Modelling, Simulation, and Large-Scale Model Checking of Biological Regulatory Networks

Loïc Paulevé, Morgan Magnin, Olivier Roux

{loic.pauleve,morgan.magnin,olivier.roux}@irccyn.ec-nantes.fr

http://www.irccyn.ec-nantes.fr/~pauleve

IRCCyN / MeForBio Team, Nantes (France)

DTU - 10th March 2011

IRCCyN laboratory, Nantes (France) - MeForBio Team

Loïc Paulevé

Morgan Magnin

Olivier Roux

Other people I work with

Andrew Phillips
Microsoft Research, Cambridge, UK
Stochastic simulation

Adrien Richard
I3S, Nice, France
Biological Regulatory Networks

Overview

Computer science for systems biology

- Models for dynamical concurrent systems.
- Validation of the model / control of the system.

- We focus on Biological Regulatory Networks (BRNs).
- We introduce a new modelling framework: the Process Hitting.

Overview

Computer science for systems biology

- Models for dynamical concurrent systems.
- Validation of the model / control of the system.
- We focus on Biological Regulatory Networks (BRNs).
- We introduce a new modelling framework: the Process Hitting.

The Process Hitting [Paulevé, Magnin, Roux in TCSB 2011]

- *Elementary* framework for dynamical complex systems;
- Applied to BRNs; not limited to.
- Stochastic and Time dimensions (simulation + standard model checking).
- Software available (PINT - `http://processhitting.wordpress.com`).

Overview

Computer science for systems biology

- Models for dynamical concurrent systems.
- Validation of the model / control of the system.
- We focus on Biological Regulatory Networks (BRNs).
- We introduce a new modelling framework: the Process Hitting.

The Process Hitting [Paulevé, Magnin, Roux in TCSB 2011]

- *Elementary* framework for dynamical complex systems;
- Applied to BRNs; not limited to.
- Stochastic and Time dimensions (simulation + standard model checking).
- Software available (PINT - `http://processhitting.wordpress.com`).

Large-scale model checking (dynamical properties)

- Cope with state space explosion.
- Our approach: Static Analysis of the model.
- Static analysis by Abstract Interpretation.

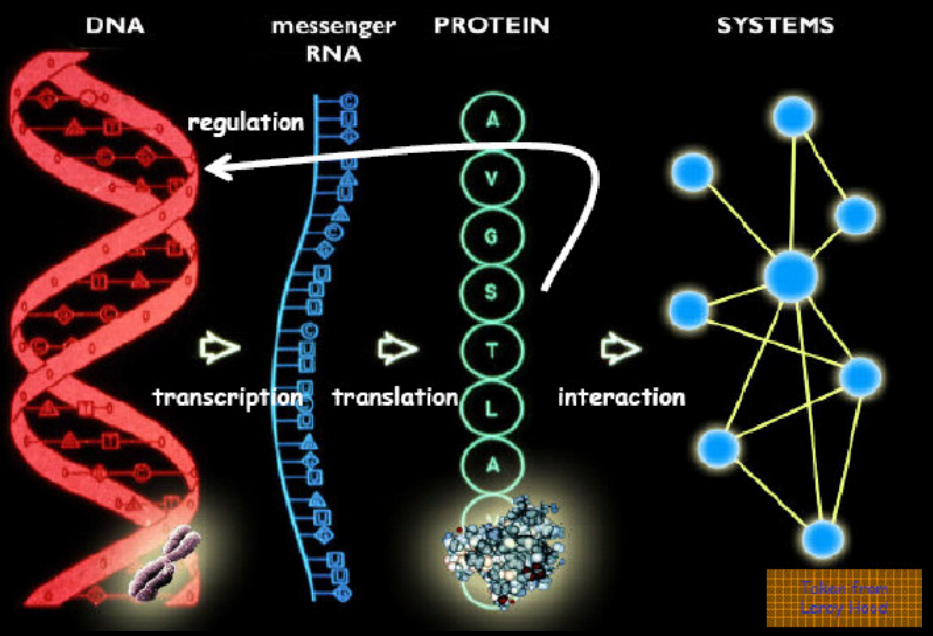
Outline

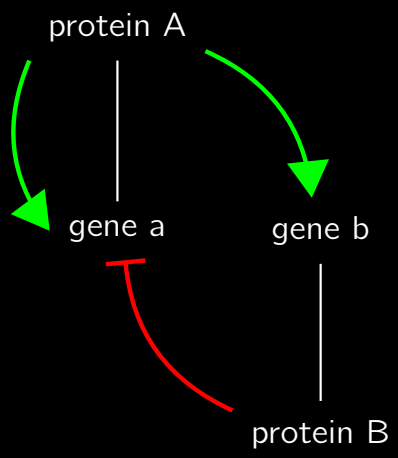
1. Introduction to BRNs
2. The Process Hitting
3. Stochastic and Time Parameters
4. Scalable Analysis of Dynamics
   - Dynamics Properties from BRN Interaction Graph
   - Fix Points
5. Abstract Interpretation of Scenarios
6. Applications
7. Discussion

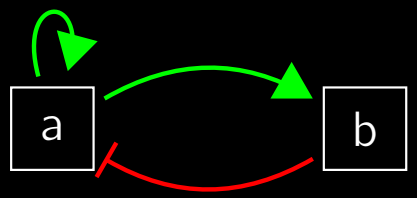
Outline

1. Introduction to BRNs
2. The Process Hitting
3. Stochastic and Time Parameters
4. Scalable Analysis of Dynamics
   - Dynamics Properties from BRN Interaction Graph
   - Fix Points
5. Abstract Interpretation of Scenarios
6. Applications
7. Discussion

Biological Regulatory Networks (BRNs)





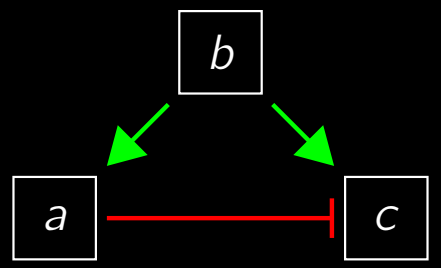


Loïc Paulevé, Morgan Magnin, Olivier Roux

Biological Regulatory Networks (BRNs)

- Relates components with actions of activation and inhibition;
- each component has a finite number of ordered states;
- the next state of a component is function of its activators/inhibitors.

Interaction graph

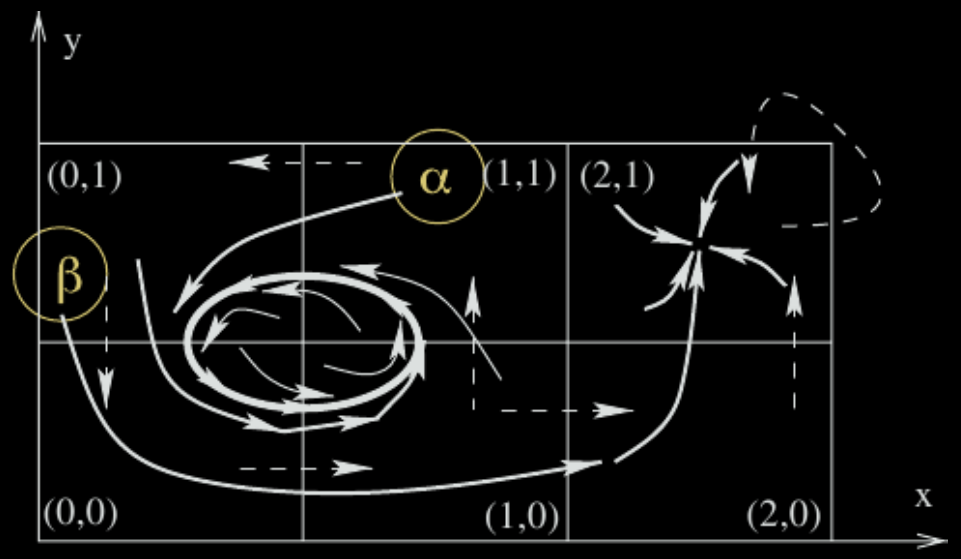


Cooperations

For instance (boolean case): $c = \neg a \wedge b$.

[René Thomas in Journal of Theoritical Biology, 1973] [A. Richard, J.-P. Comet, G. Bernot in Modern Formal Methods and Applications, 2006]

Quantitative Features



Some current work

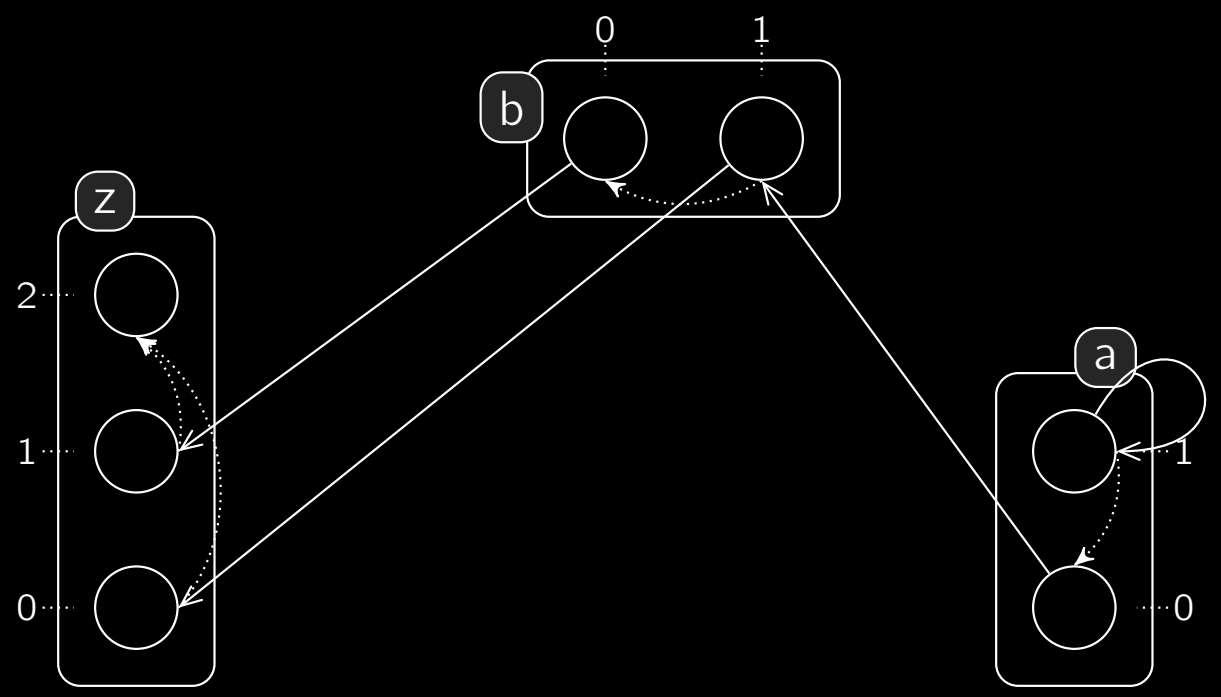
- Time(d), Stochastic Petri Nets [Heiner],
- Continuous-time Markov Chains (PRISM) [Kwiatkowska, Parker],
- Biocham [Fages], Kappa [Danos, Feret, Fontana, Krivine],
- Timed Automata [Siebert, Bockmayr],
- Linear Hybrid Automata [Ahmad, Roux].

Outline

1. Introduction to BRNs
2. The Process Hitting
3. Stochastic and Time Parameters
4. Scalable Analysis of Dynamics
   - Dynamics Properties from BRN Interaction Graph
   - Fix Points
5. Abstract Interpretation of Scenarios
6. Applications
7. Discussion

The Process Hitting Framework

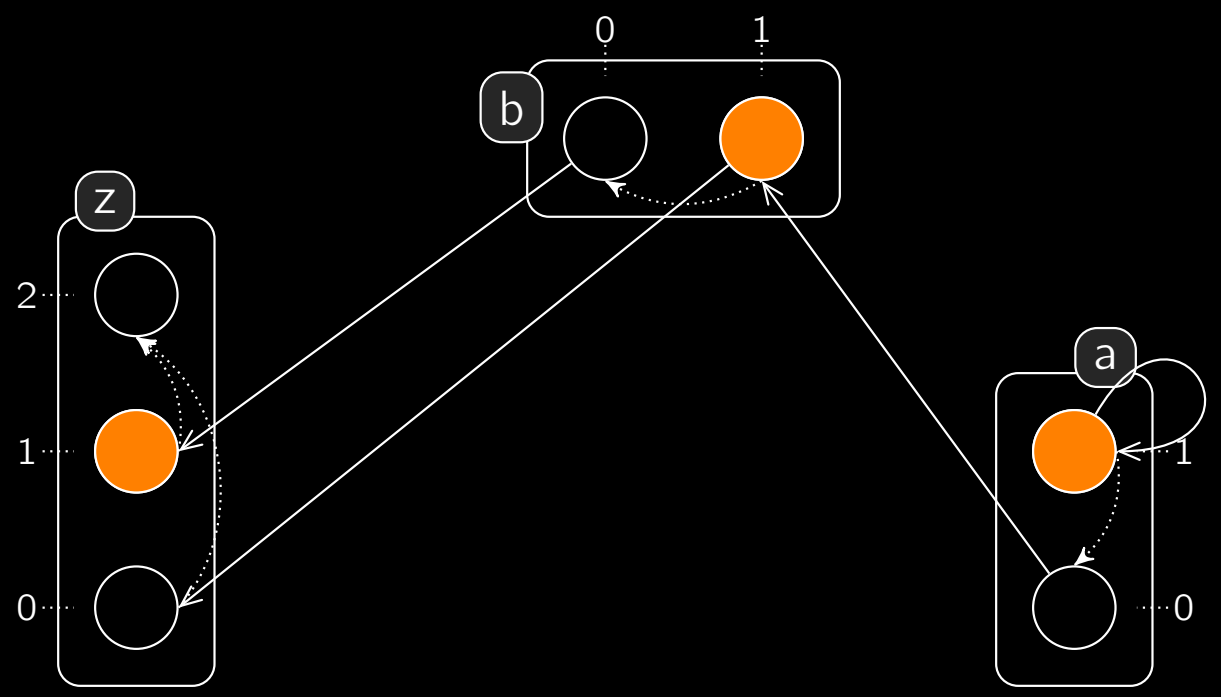
[Paulevé, Magnin, Roux in TCSB 2011]



- Sorts: a,b,z; Processes: $a_0, a_1, b_0, b_1, z_0, z_1, z_2$;
- Actions: a_0 *hits* b_1 to make it *bounce* to b_0, ...;
- States: $\langle a_1, b_1, z_1 \rangle$, $\langle a_0, b_1, z_1 \rangle$, $\langle a_0, b_0, z_1 \rangle$, ...;
- Restriction of Communicating Finite-State Machines (CFSM).

The Process Hitting Framework

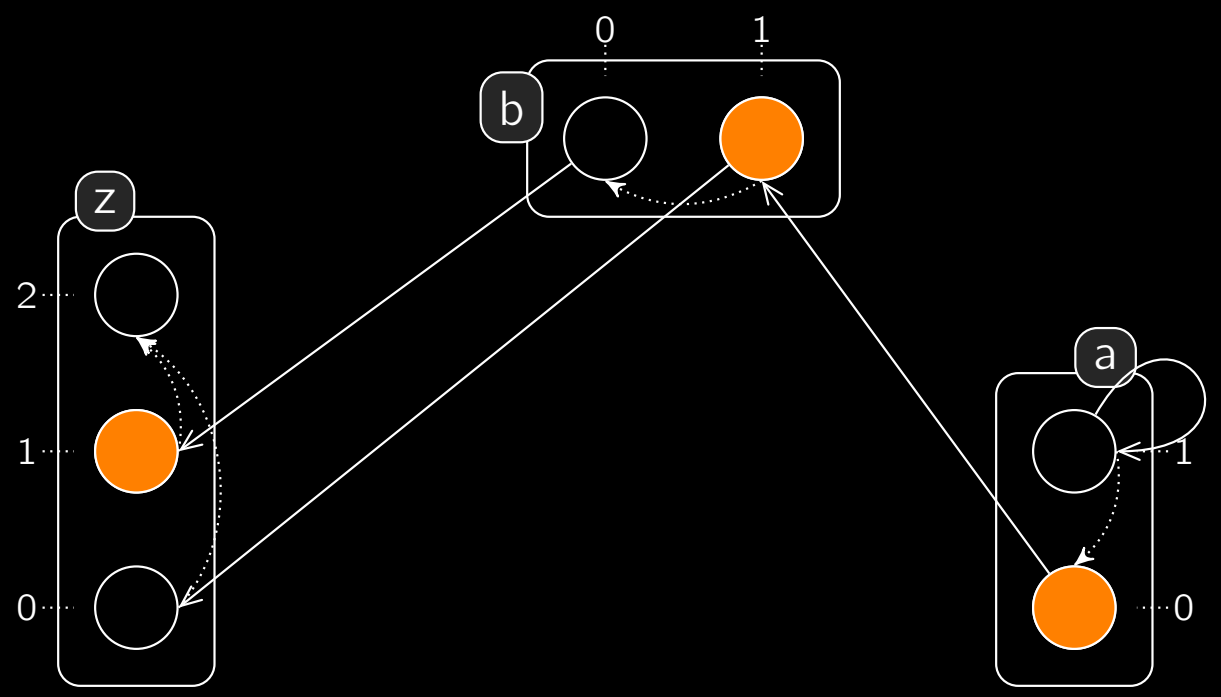
[Paulevé, Magnin, Roux in TCSB 2011]



- Sorts: a,b,z; Processes: $a_0, a_1, b_0, b_1, z_0, z_1, z_2$;
- Actions: a_0 *hits* b_1 to make it *bounce* to b_0, ...;
- States: $\langle a_1, b_1, z_1 \rangle$, $\langle a_0, b_1, z_1 \rangle$, $\langle a_0, b_0, z_1 \rangle$, ...;
- Restriction of Communicating Finite-State Machines (CFSM).

The Process Hitting Framework

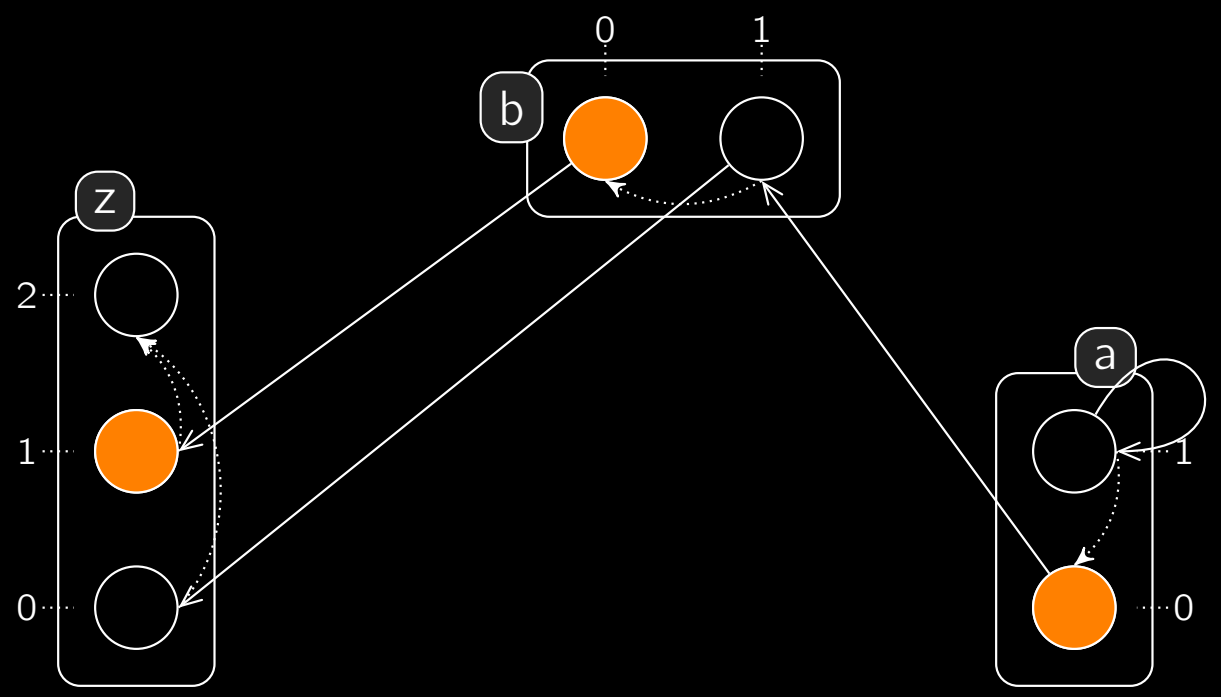
[Paulevé, Magnin, Roux in TCSB 2011]



- Sorts: a,b,z; Processes: $a_0, a_1, b_0, b_1, z_0, z_1, z_2$;
- Actions: a_0 *hits* b_1 to make it *bounce* to b_0, ...;
- States: $\langle a_1, b_1, z_1 \rangle$, $\langle a_0, b_1, z_1 \rangle$, $\langle a_0, b_0, z_1 \rangle$, ...;
- Restriction of Communicating Finite-State Machines (CFSM).

The Process Hitting Framework

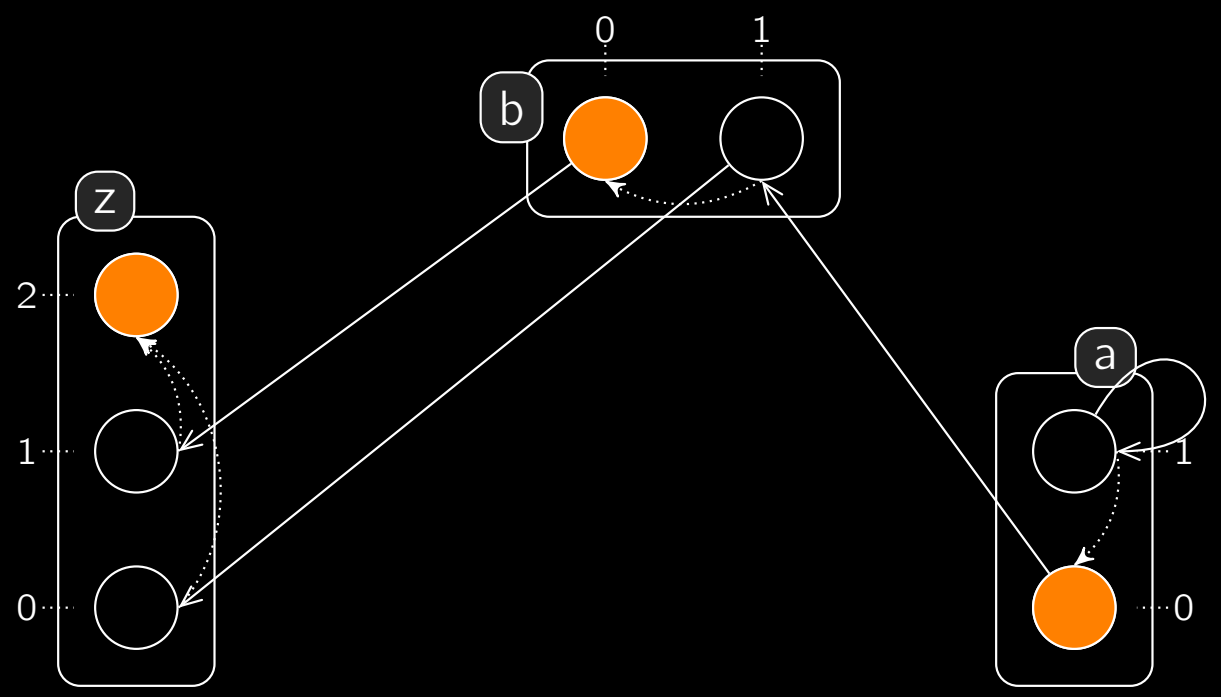
[Paulevé, Magnin, Roux in TCSB 2011]



- Sorts: a,b,z; Processes: $a_0, a_1, b_0, b_1, z_0, z_1, z_2$;
- Actions: a_0 *hits* b_1 to make it *bounce* to b_0, ...;
- States: $\langle a_1, b_1, z_1 \rangle$, $\langle a_0, b_1, z_1 \rangle$, $\langle a_0, b_0, z_1 \rangle$, ...;
- Restriction of Communicating Finite-State Machines (CFSM).

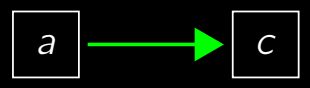
The Process Hitting Framework

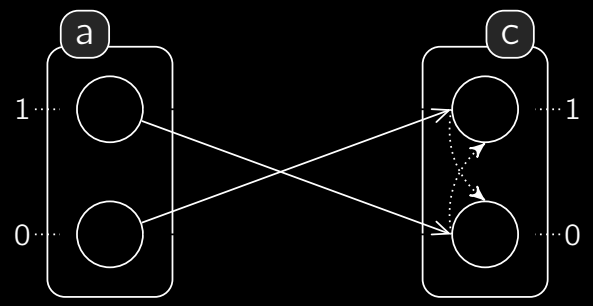
[Paulevé, Magnin, Roux in TCSB 2011]

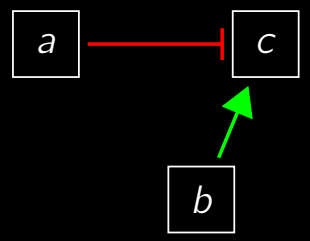


- Sorts: a,b,z; Processes: $a_0, a_1, b_0, b_1, z_0, z_1, z_2$;
- Actions: a_0 *hits* b_1 to make it *bounce* to b_0, ...;
- States: $\langle a_1, b_1, z_1 \rangle$, $\langle a_0, b_1, z_1 \rangle$, $\langle a_0, b_0, z_1 \rangle$, ...;
- Restriction of Communicating Finite-State Machines (CFSM).

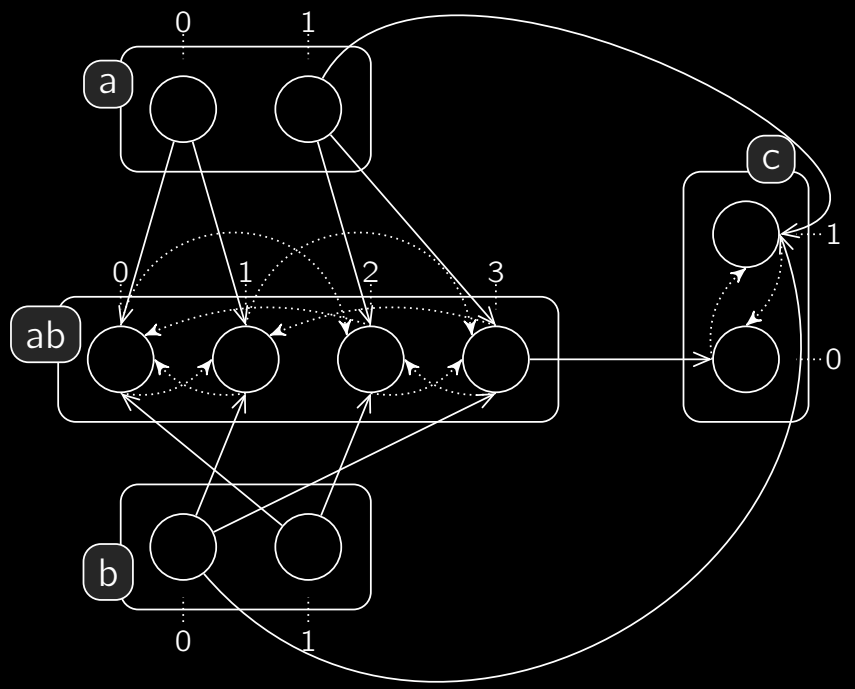
From BRNs to Process Hittings





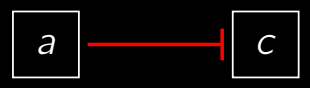


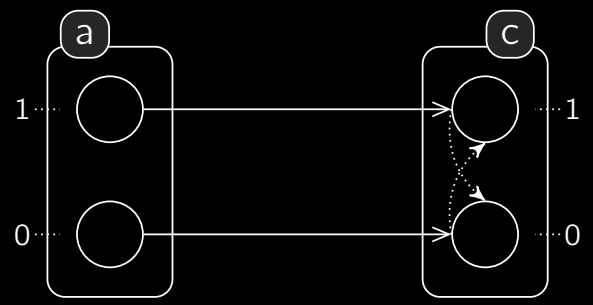
$c = \neg a \wedge b$

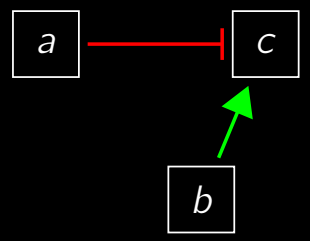


[Paulevé, Magnin, Roux in Trans. in Computational Systems Biology, 2011]

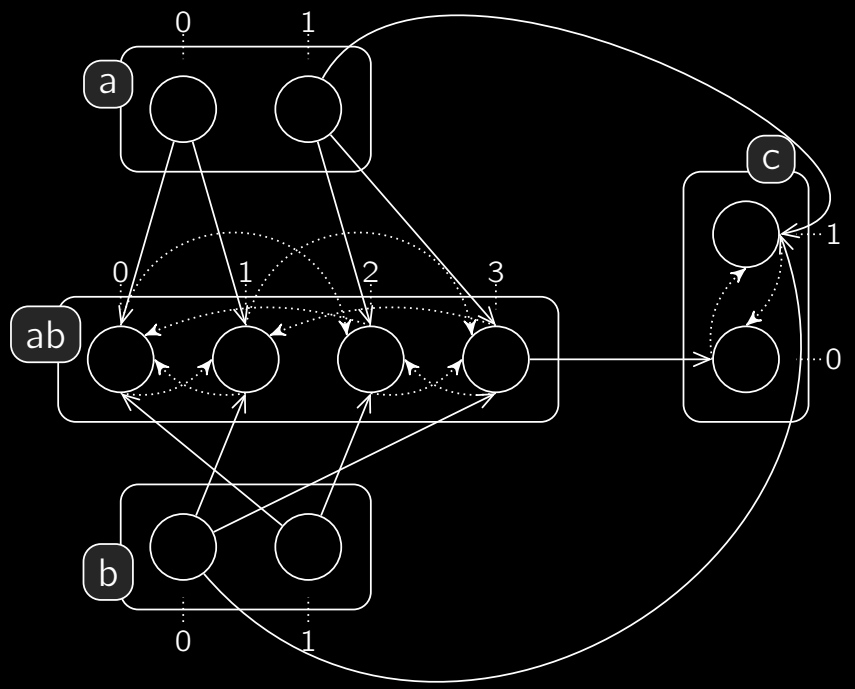
From BRNs to Process Hittings







$c = \neg a \wedge b$



[Paulevé, Magnin, Roux in Trans. in Computational Systems Biology, 2011]

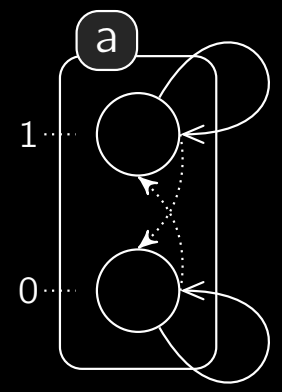
Outline

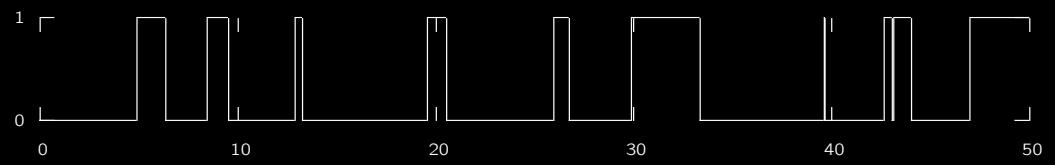
1. Introduction to BRNs
2. The Process Hitting
3. Stochastic and Time Parameters
4. Scalable Analysis of Dynamics
   - Dynamics Properties from BRN Interaction Graph
   - Fix Points
5. Abstract Interpretation of Scenarios
6. Applications
7. Discussion

Stochasticity Absorption Factor

[Paulevé, Magnin, Roux in IEEE TSE, 2010]

- Stochastic (Markovian) dimension is prominent
- ... but no precise time features;
- Exponential distribution: mean r^{-1}; variance r^{-2}.



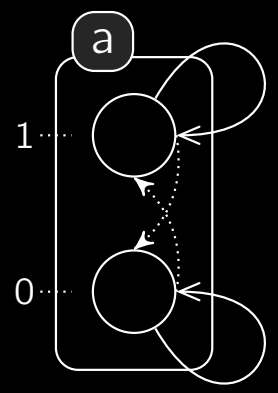


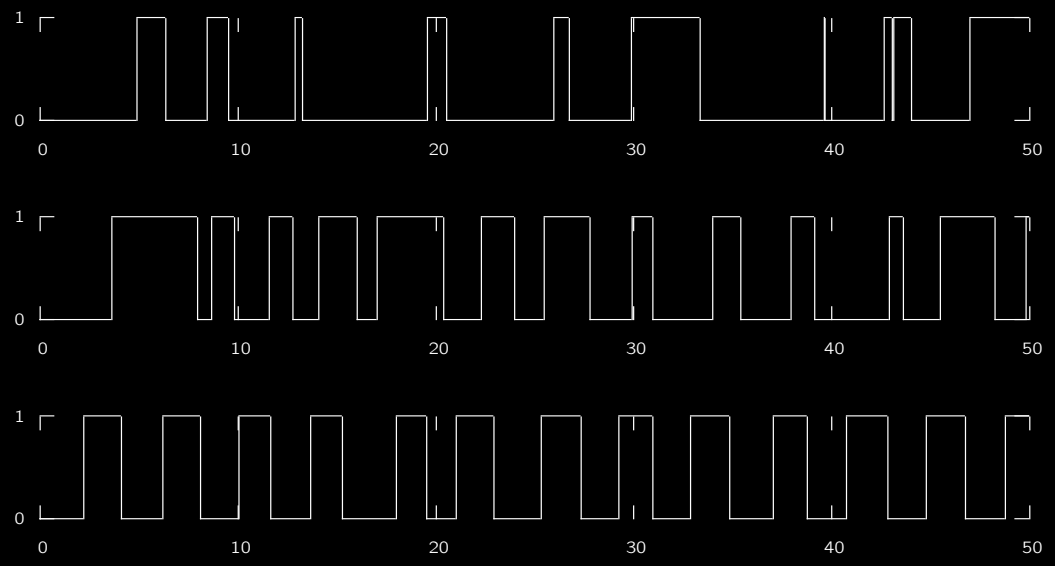
sa=1

Stochasticity Absorption Factor

[Paulevé, Magnin, Roux in IEEE TSE, 2010]

- Stochastic (Markovian) dimension is prominent
- ... but no precise time features;
- Exponential distribution: mean r^{-1}; variance r^{-2}.
- Provide a stochasticity absorption factor sa:
- duration follows the sum of sa exponentials of rate $r.sa$;
- mean r^{-1}; variance $r^{-2}sa^{-1}$ (Erlang distribution).





sa=1

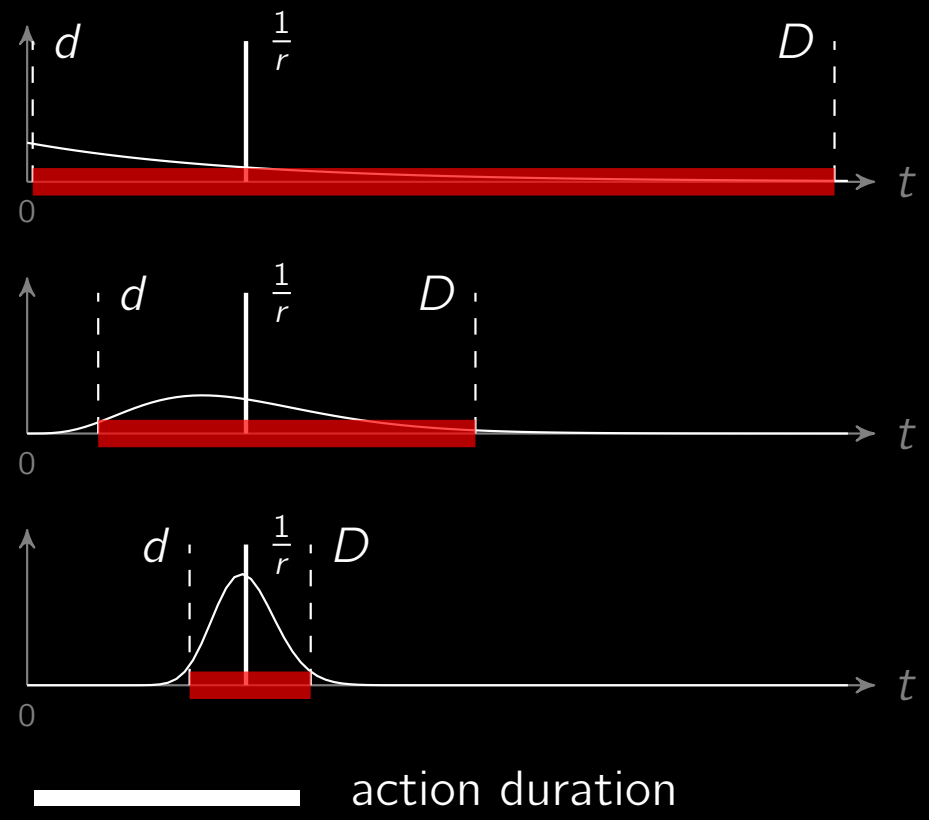
sa=5

sa=50

Stochastic and Time Parameters

[Paulevé, Magnin, Roux in IEEE TSE, 2010]

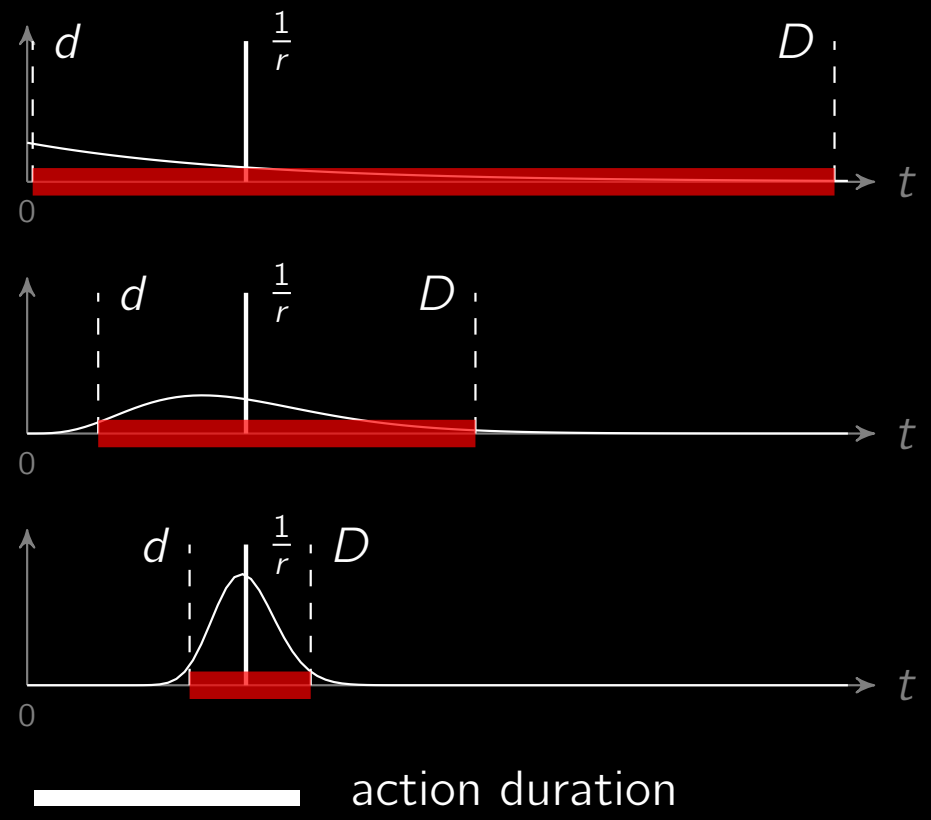
- Specify either (r, sa), or its firing interval $[d; D]$,
- which is the confidence interval at confidence coefficient $1 - \alpha$.
- We have estimators to translate $[d; D]$ to (r, sa).

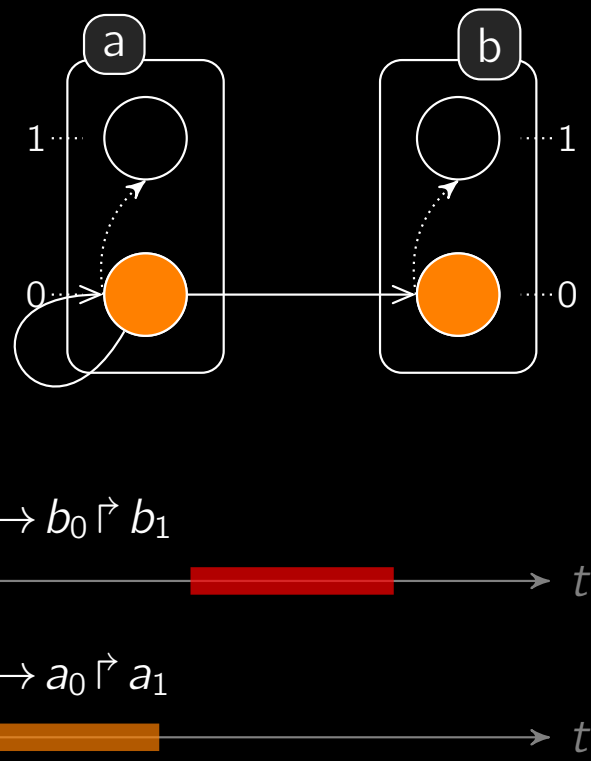


Stochastic and Time Parameters

[Paulevé, Magnin, Roux in IEEE TSE, 2010]

- Specify either (r, sa), or its firing interval $[d; D]$,
- which is the confidence interval at confidence coefficient $1 - \alpha$.
- We have estimators to translate $[d; D]$ to (r, sa).





$\Rightarrow b_1$ is reached at a very low probability.

Simulation and Model Checking

Probabilistic Model Checking

- Translation from the Erlang stochastic π-calculus to PRISM.
- Applies to the Process Hitting as well.
- Not tractable with large stochasticiy absorption factors;
- but there is hope in symmetry reductions, or abstractions of sequences of transitions, or ...

Simulation

- Non-Markovian simulation using the
- Generic abstract machine for stochastic process calculi [Paulevé, Youssef, Lakin, Phillips at CMSB 2010].
- Process Hitting simulator implemented in PINT.

Challenge: scalable inference of stochastic and time parameters...

Simulation and Model Checking

Probabilistic Model Checking

- Translation from the Erlang stochastic π-calculus to PRISM.
- Applies to the Process Hitting as well.
- Not tractable with large stochasticiy absorption factors;
- but there is hope in symmetry reductions, or abstractions of sequences of transitions, or ...

Simulation

- Non-Markovian simulation using the
- Generic abstract machine for stochastic process calculi [Paulevé, Youssef, Lakin, Phillips at CMSB 2010].
- Process Hitting simulator implemented in PINT.

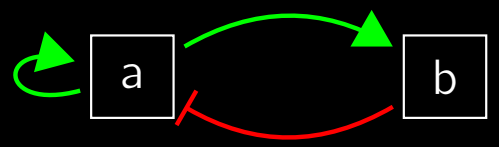
Challenge: scalable inference of stochastic and time parameters...
... still open; prior need for scalable qualitative analyses.

Outline

1. Introduction to BRNs
2. The Process Hitting
3. Stochastic and Time Parameters
4. Scalable Analysis of Dynamics
   - Dynamics Properties from BRN Interaction Graph
   - Fix Points
5. Abstract Interpretation of Scenarios
6. Applications
7. Discussion

Dynamics Properties from BRN Interaction Graph

An interaction graph can describe a large set of different dynamics (think to different cooperations between components).



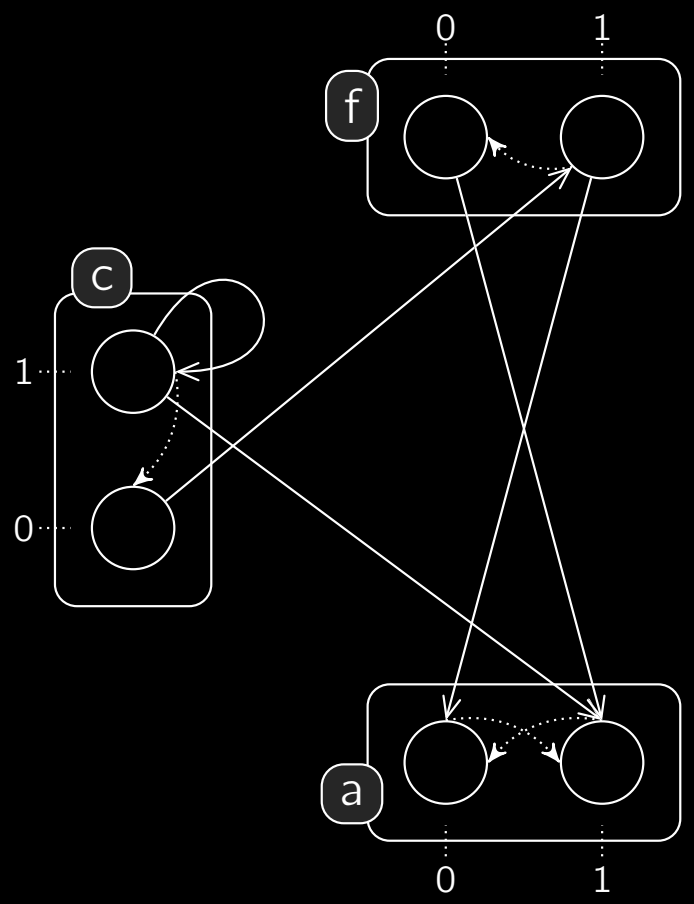
Mathematical work on relationship between the interaction graph and dynamical properties:

- Multi-stationnarity requires a positive circuit (René Thomas conjecture) [Soule in ComPlexUs, 2003] [Richard, Comet in Discrete Appl. Math., 2007].
- Sustained oscillations requires a negative circuit (René Thomas conjecture) [Remy, *et al.* in Adv. Appl. Math., 2008] [Richard in Adv. Appl. Math., 2010].
- The maximum number of fixed points can be characterized [Aracena in Bul. of Mathematical Biology, 2008]; [Richard in Discrete Appl. Math., 2009].
- Topological Fixed Points [Paulevé, Richard in CRAS 2010].

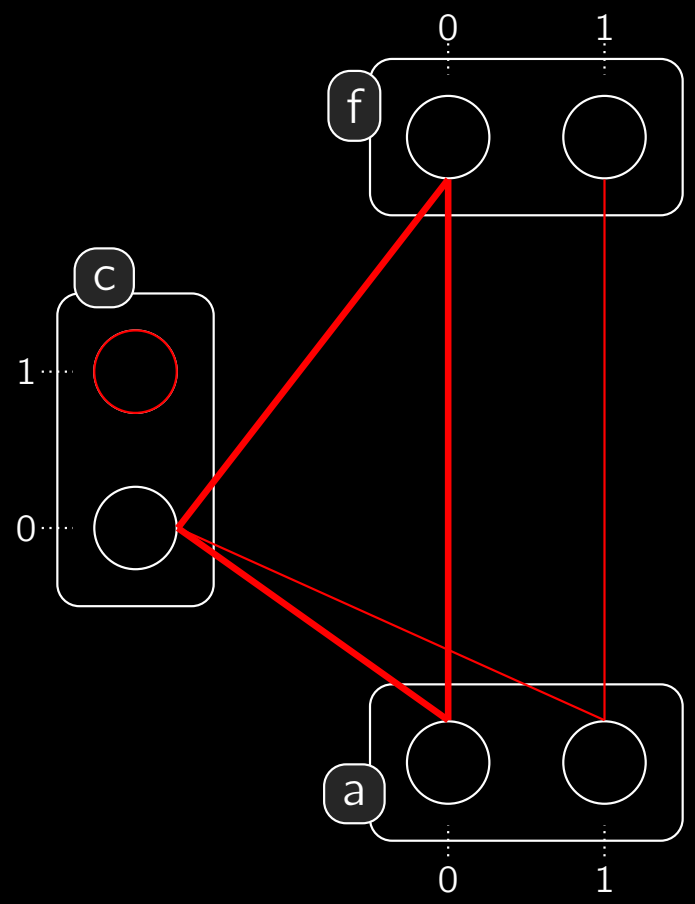
Fix Points

[Paulevé, Magnin, Roux in TCSB 2011]

Process Hitting



Hitless graph

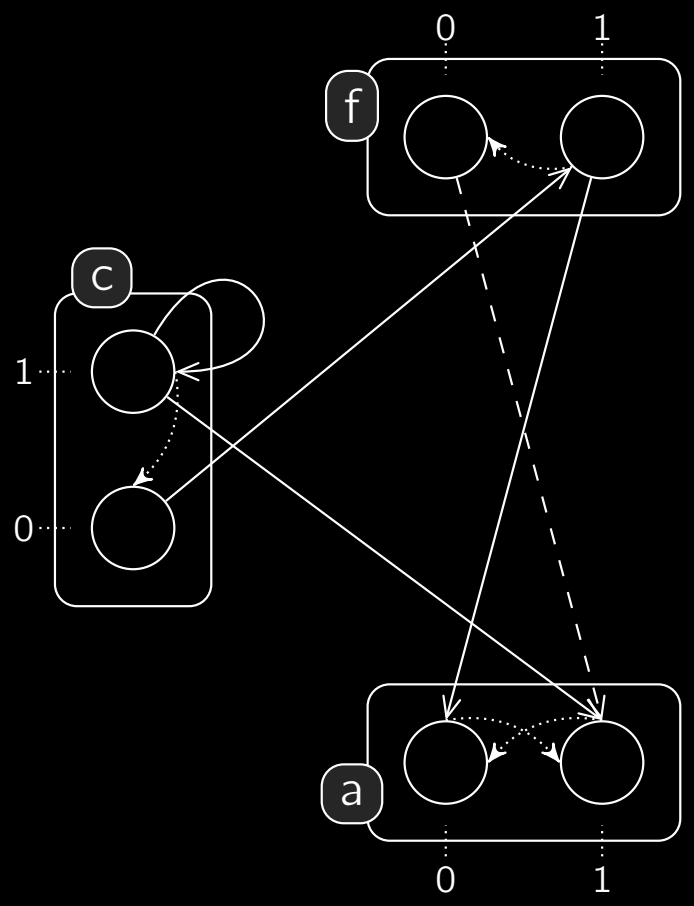


n-cliques <u>are</u> fix points

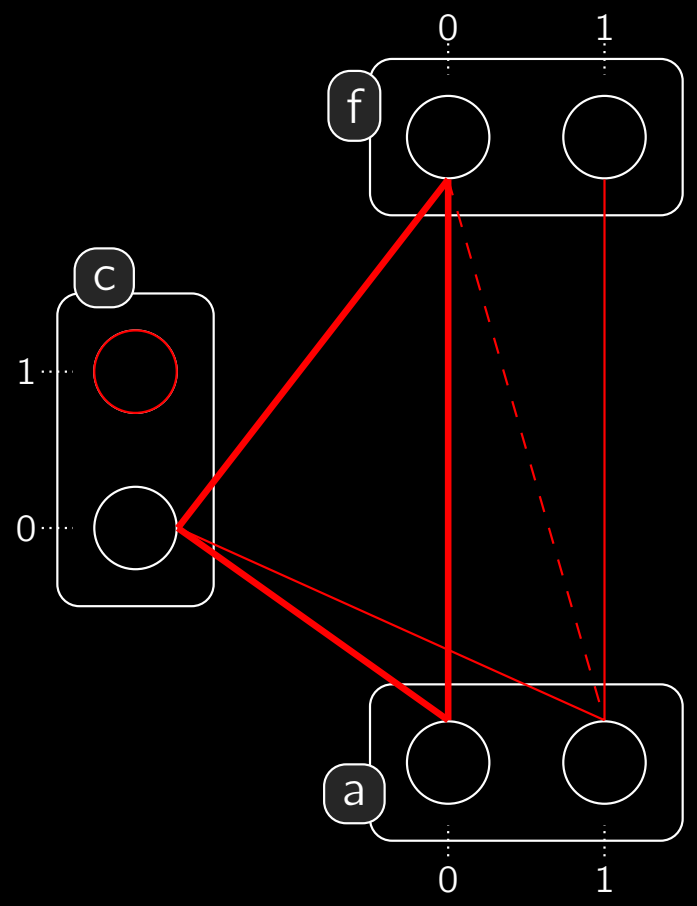
Fix Points

[Paulevé, Magnin, Roux in TCSB 2011]

Process Hitting



Hitless graph

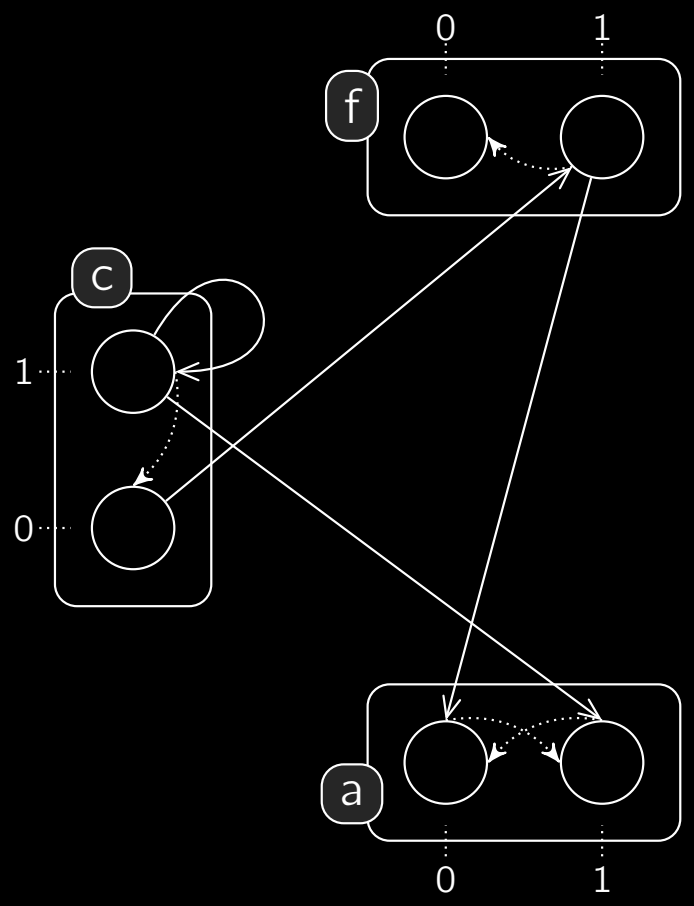


n-cliques <u>are</u> fix points

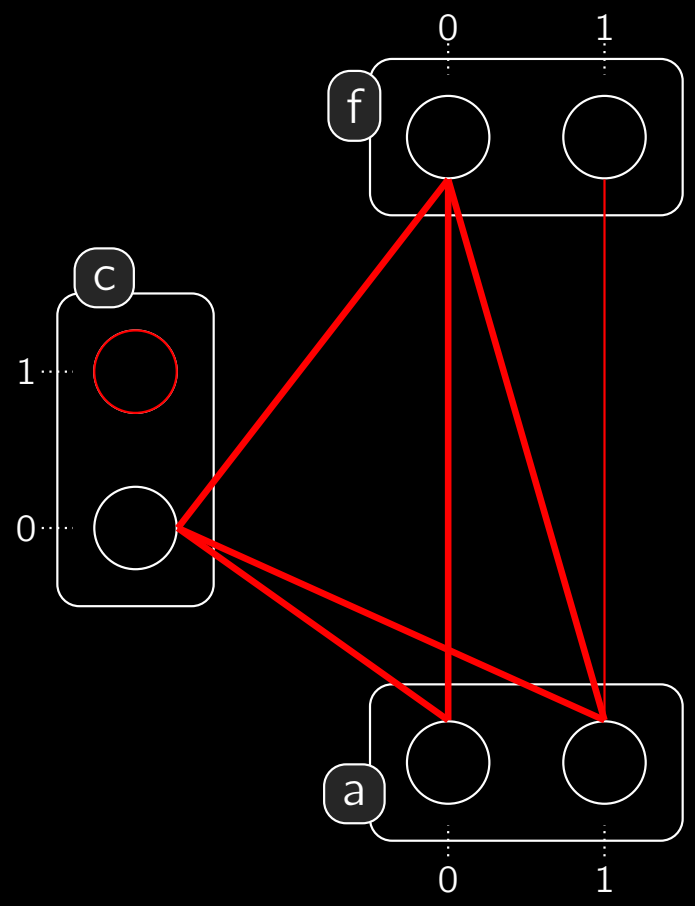
Fix Points

[Paulevé, Magnin, Roux in TCSB 2011]

Process Hitting



Hitless graph

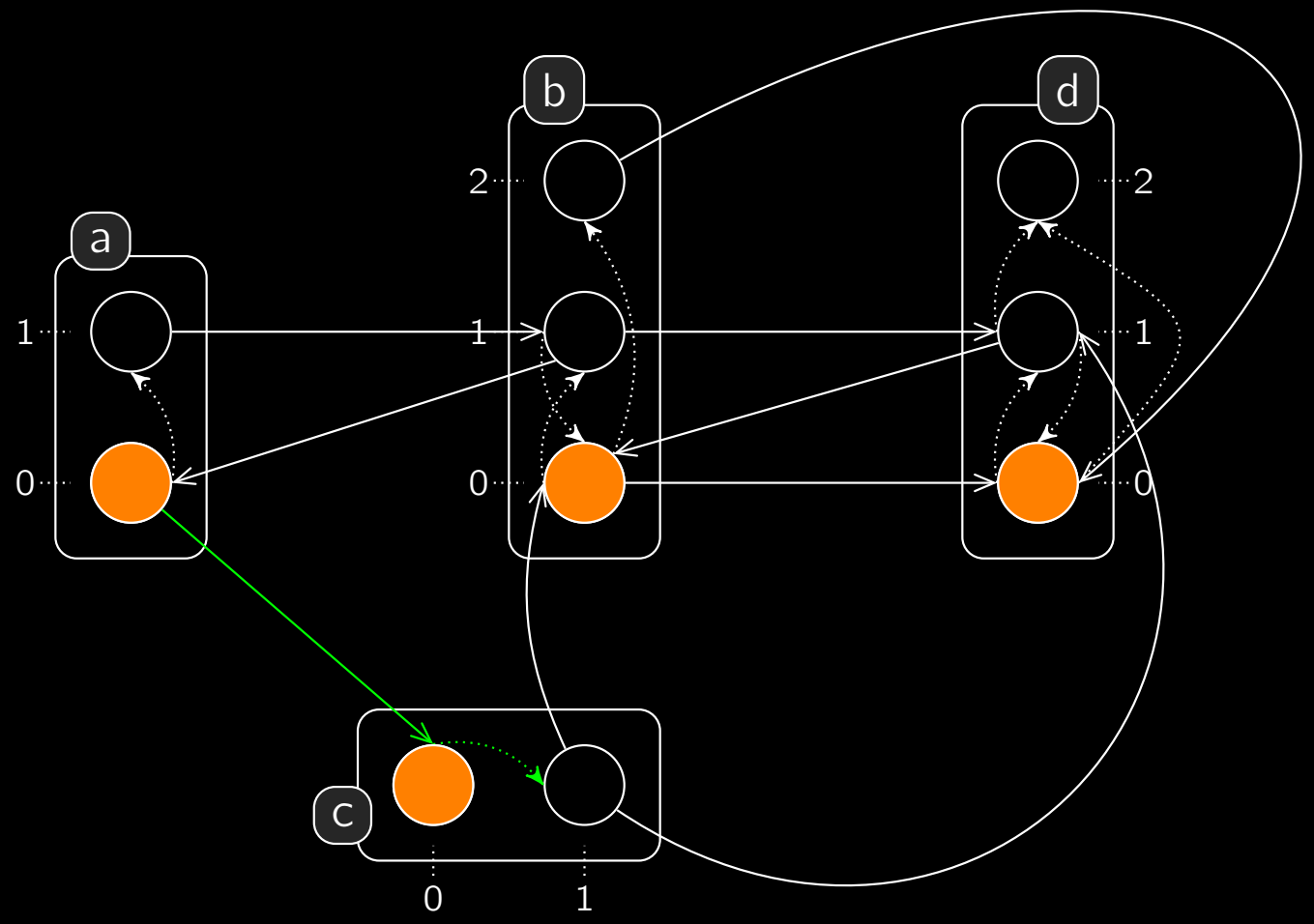


n-cliques <u>are</u> fix points

Outline

1. Introduction to BRNs
2. The Process Hitting
3. Stochastic and Time Parameters
4. Scalable Analysis of Dynamics
   - Dynamics Properties from BRN Interaction Graph
   - Fix Points
5. Abstract Interpretation of Scenarios
6. Applications
7. Discussion

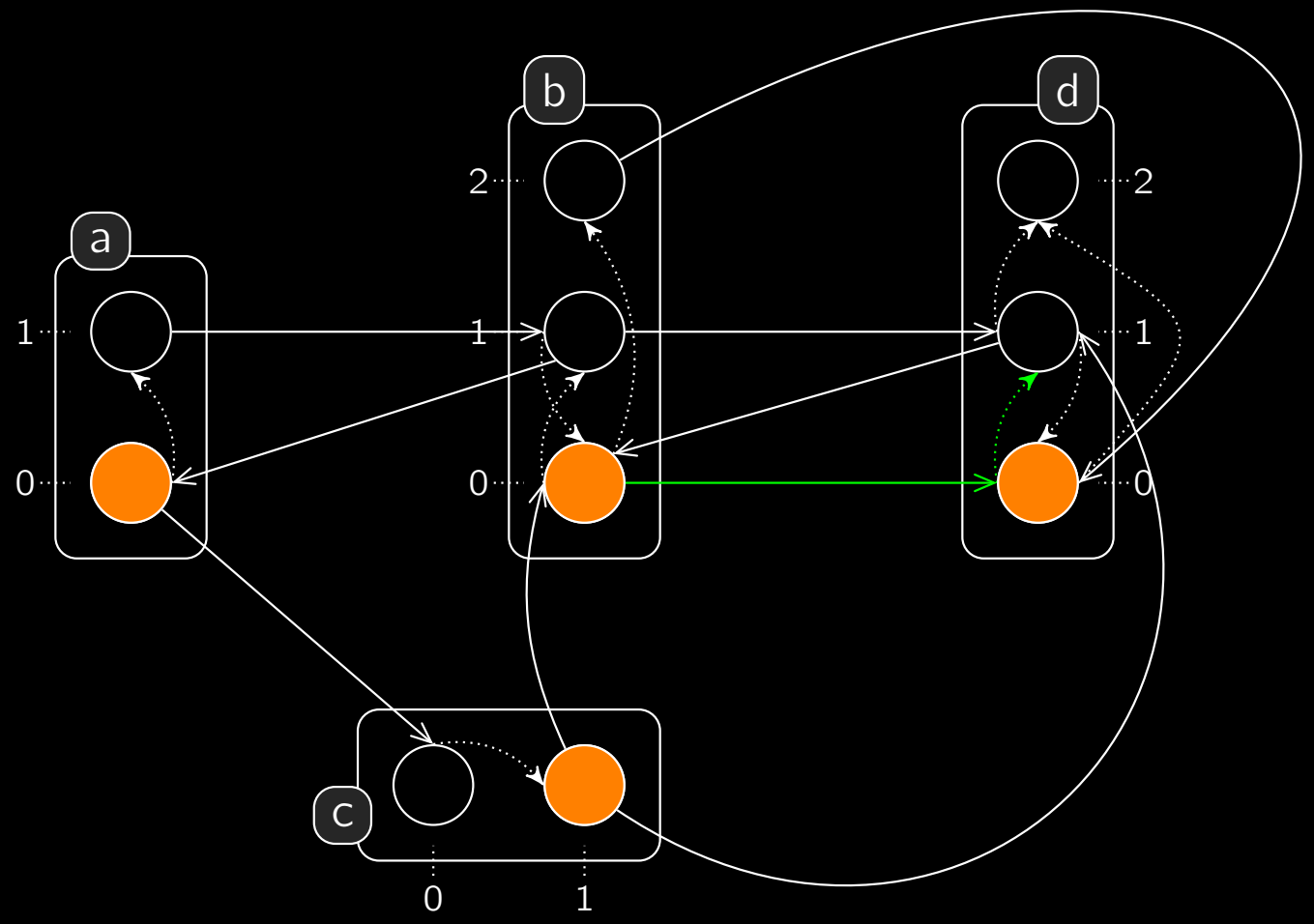
Scenarios



$a_0 \rightarrow c_0 \lceil c_1 :: b_0 \rightarrow d_0 \lceil d_1 :: c_1 \rightarrow b_0 \lceil b_1 :: b_1 \rightarrow d_1 \lceil d_2$

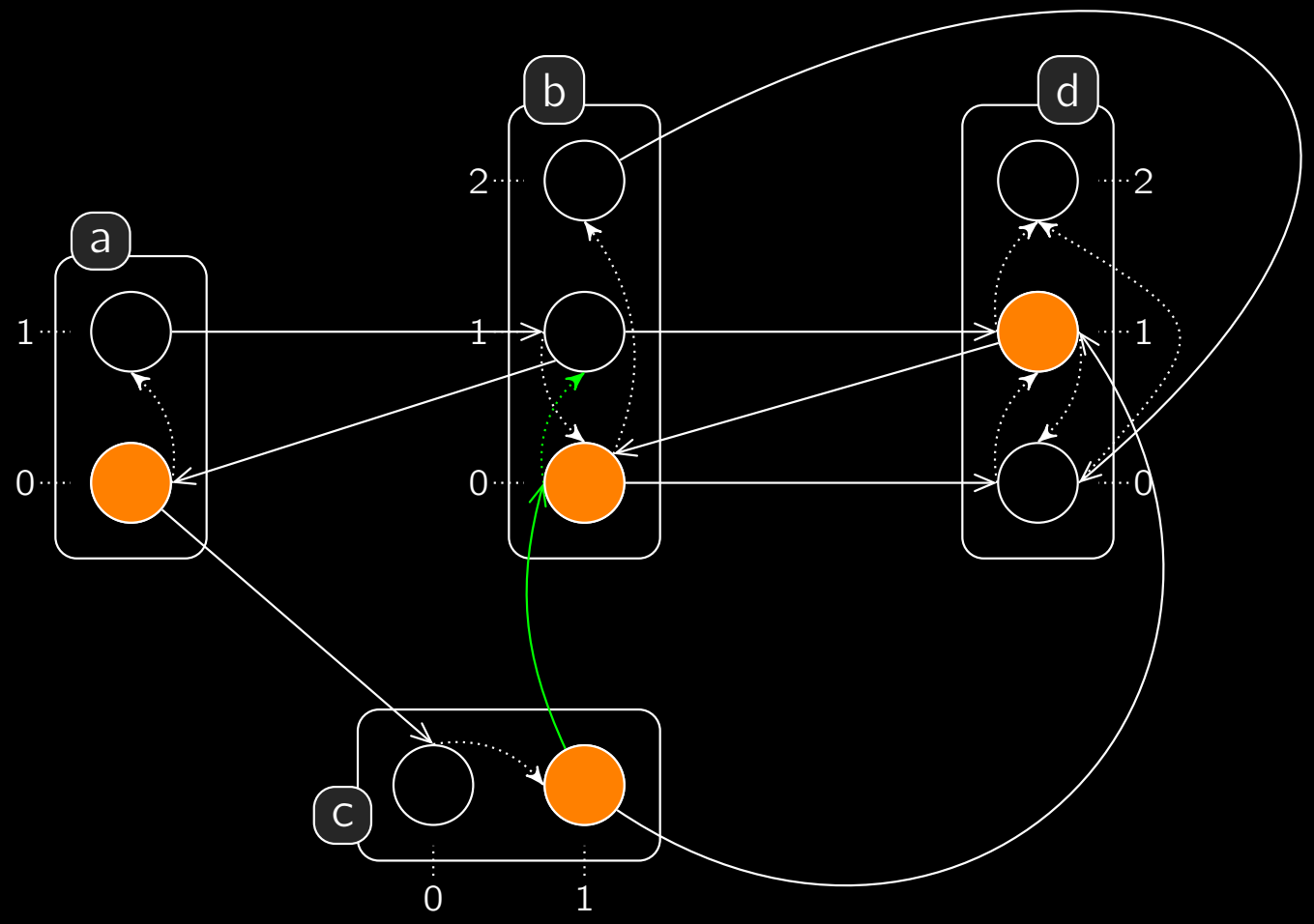
Loïc Paulevé, Morgan Magnin, Olivier Roux

Scenarios



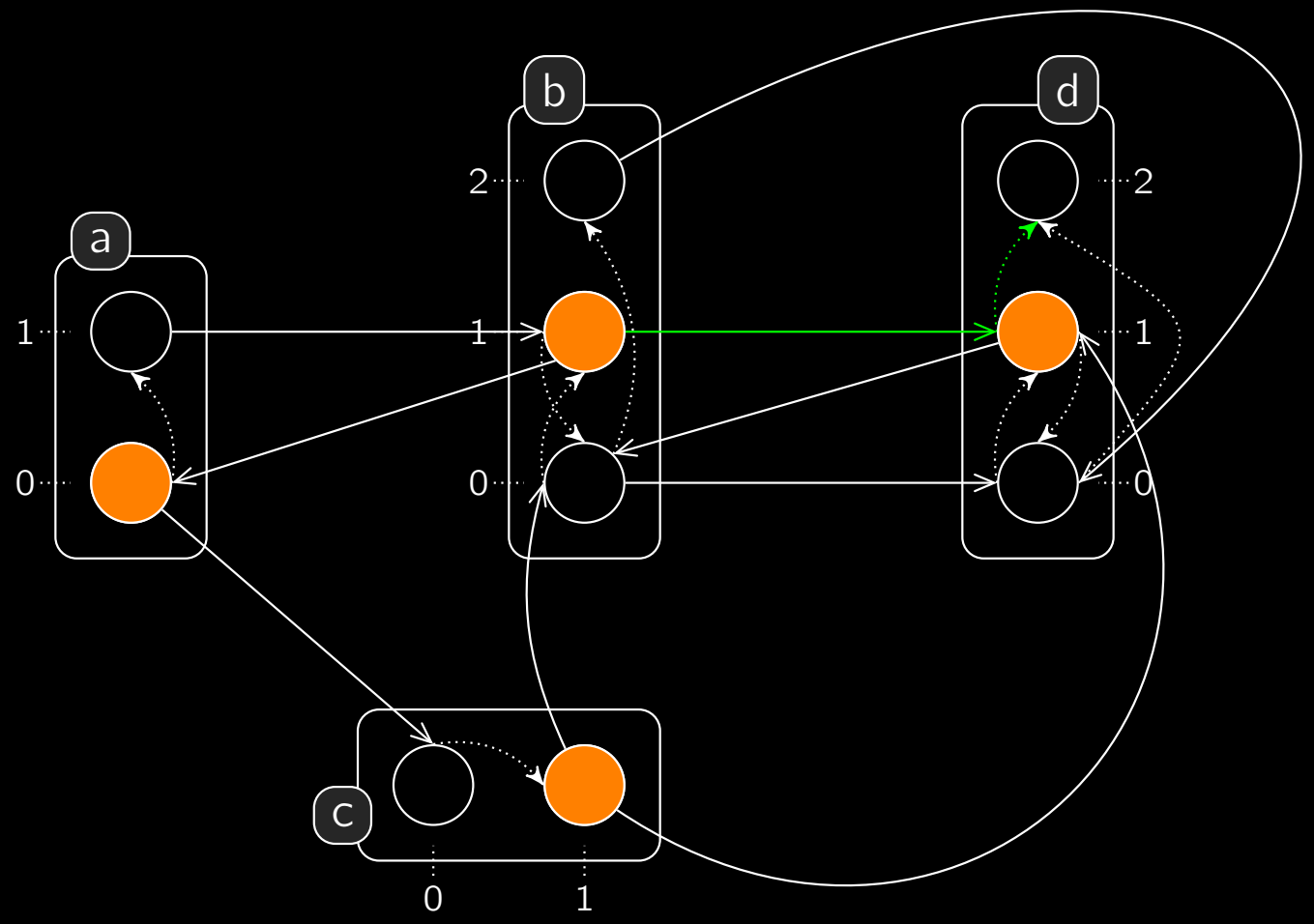
$$a_0 \rightarrow c_0 \lceil c_1 :: b_0 \rightarrow d_0 \lceil d_1 :: c_1 \rightarrow b_0 \lceil b_1 :: b_1 \rightarrow d_1 \lceil d_2$$

Scenarios



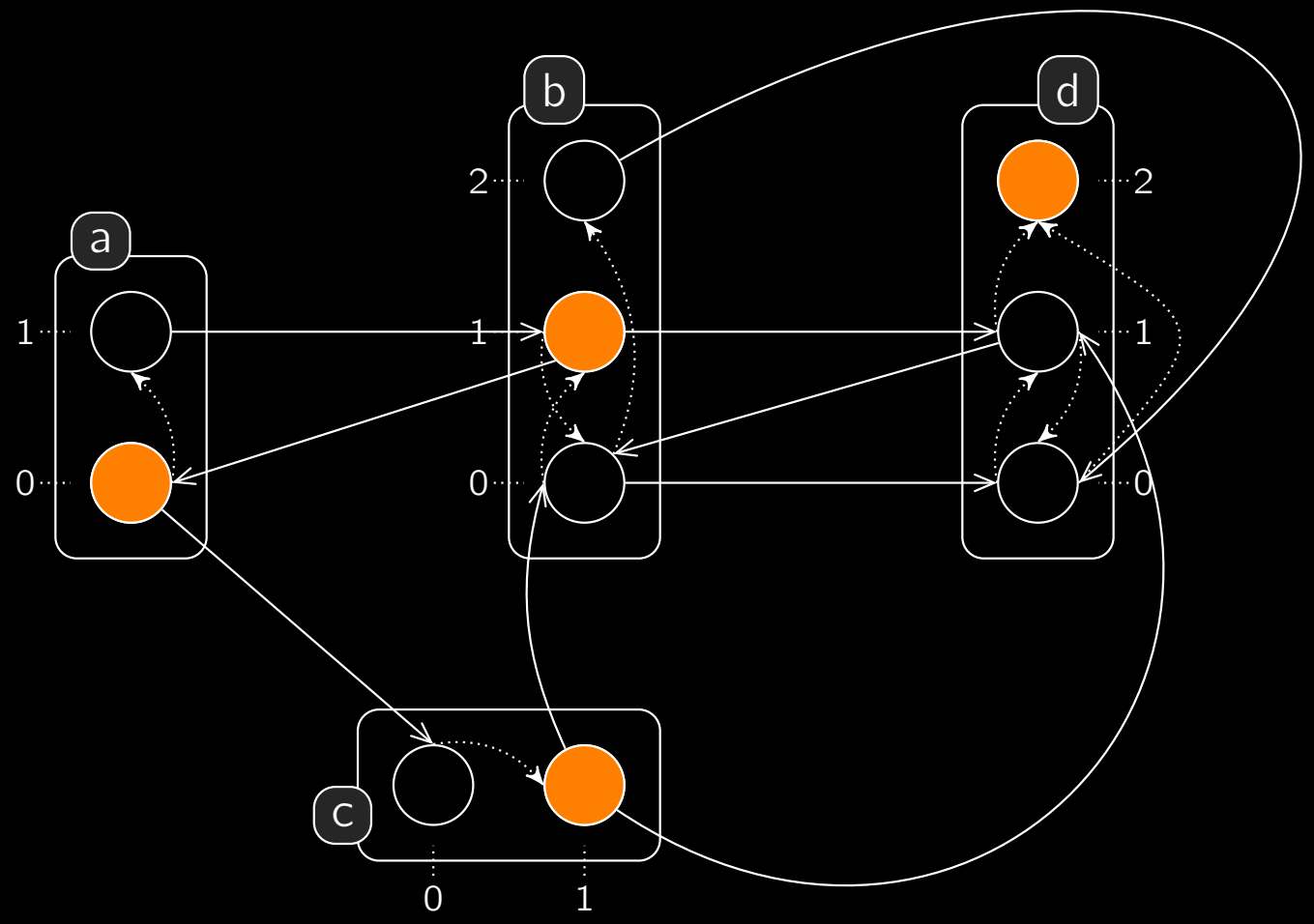
$$a_0 \rightarrow c_0 \lceil c_1 :: b_0 \rightarrow d_0 \lceil d_1 :: c_1 \rightarrow b_0 \lceil b_1 :: b_1 \rightarrow d_1 \lceil d_2$$

Scenarios



$$a_0 \rightarrow c_0 \lceil c_1 :: b_0 \rightarrow d_0 \lceil d_1 :: c_1 \rightarrow b_0 \lceil b_1 :: b_1 \rightarrow d_1 \lceil d_2$$

Scenarios



$$a_0 \rightarrow c_0 \lceil\!\lceil c_1 :: b_0 \rightarrow d_0 \lceil\!\lceil d_1 :: c_1 \rightarrow b_0 \lceil\!\lceil b_1 :: b_1 \rightarrow d_1 \lceil\!\lceil d_2$$

Abstract Interpretation of Scenarios

[Paulevé, Magnin, Roux at SASB 2010 + MSCS submitted]

Scenarios – Successively playable actions.

- E.g. $\delta = a_0 \rightarrow c_0 \Rsh c_1 :: b_0 \rightarrow d_0 \Rsh d_1 :: c_1 \rightarrow b_0 \Rsh b_1 :: b_1 \rightarrow d_1 \Rsh d_2$.

Context — For each sort, subset of initial processes.

- E.g. $\varsigma = \langle a_0, b_0, b_2, c_0, d_0 \rangle$.

Abstract Interpretation of Scenarios

[Paulevé, Magnin, Roux at SASB 2010 + MSCS submitted]

Scenarios – Successively playable actions.

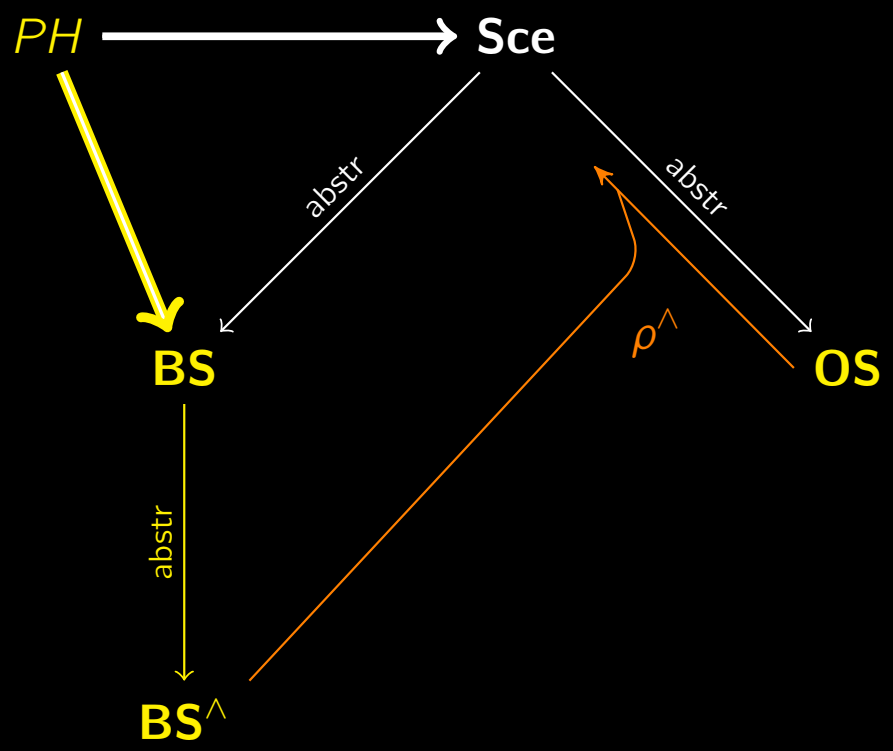
- E.g. $\delta = a_0 \rightarrow c_0 \Rsh c_1 :: b_0 \rightarrow d_0 \Rsh d_1 :: c_1 \rightarrow b_0 \Rsh b_1 :: b_1 \rightarrow d_1 \Rsh d_2$.

Context — For each sort, subset of initial processes.

- E.g. $\varsigma = \langle a_0, b_0, b_2, c_0, d_0 \rangle$.

Overall approach

- 2 orthogonal abstractions;
- Bounce Sequences **BS**;
- Objective Sequences **OS**;
- Concretization: $\gamma_\varsigma : \mathbf{OS} \mapsto \wp(\mathbf{Sce})$;
- Refinements: $\rho : \mathbf{OS} \mapsto \wp(\mathbf{OS})$;
- $\gamma_\varsigma(\omega) = \gamma_\varsigma(\rho(\omega))$.



Two Orthogonal Abstractions

$$a_0 \rightarrow c_0 \lceil^{\rightarrow} c_1 :: b_0 \rightarrow d_0 \lceil^{\rightarrow} d_1 :: c_1 \rightarrow b_0 \lceil^{\rightarrow} b_1 :: b_1 \rightarrow d_1 \lceil^{\rightarrow} d_2$$

Abstraction by Objective Sequences

- $c_0 \lceil^{\rightarrow *} c_1 :: d_0 \lceil^{\rightarrow *} d_1 :: b_0 \lceil^{\rightarrow *} b_1 :: d_1 \lceil^{\rightarrow *} d_2;$

Two Orthogonal Abstractions

$$a_0 \rightarrow c_0 \Rsh c_1 :: b_0 \rightarrow d_0 \Rsh d_1 :: c_1 \rightarrow b_0 \Rsh b_1 :: b_1 \rightarrow d_1 \Rsh d_2$$

Abstraction by Objective Sequences

- $c_0 \Rsh^* c_1 :: d_0 \Rsh^* d_1 :: b_0 \Rsh^* b_1 :: d_1 \Rsh^* d_2$;
- $b_0 \Rsh^* b_1 :: d_0 \Rsh^* d_2$

Two Orthogonal Abstractions

$$a_0 \rightarrow c_0 \lceil\!\!\rceil c_1 :: b_0 \rightarrow d_0 \lceil\!\!\rceil d_1 :: c_1 \rightarrow b_0 \lceil\!\!\rceil b_1 :: b_1 \rightarrow d_1 \lceil\!\!\rceil d_2$$

Abstraction by Objective Sequences

- $c_0 \lceil\!\!\rceil^* c_1 :: d_0 \lceil\!\!\rceil^* d_1 :: b_0 \lceil\!\!\rceil^* b_1 :: d_1 \lceil\!\!\rceil^* d_2$;
- $b_0 \lceil\!\!\rceil^* b_1 :: d_0 \lceil\!\!\rceil^* d_2$
- $d_0 \lceil\!\!\rceil^* d_2$, ...

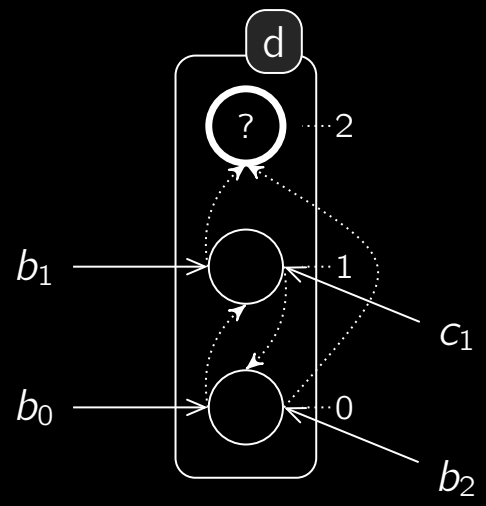
Two Orthogonal Abstractions

$$a_0 \rightarrow c_0 \lceil c_1 :: b_0 \rightarrow d_0 \lceil d_1 :: c_1 \rightarrow b_0 \lceil b_1 :: b_1 \rightarrow d_1 \lceil d_2$$

Abstraction by Objective Sequences

- $c_0 \lceil^* c_1 :: d_0 \lceil^* d_1 :: b_0 \lceil^* b_1 :: d_1 \lceil^* d_2$;
- $b_0 \lceil^* b_1 :: d_0 \lceil^* d_2$
- $d_0 \lceil^* d_2$, ...

Abstraction by Bounce Sequences



E.g.: $b_0 \rightarrow d_0 \lceil d_1 :: b_1 \rightarrow d_1 \lceil d_2 \ (d_0 \lceil^* d_2)$

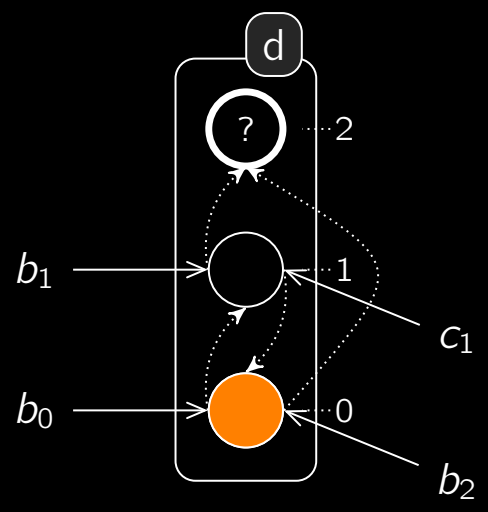
Two Orthogonal Abstractions

$$a_0 \rightarrow c_0 \lceil^{\rightarrow} c_1 :: b_0 \rightarrow d_0 \lceil^{\rightarrow} d_1 :: c_1 \rightarrow b_0 \lceil^{\rightarrow} b_1 :: b_1 \rightarrow d_1 \lceil^{\rightarrow} d_2$$

Abstraction by Objective Sequences

- $c_0 \lceil^{\rightarrow *} c_1 :: d_0 \lceil^{\rightarrow *} d_1 :: b_0 \lceil^{\rightarrow *} b_1 :: d_1 \lceil^{\rightarrow *} d_2$;
- $b_0 \lceil^{\rightarrow *} b_1 :: d_0 \lceil^{\rightarrow *} d_2$
- $d_0 \lceil^{\rightarrow *} d_2$, ...

Abstraction by Bounce Sequences



E.g.: $b_0 \rightarrow d_0 \lceil^{\rightarrow} d_1 :: b_1 \rightarrow d_1 \lceil^{\rightarrow} d_2$ $(d_0 \lceil^{\rightarrow *} d_2)$
$\Rightarrow$ can be computed off-line:

- $\mathbf{BS}(d_0 \lceil^{\rightarrow *} d_2) = \{b_0 \rightarrow d_0 \lceil^{\rightarrow} d_1 :: b_1 \rightarrow d_1 \lceil^{\rightarrow} d_2,$
  $b_2 \rightarrow d_0 \lceil^{\rightarrow} d_2]\}$;
- $\mathbf{BS}^{\wedge}(d_0 \lceil^{\rightarrow *} d_2) = \{\{b_0, b_1\}, \{b_2\}\}$.

Loïc Paulevé, Morgan Magnin, Olivier Roux

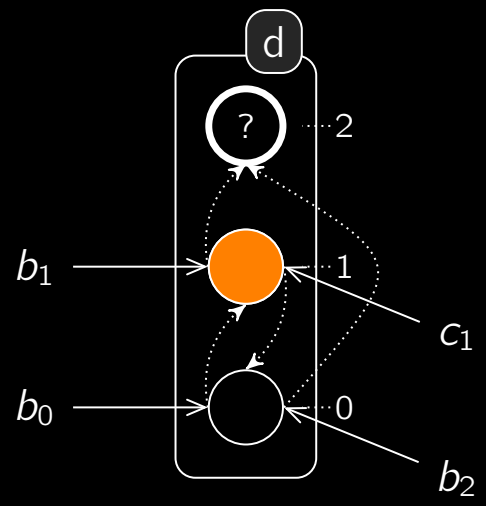
Two Orthogonal Abstractions

$$a_0 \rightarrow c_0 \Rsh c_1 :: b_0 \rightarrow d_0 \Rsh d_1 :: c_1 \rightarrow b_0 \Rsh b_1 :: b_1 \rightarrow d_1 \Rsh d_2$$

Abstraction by Objective Sequences

- $c_0 \Rsh^* c_1 :: d_0 \Rsh^* d_1 :: b_0 \Rsh^* b_1 :: d_1 \Rsh^* d_2$;
- $b_0 \Rsh^* b_1 :: d_0 \Rsh^* d_2$
- $d_0 \Rsh^* d_2$, ...

Abstraction by Bounce Sequences



E.g.: $b_0 \rightarrow d_0 \Rsh d_1 :: b_1 \rightarrow d_1 \Rsh d_2$ $(d_0 \Rsh^* d_2)$
$\Rightarrow$ can be computed off-line:

- $\mathbf{BS}(d_0 \Rsh^* d_2) = \{b_0 \rightarrow d_0 \Rsh d_1 :: b_1 \rightarrow d_1 \Rsh d_2,$
  $b_2 \rightarrow d_0 \Rsh d_2]\}$;
- $\mathbf{BS}^\wedge(d_0 \Rsh^* d_2) = \{\{b_0, b_1\}, \{b_2\}\}$.
- $\mathbf{BS}(d_1 \Rsh^* d_2) = \{b_1 \rightarrow d_1 \Rsh d_2,$
  $c_1 \rightarrow d_1 \Rsh d_0 :: b_2 \rightarrow d_0 \Rsh d_2\}$;
- $\mathbf{BS}^\wedge(d_1 \Rsh^* d_2) = \{\{b_1\}, \{b_2, c_1\}\}$.

Objective Sequence Refinements

$$\gamma_\varsigma(\omega) = \{\delta \in \textbf{Sce} \mid \omega \text{ abstracts } \delta \wedge \mathrm{support}(\delta) \subseteq \varsigma\}.$$

Idea: the more details we know, the better $\gamma_\varsigma(\omega)$ should be understood.

Objective Sequence Refinements

$$\gamma_\varsigma(\omega) = \{\delta \in \mathbf{Sce} \mid \omega \text{ abstracts } \delta \wedge \mathrm{support}(\delta) \subseteq \varsigma\}.$$

Idea: the more details we know, the better $\gamma_\varsigma(\omega)$ should be understood.

Objective Refinement by $\mathbf{BS}^\wedge$: $\rho^\wedge$

$\mathbf{Obj} \times \wp(\mathbf{BS}^\wedge)$	$\wp(\mathbf{OS})$
$d_0 \Rsh^* d_2$, $\{\{b_0, b_1\}, \{b_2\}\}$	$\star \Rsh^* b_0 :: b_0 \Rsh^* b_1 :: d_0 \Rsh^* d_2$, $\star \Rsh^* b_1 :: b_1 \Rsh^* b_0 :: d_0 \Rsh^* d_2$, $\star \Rsh^* b_2 :: d_0 \Rsh^* d_2$
$\gamma_\varsigma(d_0 \Rsh^* d_2)$	$= \gamma_\varsigma(\rho^\wedge(d_0 \Rsh^* d_2, \mathbf{BS}^\wedge(d_0 \Rsh^* d_2)))$

Objective Sequence Refinements

$$\gamma_\varsigma(\omega) = \{\delta \in \mathbf{Sce} \mid \omega \text{ abstracts } \delta \wedge \mathrm{support}(\delta) \subseteq \varsigma\}.$$

Idea: the more details we know, the better $\gamma_\varsigma(\omega)$ should be understood.

Objective Refinement by $\mathbf{BS}^\wedge$: $\rho^\wedge$

$\mathbf{Obj} \times \wp(\mathbf{BS}^\wedge)$	$\wp(\mathbf{OS})$
$d_0 \Rsh^* d_2$, $\{\{b_0, b_1\}, \{b_2\}\}$	$\star \Rsh^* b_0 :: b_0 \Rsh^* b_1 :: d_0 \Rsh^* d_2$, $\star \Rsh^* b_1 :: b_1 \Rsh^* b_0 :: d_0 \Rsh^* d_2$, $\star \Rsh^* b_2 :: d_0 \Rsh^* d_2$
$\gamma_\varsigma(d_0 \Rsh^* d_2)$	$= \gamma_\varsigma(\rho^\wedge(d_0 \Rsh^* d_2, \mathbf{BS}^\wedge(d_0 \Rsh^* d_2)))$

Generalization to **OS** refinements: $\widetilde{\rho}$

$\mathbf{OS} \times \wp(\mathbf{BS}^\wedge)$	$\wp(\mathbf{OS})$
$\omega, \mathbf{BS}^\wedge$	$\mathrm{interleave}\begin{pmatrix}\omega' \\ \omega_{1..n-1}\end{pmatrix} :: \omega_{n..\lvert\omega\rvert}$ where $n \in \mathbb{I}^\omega$ and $\omega' :: \omega_n \in \rho^\wedge(\omega_n, \mathbf{BS}^\wedge(\omega_n))$
$\gamma_\varsigma(\omega)$	$= \gamma_\varsigma(\widetilde{\rho}(\omega, \mathbf{BS}^\wedge))$

Objective Sequence Concretizability

a.k.a. Process Reachability Problem

$\gamma_s(a_i \Rsh^* a_j :: \ldots :: z_k \Rsh^* z_l) \neq \emptyset$?

CTL: $EF(s[a] = a_j \wedge EF(\cdots \wedge EF(s'[z] = z_l) \ldots))$.

Objective Sequence Concretizability

a.k.a. Process Reachability Problem

$\gamma_s(a_i \Rsh^* a_j :: \ldots :: z_k \Rsh^* z_l) \neq \emptyset$?

CTL: $EF(s[a] = a_j \wedge EF(\cdots \wedge EF(s'[z] = z_l) \ldots))$.

Over-approximation

- Un-ordered: $\gamma_s(\omega) \neq \emptyset \Longrightarrow \gamma_s(\omega_n) \neq \emptyset, \forall n \in \mathbb{I}^\omega$
  (recursive reasoning with $\rho^\wedge : \mathbf{Obj} \mapsto \wp(\mathbf{OS})$).

Objective Sequence Concretizability

a.k.a. Process Reachability Problem

$\gamma_\varsigma(a_i \lceil^* a_j :: \ldots :: z_k \lceil^* z_l) \neq \emptyset$?

CTL: $EF(s[a] = a_j \wedge EF(\cdots \wedge EF(s'[z] = z_l) \ldots))$.

Over-approximation

- Un-ordered: $\gamma_\varsigma(\omega) \neq \emptyset \Longrightarrow \gamma_\varsigma(\omega_n) \neq \emptyset, \forall n \in \mathbb{I}^\omega$ (recursive reasoning with $\rho^\wedge : \mathbf{Obj} \mapsto \wp(\mathbf{OS})$).
- Ordered: $\gamma_\varsigma(a_i \lceil^* a_j :: \omega) \neq \emptyset \Longrightarrow \gamma_{max\varsigma}(\omega) \neq \emptyset$.

Objective Sequence Concretizability

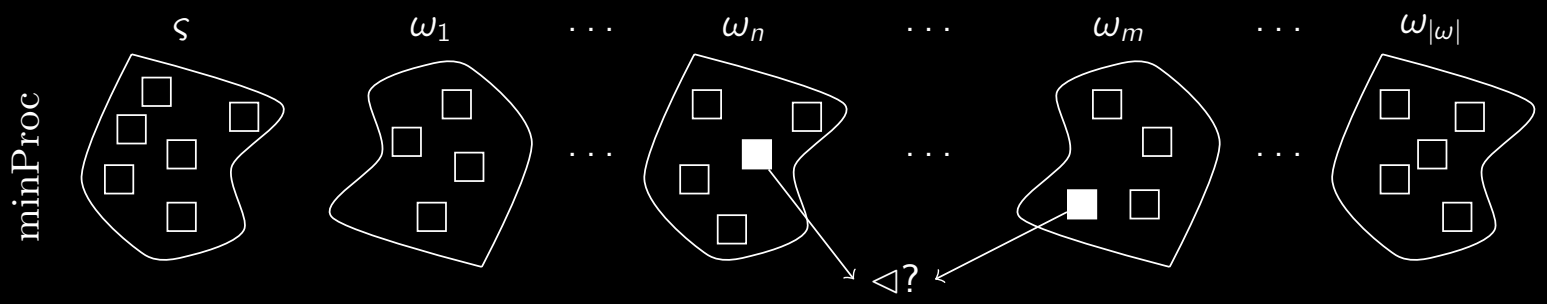
a.k.a. Process Reachability Problem

$\gamma_\varsigma(a_i \Rsh^* a_j :: \ldots :: z_k \Rsh^* z_l) \neq \emptyset$?

CTL: $EF(s[a] = a_j \wedge EF(\cdots \wedge EF(s'[z] = z_l) \ldots))$.

Over-approximation

- Un-ordered: $\gamma_\varsigma(\omega) \neq \emptyset \Longrightarrow \gamma_\varsigma(\omega_n) \neq \emptyset, \forall n \in \mathbb{I}^\omega$ (recursive reasoning with $\rho^\wedge : \mathbf{Obj} \mapsto \wp(\mathbf{OS})$).
- Ordered: $\gamma_\varsigma(a_i \Rsh^* a_j :: \omega) \neq \emptyset \Longrightarrow \gamma_{max\varsigma}(\omega) \neq \emptyset$.
- Process occurrences order constaints ($a_i \lhd a_j$):



Loïc Paulevé, Morgan Magnin, Olivier Roux

Objective Sequence Concretizability

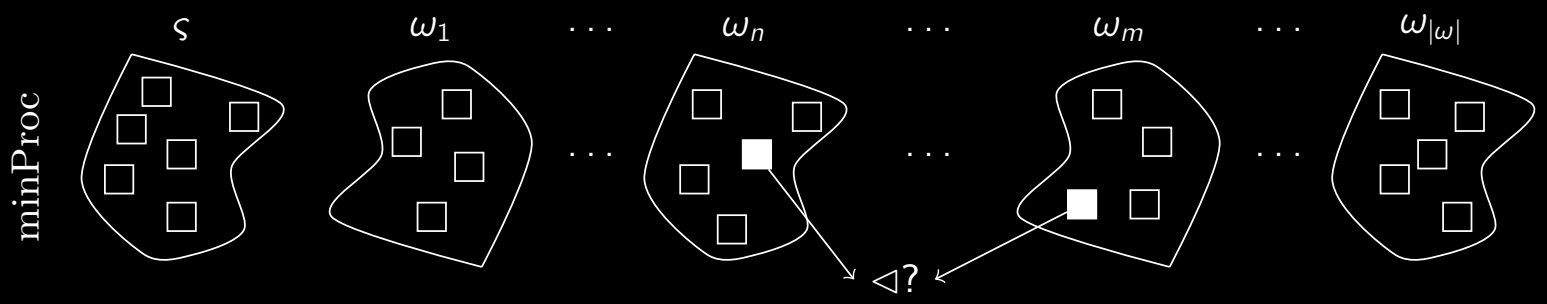
a.k.a. Process Reachability Problem

$\gamma_\varsigma(a_i \Rsh^* a_j :: \ldots :: z_k \Rsh^* z_l) \neq \emptyset$?

CTL: $EF(s[a] = a_j \wedge EF(\cdots \wedge EF(s'[z] = z_l) \ldots))$.

Over-approximation

- Un-ordered: $\gamma_\varsigma(\omega) \neq \emptyset \Longrightarrow \gamma_\varsigma(\omega_n) \neq \emptyset, \forall n \in \mathbb{I}^\omega$ (recursive reasoning with $\rho^\wedge : \mathbf{Obj} \mapsto \wp(\mathbf{OS})$).
- Ordered: $\gamma_\varsigma(a_i \Rsh^* a_j :: \omega) \neq \emptyset \Longrightarrow \gamma_{max\varsigma}(\omega) \neq \emptyset$.
- Process occurrences order constaints ($a_i \lhd a_j$):



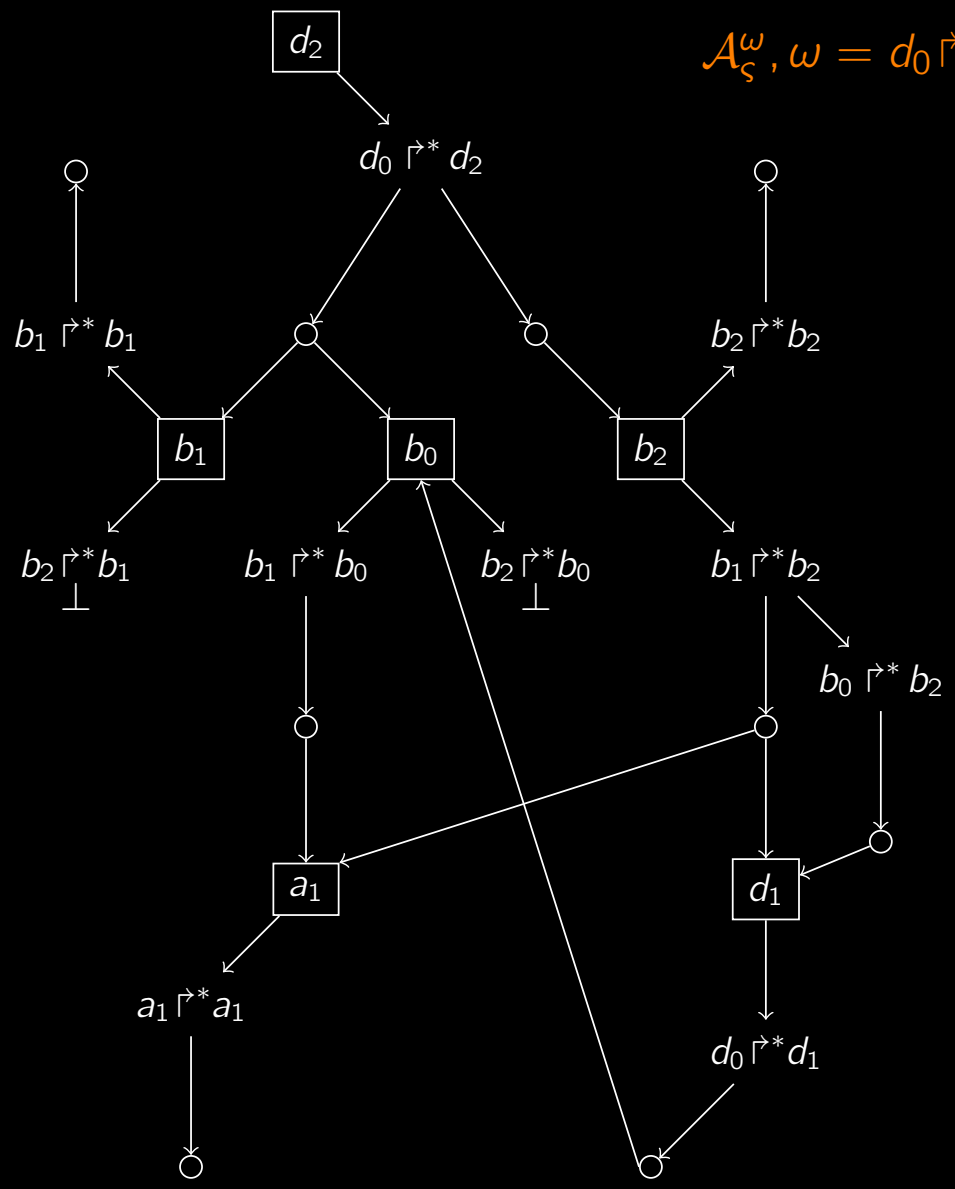
Under-approximation

- Sufficient condition: impose a particular structure of scenarios.

Loïc Paulevé, Morgan Magnin, Olivier Roux

Implementation of Over-approximations

$\mathcal{A}^{\omega}_{\varsigma}, \omega = d_0 \Rsh^* d_2, \varsigma = \langle a_1, b_1, b_2, c_1, d_0 \rangle$



Legend

Requirement

$a_j \longrightarrow a_i \Rsh^* a_j$

Solution

$(\{b_i, c_j\} \in \mathbf{BS}^{\wedge}(a_i \Rsh^* a_j))$

$a_i \Rsh^* a_j \longrightarrow \bigcirc \to b_i, c_j$

Continuity

$a_i \Rsh^* a_j \longrightarrow a_k \Rsh^* a_j$

Trivial solution

$a_i \Rsh^* a_j \longrightarrow \bigcirc$

No solution

$a_i \Rsh^* a_j$
$\perp$

Loïc Paulevé, Morgan Magnin, Olivier Roux

Complexities

Data structures

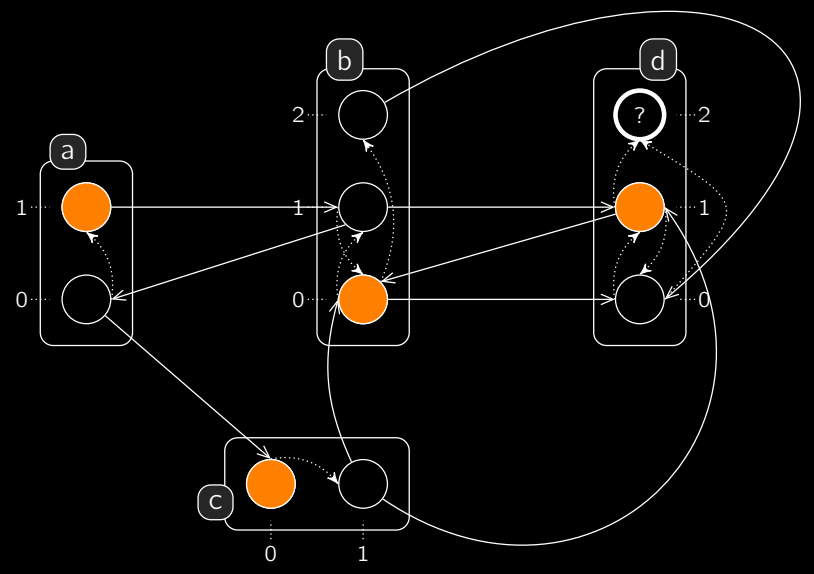
- Computing **BS** is exponential in the number of processes in a single sort.
- Computing $\mathbf{BS}^{\wedge}$ is faster, but still exponential.
- The size of $\mathcal{A}_{\varsigma}^{\omega}$ is $\approx$ polynomial in the number of processes.

Analyses

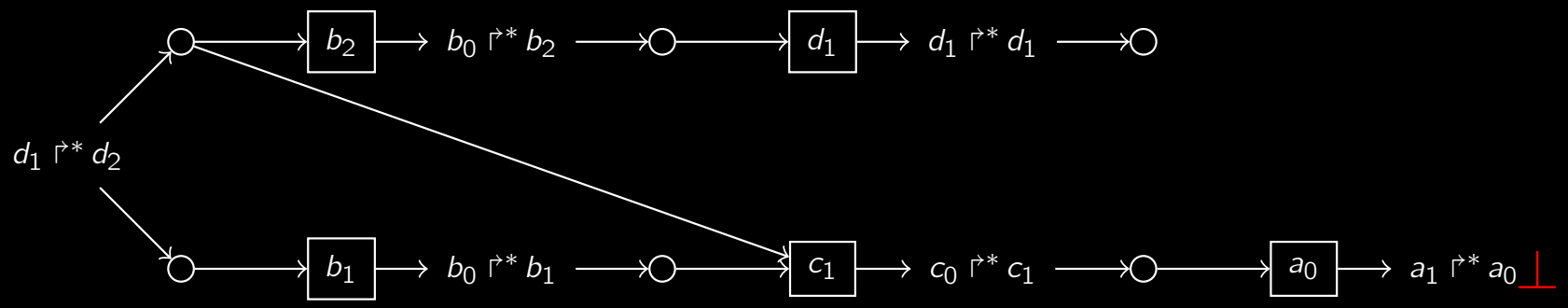
- Over-approximations are polynomial in the size of $\mathcal{A}_{\varsigma}^{\omega}$.
- Under-approximation is polynomial in the size of $\mathcal{A}_{\varsigma}^{\omega}$; and can be applied a number of times exponential in the number of solutions of a single objective.

$\Longrightarrow$ efficient with a small number of processes per sort; while a very large number of sorts can be handled.

Un-ordered Over-approximation

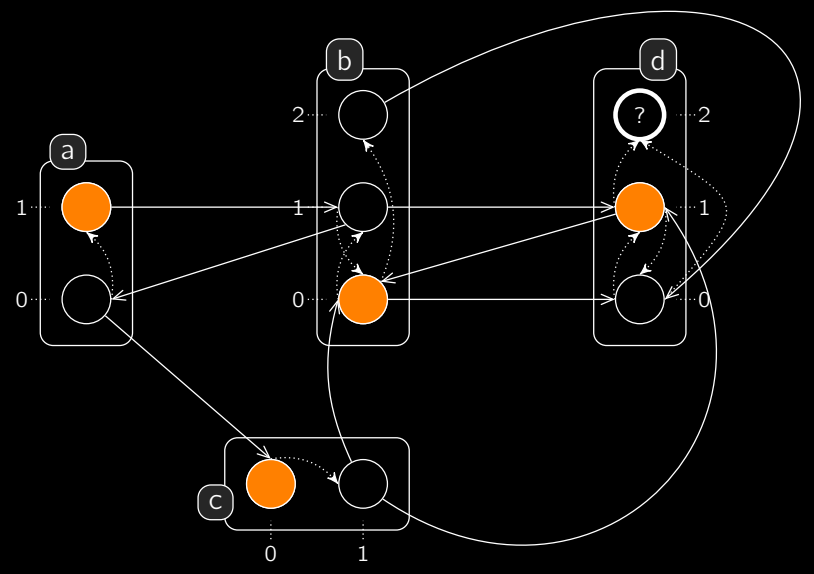


- $\gamma_\varsigma(\omega) \neq \emptyset \Longrightarrow \gamma_\varsigma(\omega_n) \neq \emptyset, \forall n \in \mathbb{I}^\omega$ (recursive reasoning with $\rho^\wedge : \mathbf{Obj} \mapsto \wp(\mathbf{OS})$).
- Necessary condition: there always exists a solution ending with a trivial objective.

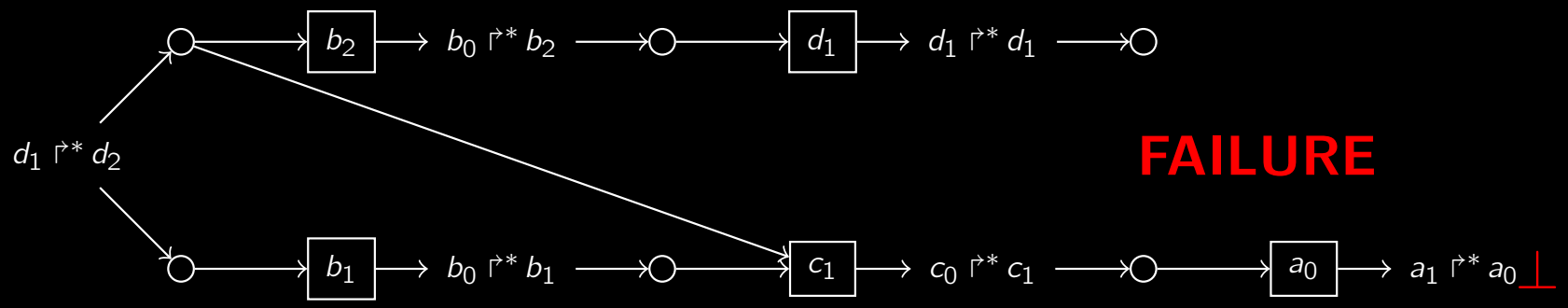


Loïc Paulevé, Morgan Magnin, Olivier Roux

Un-ordered Over-approximation

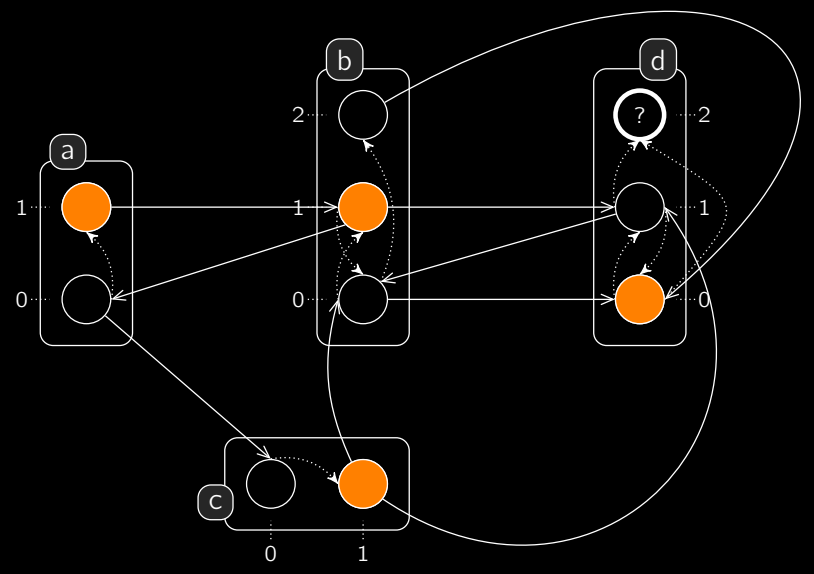


- $\gamma_\varsigma(\omega) \neq \emptyset \Longrightarrow \gamma_\varsigma(\omega_n) \neq \emptyset, \forall n \in \mathbb{I}^\omega$ (recursive reasoning with $\rho^\wedge : \mathbf{Obj} \mapsto \wp(\mathbf{OS})$).
- Necessary condition: there always exists a solution ending with a trivial objective.

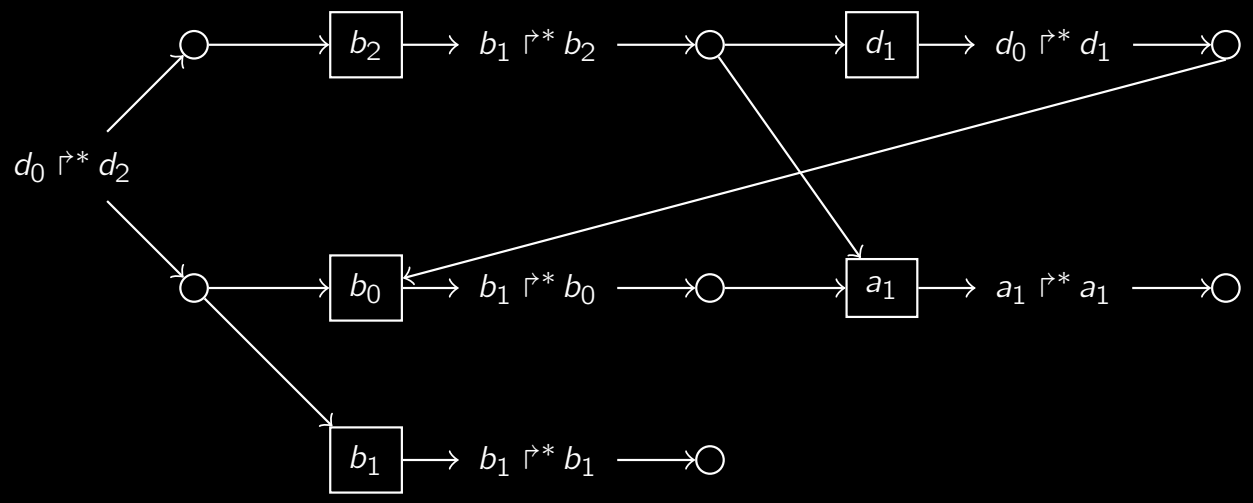


Loïc Paulevé, Morgan Magnin, Olivier Roux

Un-ordered Over-approximation



- $\gamma_\varsigma(\omega) \neq \emptyset \Longrightarrow \gamma_\varsigma(\omega_n) \neq \emptyset, \forall n \in \mathbb{I}^\omega$ (recursive reasoning with $\rho^\wedge : \mathbf{Obj} \mapsto \wp(\mathbf{OS})$).
- Necessary condition: there always exists a solution ending with a trivial objective.



OK

Ordered Over-approximation

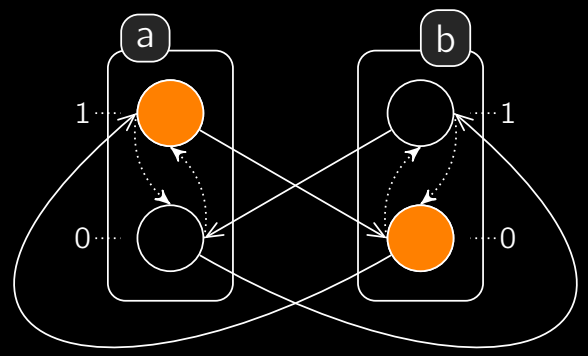
Goal: $\gamma_\varsigma(a_1 \Rsh^* a_0 :: \omega) \neq \emptyset \Longrightarrow \gamma_{max\varsigma}(\omega) \neq \emptyset$

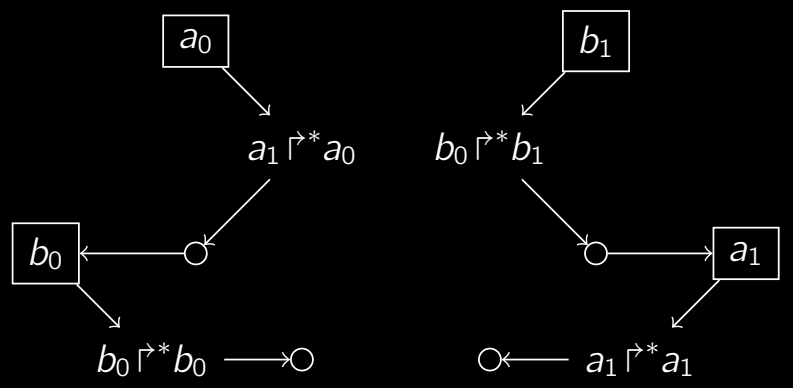
- If $a_0 \notin \varsigma$, $\exists c_i \rightarrow a_? \Rsh a_0$, hence $max\varsigma[a] = \{a_0\}$ and $max\varsigma[c] = \{c_i\}$:
$$\delta_{1..n} :: c_i \rightarrow a_? \Rsh a_0 :: \delta_{m..|\delta|} \text{ — } \delta_{m..|\delta|} \in \gamma_{\varsigma'}(\omega) \ .$$
- For all other sorts, use the saturated context of $\lceil \mathcal{A}^\omega_\varsigma \rceil$:
$$\lceil \mathcal{A}^\omega_\varsigma \rceil = \mathrm{lfp}\ \left(\mathcal{A}^\omega_\varsigma \mapsto \mathcal{A}^\omega_{\varsigma \Cap \mathrm{procs}(\mathcal{A}^\omega_\varsigma)} \right) \ .$$
- Un-ordered over-approximation should be satisfied.

Ordered Over-approximation

Goal: $\gamma_\varsigma(a_1 \lceil^* a_0 :: \omega) \neq \emptyset \Longrightarrow \gamma_{max\varsigma}(\omega) \neq \emptyset$

- If $a_0 \notin \varsigma$, $\exists c_i \rightarrow a_? \lceil a_0$, hence $max\varsigma[a] = \{a_0\}$ and $max\varsigma[c] = \{c_i\}$:
$$\delta_{1..n} :: c_i \rightarrow a_? \lceil a_0 :: \delta_{m..|\delta|} \text{ — } \delta_{m..|\delta|} \in \gamma_{\varsigma'}(\omega) \ .$$
- For all other sorts, use the saturated context of $\lceil \mathcal{A}^\omega_\varsigma \rceil$:
$$\lceil \mathcal{A}^\omega_\varsigma \rceil = \mathrm{lfp}\ \left(\mathcal{A}^\omega_\varsigma \mapsto \mathcal{A}^\omega_{\varsigma \Cap \mathrm{procs}(\mathcal{A}^\omega_\varsigma)} \right) \ .$$
- Un-ordered over-approximation should be satisfied.



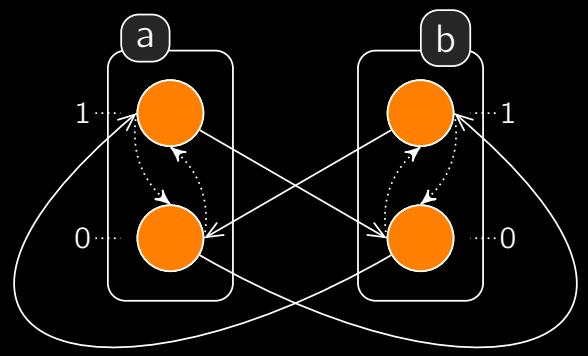


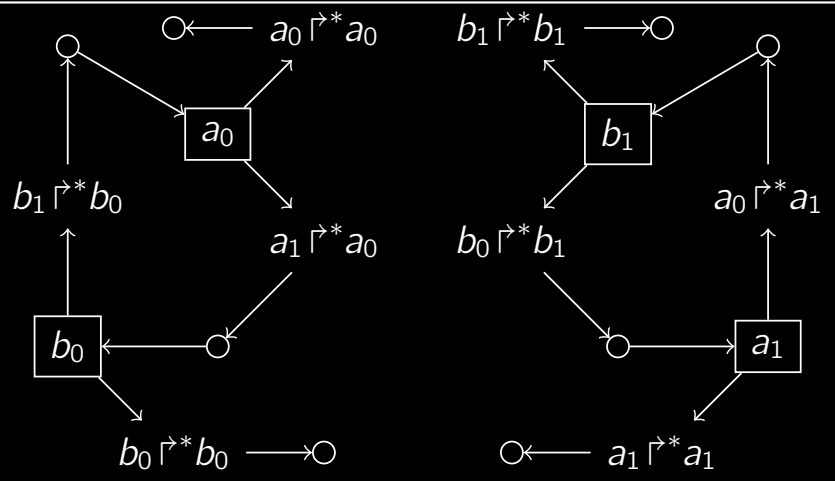
$\gamma_{\langle a_1, b_0 \rangle}(a_1 \lceil^* a_0 :: b_0 \lceil^* b_1) \neq \emptyset$

Ordered Over-approximation

Goal: $\gamma_\varsigma(a_1 \Rsh^* a_0 :: \omega) \neq \emptyset \implies \gamma_{max\varsigma}(\omega) \neq \emptyset$

- If $a_0 \notin \varsigma$, $\exists c_i \rightarrow a_? \Rsh a_0$, hence $max\varsigma[a] = \{a_0\}$ and $max\varsigma[c] = \{c_i\}$:
$$\delta_{1..n} :: c_i \rightarrow a_? \Rsh a_0 :: \delta_{m..|\delta|} \text{ — } \delta_{m..|\delta|} \in \gamma_{\varsigma'}(\omega) \ .$$
- For all other sorts, use the saturated context of $\lceil \mathcal{A}^\omega_\varsigma \rceil$:
$$\lceil \mathcal{A}^\omega_\varsigma \rceil = \mathrm{lfp}\ \left(\mathcal{A}^\omega_\varsigma \mapsto \mathcal{A}^\omega_{\varsigma \Cap \mathrm{procs}(\mathcal{A}^\omega_\varsigma)} \right) \ .$$
- Un-ordered over-approximation should be satisfied.



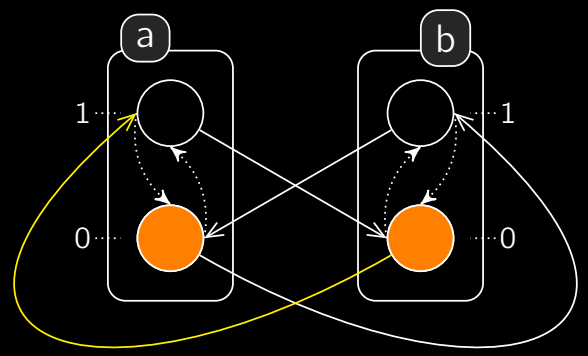


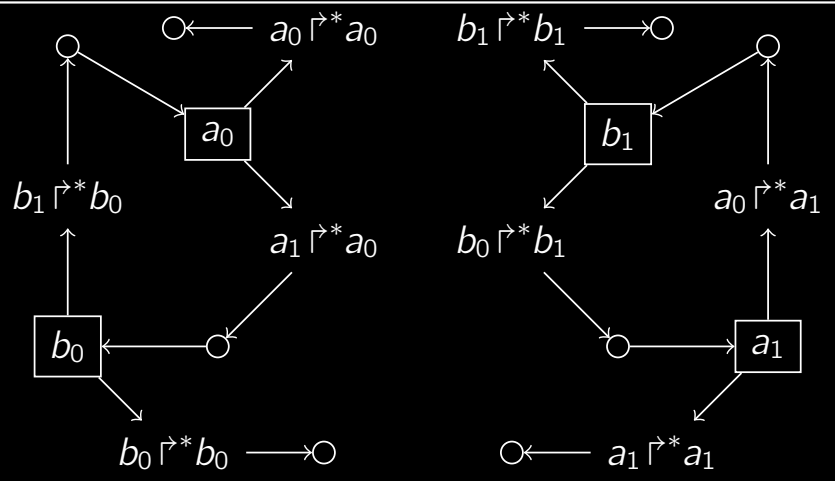
$\gamma_{\langle a_1, b_0 \rangle}(a_1 \Rsh^* a_0 :: b_0 \Rsh^* b_1) \neq \emptyset$

Ordered Over-approximation

Goal: $\gamma_\varsigma(a_1 \Rsh^* a_0 :: \omega) \neq \emptyset \implies \gamma_{max\varsigma}(\omega) \neq \emptyset$

- If $a_0 \notin \varsigma$, $\exists c_i \rightarrow a_? \Rsh a_0$, hence $max\varsigma[a] = \{a_0\}$ and $max\varsigma[c] = \{c_i\}$:
$$\delta_{1..n} :: c_i \rightarrow a_? \Rsh a_0 :: \delta_{m..|\delta|} \text{ — } \delta_{m..|\delta|} \in \gamma_{\varsigma'}(\omega) \ .$$
- For all other sorts, use the saturated context of $\lceil \mathcal{A}^\omega_\varsigma \rceil$:
$$\lceil \mathcal{A}^\omega_\varsigma \rceil = \mathrm{lfp}\ \left(\mathcal{A}^\omega_\varsigma \mapsto \mathcal{A}^\omega_{\varsigma \Cap \mathrm{procs}(\mathcal{A}^\omega_\varsigma)} \right) \ .$$
- Un-ordered over-approximation should be satisfied.





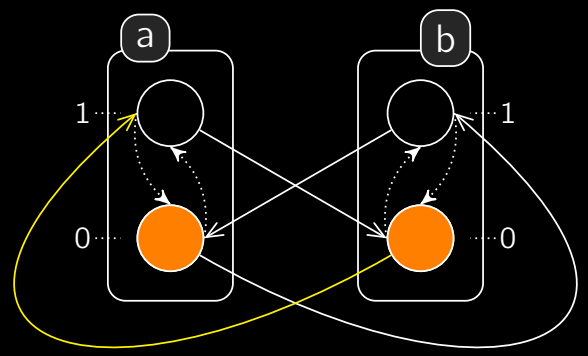
$$\gamma_{\langle a_1, b_0 \rangle}(a_1 \Rsh^* a_0 :: b_0 \Rsh^* b_1) \neq \emptyset \implies \gamma_{\langle a_0, b_0 \rangle}(b_0 \Rsh^* b_1) \neq \emptyset$$

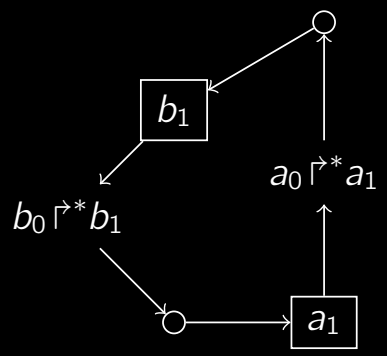
Loïc Paulevé, Morgan Magnin, Olivier Roux

Ordered Over-approximation

Goal: $\gamma_\varsigma(a_1 \Rsh^* a_0 :: \omega) \neq \emptyset \implies \gamma_{max\varsigma}(\omega) \neq \emptyset$

- If $a_0 \notin \varsigma$, $\exists c_i \rightarrow a_? \Rsh a_0$, hence $max\varsigma[a] = \{a_0\}$ and $max\varsigma[c] = \{c_i\}$:
$$\delta_{1..n} :: c_i \rightarrow a_? \Rsh a_0 :: \delta_{m..|\delta|} \text{ — } \delta_{m..|\delta|} \in \gamma_{\varsigma'}(\omega) \ .$$
- For all other sorts, use the saturated context of $\lceil \mathcal{A}^\omega_\varsigma \rceil$:
$$\lceil \mathcal{A}^\omega_\varsigma \rceil = \mathrm{lfp}\ \left(\mathcal{A}^\omega_\varsigma \mapsto \mathcal{A}^\omega_{\varsigma \Cap \mathrm{procs}(\mathcal{A}^\omega_\varsigma)} \right) \ .$$
- Un-ordered over-approximation should be satisfied.





$\gamma_{\langle a_1, b_0 \rangle}(a_1 \Rsh^* a_0 :: b_0 \Rsh^* b_1) \neq \emptyset \implies \gamma_{\langle a_0, b_0 \rangle}(b_0 \Rsh^* b_1) \neq \emptyset$ **FAILURE**

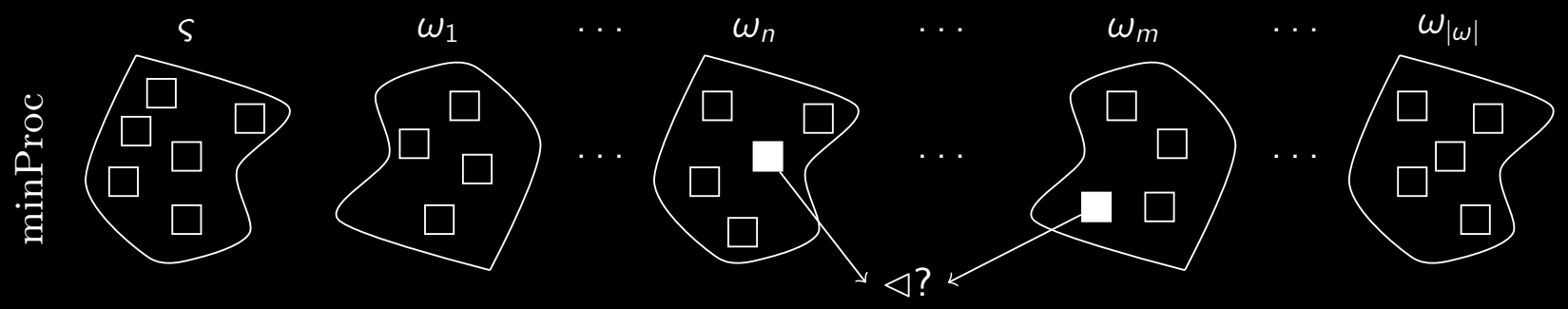
Process Occurrences Order Constraints

$a_j \lhd a_i \iff$ no scenario can be abstracted by $a_i \lceil^{*} a_j$.

Uncovering Order Constraints

$\mathbf{BS}(a_i \lceil^{*} a_j) = \emptyset \implies a_j \lhd a_i$

Idea of Over-Approximation



Based on the ordered over-approximation ($\lceil \mathcal{A}^{\omega}_{\varsigma} \rceil$):

- $\mathrm{minProc}_{\varsigma}(\omega_n) = \{p \in \mathbf{Proc} \mid p$ occurs in all solutions of $\omega_n\}$;
- $\begin{cases} \{a_i \in \varsigma\} & \text{if } n = 0 \\ \mathrm{minProc}_{max\varsigma}(\omega_n) & \text{otherwise,} \\ & \text{with } max\varsigma = \mathrm{maxCtx}(\varsigma, w, n-1) \end{cases}$.

Under-approximation

Idea: force a particular shape for scenarios.

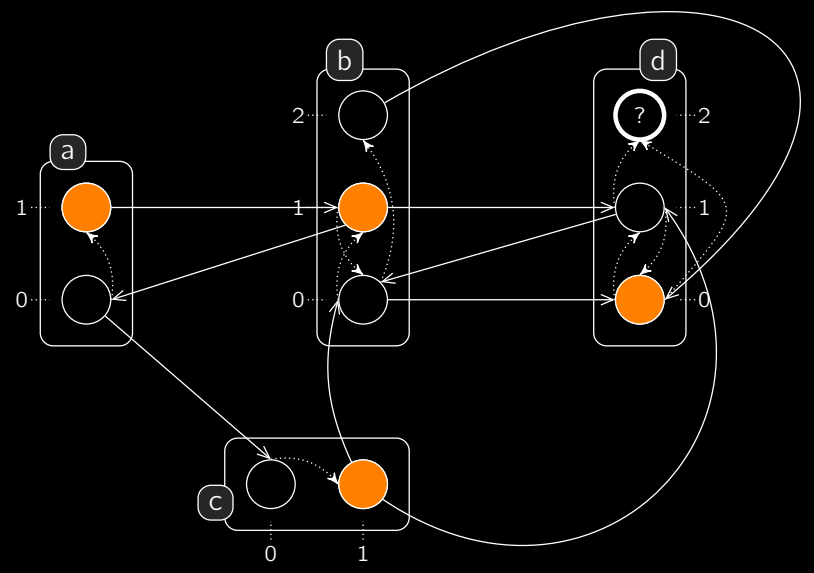
- "top-down" resolution: ignore interleaving requirements:
- if $a_i \Rsh^* a_j$ needs b_i and c_i, resolve $\star \Rsh^* b_i$, then $\star \Rsh^* c_i$ (and conversely), etc.

Implementation

- Abstract structure $\lceil \mathcal{B}_\varsigma^\omega \rceil$ very similar to $\lceil \mathcal{A}_\varsigma^\omega \rceil$, *except that*:
- some solutions can be ignored;
- (maximal continuity, instead of minimal).

Sufficient condition

- $\lceil \mathcal{B}_\varsigma^\omega \rceil$ has no cycle; and
- all referenced objectives have at least one solution.

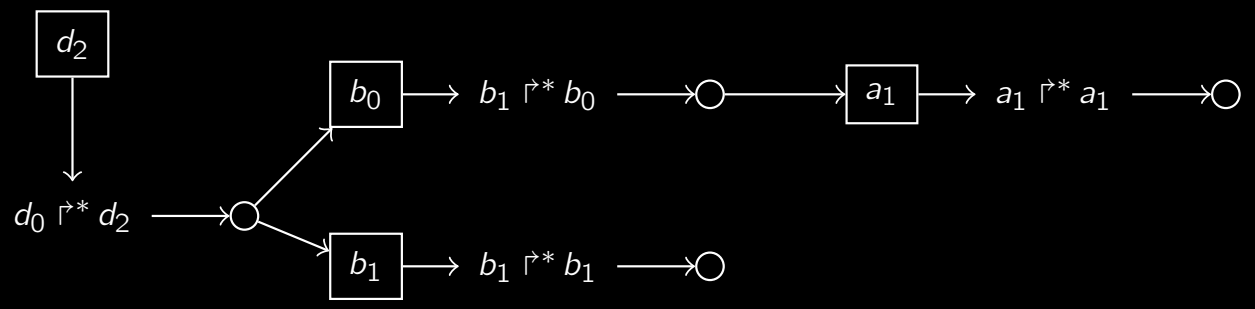


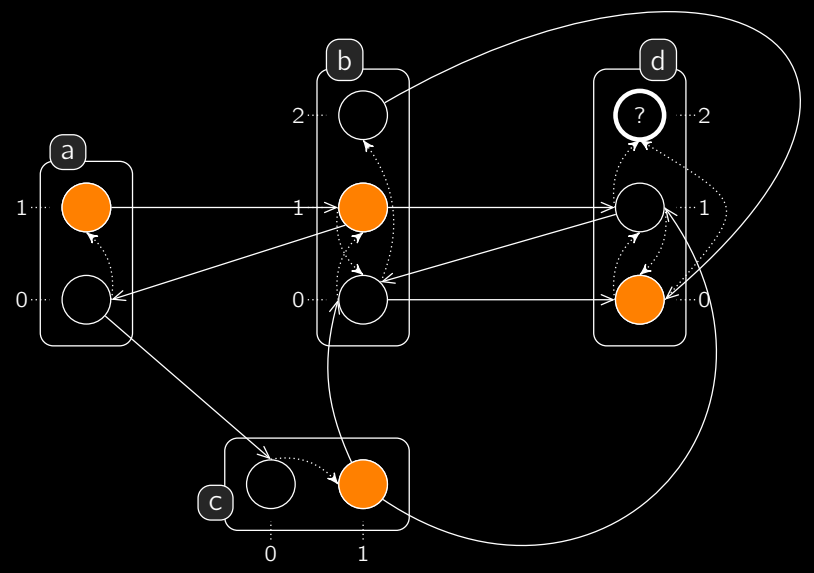
Under-approximation

Sufficient condition:

- $\lceil \mathcal{B}_\varsigma^\omega \rceil$ has no cycle; and
- all referenced objectives have at least one solution.

Remark: may be more conclusive on refined ω.



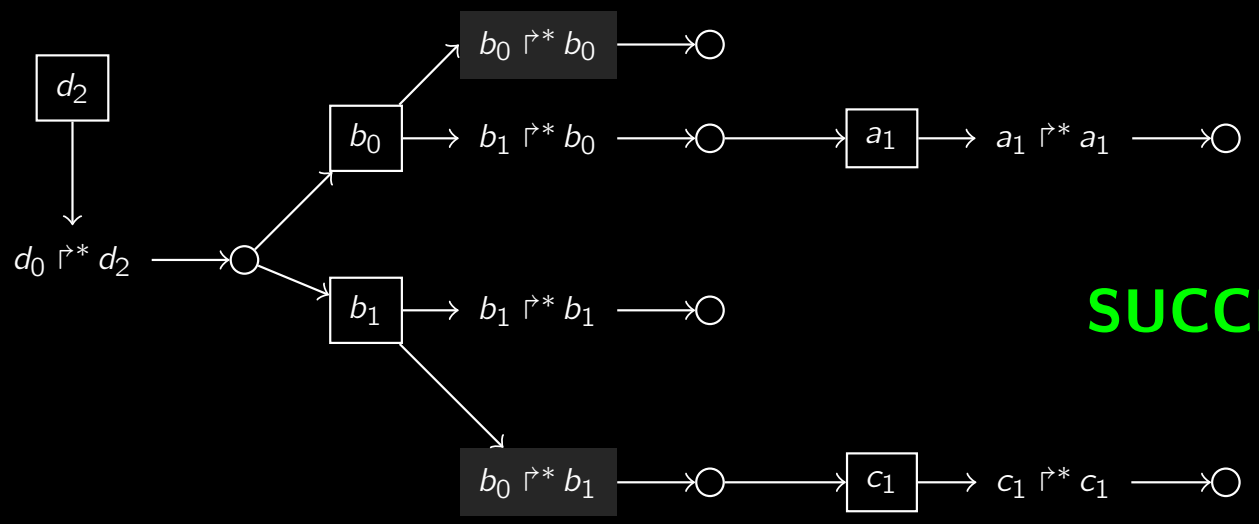


Under-approximation

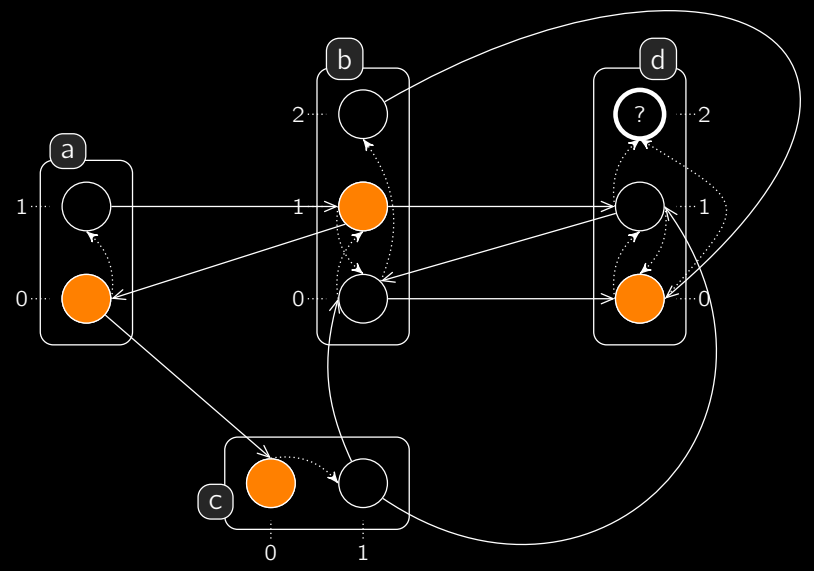
Sufficient condition:

- $\lceil \mathcal{B}_\varsigma^\omega \rceil$ has no cycle; and
- all referenced objectives have at least one solution.

Remark: may be more conclusive on refined ω.



SUCCESS

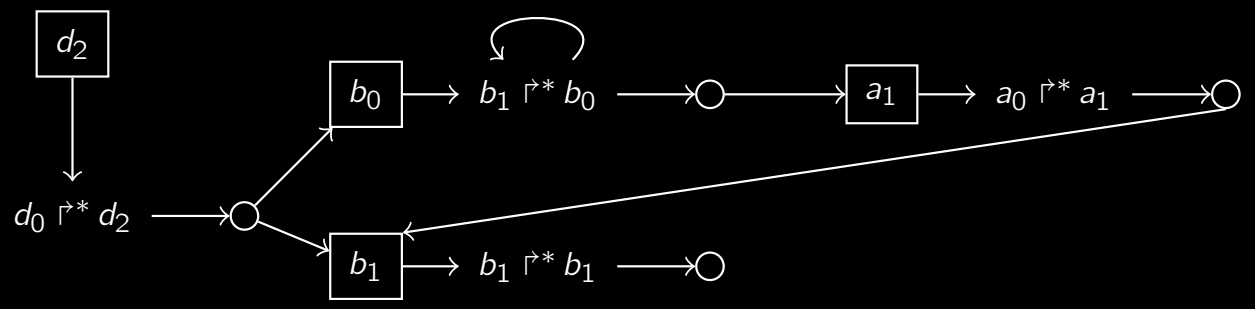


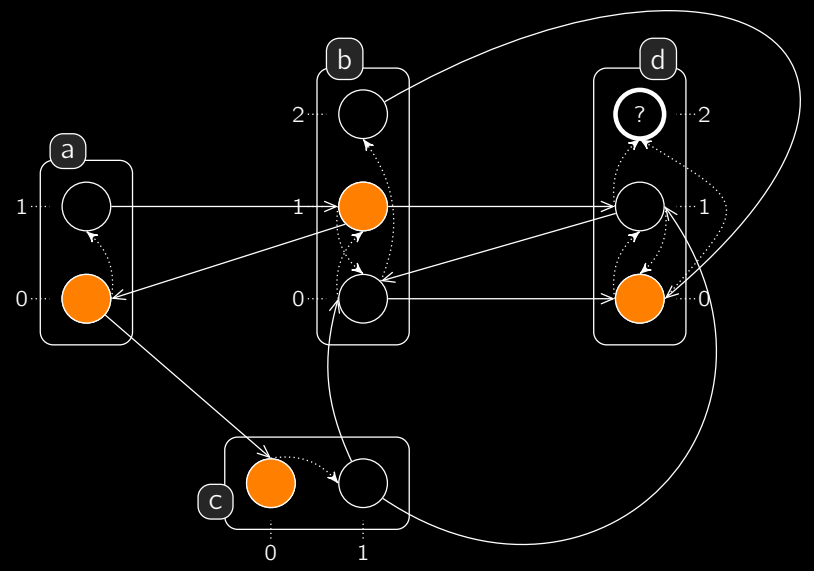
Under-approximation

Sufficient condition:

- $\lceil \mathcal{B}_\varsigma^\omega \rceil$ has no cycle; and
- all referenced objectives have at least one solution.

Remark: may be more conclusive on refined ω.



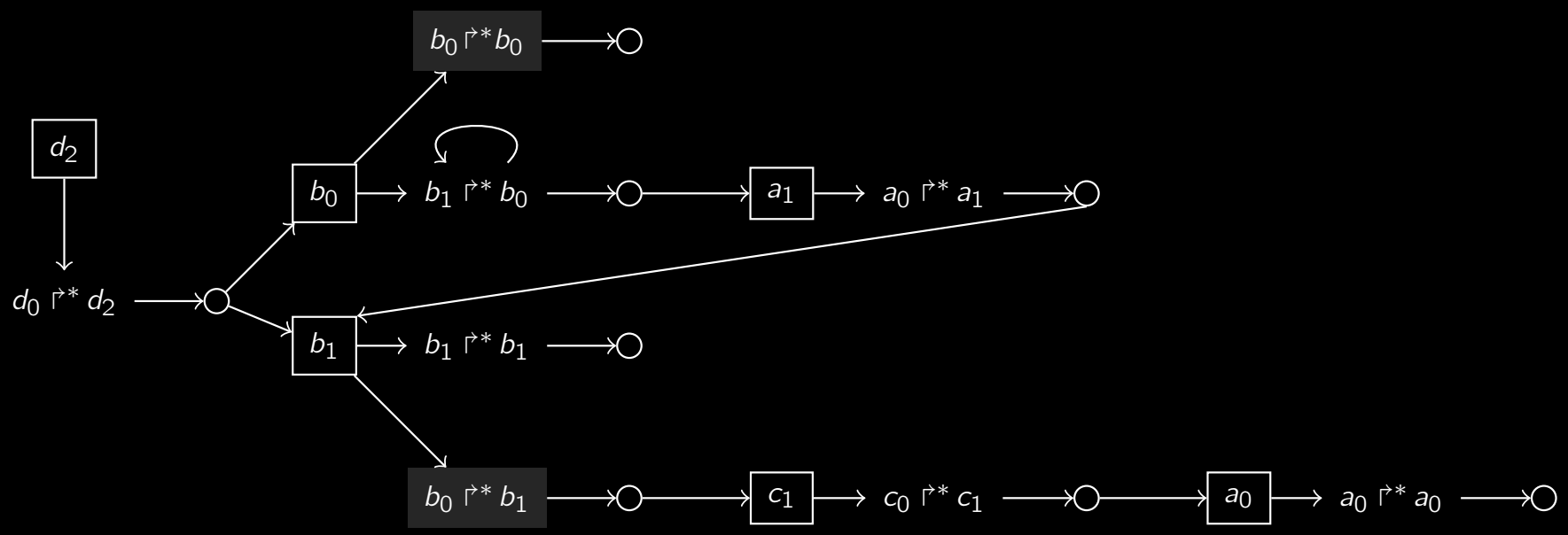


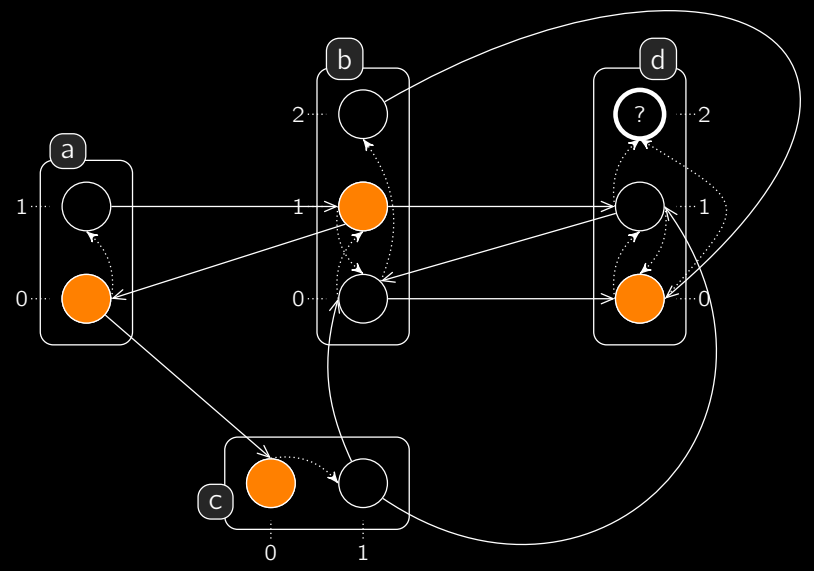
Under-approximation

Sufficient condition:

- $\lceil \mathcal{B}_\varsigma^\omega \rceil$ has no cycle; and
- all referenced objectives have at least one solution.

Remark: may be more conclusive on refined ω.



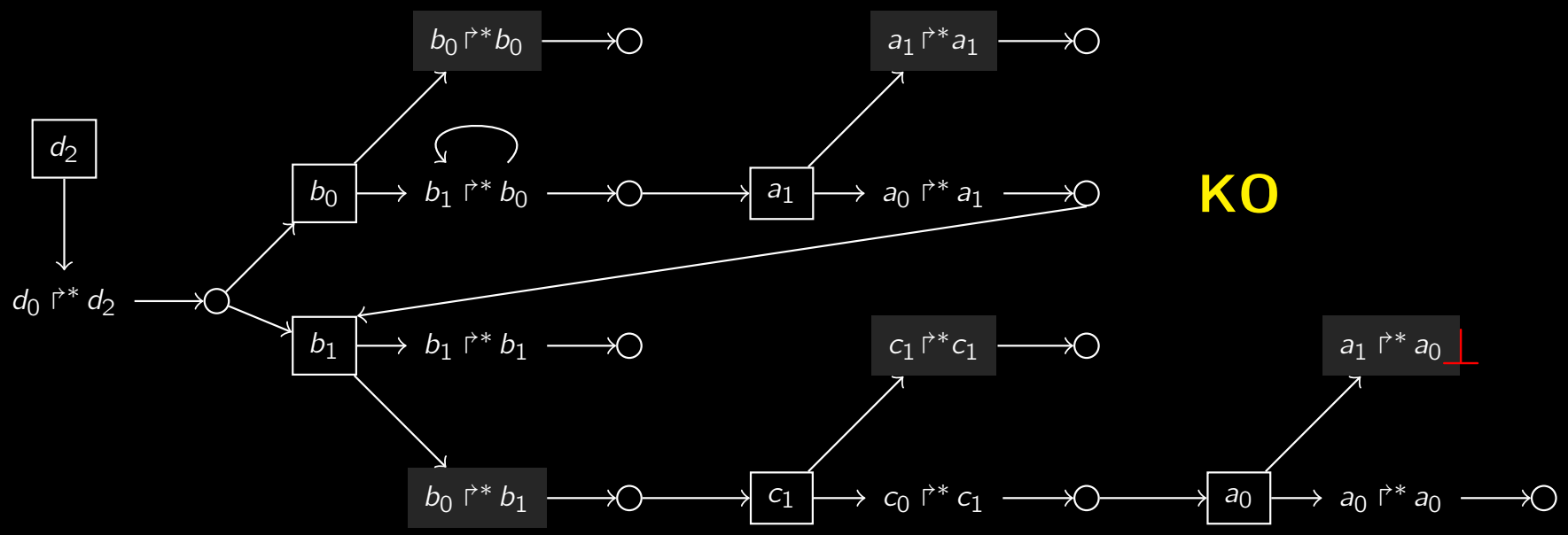


Under-approximation

Sufficient condition:

- $\lceil \mathcal{B}_\varsigma^\omega \rceil$ has no cycle; and
- all referenced objectives have at least one solution.

Remark: may be more conclusive on refined ω.



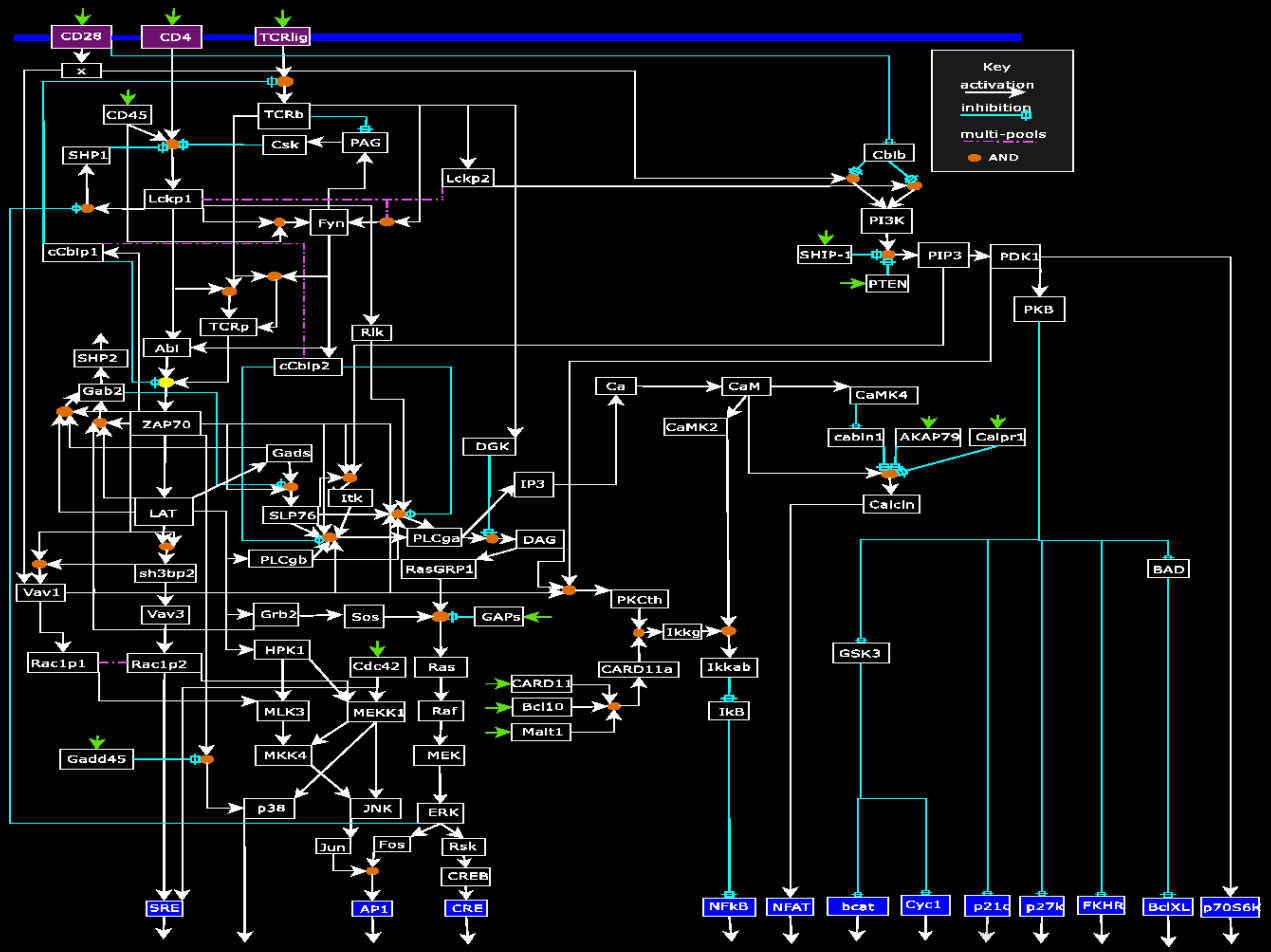
Loïc Paulevé, Morgan Magnin, Olivier Roux

Outline

1. Introduction to BRNs
2. The Process Hitting
3. Stochastic and Time Parameters
4. Scalable Analysis of Dynamics
   - Dynamics Properties from BRN Interaction Graph
   - Fix Points
5. Abstract Interpretation of Scenarios
6. Applications
7. Discussion

T-Cell Receptor Signalling Pathway

(94 components)



[Saez-Rodriguez, *et al.* in PLoS Comput Biol, 07]

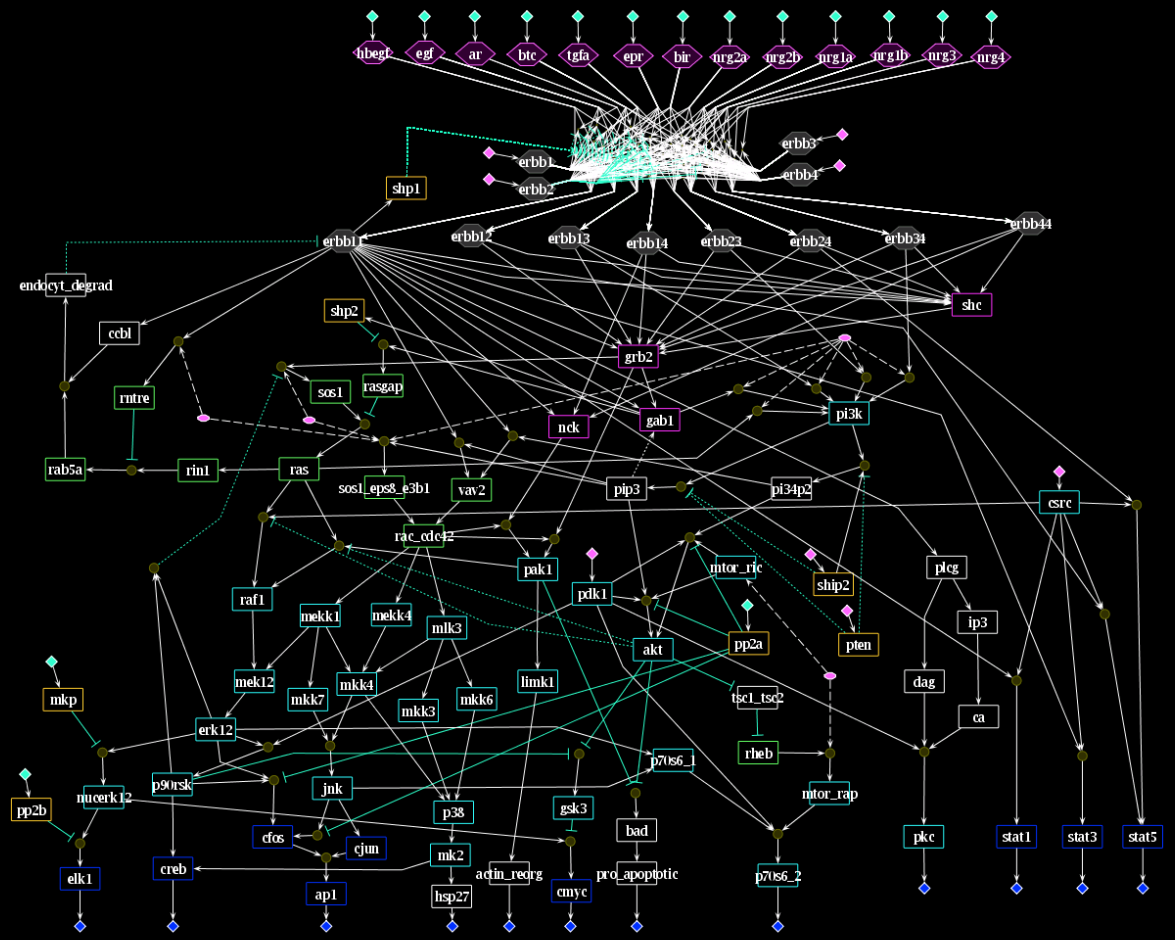
Process Hitting
133 sorts,
448 processes,
1124 actions:
$\approx 2 \cdot 10^{58}$ states.

Reachability analysis always conclusive; around 0.01*s* (compared to *libddd*: out of memory). [http://ddd.lip6.fr]

Loïc Paulevé, Morgan Magnin, Olivier Roux

EGFR/ErbB Signalling

(104 components)



[Samaga, *et al.* in PLoS Comput Biol, 2009]

Process Hitting
193 sorts,
748 processes,
2356 actions:
$\approx 2 \cdot 10^{96}$ states.

Reachability analysis always conclusive; around 0.05*s* (compared to *libddd*: out of memory). [http://ddd.lip6.fr]

Loïc Paulevé, Morgan Magnin, Olivier Roux

Outline

1. Introduction to BRNs
2. The Process Hitting
3. Stochastic and Time Parameters
4. Scalable Analysis of Dynamics
   - Dynamics Properties from BRN Interaction Graph
   - Fix Points
5. Abstract Interpretation of Scenarios
6. Applications
7. Discussion

Conclusion and Outlook

The Process Hitting

- *Elementary* framework for dynamical complex systems;
- Applied to BRNs; not limited to.
- Generic tuning of time features within stochastic models (simulation + standard model checking).
- Software available (PINT - `http://processhitting.wordpress.com`).

Static Analysis by Abstract Interpretation of Process Hitting

- Very efficient over- and under-approximations of process reachability;
- Extract necessary processes for achieving reachabilities: towards control.
- Future work may establish other dynamical properties: attractors, etc.

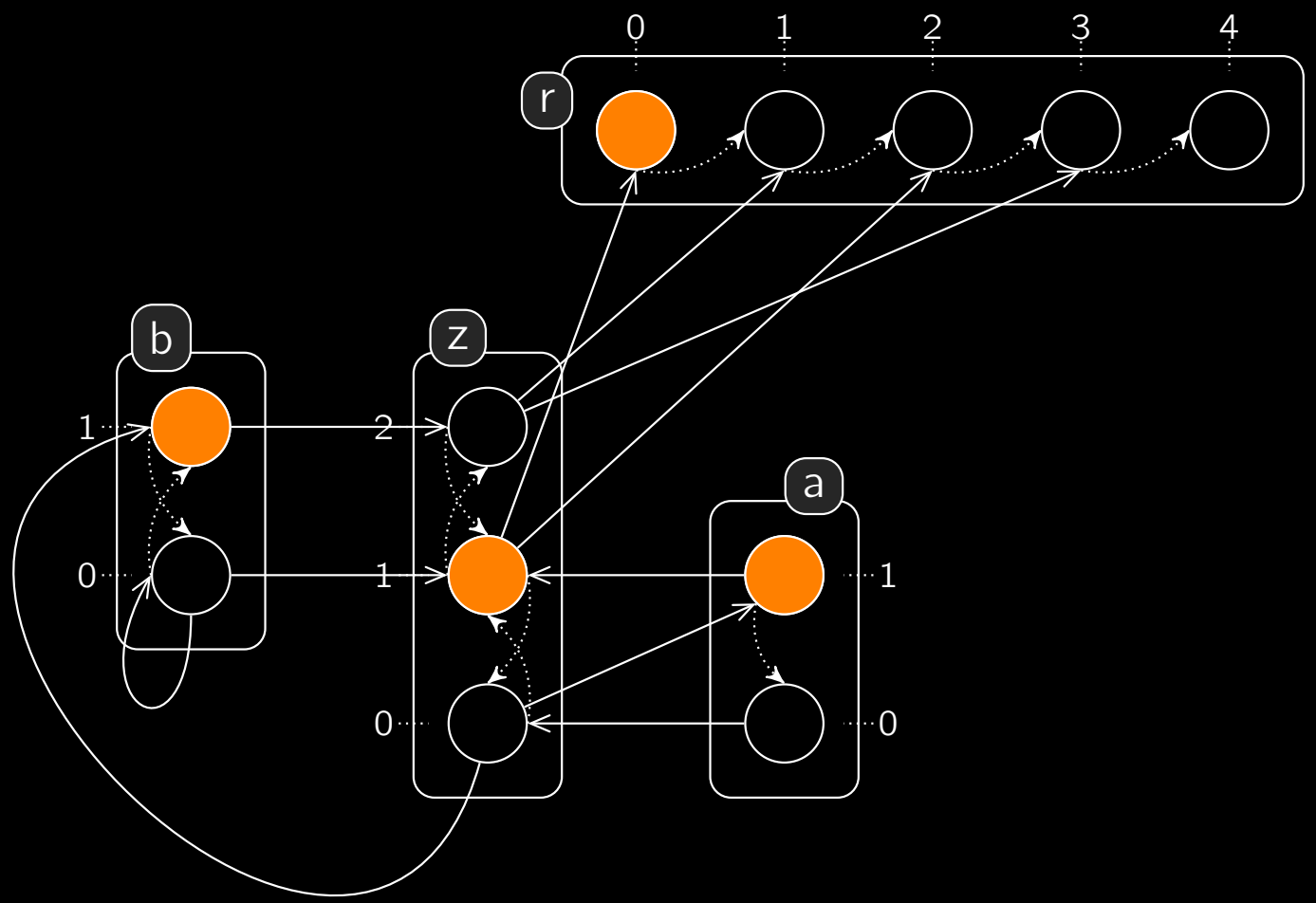
Towards Quantitative Analysis

- Extension to the Process Hitting with Priorities.
- Two classes of actions: instantaneous and non-instantaneous.

Bonus

...

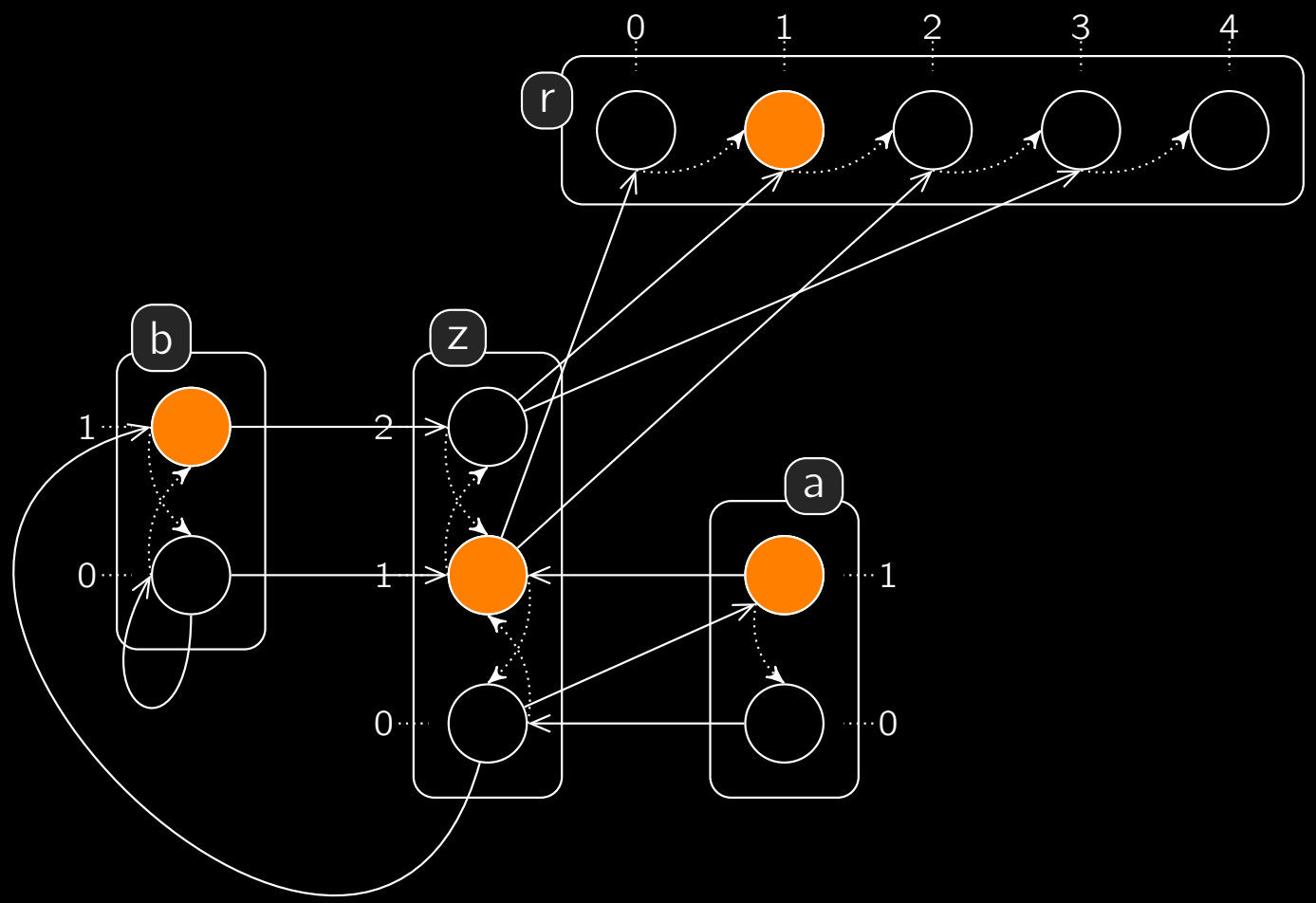
Toy Example



$\mathbf{BS}(r_0 \lceil^{*} r_4) = \{[z_1 \rightarrow r_0 \lceil r_1, z_2 \rightarrow r_1 \lceil r_2, z_1 \rightarrow r_2 \lceil r_3, z_2 \rightarrow r_3 \lceil r_4]\}$

Objective sequence to check: $z_1 \lceil^{*} z_2, z_2 \lceil^{*} z_1, z_1 \lceil^{*} z_2$.

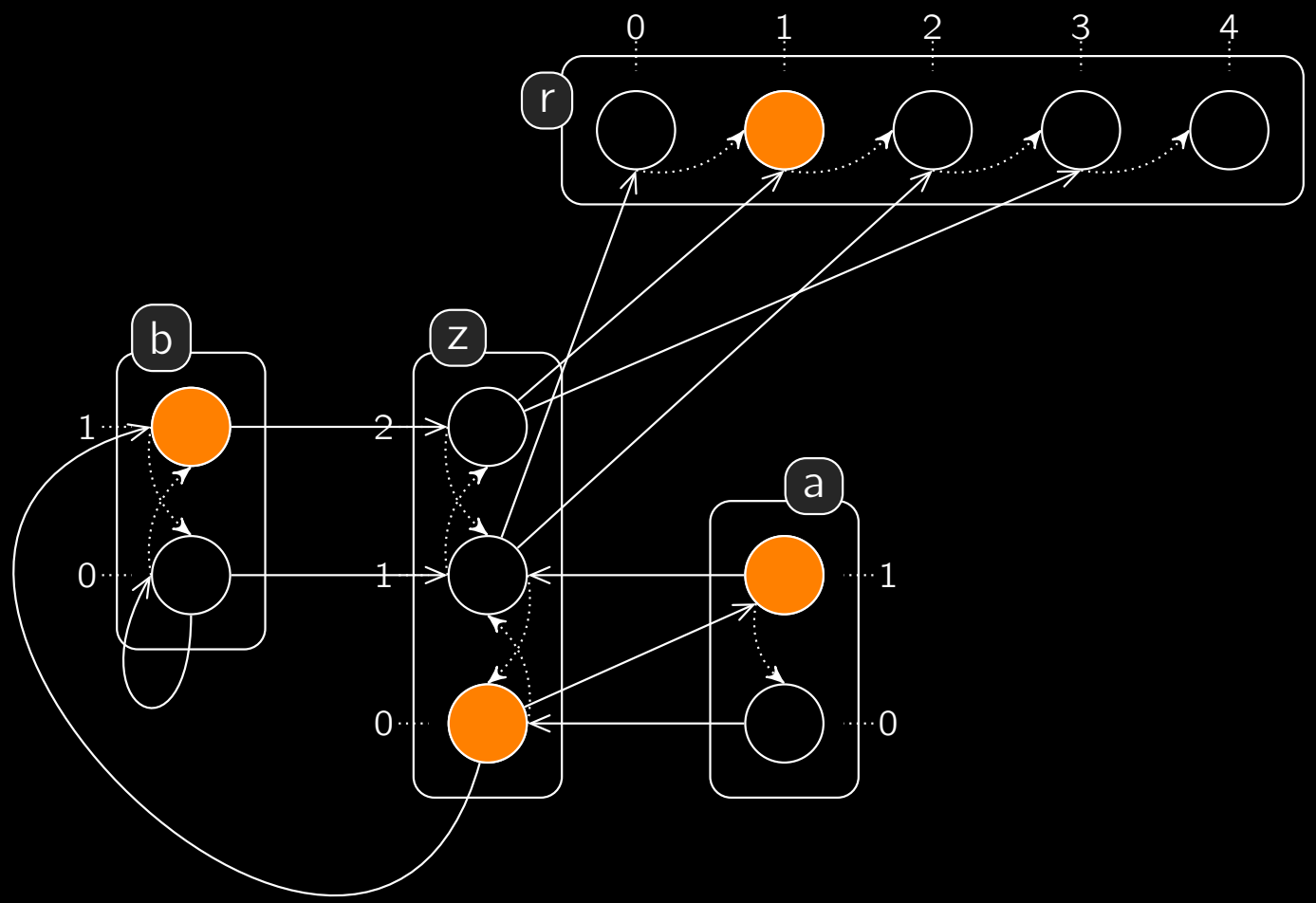
Toy Example



$\mathbf{BS}(r_0 \lceil^* r_4) = \{[z_1 \rightarrow r_0 \lceil r_1, z_2 \rightarrow r_1 \lceil r_2, z_1 \rightarrow r_2 \lceil r_3, z_2 \rightarrow r_3 \lceil r_4]\}$

Objective sequence to check: $z_1 \lceil^* z_2, z_2 \lceil^* z_1, z_1 \lceil^* z_2$.

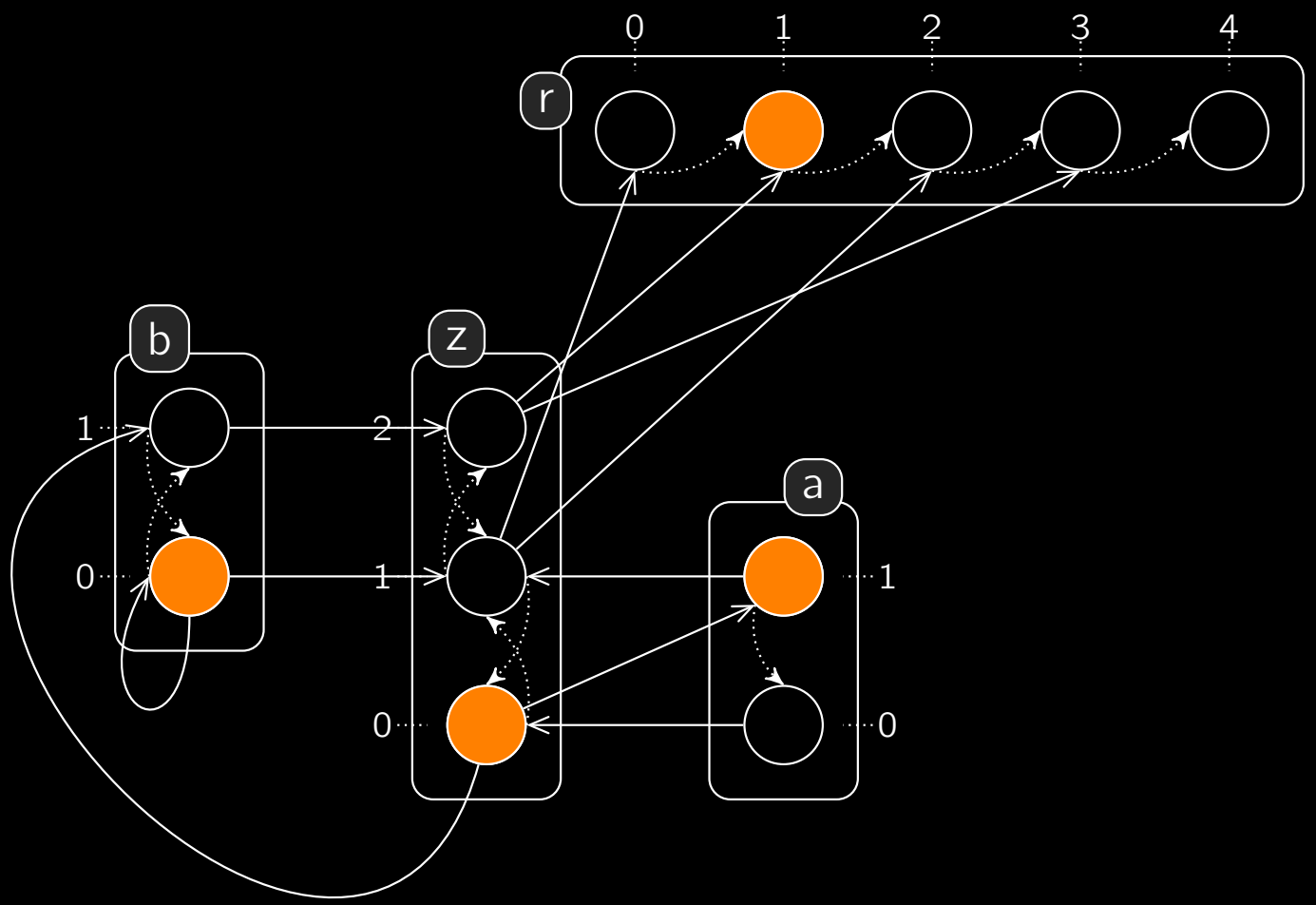
Toy Example



$\mathbf{BS}(r_0 \Rsh^* r_4) = \{[z_1 \rightarrow r_0 \Rsh r_1, z_2 \rightarrow r_1 \Rsh r_2, z_1 \rightarrow r_2 \Rsh r_3, z_2 \rightarrow r_3 \Rsh r_4]\}$

Objective sequence to check: $z_1 \Rsh^* z_2, z_2 \Rsh^* z_1, z_1 \Rsh^* z_2$.

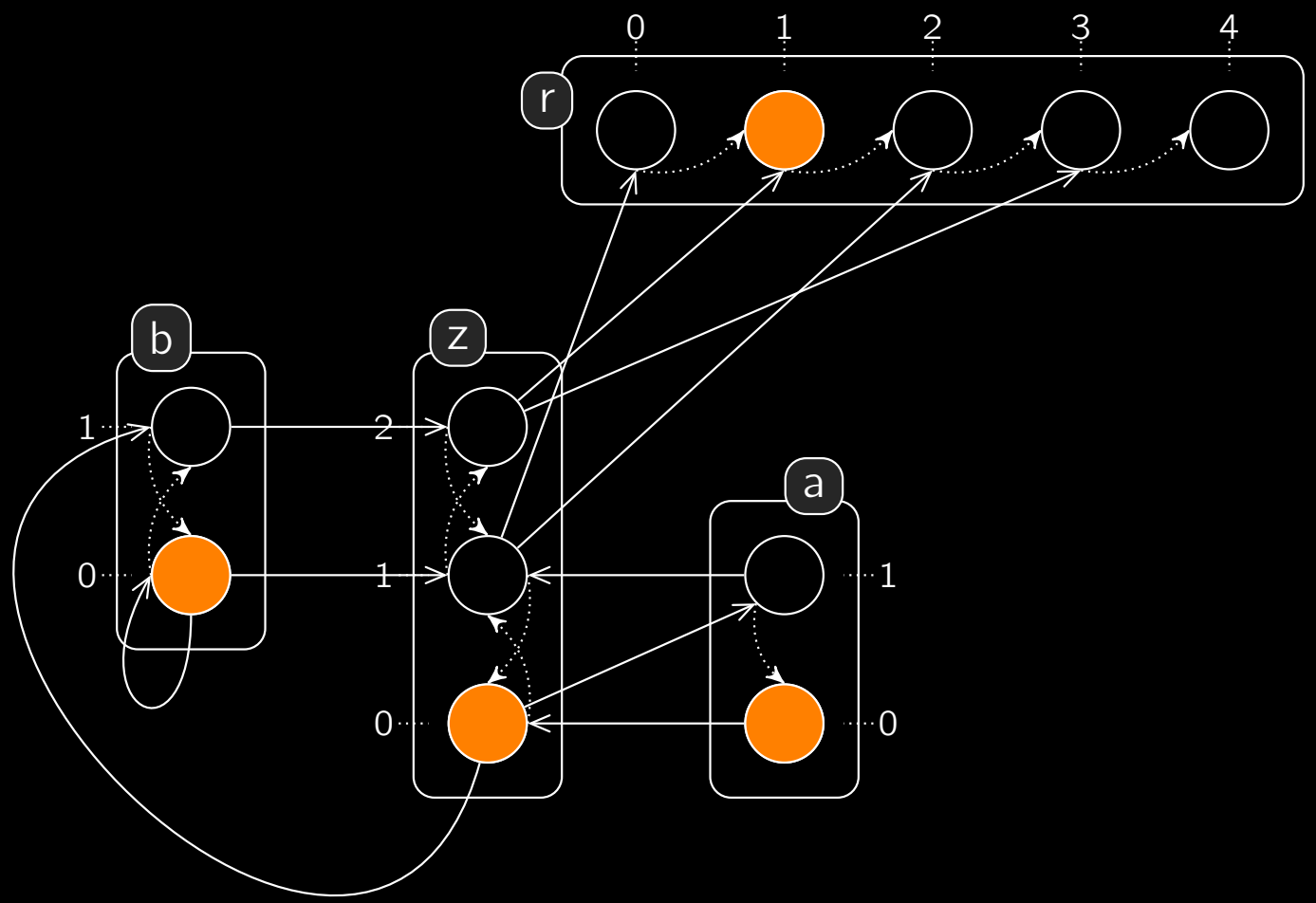
Toy Example



$\mathbf{BS}(r_0 \Rsh^* r_4) = \{[z_1 \rightarrow r_0 \Rsh r_1, z_2 \rightarrow r_1 \Rsh r_2, z_1 \rightarrow r_2 \Rsh r_3, z_2 \rightarrow r_3 \Rsh r_4]\}$

Objective sequence to check: $z_1 \Rsh^* z_2, z_2 \Rsh^* z_1, z_1 \Rsh^* z_2$.

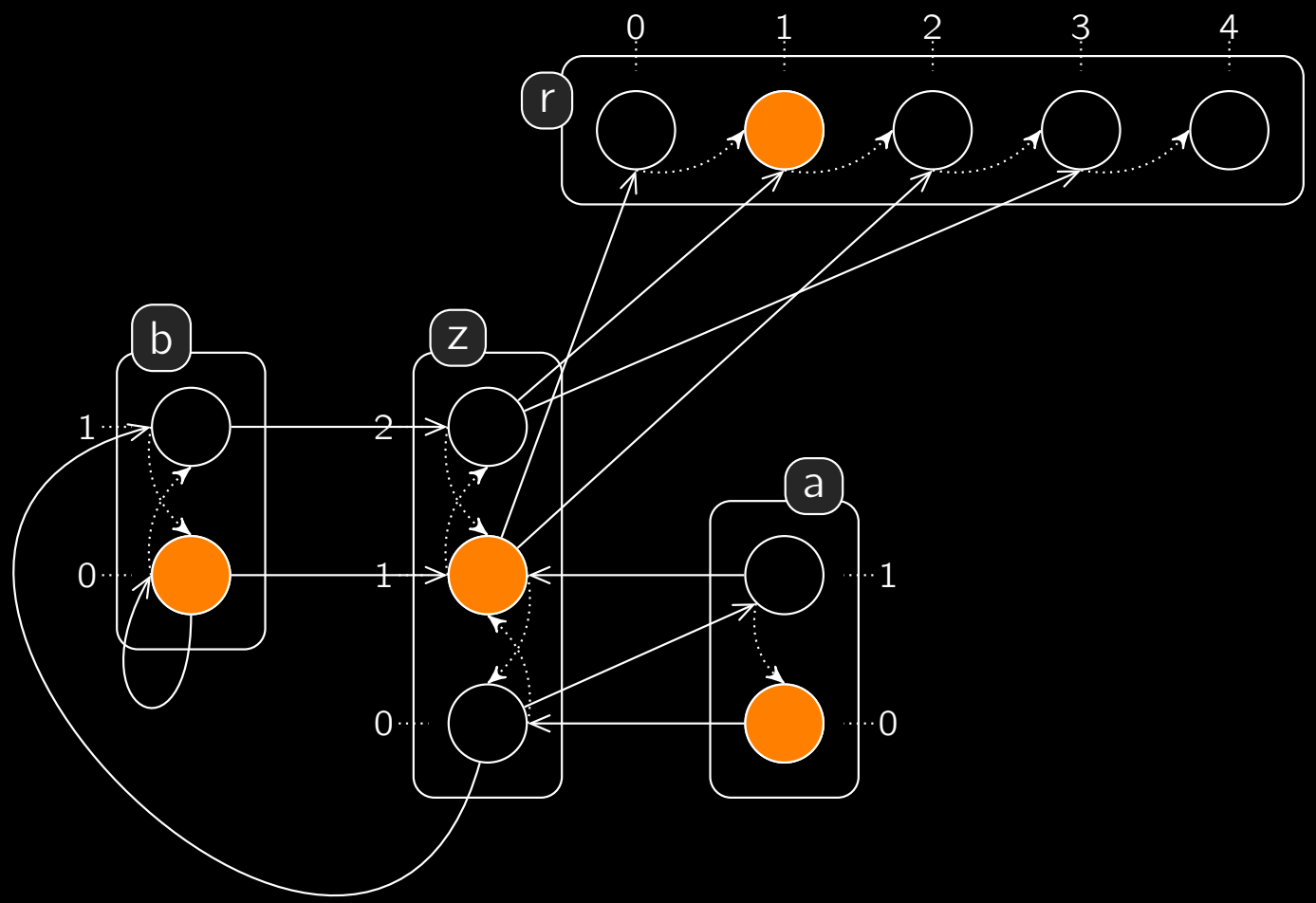
Toy Example



$\mathbf{BS}(r_0 \lceil^* r_4) = \{[z_1 \rightarrow r_0 \lceil r_1, z_2 \rightarrow r_1 \lceil r_2, z_1 \rightarrow r_2 \lceil r_3, z_2 \rightarrow r_3 \lceil r_4]\}$

Objective sequence to check: $z_1 \lceil^* z_2, z_2 \lceil^* z_1, z_1 \lceil^* z_2$.

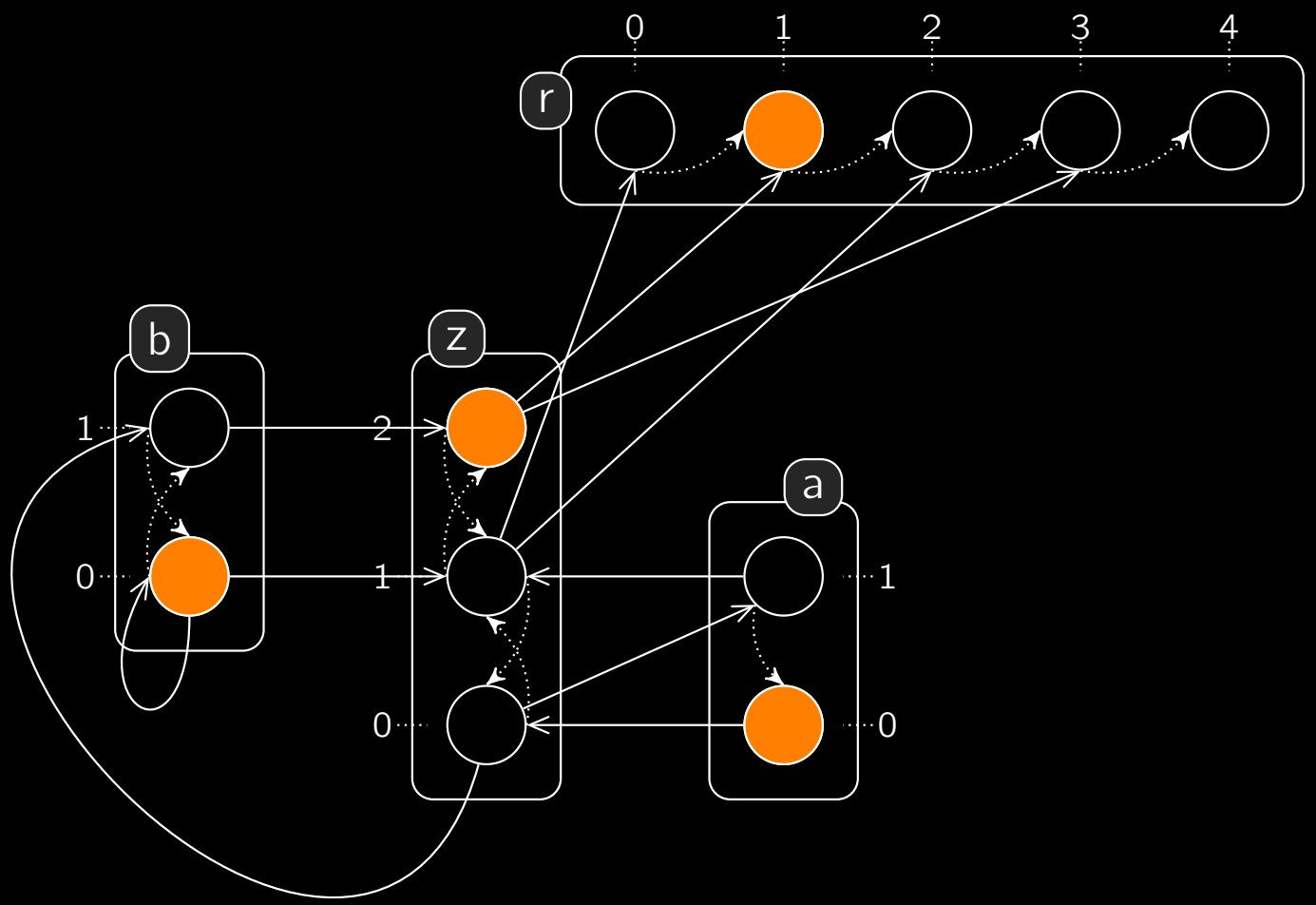
Toy Example



$\mathbf{BS}(r_0 \Rsh^* r_4) = \{[z_1 \rightarrow r_0 \Rsh r_1, z_2 \rightarrow r_1 \Rsh r_2, z_1 \rightarrow r_2 \Rsh r_3, z_2 \rightarrow r_3 \Rsh r_4]\}$

Objective sequence to check: $z_1 \Rsh^* z_2, z_2 \Rsh^* z_1, z_1 \Rsh^* z_2$.

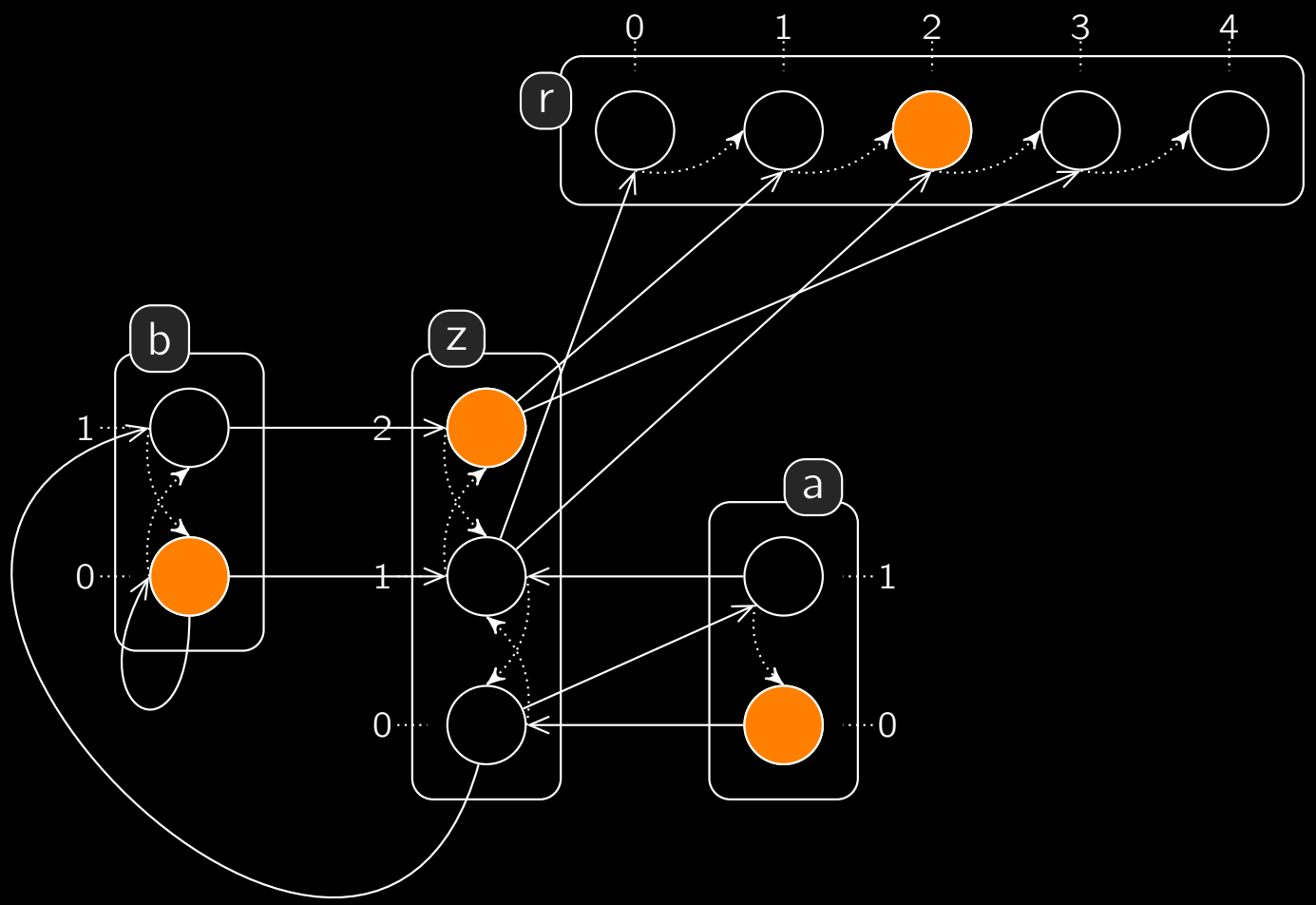
Toy Example



$\mathbf{BS}(r_0 \lceil^* r_4) = \{[z_1 \rightarrow r_0 \lceil r_1, z_2 \rightarrow r_1 \lceil r_2, z_1 \rightarrow r_2 \lceil r_3, z_2 \rightarrow r_3 \lceil r_4]\}$

Objective sequence to check: $z_1 \lceil^* z_2, z_2 \lceil^* z_1, z_1 \lceil^* z_2$.

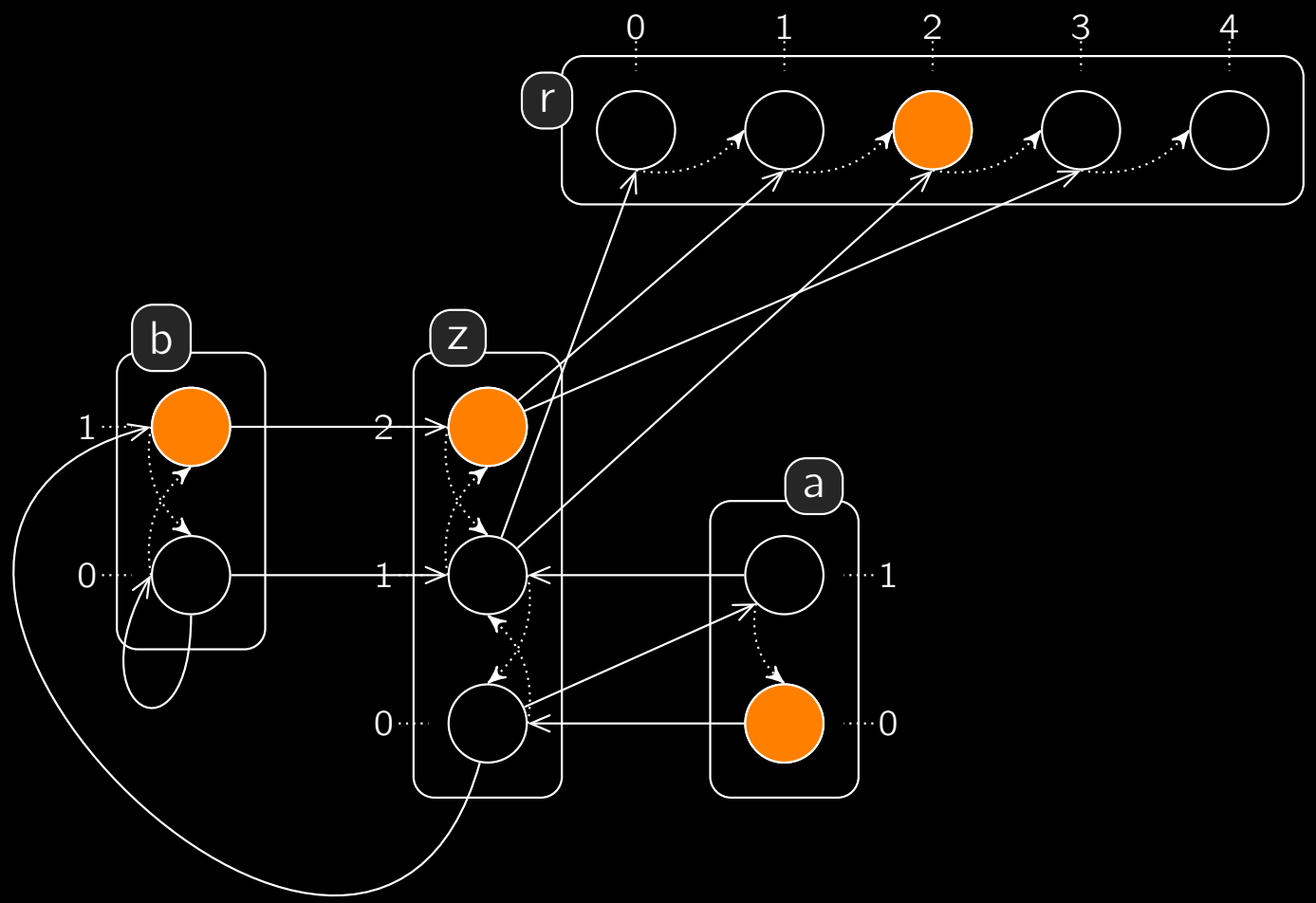
Toy Example



$\mathbf{BS}(r_0 \Rsh^* r_4) = \{[z_1 \rightarrow r_0 \Rsh r_1, z_2 \rightarrow r_1 \Rsh r_2, z_1 \rightarrow r_2 \Rsh r_3, z_2 \rightarrow r_3 \Rsh r_4]\}$

Objective sequence to check: $z_1 \Rsh^* z_2, z_2 \Rsh^* z_1, z_1 \Rsh^* z_2$.

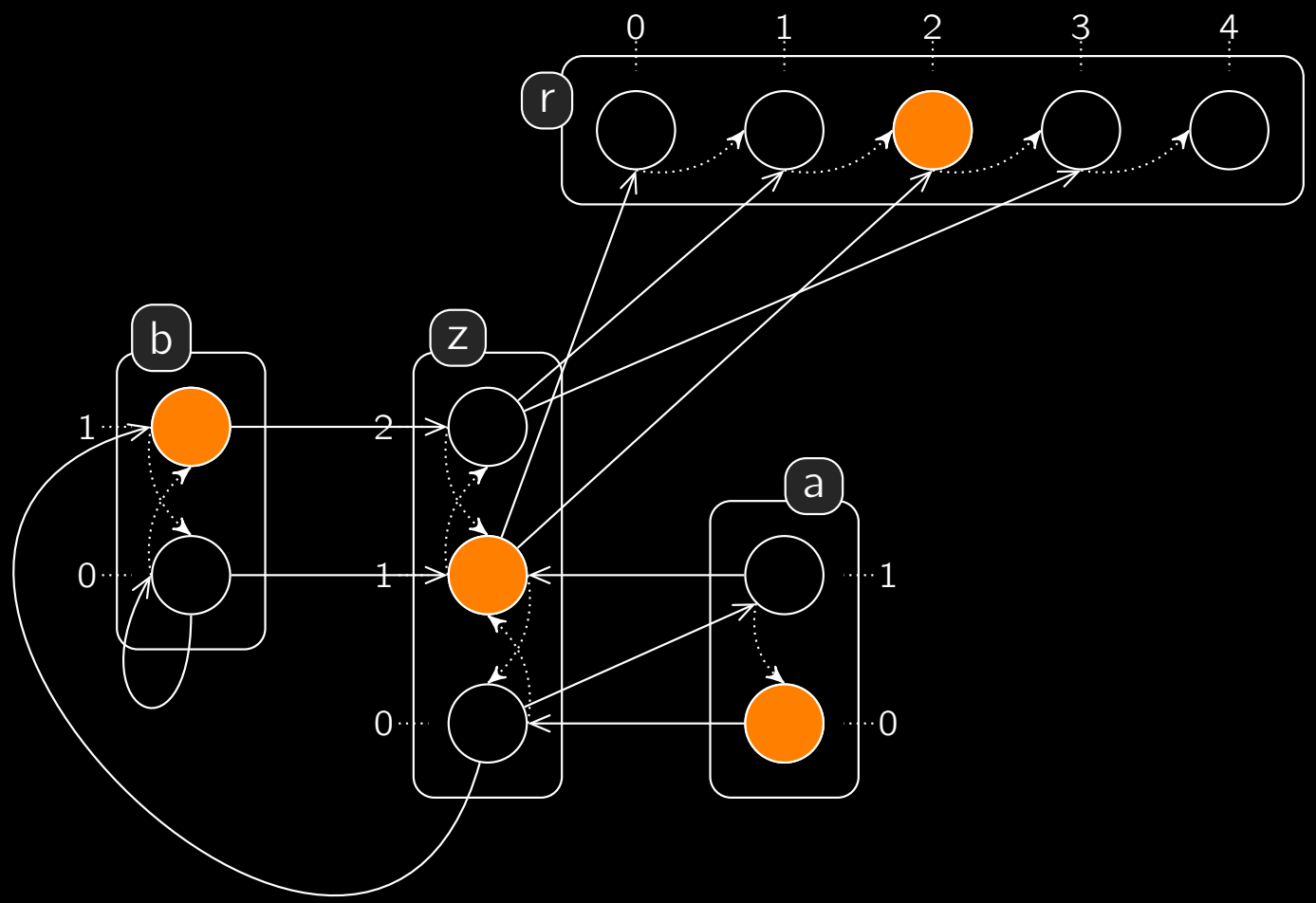
Toy Example



$\mathbf{BS}(r_0 \Rsh^* r_4) = \{[z_1 \rightarrow r_0 \Rsh r_1, z_2 \rightarrow r_1 \Rsh r_2, z_1 \rightarrow r_2 \Rsh r_3, z_2 \rightarrow r_3 \Rsh r_4]\}$

Objective sequence to check: $z_1 \Rsh^* z_2, z_2 \Rsh^* z_1, z_1 \Rsh^* z_2$.

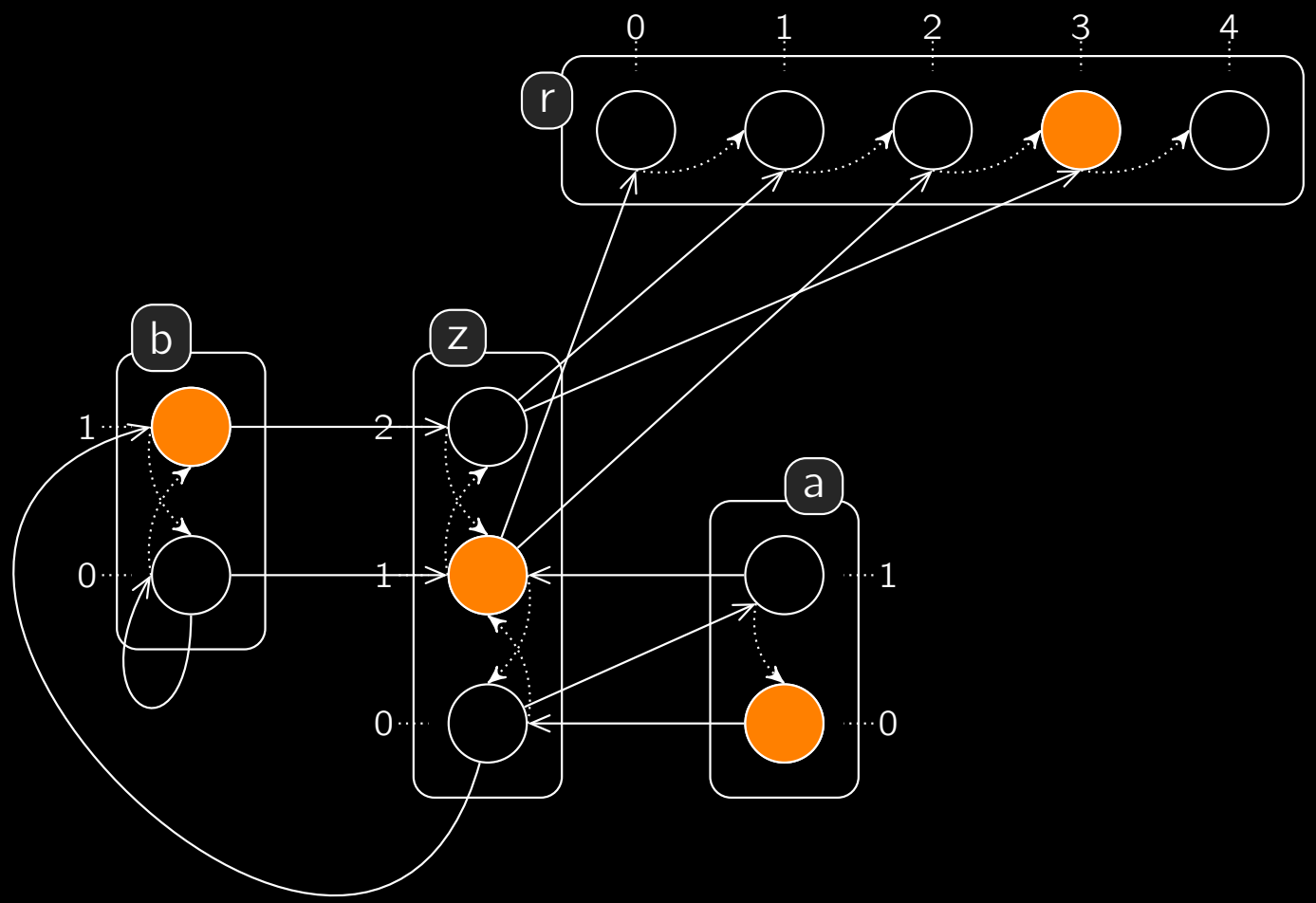
Toy Example



$\mathbf{BS}(r_0 \lceil^* r_4) = \{[z_1 \rightarrow r_0 \lceil r_1, z_2 \rightarrow r_1 \lceil r_2, z_1 \rightarrow r_2 \lceil r_3, z_2 \rightarrow r_3 \lceil r_4]\}$

Objective sequence to check: $z_1 \lceil^* z_2, z_2 \lceil^* z_1, z_1 \lceil^* z_2$.

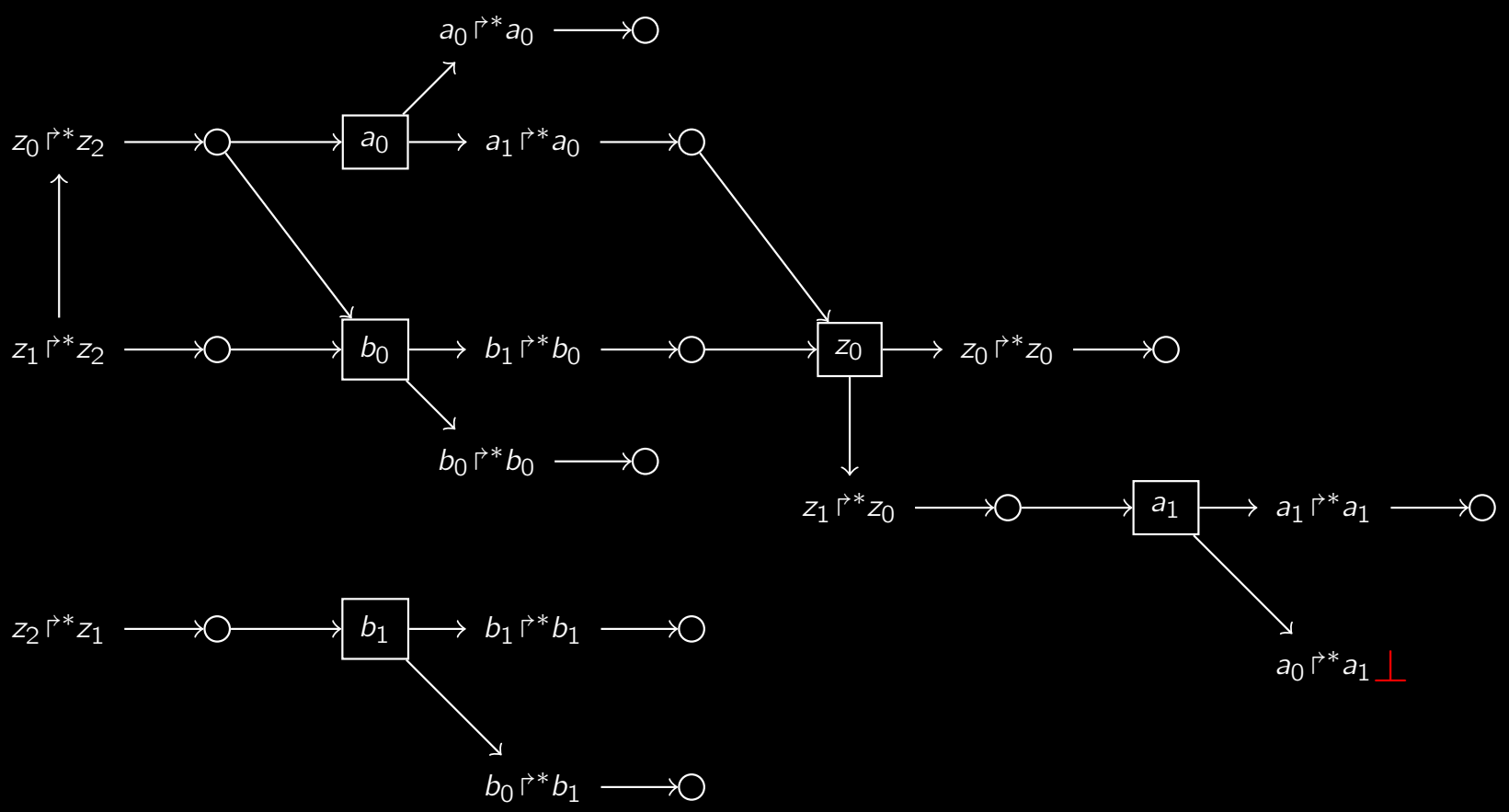
Toy Example



$\mathbf{BS}(r_0 \lceil^* r_4) = \{[z_1 \rightarrow r_0 \lceil r_1, z_2 \rightarrow r_1 \lceil r_2, z_1 \rightarrow r_2 \lceil r_3, z_2 \rightarrow r_3 \lceil r_4]\}$

Objective sequence to check: $z_1 \lceil^* z_2, z_2 \lceil^* z_1, z_1 \lceil^* z_2$.

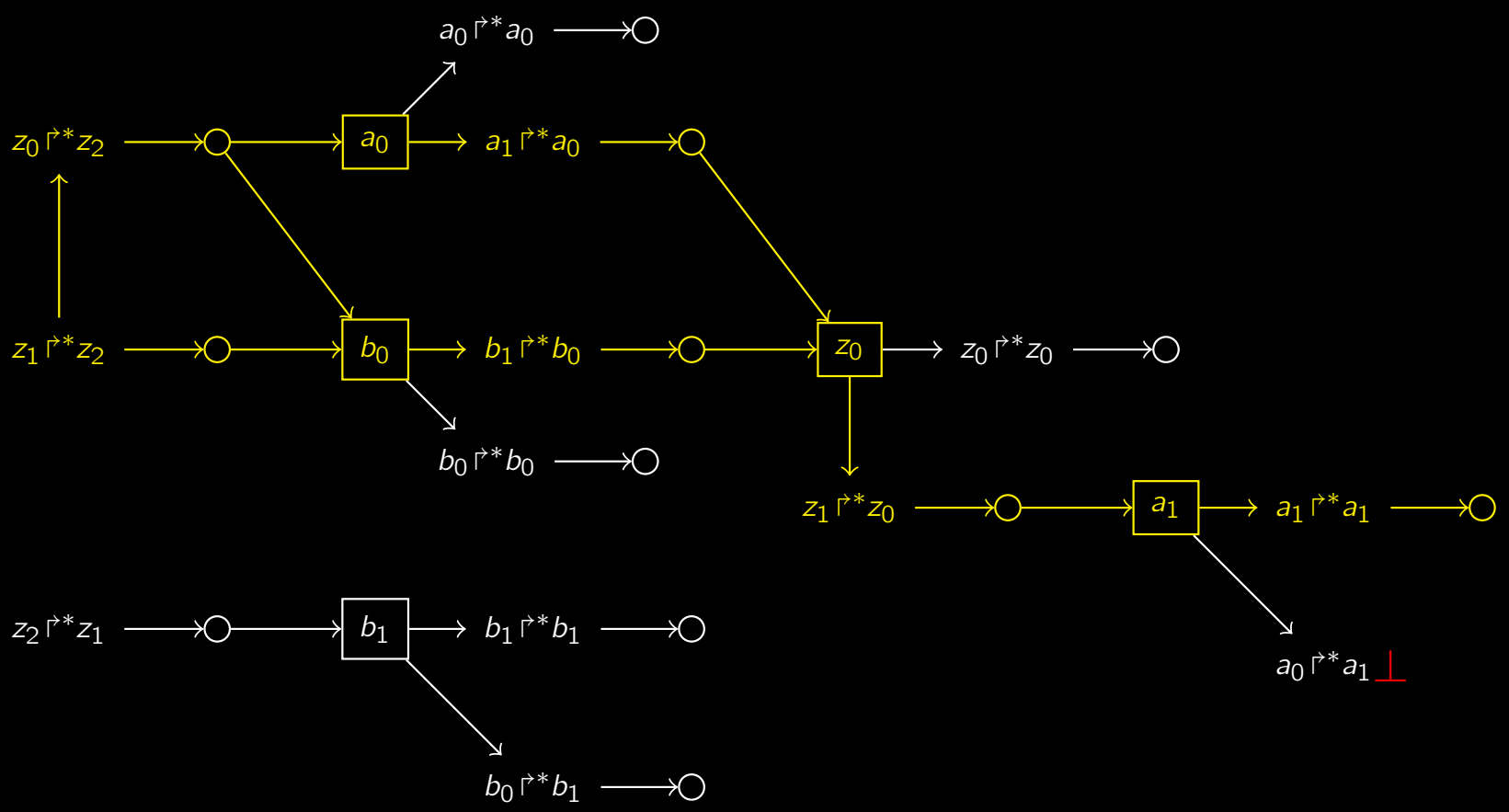
Saturated Abstract Structure of Example



$a_1 \lhd a_0$

	s	$z_1 \Rsh^* z_2$	$z_2 \Rsh^* z_1$	$z_1 \Rsh^* z_2$
minPROC	a_1, b_1, z_1			
$max\varsigma$	a_1, b_1, z_1			

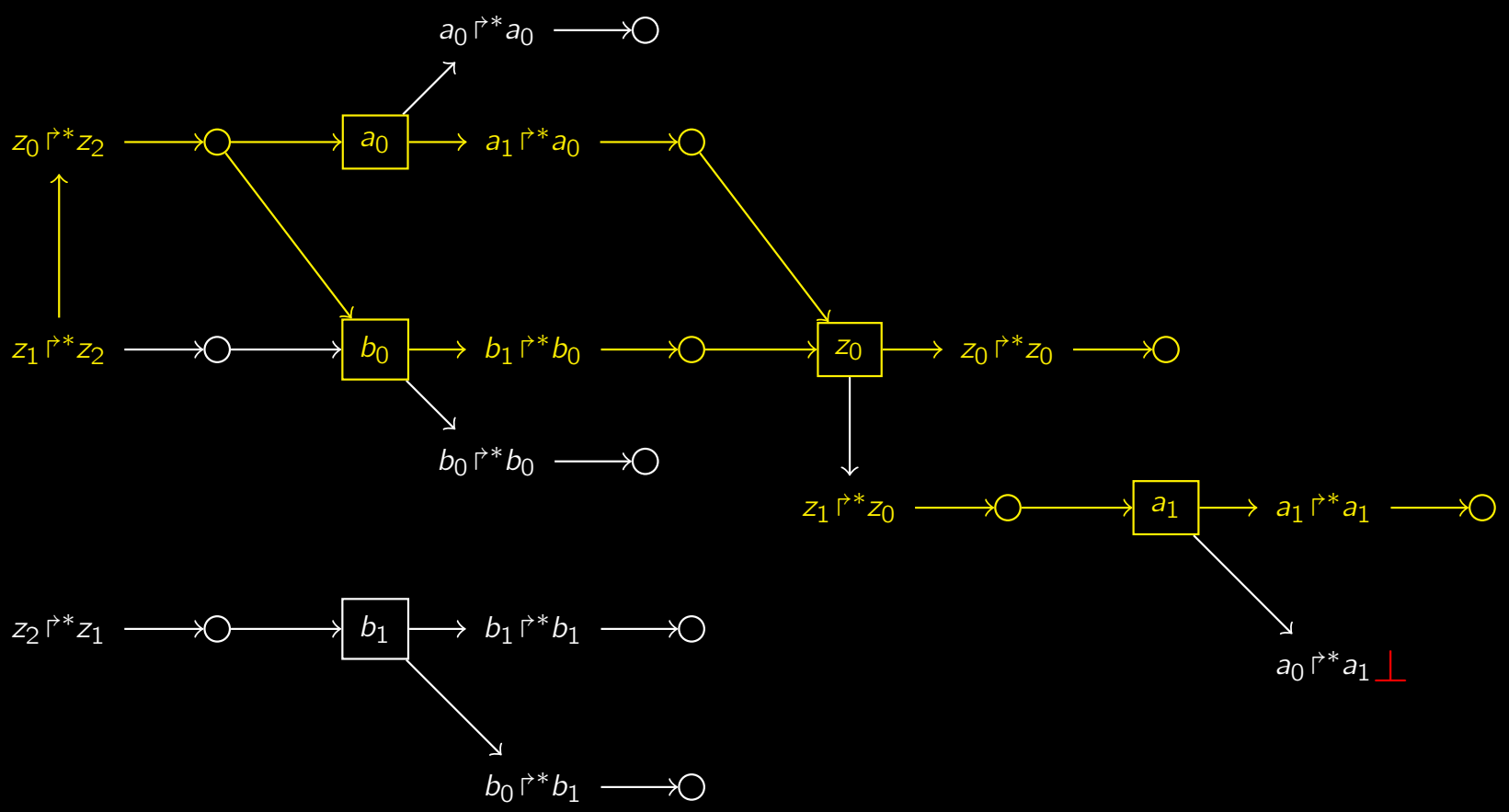
Saturated Abstract Structure of Example



$a_1 \lhd a_0$

	s	$z_1 \Rsh^* z_2$	$z_2 \Rsh^* z_1$	$z_1 \Rsh^* z_2$
minPROC	a_1, b_1, z_1			
$max\varsigma$	a_1, b_1, z_1			

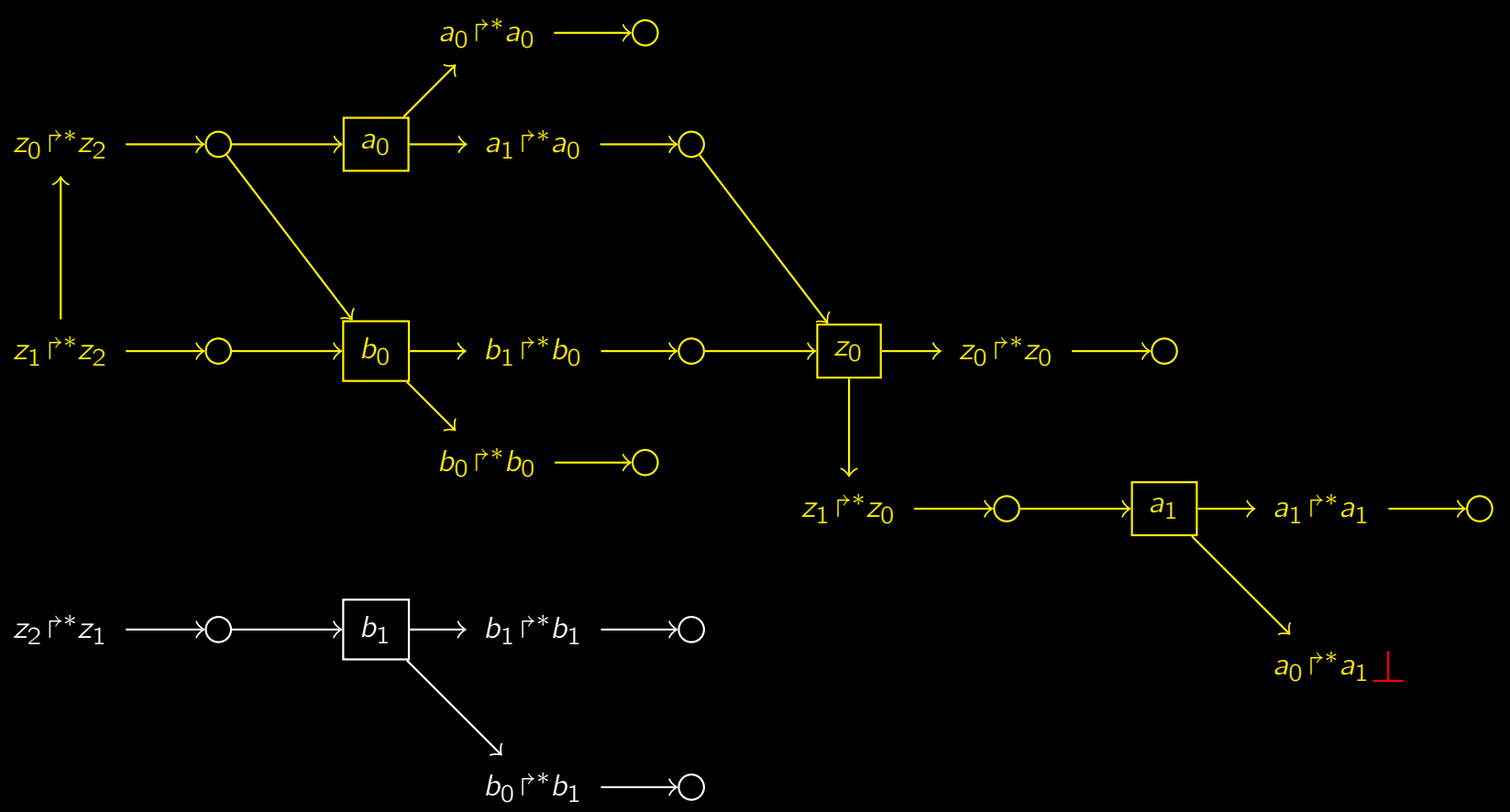
Saturated Abstract Structure of Example



$a_1 \lhd a_0$

	s	$z_1 \Rsh^* z_2$	$z_2 \Rsh^* z_1$	$z_1 \Rsh^* z_2$
minPROC	a_1, b_1, z_1	a_0, a_1, b_0 z_0, z_2		
$max\varsigma$	a_1, b_1, z_1			

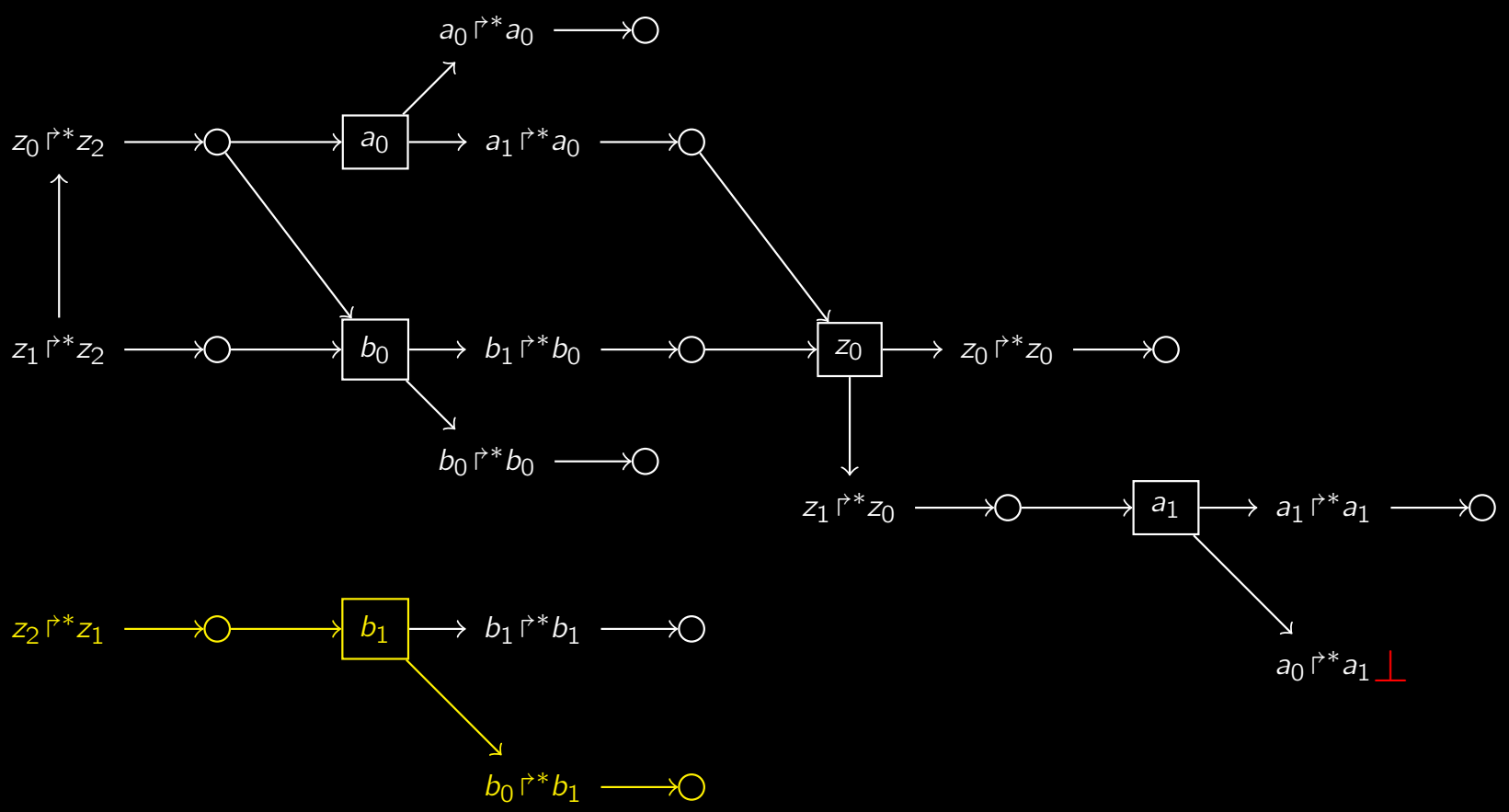
Saturated Abstract Structure of Example



$a_1 \lhd a_0$

	s	$z_1 \Rsh^* z_2$	$z_2 \Rsh^* z_1$	$z_1 \Rsh^* z_2$
minPROC	a_1, b_1, z_1	a_0, a_1, b_0 z_0, z_2		
$max\varsigma$	a_1, b_1, z_1	a_0, a_1, b_0 z_2		

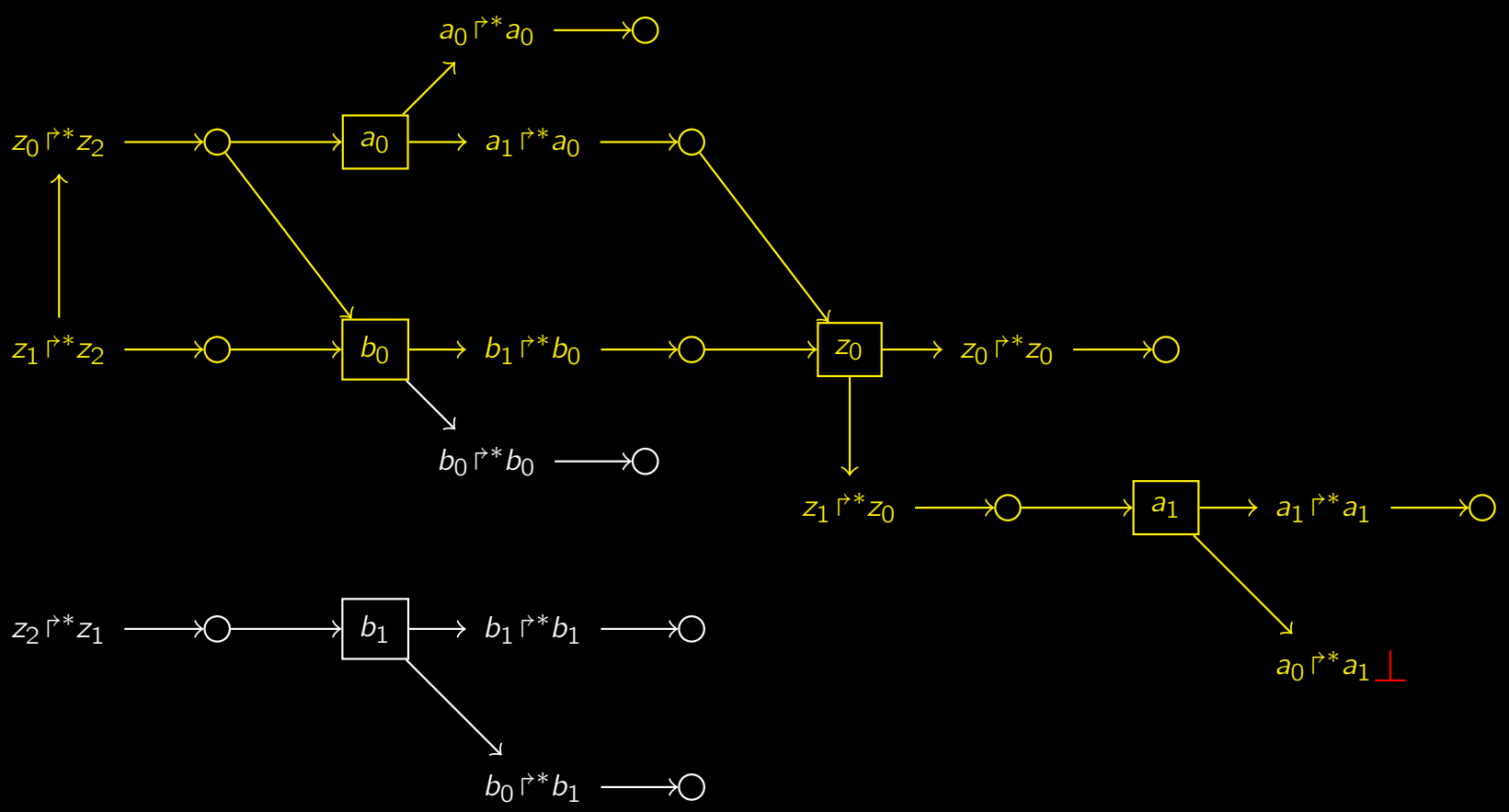
Saturated Abstract Structure of Example



$a_1 \lhd a_0$

	s	$z_1 \Rsh^* z_2$	$z_2 \Rsh^* z_1$	$z_1 \Rsh^* z_2$
minPROC	a_1, b_1, z_1	a_0, a_1, b_0 z_0, z_2	b_1, z_1	
$max\varsigma$	a_1, b_1, z_1	a_0, a_1, b_0 z_2	a_0, a_1, b_1 z_1	

Saturated Abstract Structure of Example



$a_1 \lhd a_0$

	s	$z_1 \Rsh^* z_2$	$z_2 \Rsh^* z_1$	$z_1 \Rsh^* z_2$
minPROC	a_1, b_1, z_1	a_0, a_1, b_0 z_0, z_2	b_1, z_1	a_0, a_1, b_0 z_0, z_2
$max\varsigma$	a_1, b_1, z_1	a_0, a_1, b_0 z_2	a_0, a_1, b_1 z_1	