stable patterns direct reachability: 4 fixed inputs



Segment-polarity model in Drosophila: 4 input variables

Stable patterns direct reachability: 4 varying inputs



Legend:
With varying inputs, $\exists$ one path connecting two states
With varying inputs, $\nexists$ one path connecting two states

Segment-polarity model in Drosophila: 4 input variables

Stable patterns direct reachability: 4 varying inputs

	TT	TC	TN	CT	CC	CE	CN	EC	EE	EW	NT	NC	NN	NW	WE	WN	WW
TT																	
TC																	
TN																	
CT																	
CC																	
CE																	
CN																	
EC																	
EE																	
EW																	
NT																	
NC																	
NN																	
NW																	
WE																	
WN																	
WW																	

Legend:
- With varying inputs, ∃ one path connecting two states
- With varying inputs, ∄ one path connecting two states

- Reduction of the state space without loss of information
- Identification of WE and EW patterns as strong stable states

Outline

1 Introduction

2 Reduction/NuSMV enconding

3 Application

4 Conclusions and Prospects

Conclusions

In GINsim

- Input components have constant values
- Lossless reduction of (pseudo-)output components
  Preserve attractors and reachability, compute output values on demand

Conclusions

In GINsim

- Input components have constant values
- Lossless reduction of (pseudo-)output components
  Preserve attractors and reachability, compute output values on demand

In the NuSMV export

- Symbolic representation
  Profit from NuSMV internal OMDD representation
- State space reduction:
  - Outputs defined as macros
  - Projection of input components on transitions

Conclusions

In GINsim

- Input components have constant values
- Lossless reduction of (pseudo-)output components
  Preserve attractors and reachability, compute output values on demand

In the NuSMV export

- Symbolic representation
  Profit from NuSMV internal OMDD representation
- State space reduction:
  - Outputs defined as macros
  - Projection of input components on transitions

Study of the structure of the system's dynamics

- Counterexample may contain information about necessary environmental conditions to ensure specific reachability properties
- Impact of input components on attractor switches:
  - Definition of strong/weak stable states
  - Definition of strong/weak stable core ensembles

Prospects

Study of the structure of the system's dynamics

- Automated uncovering of necessary conditions for attractor reachability

Complex attractors

- Efficient methods for complex attractor identification (without performing simulation)
- Complex attractor characterization in terms of strong/weak patterns

Thank you!

Funding

- FCT - Fundação para a Ciência e a Tecnologia
  - Postdoctoral grant (SFRH/BPD/75124/2010)
  - Project grant (PTDC/EIACCO/099229/2008)
- SystemsX - Swiss Initiative in Systems Biology
  - CycliX project grant

FUNDAÇÃO CALOUSTE GULBENKIAN
Instituto Gulbenkian de Ciência

UNIL | Université de Lausanne

Questions?!