Transient Dynamics of Automata Networks; Towards Cell Reprogramming

Loïc Paulevé

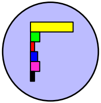
CNRS/LRI, Univ. Paris-Sud, Univ. Paris-Saclay – BioInfo team

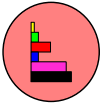
loic.pauleve@lri.fr

http://loicpauleve.name

CIRM - 4 January 2017

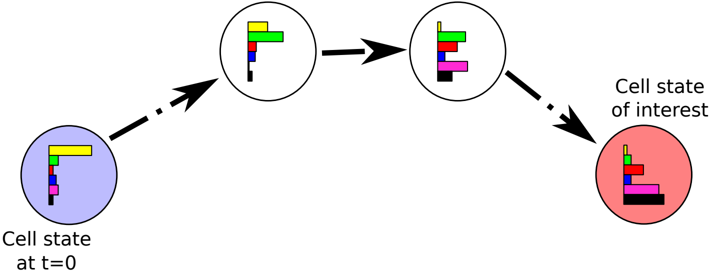
Transient Reachability





Initial state(s)/Goal state(s)

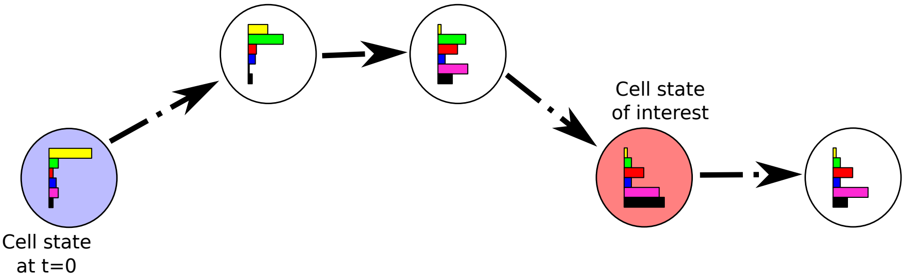
Transient Reachability



Initial state(s)/Goal state(s)

- Trajectory existence

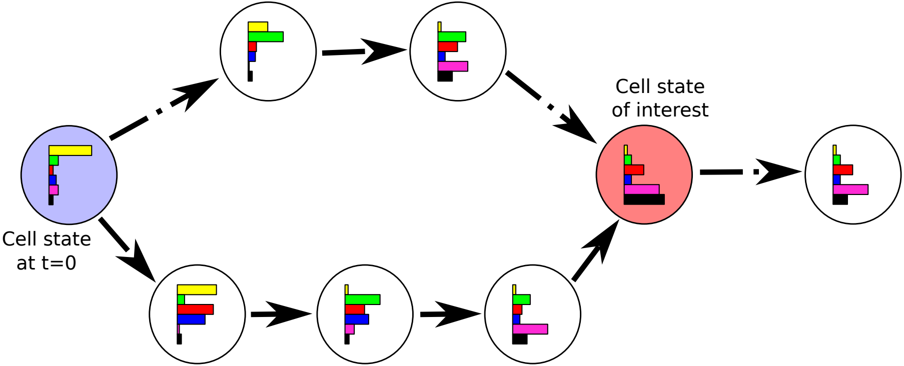
Transient Reachability



Initial state(s)/Goal state(s)

- Trajectory existence

Transient Reachability



Initial state(s)/Goal state(s)

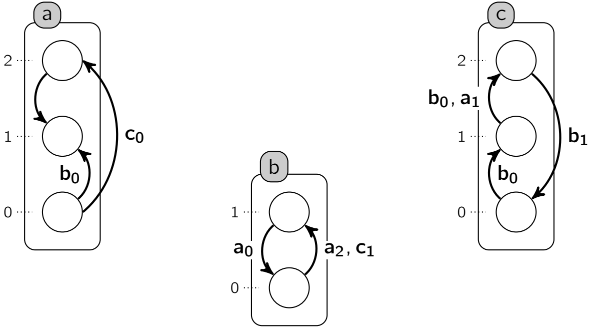
- Trajectory existence
- Reasoning on all trajectories

Outline

1. Automata Networks
2. Approximations of transient dynamics
   - Abstraction of traces
   - Reachability: cut sets, bifurcations
   - Model reduction preserving transient properties
   - Software Pint
3. Starting project: cell reprogramming

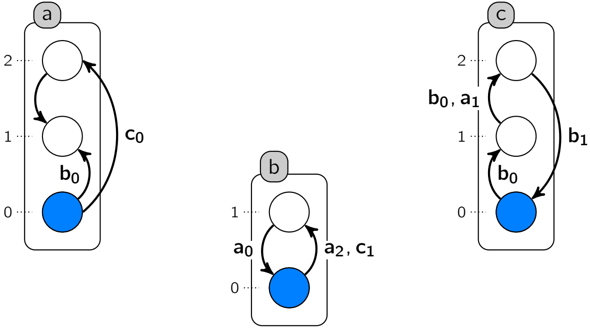
Outline

Automata Networks



Asynchronous semantics (one transition at a time):

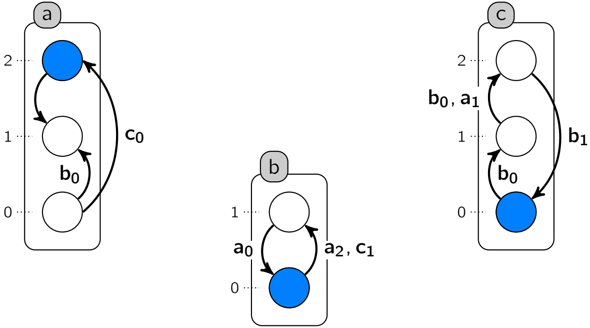
Automata Networks



Asynchronous semantics (one transition at a time):

$\langle a_0, b_0, c_0 \rangle$

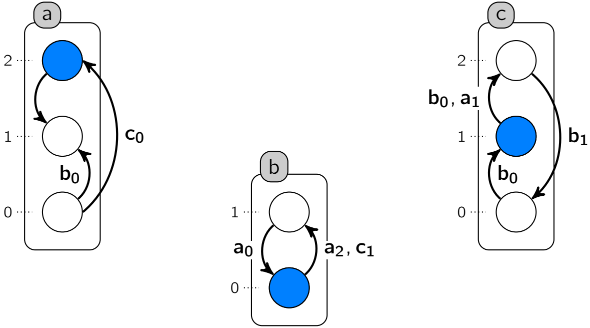
Automata Networks



Asynchronous semantics (one transition at a time):

$$\begin{array}{ccc} & & \langle a_2, b_0, c_0 \rangle \\ & \nearrow & \\ \langle a_0, b_0, c_0 \rangle & & \\ & \searrow & \\ & & \langle a_1, b_0, c_0 \rangle \end{array}$$

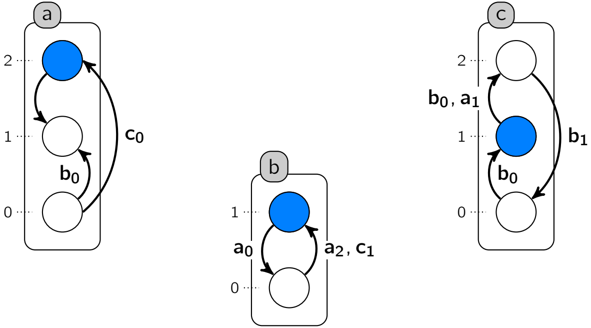
Automata Networks



Asynchronous semantics (one transition at a time):

$$\begin{array}{lll} & \langle a_2, b_0, c_0\rangle \longrightarrow \langle a_2, b_0, c_1\rangle \\ & \nearrow & \\ \langle a_0, b_0, c_0\rangle & & \\ & \searrow & \\ & \langle a_1, b_0, c_0\rangle & \end{array}$$

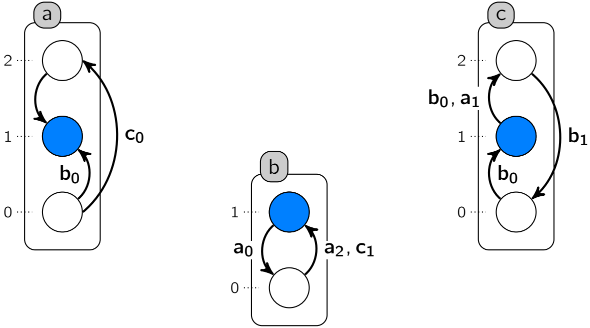
Automata Networks



Asynchronous semantics (one transition at a time):

$$
\begin{array}{lllll}
 & \langle a_2, b_0, c_0\rangle & \longrightarrow & \langle a_2, b_0, c_1\rangle \longrightarrow \langle a_2, b_1, c_1\rangle \\
 & \nearrow & & \\
\langle a_0, b_0, c_0\rangle & & & \\
 & \searrow & & \\
 & \langle a_1, b_0, c_0\rangle & &
\end{array}
$$

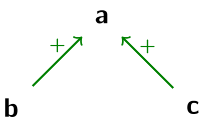
Automata Networks



Asynchronous semantics (one transition at a time):

$$
\begin{array}{lllllll}
 & \langle a_2, b_0, c_0\rangle \longrightarrow \langle a_2, b_0, c_1\rangle \longrightarrow \langle a_2, b_1, c_1\rangle \longrightarrow \langle a_1, b_1, c_1\rangle \\
 \nearrow & \\
\langle a_0, b_0, c_0\rangle & \\
 \searrow & \\
 & \langle a_1, b_0, c_0\rangle \longrightarrow \cdots
\end{array}
$$

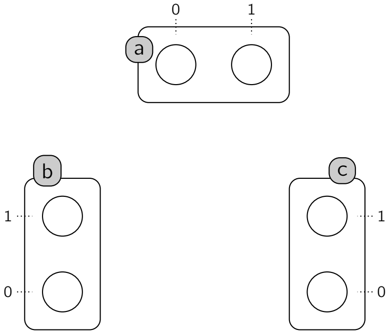
Transition-centered specification



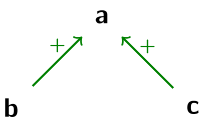
1. $f^a(x) = x[b] \wedge x[c]$
transitions:

$a_0 \rightarrow a_1$: $b_1 \wedge c_1$

$a_1 \rightarrow a_0$: $b_0 \vee c_0$



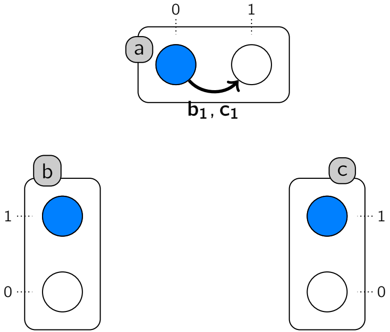
Transition-centered specification



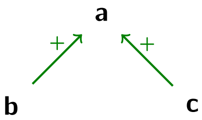
1. $f^a(x) = x[b] \wedge x[c]$
transitions:

$a_0 \rightarrow a_1 \colon\ b_1 \wedge c_1$

$a_1 \rightarrow a_0 \colon\ b_0 \vee c_0$



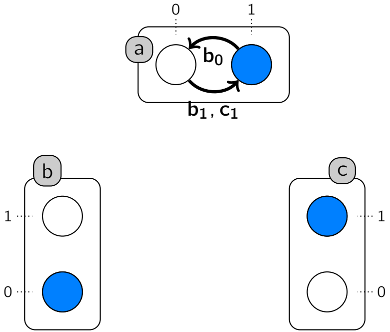
Transition-centered specification



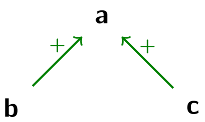
1. $f^a(x) = x[b] \wedge x[c]$
transitions:

$a_0 \rightarrow a_1\colon\ b_1 \wedge c_1$

$a_1 \rightarrow a_0\colon\ b_0 \vee c_0$



Transition-centered specification

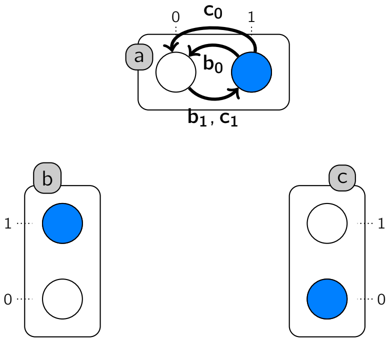


1. $f^a(x) = x[b] \wedge x[c]$

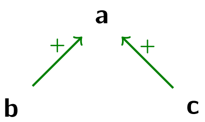
transitions:

$a_0 \rightarrow a_1$: $b_1 \wedge c_1$

$a_1 \rightarrow a_0$: $b_0 \vee c_0$



Transition-centered specification



1. $f^a(x) = x[b] \wedge x[c]$
transitions:

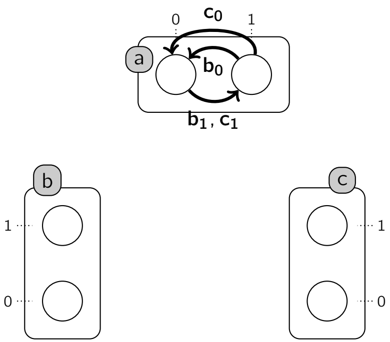
$a_0 \rightarrow a_1$: $b_1 \wedge c_1$

$a_1 \rightarrow a_0$: $b_0 \vee c_0$

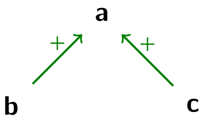
2. Non-deterministic f^a
transitions:

$a_0 \rightarrow a_1$: $b_1 \vee c_1$

$a_1 \rightarrow a_0$: $b_0 \vee c_0$



Transition-centered specification



1. $f^a(x) = x[b] \wedge x[c]$
transitions:

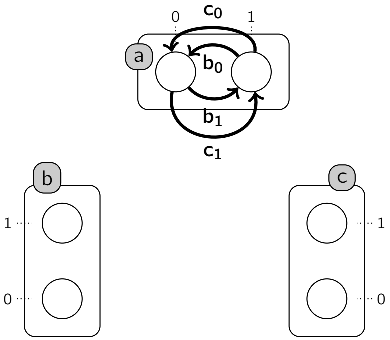
$a_0 \rightarrow a_1$: $b_1 \wedge c_1$

$a_1 \rightarrow a_0$: $b_0 \vee c_0$

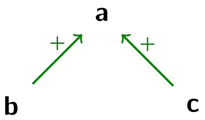
2. Non-deterministic f^a
transitions:

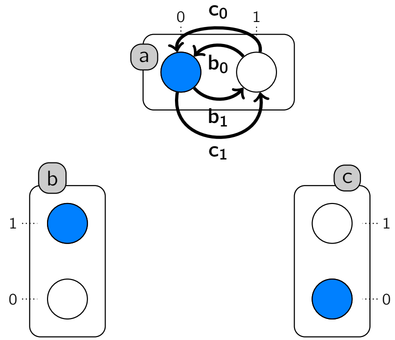
$a_0 \rightarrow a_1$: $b_1 \vee c_1$

$a_1 \rightarrow a_0$: $b_0 \vee c_0$



Transition-centered specification





1. $f^a(x) = x[b] \wedge x[c]$
transitions:

$a_0 \rightarrow a_1\colon\ b_1 \wedge c_1$

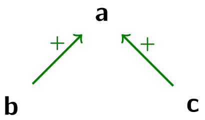
$a_1 \rightarrow a_0\colon\ b_0 \vee c_0$

2. Non-deterministic f^a
transitions:

$a_0 \rightarrow a_1\colon\ b_1 \vee c_1$

$a_1 \rightarrow a_0\colon\ b_0 \vee c_0$

Transition-centered specification



1. $f^a(x) = x[b] \wedge x[c]$
transitions:

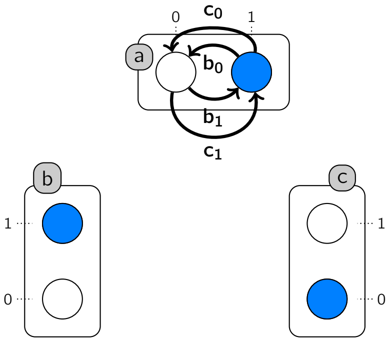
$a_0 \to a_1$: $b_1 \wedge c_1$

$a_1 \to a_0$: $b_0 \vee c_0$

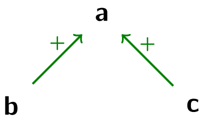
2. Non-deterministic f^a
transitions:

$a_0 \to a_1$: $b_1 \vee c_1$

$a_1 \to a_0$: $b_0 \vee c_0$



Transition-centered specification



1. $f^a(x) = x[b] \wedge x[c]$
transitions:

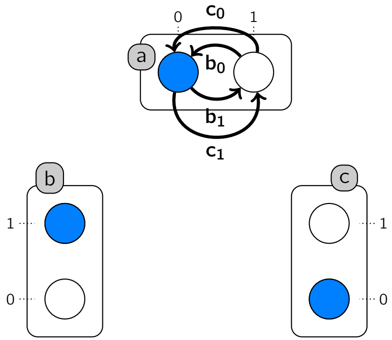
$a_0 \rightarrow a_1$: $b_1 \wedge c_1$

$a_1 \rightarrow a_0$: $b_0 \vee c_0$

2. Non-deterministic f^a
transitions:

$a_0 \rightarrow a_1$: $b_1 \vee c_1$

$a_1 \rightarrow a_0$: $b_0 \vee c_0$



Outline

1. Automata Networks

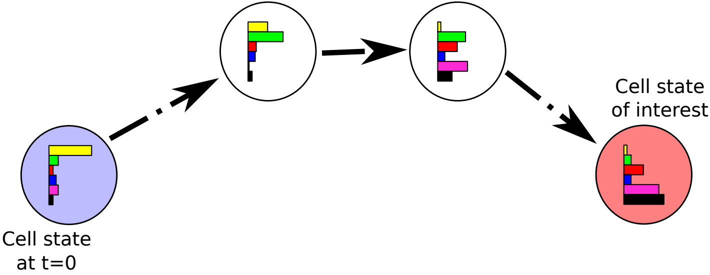
2. Approximations of transient dynamics
   - Abstraction of traces
   - Reachability: cut sets, bifurcations
   - Model reduction preserving transient properties
   - Software Pint

3. Starting project: cell reprogramming

Transient Reachability

Initial state(s)/Goal state(s)

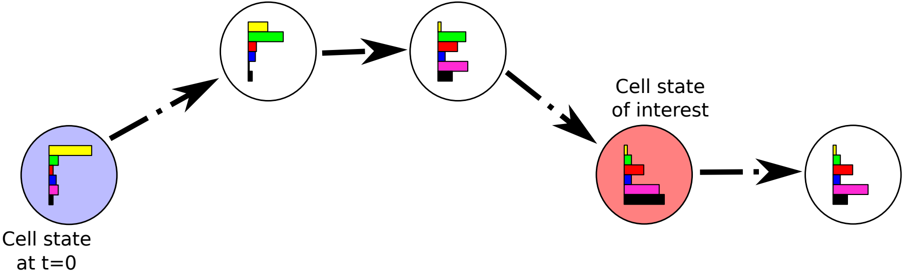
Transient Reachability



Initial state(s)/Goal state(s)

- Trajectory existence

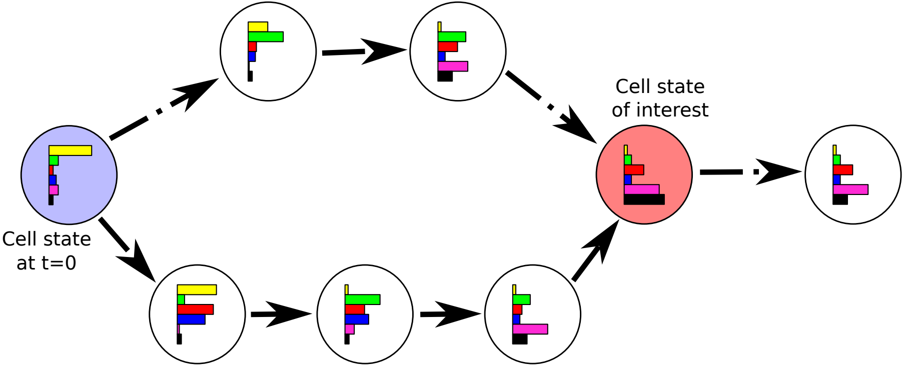
Transient Reachability



Initial state(s)/Goal state(s)

- Trajectory existence

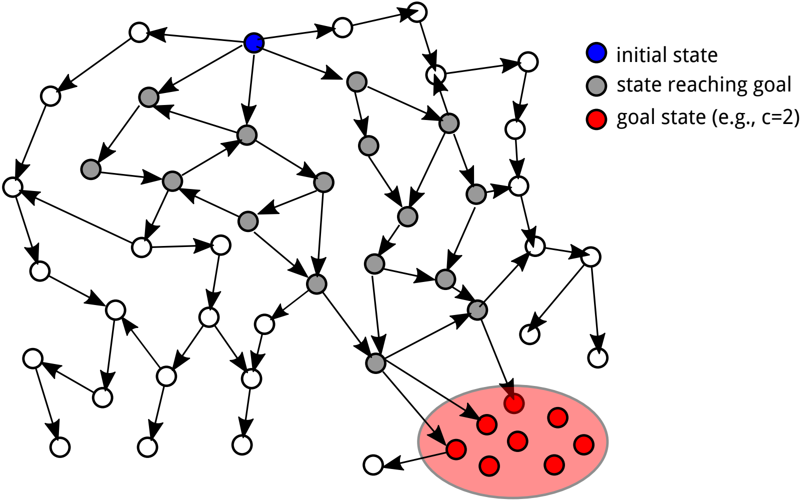
Transient Reachability



Initial state(s)/Goal state(s)

- Trajectory existence
- Reasoning on all trajectories

State Transition Graph



$\Rightarrow$ **avoid building it! (even symbolically): abstractions**

(reachability is PSPACE-complete)

Summary

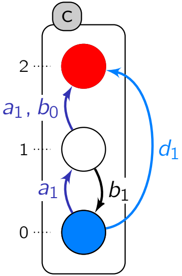
Abstractions for transient dynamics of Automata Networks

Intuition: exploit the low scope of transitions

- Static analysis by abstract interpretation [Cousot and Cousot 77]
- Intermediate representation (Local Causality Graph) to reason on necessary/sufficient conditions for transitions
- Implementation mixes algorithms on graphs and SAT (ASP).

Basically:
Approx. of PSPACE problems with P.$e^{|a|-1}$ or NP.$e^{|a|-1}$ problems
where $|a|$ is the number of local states within a single automaton (typically 2-4)

Local Causality



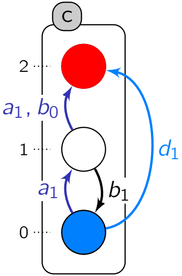
Objective: pair of local states of a same automaton
E.g., $c_0 \rightsquigarrow c_2$, $c_0 \rightsquigarrow c_0$, $d_0 \rightsquigarrow d_1$, ...

Local path: set of acyclic seq of local transitions

$$\text{local-paths}(c_0 \rightsquigarrow c_2) = \{c_0 \xrightarrow{a_1} c_1 \xrightarrow{a_1, b_0} c_2, \\ c_0 \xrightarrow{d_1} c_2\}$$

nb local paths: poly(nb local trs),exp(nb levels)

Local Causality



Objective: pair of local states of a same automaton
E.g., $c_0 \rightsquigarrow c_2$, $c_0 \rightsquigarrow c_0$, $d_0 \rightsquigarrow d_1$, ...

Local path: set of acyclic seq of local transitions

$$\text{local-paths}(c_0 \rightsquigarrow c_2) = \{ c_0 \xrightarrow{a_1} c_1 \xrightarrow{a_1, b_0} c_2, \; c_0 \xrightarrow{d_1} c_2 \}$$

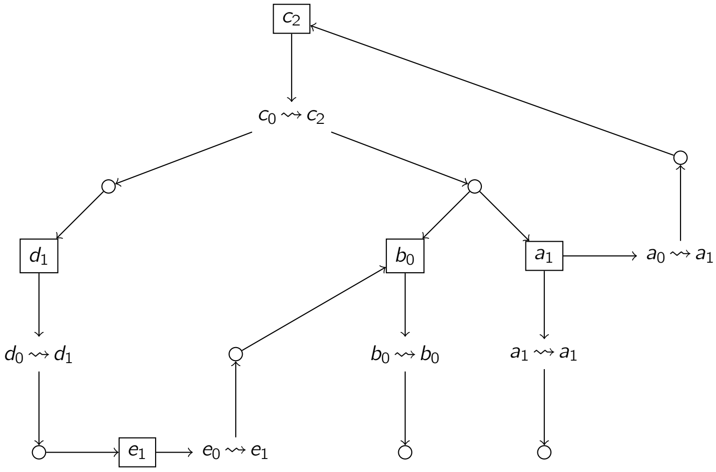
nb local paths: poly(nb local trs),exp(nb levels)

For any trace π starting at some global state s with $c_0 \in s$ and reaching c_2:

- either $c_0 \xrightarrow{a_1} c_1 \xrightarrow{a_1, b_0} c_2$ or $c_0 \xrightarrow{d_1} c_2$ is a sub-trace of π;
- either a_1 and b_0, or d_1 are reached before c_2 in π.

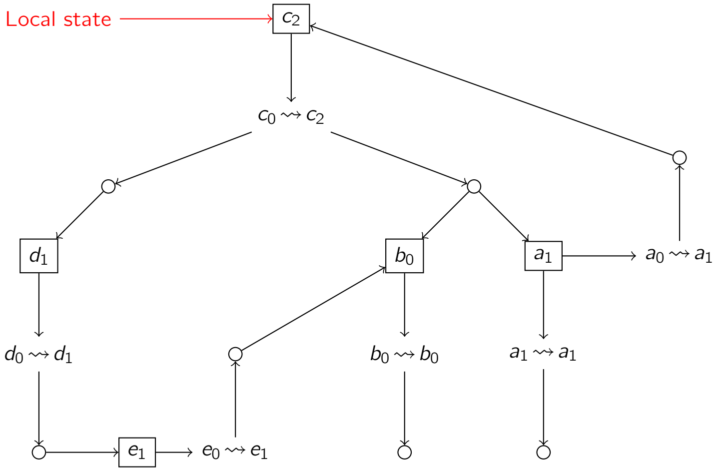
Local Causality Graph

- Initial context $\varsigma = \{a \mapsto \{0,1\}; b \mapsto \{0\}; c \mapsto \{0\}; d \mapsto \{0\}\}$.



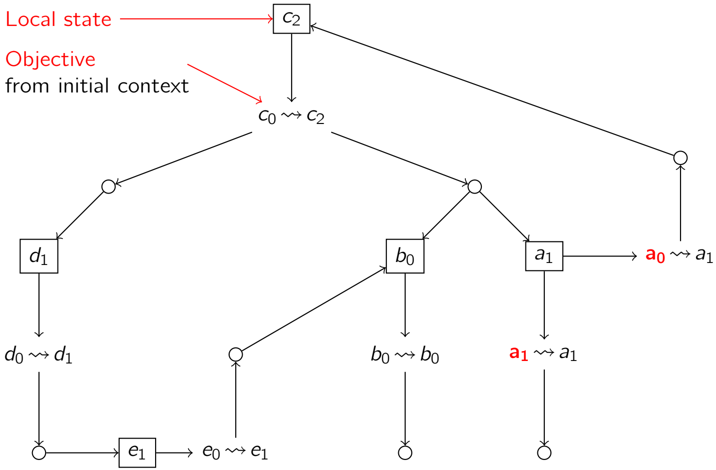
Local Causality Graph

- Initial context $\varsigma = \{a \mapsto \{0,1\}; b \mapsto \{0\}; c \mapsto \{0\}; d \mapsto \{0\}\}$.



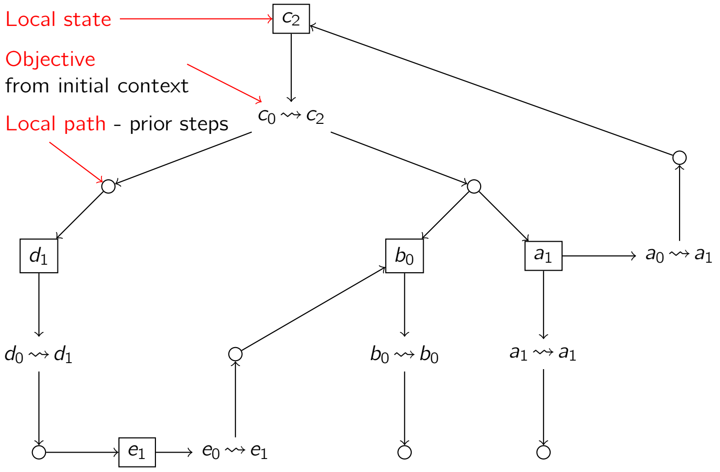
Local Causality Graph

- Initial context $\varsigma = \{a \mapsto \{\mathbf{0}, \mathbf{1}\}; b \mapsto \{0\}; c \mapsto \{0\}; d \mapsto \{0\}\}$.



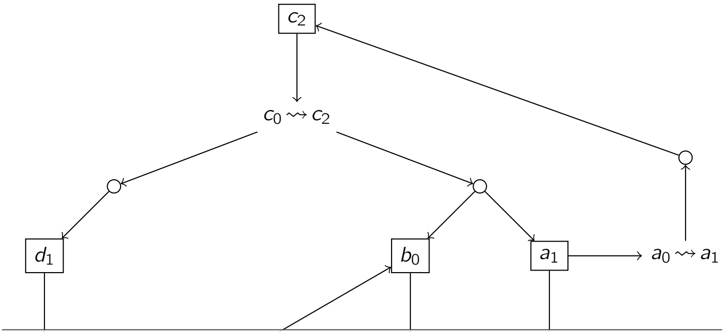
Local Causality Graph

- Initial context $\varsigma = \{a \mapsto \{0,1\}; b \mapsto \{0\}; c \mapsto \{0\}; d \mapsto \{0\}\}$.



Local Causality Graph

- Initial context $\varsigma = \{a \mapsto \{0,1\}; b \mapsto \{0\}; c \mapsto \{0\}; d \mapsto \{0\}\}$.



Size

Nb of objectives: poly(automata size) x nb automata

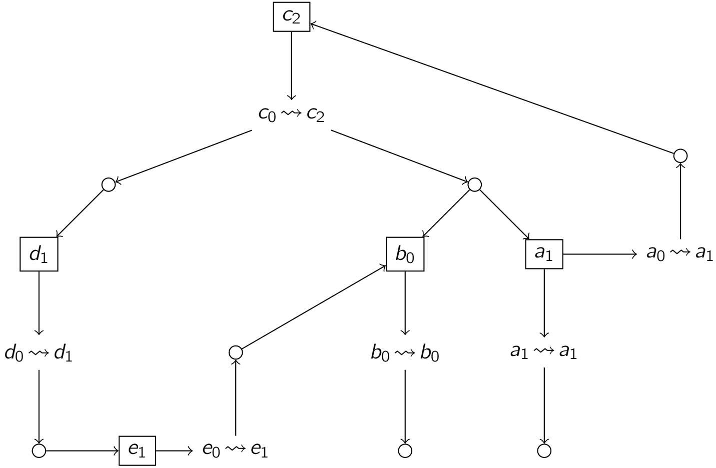
Nb of local paths: exp(automata size), poly(local transitions)

Usually, automata size is very small (2 for Boolean networks)

$\Rightarrow$ highly tractable for large networks of small automata

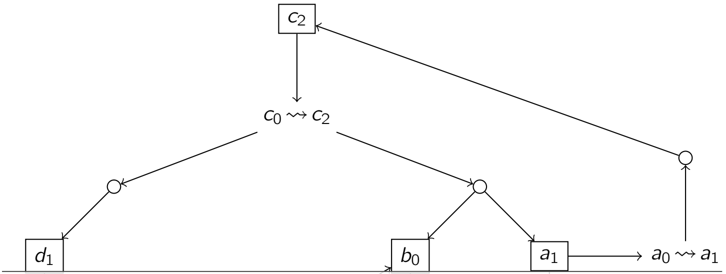
Local Causality Graph

- Initial context $\varsigma = \{a \mapsto \{0,1\}; b \mapsto \{0\}; c \mapsto \{0\}; d \mapsto \{0\}\}$.



Local Causality Graph

- Initial context $\varsigma = \{a \mapsto \{0,1\}; b \mapsto \{0\}; c \mapsto \{0\}; d \mapsto \{0\}\}$.



Necessary condition for reachability

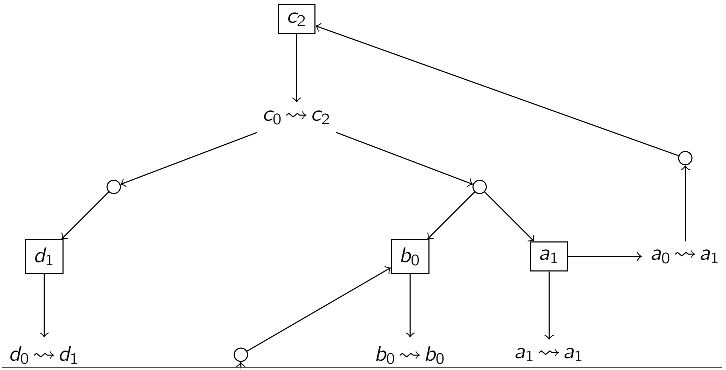
OA($\varsigma \rightarrow^* c_2$) $\equiv$ there is an acyclic traversal from c_2 s.t.

- node objective $\rightarrow$ follow at least one child;
- other nodes $\rightarrow$ follow all children;
- terminates on "local path" leafs.

(can be verified linearly in the size of the LCG).

Local Causality Graph

- Initial context $\varsigma = \{a \mapsto \{0,1\}; b \mapsto \{0\}; c \mapsto \{0\}; d \mapsto \{0\}\}$.



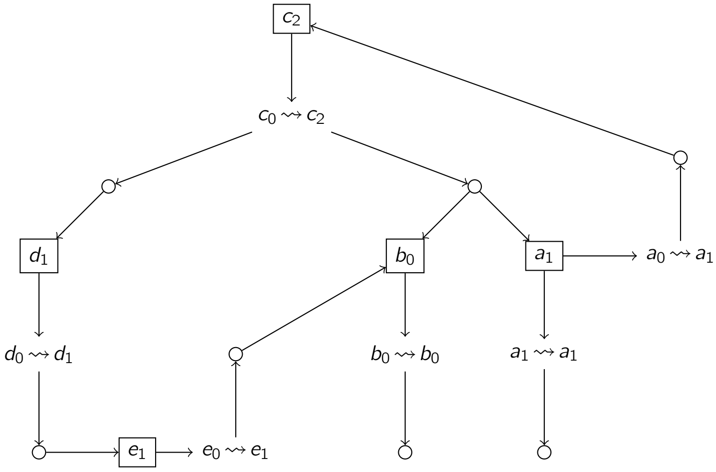
Sufficient condition for reachability

UA($\varsigma \rightarrow^* c_2$) $\equiv \exists$ particular acyclic sub-LCG with *saturated* ς.

NP formulation (find the right combination of local paths).

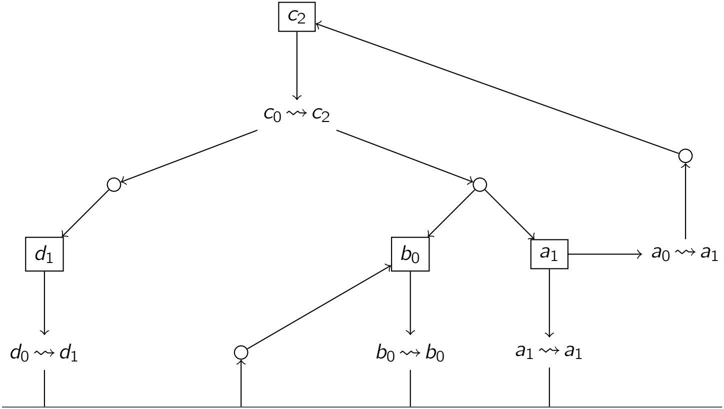
Local Causality Graph

- Initial context $\varsigma = \{a \mapsto \{0,1\}; b \mapsto \{0\}; c \mapsto \{0\}; d \mapsto \{0\}\}$.



Local Causality Graph

- Initial context $\varsigma = \{a \mapsto \{0,1\}; b \mapsto \{0\}; c \mapsto \{0\}; d \mapsto \{0\}\}$.

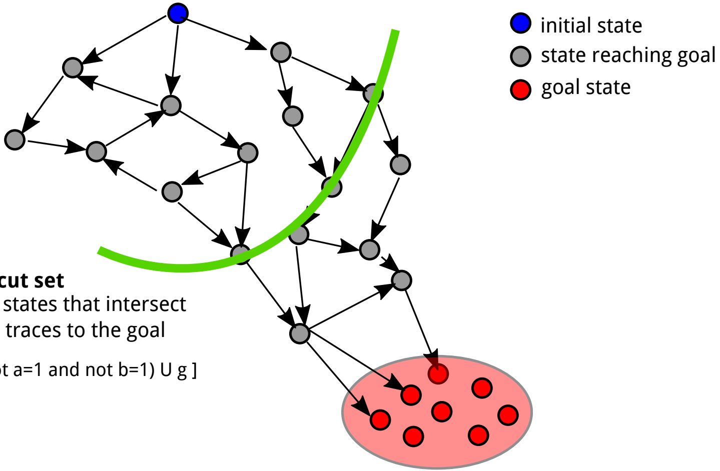


Approximations of reachability

$\mathrm{UA}(s \to^* c_2) \Rightarrow s \to^* c_2 \Rightarrow \mathrm{OA}(s \to^* c_2)$

Cut sets for (transient) reachability

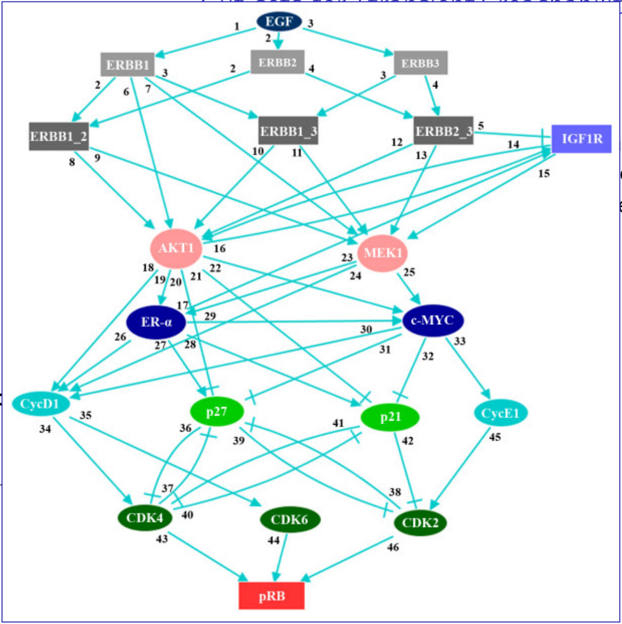
Global state graph



cut set
set of local states that intersect
with all the traces to the goal

not E [(not a=1 and not b=1) U g]

Cut sets for (transient) reachability



te
ching goal
e

c
set of local
with all the

not E [(no

Cut sets for (transient) reachability

Experiments

Under-approximation of N-cut sets (cardinality at most N)
Alternative implementations:

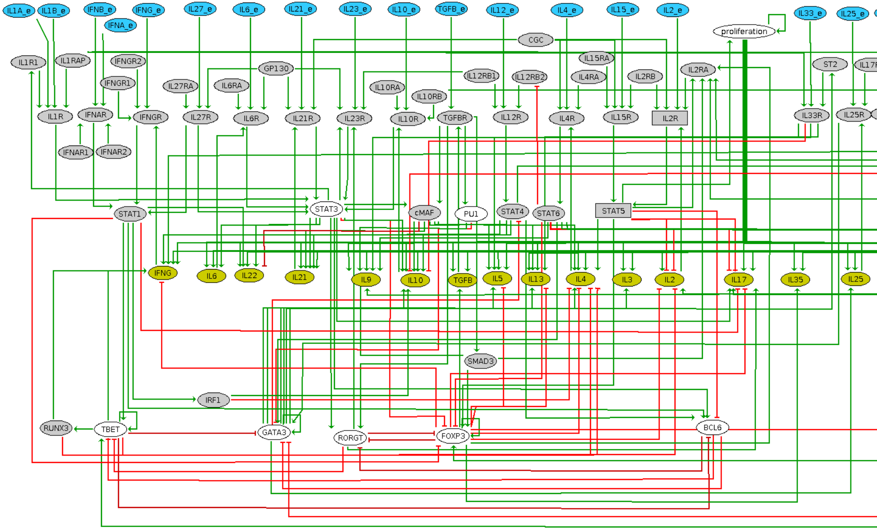
- Computation on Local Causality Graph
- Set of local states *ls* such that OA($s \rightarrow^* g$) is wrong in $\mathcal{A} \setminus ls$ (NP formulation)

```
$ pint-reach --cutsets 4 --no-init-cutsets -i TCell-d.an BCL6=1
"GP130"=1
"STAT3"=1
"CD28"=1,"IL6R"=1
...
"IL6RA"=1,"TCR"=1
```

	TCell-d (101)	RBE2F (370)	MAPK-Schoeberl (309)	PID (21,000)
4-cut sets	0.03s (27)	0.06s (57)	0.1s (34)	39s (37)
6-cut sets	0.03s (27)	0.76s (334)	0.5s (43)	2.6h (1257)

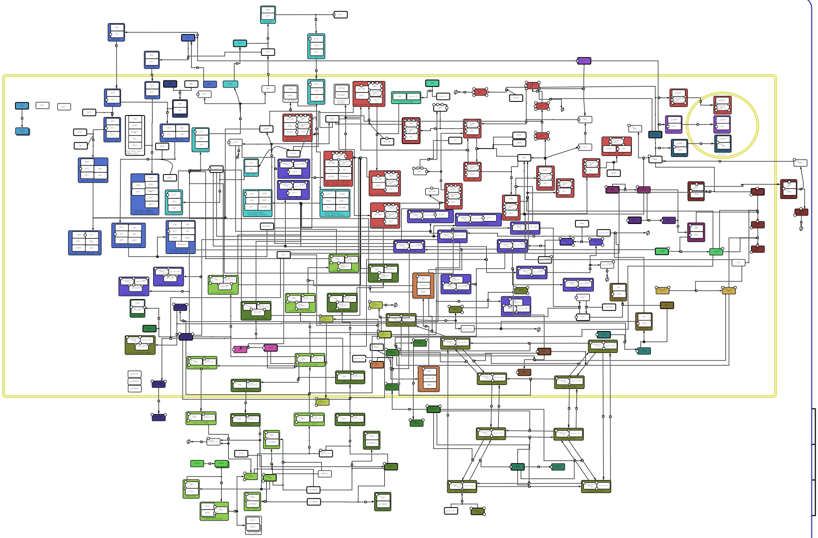
[Paulevé et al at CAV 2013]

Cut sets for (transient) reachability



[Abou-Jaoudé et al, Frontiers in Bioengineering and Biotechnology, 2015]

Cut sets for (transient) reachability



[Calzone et al, Mol Syst Biol, 2008]

Cut sets for (transient) reachability

Experiments

Under-approximation of N-cut sets (cardinality at most N)
Alternative implementations:

- Computation on Local Causality Graph
- Set of local states *ls* such that $\mathsf{OA}(s \to^* g)$ is wrong in $\mathcal{A} \setminus ls$ (NP formulation)

```
$ pint-reach --cutsets 4 --no-init-cutsets -i TCell-d.an BCL6=1
"GP130"=1
"STAT3"=1
"CD28"=1,"IL6R"=1
...
"IL6RA"=1,"TCR"=1
```

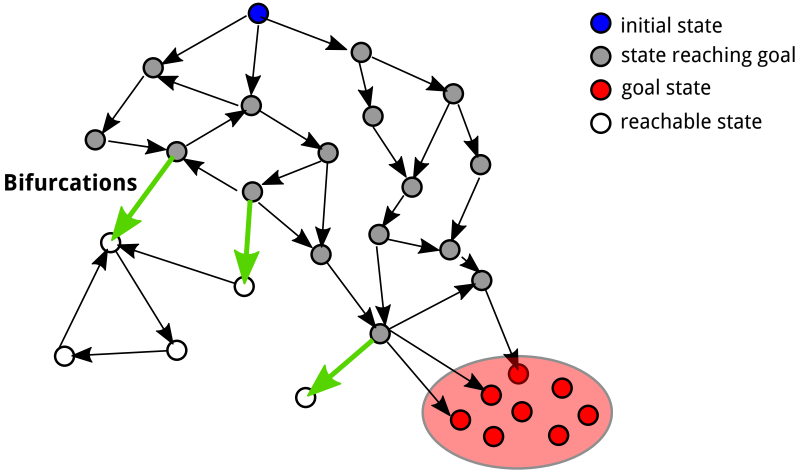
	TCell-d (101)	RBE2F (370)	MAPK-Schoeberl (309)	PID (21,000)
4-cut sets	0.03s (27)	0.06s (57)	0.1s (34)	39s (37)
6-cut sets	0.03s (27)	0.76s (334)	0.5s (43)	2.6h (1257)

[Paulevé et al at CAV 2013]

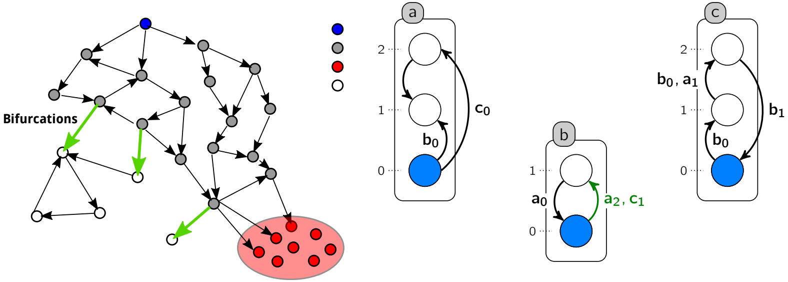
Bifurcation transitions for reachability

Identify when and how a system loses a capability

Global state graph



Bifurcation transitions for reachability



Under-approximation with NP formulation: find transition t, s_b such that

$$\mathrm{UA}(s_0 \to^* s_b) \wedge \mathrm{UA}(s_b \to^* g) \wedge \neg\, \mathrm{OA}(s_b \cdot t \to^* g)$$

ASP (SAT) implementation

Joint work with L. F. Fitime, C. Guziolowski, O. Roux [WCB'16; journal submitted]

Bifurcations for reachability

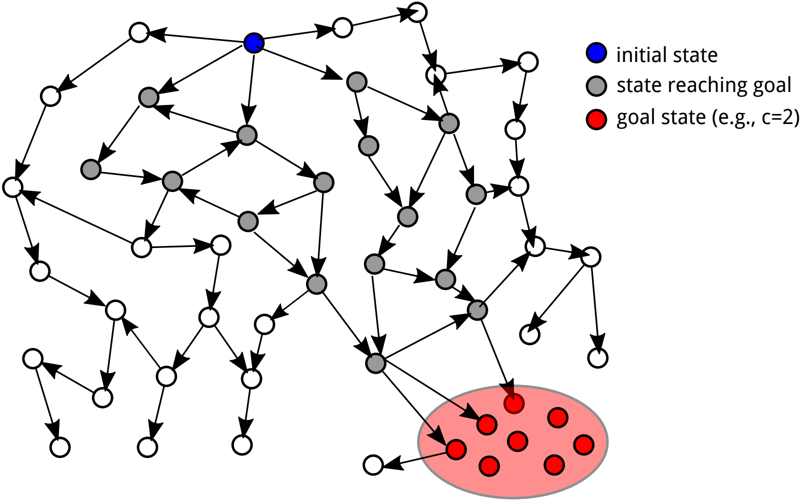
Experiments

```
$ pint-reach --bifurcations -i th_pluri.an FOXP3=1
"STAT6" 0 -> 1 when "IL4R"=1
"RORGT" 0 -> 1 when "BCL6"=0 and "FOXP3"=0 and "STAT3"=1 and "TGFBR"=1
"STAT1" 0 -> 1 when "IL27R"=1
"STAT1" 0 -> 1 when "IFNGR"=1
```

Automata Network	|states|	Goal	MC (NuSMV)		Pint	
			$\|t_b\|$	Time	$\|t_b\|$	Time
Lambda phage $\|\Sigma\| = 4$ $\|T\| = 11$	14	CI_2	10	0.1s	0	0.2s
		Cro_2	3	0.1s	2	0.3s
Th_th1 $\|\Sigma\| = 101$ $\|T\| = 381$	$\approx 3.10^{11}$	$\mathrm{BCL6}_1$	8	13s	5	23s
		TBET_1	11	14s	4	24s
Th_pluri $\|\Sigma\| = 101$ $\|T\| = 381$	$> 5.10^{14}$	$\mathrm{BCL6}_1$	out-of-time		2	32s
		$\mathrm{IL21}_1$			0	26s
		$\mathrm{FOXP3}_1$			4	56s
		TGFB_1			5	96s

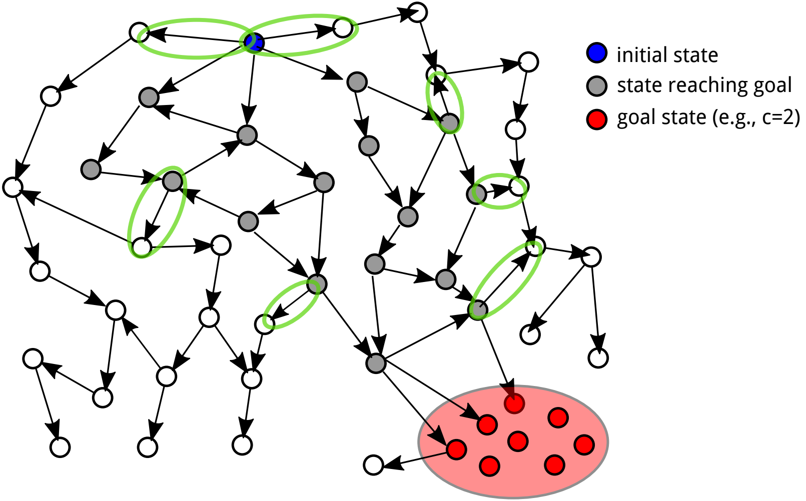
Goal-oriented Reduction

[Paulevé at CMSB'16]



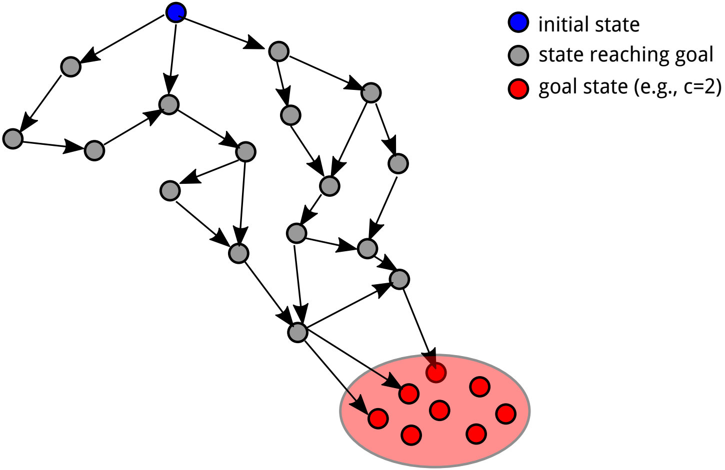
Goal-oriented Reduction

[Paulevé at CMSB'16]



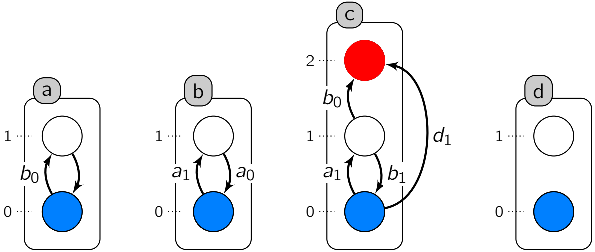
Goal-oriented Reduction

[Paulevé at CMSB'16]

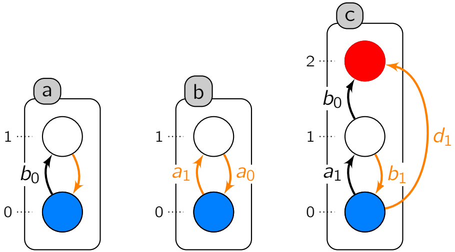


⇒ identify **useless transitions in Automata Network definition**
(no transition graph computation!)

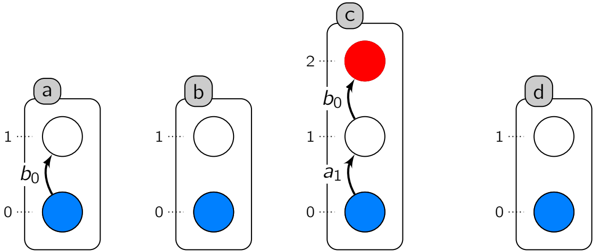
Goal-oriented reduction



Goal-oriented reduction



Goal-oriented reduction



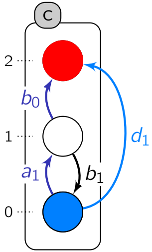
Theorem

Goal-oriented reduction **preserves all simple traces** *from initial state to goal.*

Refining local paths

Given an initial state s, ignore local paths requiring **impossible** objectives:

$$\text{filtered-local-paths}_s(a_i \rightsquigarrow a_j) \stackrel{\Delta}{=} \{\eta \in \text{local-paths}(a_i \rightsquigarrow a_j) \mid \forall n \in \mathbb{I}^\eta, \\ \forall b_k \in \text{enab}(\eta^n), \text{OA}(s \rightarrow^* b_k)\}$$



$$\text{local-paths}(c_0 \rightsquigarrow c_2) = \{c_0 \xrightarrow{a_1} c_1 \xrightarrow{b_0} c_2, c_0 \xrightarrow{d_1} c_2\}$$

If $\neg\,\text{OA}(s \rightarrow^* d_1)$, then

$$\text{filtered-local-paths}_s(c_0 \rightsquigarrow c_2) = \{c_0 \xrightarrow{a_1} c_1 \xrightarrow{b_0} c_2\}$$

Reduction procedure

Smallest set of objectives $\mathcal{B}$ satisfying:

1. $g_0 \rightsquigarrow g_\top \in \mathcal{B}$ (main objective)
2. $b_j \xrightarrow{\ell} b_k \in \mathbf{tr}(\mathcal{B}) \Rightarrow \forall a_i \in \ell, a_0 \rightsquigarrow a_i \in \mathcal{B}$
3. $b_j \xrightarrow{\ell} b_k \in \mathbf{tr}(\mathcal{B}) \wedge b_\star \rightsquigarrow b_i \in \mathcal{B} \Rightarrow b_k \rightsquigarrow b_i \in \mathcal{B}$

with $\mathbf{tr}(\mathcal{B}) \stackrel{\Delta}{=} \bigcup_{P \in \mathcal{B}} \mathbf{tr}(\textbf{filtered-local-paths}_s(P))$

Transitions not in $\mathrm{tr}(\mathcal{B})$ **can be removed.**

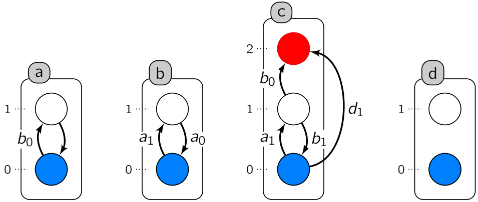
Reduction procedure

Smallest set of objectives $\mathcal{B}$ satisfying:

1. $g_0 \rightsquigarrow g_\top \in \mathcal{B}$ (main objective)
2. $b_j \xrightarrow{\ell} b_k \in \mathbf{tr}(\mathcal{B}) \Rightarrow \forall a_i \in \ell, a_0 \rightsquigarrow a_i \in \mathcal{B}$
3. $b_j \xrightarrow{\ell} b_k \in \mathbf{tr}(\mathcal{B}) \wedge b_\star \rightsquigarrow b_i \in \mathcal{B} \Rightarrow b_k \rightsquigarrow b_i \in \mathcal{B}$

with $\mathbf{tr}(\mathcal{B}) \triangleq \bigcup_{P \in \mathcal{B}} \mathbf{tr}(\text{filtered-local-paths}_s(P))$

Transitions not in $\mathrm{tr}(\mathcal{B})$ can be removed.



$\mathcal{B}$	$\mathrm{tr}(\mathcal{B})$
$c_0 \rightsquigarrow c_2$	

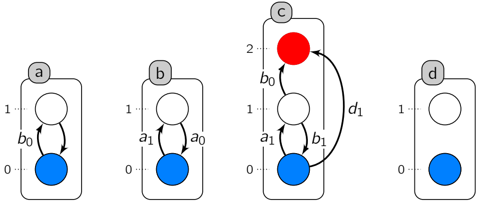
Reduction procedure

Smallest set of objectives $\mathcal{B}$ satisfying:

1. $g_0 \rightsquigarrow g_\top \in \mathcal{B}$ (main objective)
2. $b_j \xrightarrow{\ell} b_k \in \mathrm{tr}(\mathcal{B}) \Rightarrow \forall a_i \in \ell, a_0 \rightsquigarrow a_i \in \mathcal{B}$
3. $b_j \xrightarrow{\ell} b_k \in \mathrm{tr}(\mathcal{B}) \wedge b_\star \rightsquigarrow b_i \in \mathcal{B} \Rightarrow b_k \rightsquigarrow b_i \in \mathcal{B}$

with $\mathrm{tr}(\mathcal{B}) \stackrel{\Delta}{=} \bigcup_{P \in \mathcal{B}} \mathrm{tr}(\text{filtered-local-paths}_s(P))$

Transitions not in $\mathrm{tr}(\mathcal{B})$ can be removed.



$\mathcal{B}$	$\mathrm{tr}(\mathcal{B})$
$c_0 \rightsquigarrow c_2$	$c_0 \xrightarrow{a_1} c_1,\ c_1 \xrightarrow{b_0} c_2$

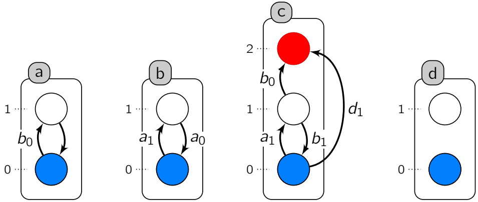
Reduction procedure

Smallest set of objectives $\mathcal{B}$ satisfying:

1. $g_0 \rightsquigarrow g_\top \in \mathcal{B}$ (main objective)
2. $b_j \xrightarrow{\ell} b_k \in \mathbf{tr}(\mathcal{B}) \Rightarrow \forall a_i \in \ell, a_0 \rightsquigarrow a_i \in \mathcal{B}$
3. $b_j \xrightarrow{\ell} b_k \in \mathbf{tr}(\mathcal{B}) \wedge b_\star \rightsquigarrow b_i \in \mathcal{B} \Rightarrow b_k \rightsquigarrow b_i \in \mathcal{B}$

with $\mathbf{tr}(\mathcal{B}) \triangleq \bigcup_{P \in \mathcal{B}} \mathbf{tr}(\text{filtered-local-paths}_s(P))$

Transitions not in $\mathrm{tr}(\mathcal{B})$ can be removed.



$\mathcal{B}$	$\mathrm{tr}(\mathcal{B})$
$c_0 \rightsquigarrow c_2$ $a_0 \rightsquigarrow a_1$, $b_0 \rightsquigarrow b_0$	$c_0 \xrightarrow{a_1} c_1$, $c_1 \xrightarrow{b_0} c_2$

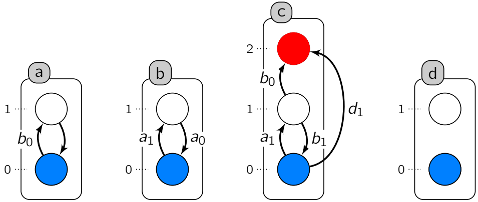
Reduction procedure

Smallest set of objectives $\mathcal{B}$ satisfying:

1. $g_0 \rightsquigarrow g_\top \in \mathcal{B}$ (main objective)
2. $b_j \xrightarrow{\ell} b_k \in \mathrm{tr}(\mathcal{B}) \Rightarrow \forall a_i \in \ell, a_0 \rightsquigarrow a_i \in \mathcal{B}$
3. $b_j \xrightarrow{\ell} b_k \in \mathrm{tr}(\mathcal{B}) \wedge b_\star \rightsquigarrow b_i \in \mathcal{B} \Rightarrow b_k \rightsquigarrow b_i \in \mathcal{B}$

with $\mathrm{tr}(\mathcal{B}) \triangleq \bigcup_{P \in \mathcal{B}} \mathrm{tr}(\text{filtered-local-paths}_s(P))$

Transitions not in $\mathrm{tr}(\mathcal{B})$ can be removed.



$\mathcal{B}$	$\mathrm{tr}(\mathcal{B})$
$c_0 \rightsquigarrow c_2$	$c_0 \xrightarrow{a_1} c_1,\ c_1 \xrightarrow{b_0} c_2$
$a_0 \rightsquigarrow a_1,\ b_0 \rightsquigarrow b_0$	$a_0 \xrightarrow{b_0} a_1$

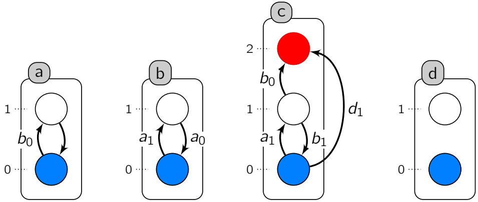
Reduction procedure

Smallest set of objectives $\mathcal{B}$ satisfying:

1. $g_0 \rightsquigarrow g_\top \in \mathcal{B}$ (main objective)
2. $b_j \xrightarrow{\ell} b_k \in \mathbf{tr}(\mathcal{B}) \Rightarrow \forall a_i \in \ell, a_0 \rightsquigarrow a_i \in \mathcal{B}$
3. $b_j \xrightarrow{\ell} b_k \in \mathbf{tr}(\mathcal{B}) \wedge b_\star \rightsquigarrow b_i \in \mathcal{B} \Rightarrow b_k \rightsquigarrow b_i \in \mathcal{B}$

with $\mathbf{tr}(\mathcal{B}) \triangleq \bigcup_{P \in \mathcal{B}} \mathbf{tr}(\text{filtered-local-paths}_s(P))$

Transitions not in $\mathrm{tr}(\mathcal{B})$ can be removed.



$\mathcal{B}$	$\mathrm{tr}(\mathcal{B})$
$c_0 \rightsquigarrow c_2,\ c_1 \rightsquigarrow c_2$ $a_0 \rightsquigarrow a_1,\ b_0 \rightsquigarrow b_0$	$c_0 \xrightarrow{a_1} c_1,\ c_1 \xrightarrow{b_0} c_2$ $a_0 \xrightarrow{b_0} a_1$

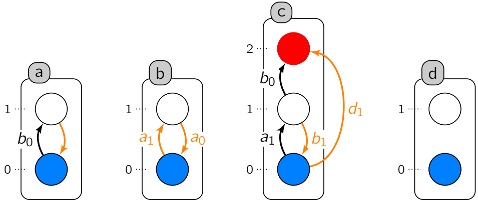
Reduction procedure

Smallest set of objectives $\mathcal{B}$ satisfying:

1. $g_0 \rightsquigarrow g_\top \in \mathcal{B}$ (main objective)
2. $b_j \xrightarrow{\ell} b_k \in \mathbf{tr}(\mathcal{B}) \Rightarrow \forall a_i \in \ell, a_0 \rightsquigarrow a_i \in \mathcal{B}$
3. $b_j \xrightarrow{\ell} b_k \in \mathbf{tr}(\mathcal{B}) \wedge b_\star \rightsquigarrow b_i \in \mathcal{B} \Rightarrow b_k \rightsquigarrow b_i \in \mathcal{B}$

with $\mathbf{tr}(\mathcal{B}) \stackrel{\Delta}{=} \bigcup_{P \in \mathcal{B}} \mathbf{tr}(\text{filtered-local-paths}_s(P))$

Transitions not in $\mathrm{tr}(\mathcal{B})$ can be removed.



$\mathcal{B}$	$\mathrm{tr}(\mathcal{B})$
$c_0 \rightsquigarrow c_2,\ c_1 \rightsquigarrow c_2$ $a_0 \rightsquigarrow a_1,\ b_0 \rightsquigarrow b_0$	$c_0 \xrightarrow{a_1} c_1,\ c_1 \xrightarrow{b_0} c_2$ $a_0 \xrightarrow{b_0} a_1$

Experiments

For each model: select an initial state; select a goal (activation of a node).

Goal reachability verification - equivalent in reduced model

```
$ pint-export -i model.an --reduce-for-goal g=1 -o reduced.an
$ pint-nusmv -i reduced.an g=1
```

			Verification of goal reachability	
Model	# local trs	# states	`NuSMV` (EF g)	`its-reach`
VPC (88)	332	KO	KO	1s 50Mb
	219	$\mathbf{1.8 \cdot 10^9}$	**236s 156Mb**	**0.8s 21Mb**
TCell-d (101)	384	$\approx 2.7 \cdot 10^8$	3s 40Mb	0.5s 24Mb
profile 1	**0**	**1**		
TCell-d (101)	384	KO	KO	0.5s 23Mb
profile 2	**161**	**75,947,684**	**474s 260Mb**	**0.3s 19Mb**
EGF-r (104)	378	$\approx 2.7 \cdot 10^{16}$	KO	1.36s 60Mb
	69	**62,914,560**	**11s 33Mb**	**0.3s 17Mb**
RBE2F (370)	742	KO	KO	KO
	56	**2,350,494**	**5s 377Mb**	**5s 170Mb**

In all cases, reduction step took less than 0.1s

Experiments

Verification of cut sets (checkpoints)

- requires all the simple traces
- $\{a_1, b_1\}$ is a cut set for g_1 iff `not E [ (`$a \neq 1 \wedge b \neq 1$`) U` $g = 1$ `]`
- equivalent in the reduced model

```
$ pint-export -i model.an --reduce-for-goal g=1 -o reduced.an
$ pint-nusmv -i reduced.an --is-cutset a=1,b=1 g=1
```

	Wnt (32)	TCell-r (40)	EGF-r (104)	TCell-d (101)	RBE2F (370)
NuSMV	44s 55Mb **9.1s 27Mb**	KO **2.4s 34Mb**	KO **13s 33Mb**	KO **600s 360Mb**	KO **6s 29Mb**
its-ctl	105s 2.1Gb **16s 720Mb**	492s 10Gb **11s 319Mb**	KO **21s 875Mb**	KO **KO**	KO **179s 1.8Gb**

In all cases, reduction step took less than 0.1s

Goal-oriented reduction

- Automata networks with asynchronous or general step semantics
- Goal: sub-state reachability; sequences of sub-state reachability
- Removes local transitions identified as useless for the goal
- Low complexity: poly(automata, local trs); exp(nb levels)

Properties of the reduced model

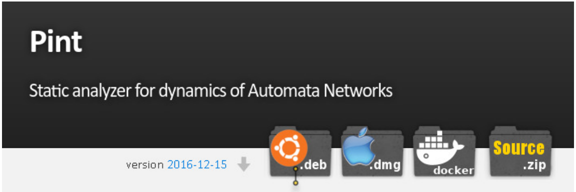
- Preserves all simple traces for goal reachability from initial state
  $\Rightarrow$ existence of a trace to the goal is preserved
  $\Rightarrow$ properties shared by all the traces to the goal are preserved
- Experiments show drastic improvement for model-checking of biological nets

On-going work

- Embed in Petri net unfolding; model identification
- Fast updating after one transition

Software: Pint

http://loicpauleve.name/pint



- Input: automata networks
  - convert SBML-qual/GINsim with `LogicalModels`
  - scripts for CellNetAnalyser, Biocham, etc.
- Command line tools:
  - Static analysis for reachability, cut sets, fixed points
  - Model reduction w.r.t. reachability property
  - Inference of Interaction graph/Thomas parameters
  - Interface with model-checkers (NuSMV, ITS, mole).
- OCaml library (possible C/C++ bindings)

```
model.an:
a [0, 1]
b [0, 1, 2]
c [0, 1]

a 0 -> 1 when b=0 and c=1
a 1 -> 0 when b=1
a 1 -> 0 when b=2
a 1 -> 0 when c=0

b 0 -> 1 when a=1
b 1 -> 2 when a=1
```

Coming soon: Pint notebook

jupyter demo_ErbB2 Last Checkpoint: 30 minutes ago (autosaved)

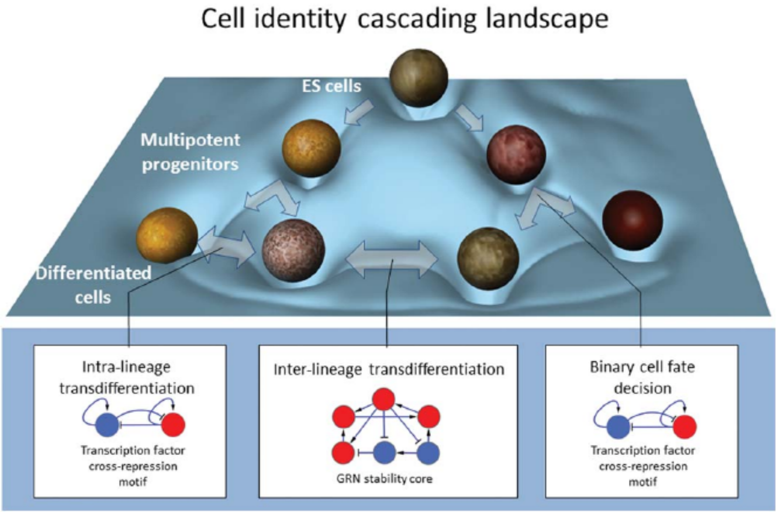
File Edit View Insert Cell Kernel Help | Python 3

Code CellToolbar

```
In [1]: import pint
You are using Pint version 2016-09-16

In [2]: erbb = pint.load("http://ginsim.org/sites/default/files/ErbB2_model.zginml")
Downloading 'http://ginsim.org/sites/default/files/ErbB2_model.zginml' to 'gen/pintodunp3mvErbB2_model.zginml'
Source file is in zginml format, importing with logicalmodel
Invoking GINsim...
Simplifying model...

gen/pinteeuqb4mzErbB2_model.an

1 state(s) have been registered: Init_WT

In [3]: erbb.having(EGF=1).cutsets("pRB1=1")
# Running command pint-reach --json-stdout --cutsets 5 pRB1=1 --no-init-cutsets -i gen/pinteeuqb4mzErbB2_model.an --
initial-context "EGF"=1

Out[3]: [{'CDK4': 1},
 {'CDK6': 1},
 {'CyclinD1': 1},
 {'ERalpha': 1},
 {'MYC': 1},
 {'pRB1': 1},
 {'AKT1': 1, 'MEK1': 1},
 {'ERBB1': 1, 'ERBB2': 1, 'IGF1R': 1},
 {'ERBB1': 1, 'ERBB2_3': 1, 'IGF1R': 1},
 {'ERBB1': 1, 'ERBB3': 1, 'IGF1R': 1}]

In [ ]:
```

Outline

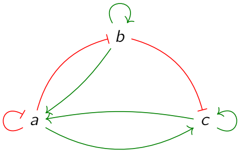
Cellular Reprogramming



(source: Crespo et al. Stem cells 2013; 31:2127-2135)

Reprogramming Determinants Prediction

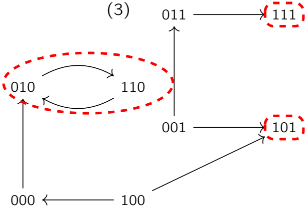
(1)



(2)

$f_a(x) = x_c$ or (not x_a and not x_b)
$f_b(x) =$ not x_a or x_b
$f_c(x) = x_c$ or (x_a and not x_b)

(3)



Reprogramming Determinants (RDs): set of nodes and perturbations

2 settings

- **Permanent perturbations** (mutations): function is changed to constant
- **Temporary perturbations**: enforced transitions

2 problems

- **Existential reprogramming**
  after perturbation the target attractor is reachable.
- **Inevitable reprogramming**
  after perturbation the target attractor is the only reachable attractor

Reprogramming Determinants Prediction

Preliminary results

Relationship between the Reprogramming Determinants of Boolean Networks and their Interaction Graph
Hugues Mandon, Stefan Haar, Loïc Paulevé at HSB 2016.

For permanent perturbations:

- existential reprog: RDs are all in (particular) SCCs of the IG;
- inevitable reprog: RDs can be outside the cycles;
- in all cases, reachability checking is key.

Algorithms for RDs characterization combines Interaction Graph analysis and model-checking.

ANR-FNR AlgoReCell 2017-2019

Computational Models and Algorithms for the Prediction of Cell Reprogrammig Determinants with High Efficiency and High Fidelity

AlgoReCell Objectives

- Design a **generic computational framework** for predicting perturbations leading to a cellular de-differentiation or trans-differentiation.
- The **predictions** will consist of combinations of targets (notably genes), referred to **Reprogramming Determinants (RDs)**.
- The predictions will be **based on a computational dynamical model** of the cell regulation network, and on the initial and targeted cell type.
- The resulting framework will be **evaluated experimentally** for the reprogramming of adipocyte and osteoblast cells.

ANR-FNR AlgoReCell 2017-2019

France

LRI

- Loïc Paulevé (leader)

LSV

- Stefan Haar
- Thomas Chatain
- Stefan Schwoon
- Hugues Mandon (PhD student LSV-LRI)
- Juraj Kolcak (future PhD student LSV-LRI)

Institut Curie

- Andrei Zinovyev
- Laurence Calzone
- + postdoc

Luxembourg

FSTC Life

- Thomas Sauter (leader)
- Lasse Sinkkonen
- + PhD student

FSTC Computer Science

- Jun Pang
- Andrzej Mizera
- + postdoc

LCSB Centre for Systems Biomedicine

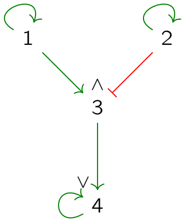
- Antonio del Sol

Perspectives

Next direction: **Temporal Reprogramming**

> One-shot reprogramming can require more perturbations than temporal reprogramming

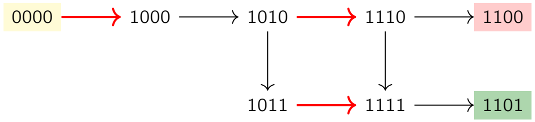
Example:



Reprogramming from 0000 to 1101 (fixpoints)

One-shot reprogramming requires 3 mutations.

Temporal reprogramming requires 2 mutations:



PhD thesis work of Hugues Mandon (co-supervised with Stefan Haar).

Preliminary results in writting. . .