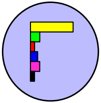
Goal-Oriented Reduction of Automata Networks

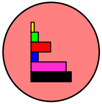
Loïc Paulevé

CNRS/LRI, Univ. Paris-Sud, Paris-Saclay, France
loic.pauleve@lri.fr
http://loicpauleve.name

CMSB 2016 - Cambridge, UK

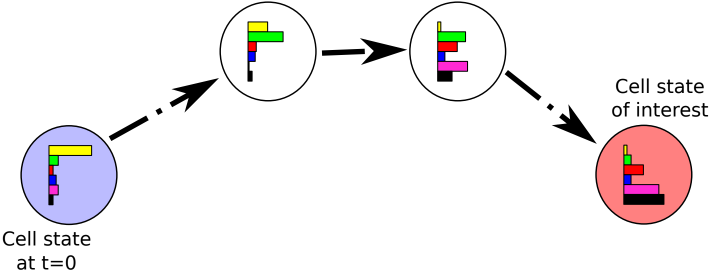
Transient Reachability





Initial state(s)/Goal state(s)

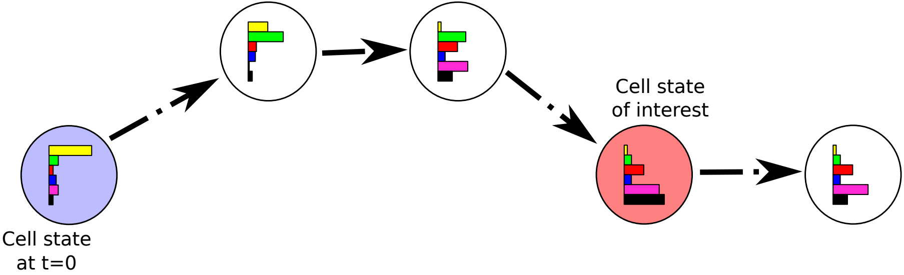
Transient Reachability



Initial state(s)/Goal state(s)

- Trajectory existence

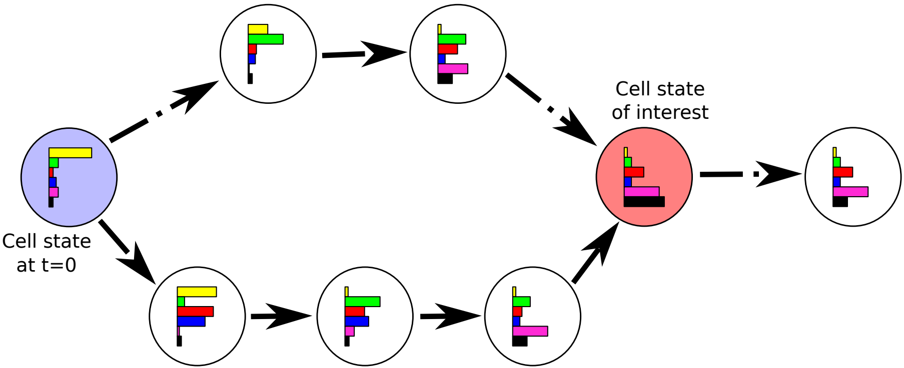
Transient Reachability



Initial state(s)/Goal state(s)

- Trajectory existence

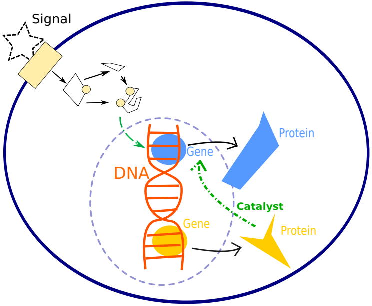
Transient Reachability



Initial state(s)/Goal state(s)

- Trajectory existence
- Reasoning on all trajectories

Reachability in models of biological networks



Validation

- Ability to reproduce time-series data

Prediction

- Cell response w.r.t. signal+environment
- Long-term behaviours (differentiation)

Control

- Mutations/Perturbations for modifying cell behaviour, Trans/De-differentiation

Reachability in logical networks

Logical models of biological networks

- Boolean networks
- Multi-valued/Thomas networks
- Automata networks

Pros

- Few parameters: applicable for large-scale networks
- Finite state space; small compared to population models
- Coarse-grained but exhaustive view of dynamics

Reachability in logical networks

Logical models of biological networks

- Boolean networks
- Multi-valued/Thomas networks
- Automata networks

Pros

- Few parameters: applicable for large-scale networks
- Finite state space; small compared to population models
- Coarse-grained but exhaustive view of dynamics

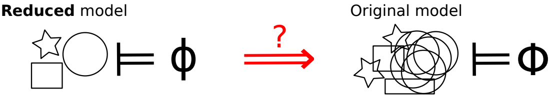
Bad news: verifying reachability is PSPACE-complete

Model reduction

Aim compute a new model, hopefully more tractable

- remove dimensions (variables)
- remove transitions (restrict trajectories)

Challenge: which properties are preserved in the reduced model?

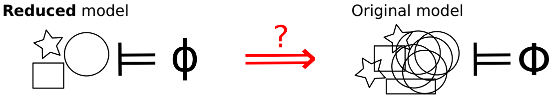


Model reduction

Aim compute a new model, hopefully more tractable

- remove dimensions (variables)
- remove transitions (restrict trajectories)

Challenge: which properties are preserved in the reduced model?



Excerpt of **state of the art** for logical networks:

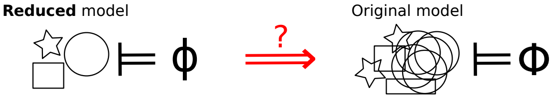
- Reduction of logical regulatory graphs [Naldi et al at CMSB'09] (dimension reduction)
  $\Rightarrow$ breaks reachability properties

Model reduction

Aim compute a new model, hopefully more tractable

- remove dimensions (variables)
- remove transitions (restrict trajectories)

Challenge: which properties are preserved in the reduced model?



Excerpt of **state of the art** for logical networks:

- Reduction of logical regulatory graphs [Naldi et al at CMSB'09]
  (dimension reduction)
  $\Rightarrow$ breaks reachability properties
- Cone of influence reduction [Biere et al at CAV'99];
  Relevant subnet computation [Talcott and Dill in TCSB 2006]
  (remove variables/transitions having no impact on a given property)
  $\Rightarrow$ preserve LTL properties

Contribution

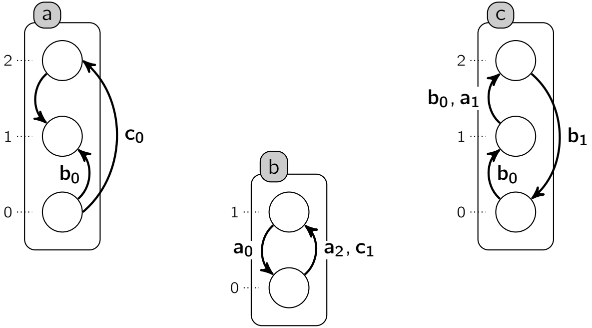
Goal-oriented reduction

- Goal: state of component (e.g., $c = 2$); sub-state ($a = 1, c = 2$); + sequence
- Preserves all minimal trajectories to the goal from a given initial state
  minimality: no sub-sequence of transitions
  (no loop, no non-contributing transitions).
- Low complexity: poly(automata, local transitions), exp(levels)

Automata networks

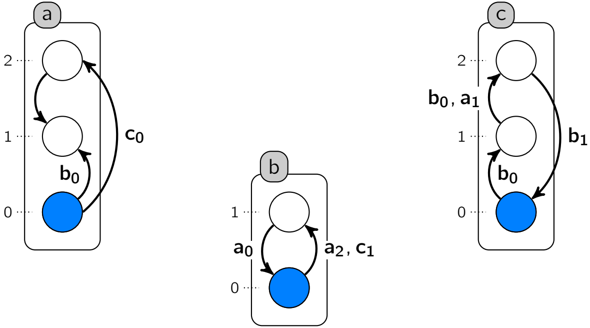
- Transition-centered specification (à la Petri net);
  (in opposition to function-centered Boolean/Thomas networks [Talk of Fages of yesterday])
- any Boolean/Thomas networks can be encoded;
- encoding of SBGN Process Description models [Rougny et al. BMC Systems Biology 2016] (includes reaction networks, e.g., Biocham models).

Automata Networks



Asynchronous semantics (one transition at a time):

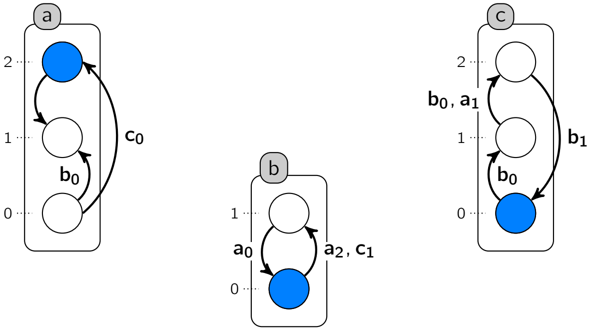
Automata Networks



Asynchronous semantics (one transition at a time):

$\langle a_0, b_0, c_0 \rangle$

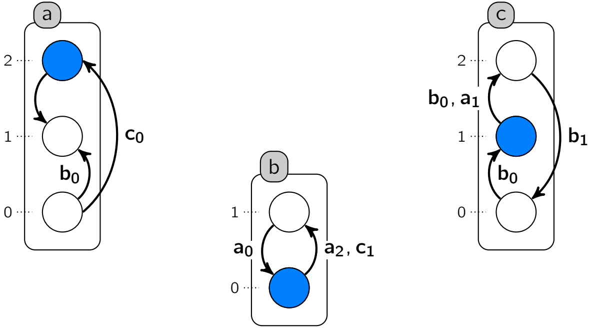
Automata Networks



Asynchronous semantics (one transition at a time):

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

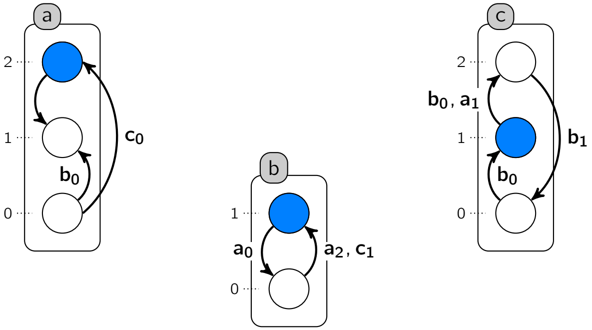
Automata Networks



Asynchronous semantics (one transition at a time):

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

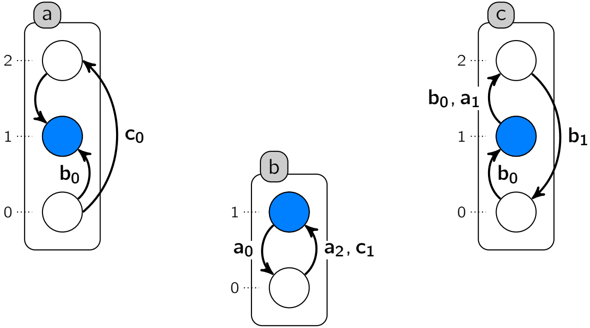
Automata Networks



Asynchronous semantics (one transition at a time):

$$\begin{array}{l} \qquad \langle a_2, b_0, c_0\rangle \longrightarrow \langle a_2, b_0, c_1\rangle \longrightarrow \langle a_2, b_1, c_1\rangle \\ \qquad \nearrow \\ \langle a_0, b_0, c_0\rangle \\ \qquad \searrow \\ \qquad \langle a_0, b_0, c_1\rangle \end{array}$$

Automata Networks



Asynchronous semantics (one transition at a time):

$$\langle a_2, b_0, c_0\rangle \longrightarrow \langle a_2, b_0, c_1\rangle \longrightarrow \langle a_2, b_1, c_1\rangle \longrightarrow \langle a_1, b_1, c_1\rangle$$

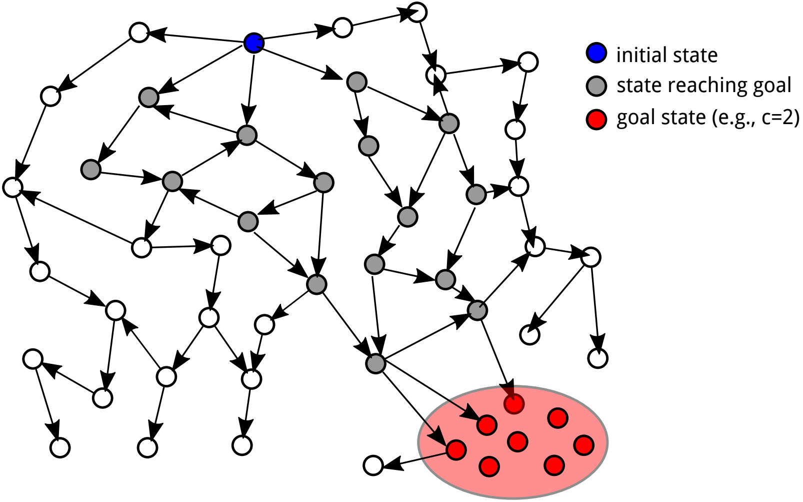
$$\nearrow$$

$$\langle a_0, b_0, c_0\rangle$$

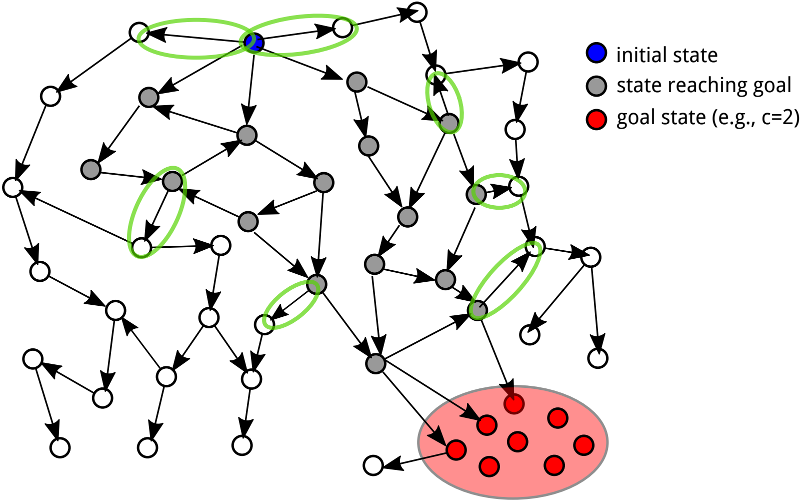
$$\searrow$$

$$\langle a_0, b_0, c_1\rangle \longrightarrow \cdots$$

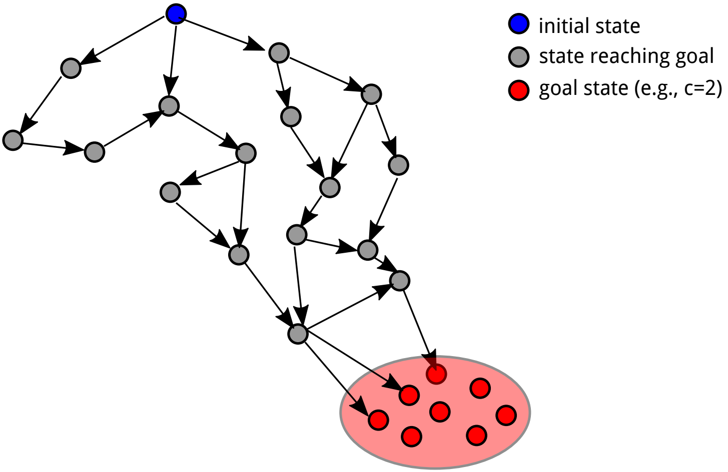
State transition graph



State transition graph



State transition graph



Minimal traces in Automata Networks

Asynchronous semantics

Trace: sequence of local transitions
A trace $\pi \vDash P$ is minimal w.r.t. P iff there is no sub-sequence $\pi' \subsetneq \pi$ s.t. $\pi' \vDash P$.

Examples with $P =$ reach a_1:

$$b_0 \xrightarrow{c_0} b_1\,,\, c_0 \xrightarrow{b_1} c_1\,,\, b_1 \xrightarrow{c_1} b_2\,,\, b_2 \xrightarrow{a_0} b_1\,,\, a_0 \xrightarrow{b_1\,,\,c_1} a_1$$

$$b_0 \xrightarrow{c_0} b_1\,,\, c_0 \xrightarrow{b_1} c_1\,,\, d_0 \xrightarrow{c_1} d_1\,,\, a_0 \xrightarrow{b_1\,,\,c_1} a_1$$

$$b_0 \xrightarrow{c_0} b_1\,,\, c_0 \xrightarrow{b_1} c_1\,,\, a_0 \xrightarrow{b_1\,,\,c_1} a_1$$

Minimal traces in Automata Networks

Asynchronous semantics

Trace: sequence of local transitions
A trace $\pi \vDash P$ is minimal w.r.t. P iff there is no sub-sequence $\pi' \subsetneq \pi$ s.t. $\pi' \vDash P$.

Examples with $P =$ reach a_1:

$$b_0 \xrightarrow{c_0} b_1 \,,\, c_0 \xrightarrow{b_1} c_1 \,,\, \mathbf{b_1} \xrightarrow{\mathbf{c_1}} \mathbf{b_2} \,,\, \mathbf{b_2} \xrightarrow{\mathbf{a_0}} \mathbf{b_1} \,,\, a_0 \xrightarrow{b_1 \,,\, c_1} a_1$$

non-minimal

$$b_0 \xrightarrow{c_0} b_1 \,,\, c_0 \xrightarrow{b_1} c_1 \,,\, d_0 \xrightarrow{c_1} d_1 \,,\, a_0 \xrightarrow{b_1 \,,\, c_1} a_1$$

$$b_0 \xrightarrow{c_0} b_1 \,,\, c_0 \xrightarrow{b_1} c_1 \,,\, a_0 \xrightarrow{b_1 \,,\, c_1} a_1$$

Minimal traces in Automata Networks

Asynchronous semantics

Trace: sequence of local transitions
A trace $\pi \vDash P$ is minimal w.r.t. P iff there is no sub-sequence $\pi' \subsetneq \pi$ s.t. $\pi' \vDash P$.

Examples with $P =$ reach a_1:

$$b_0 \xrightarrow{c_0} b_1, c_0 \xrightarrow{b_1} c_1, \mathbf{b_1} \xrightarrow{\mathbf{c_1}} \mathbf{b_2}, \mathbf{b_2} \xrightarrow{\mathbf{a_0}} \mathbf{b_1}, a_0 \xrightarrow{b_1, c_1} a_1$$

non-minimal

$$b_0 \xrightarrow{c_0} b_1, c_0 \xrightarrow{b_1} c_1, \mathbf{d_0} \xrightarrow{\mathbf{c_1}} \mathbf{d_1}, a_0 \xrightarrow{b_1, c_1} a_1$$

non-minimal

$$b_0 \xrightarrow{c_0} b_1, c_0 \xrightarrow{b_1} c_1, a_0 \xrightarrow{b_1, c_1} a_1$$

Minimal traces in Automata Networks

Asynchronous semantics

Trace: sequence of local transitions
A trace $\pi \vDash P$ is minimal w.r.t. P iff there is no sub-sequence $\pi' \subsetneq \pi$ s.t. $\pi' \vDash P$.

Examples with $P =$ reach a_1:

$$b_0 \xrightarrow{c_0} b_1, c_0 \xrightarrow{b_1} c_1, \mathbf{b_1} \xrightarrow{\mathbf{c_1}} \mathbf{b_2}, \mathbf{b_2} \xrightarrow{\mathbf{a_0}} \mathbf{b_1}, a_0 \xrightarrow{b_1, c_1} a_1$$

non-minimal

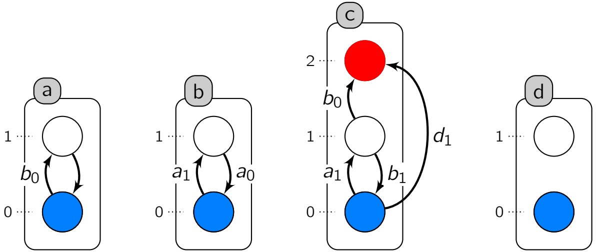
$$b_0 \xrightarrow{c_0} b_1, c_0 \xrightarrow{b_1} c_1, \mathbf{d_0} \xrightarrow{\mathbf{c_1}} \mathbf{d_1}, a_0 \xrightarrow{b_1, c_1} a_1$$

non-minimal

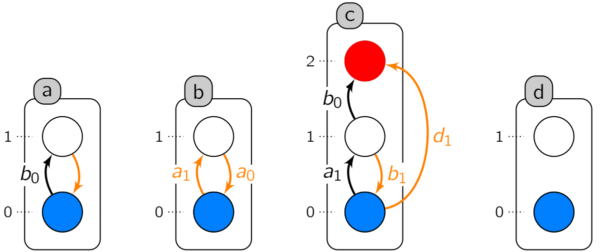
$$b_0 \xrightarrow{c_0} b_1, c_0 \xrightarrow{b_1} c_1, a_0 \xrightarrow{b_1, c_1} a_1$$

minimal

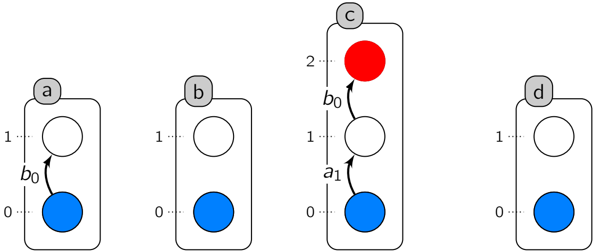
Goal-oriented reduction



Goal-oriented reduction



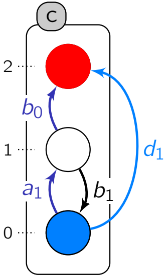
Goal-oriented reduction



Theorem

Goal-oriented reduction **preserves all minimal traces** *from initial states to goal.*

Local Causality



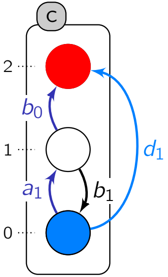
Objective: pair of local states of a same automaton
E.g., $c_0 \rightsquigarrow c_2$, $c_0 \rightsquigarrow c_0$, $d_0 \rightsquigarrow d_1$, ...

Local path: set of acyclic seq of local transitions

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

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

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

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 $c_0 \xrightarrow{a_1} c_1 \xrightarrow{b_0} c_2$ 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.

Helper: Necessary condition for reachability

Let us assume a predicate $\mathbf{valid}_s(a_i \rightsquigarrow a_j)$ such that:

$$\neg\mathbf{valid}_s(a_i \rightsquigarrow a_j) \Longrightarrow \nexists \text{ trace } \pi \text{ from } s \text{ reaching } a_i \text{ then } a_j$$

Helper: Necessary condition for reachability

Let us assume a predicate $\textbf{valid}_s(a_i \rightsquigarrow a_j)$ such that:

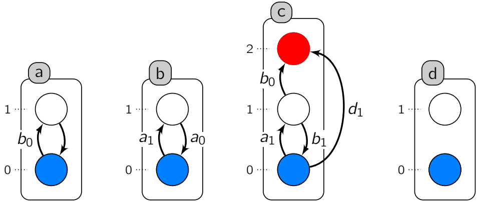
$$\neg\textbf{valid}_s(a_i \rightsquigarrow a_j) \implies \nexists \text{ trace } \pi \text{ from } s \text{ reaching } a_i \text{ then } a_j$$

Example of implementation

For this talk:

$$\textbf{valid}_s(a_i \rightsquigarrow a_j) \overset{\Delta}{\Leftrightarrow} \text{local-paths}(a_i \rightsquigarrow a_j) \neq \emptyset$$

For finer impl. see paper, and even finer see [Paulevé et al. in MSCS 2012].

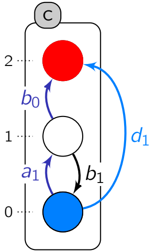


In particular: $\neg\textbf{valid}_s(d_0 \rightsquigarrow d_1)$.

Refining local paths

Given an initial state s, ignore local paths requiring non-valid objectives:

$$\text{filtered-local-paths}_s(a_i \rightsquigarrow a_j) \triangleq \{\eta \in \text{local-paths}(a_i \rightsquigarrow a_j) \mid \forall n \in \mathbb{I}^\eta, \forall b_k \in \text{enab}(\eta^n), \textbf{valid}_s(b_0 \rightsquigarrow b_k)\}$$



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

If $\neg\textbf{valid}_s(d_0 \rightsquigarrow d_1)$, 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. $b_j \xrightarrow{\ell} b_k \in \mathrm{tr}(\mathcal{B}) \Rightarrow \forall a_i \in \ell, a_0 \rightsquigarrow a_i \in \mathcal{B}$
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.**

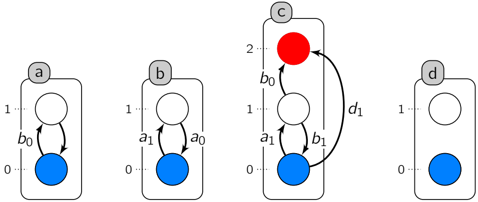
Reduction procedure

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.



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

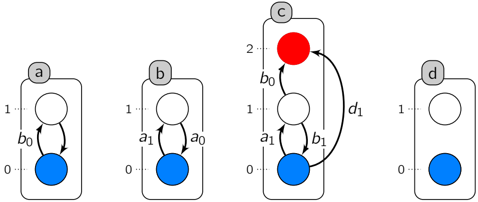
Reduction procedure

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.



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

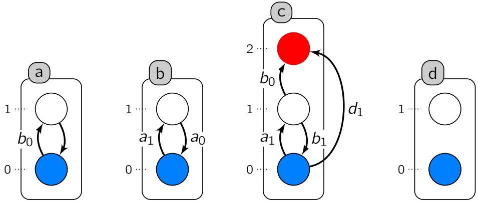
Reduction procedure

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.



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

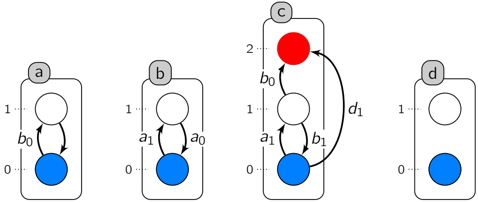
Reduction procedure

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.



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

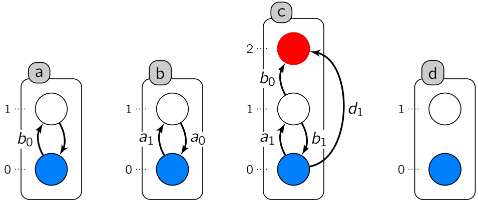
Reduction procedure

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

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



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

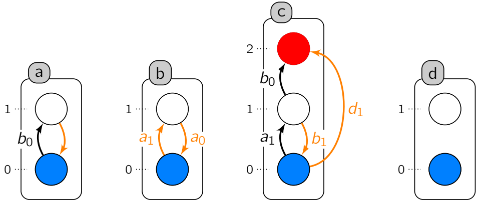
Reduction procedure

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

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



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

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**

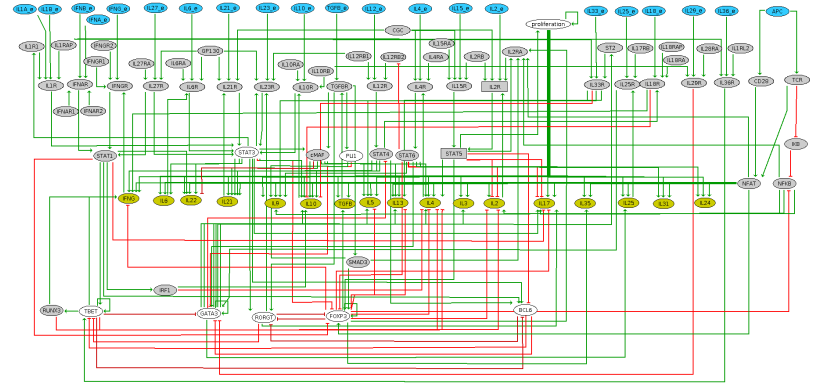
In all cases, reduction step took less than 0.1s

Experiments

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
```



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

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	**$1.8 \cdot 10^9$**	**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 minimal 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

Conclusion

Goal-oriented reduction of automata networks

- 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 minimal 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

Implemented in Pint – `http://loicpauleve.name/pint`

On-going work

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