Computational approaches to analyze complex dynamic systems: model-checking and its applications.

Part 4: Models and algorithms to analyze large-scale concurrent systems: approaches inspired by pi-calculus and static analysis

Morgan Magnin
morgan.magnin@irccyn.ec-nantes.fr | www.morganmagnin.net

NII - *Inoue Laboratory*
École Centrale de Nantes - IRCCyN - *MeForBio team*

Lecture Series - Lecture 4 / NII - 2013/04/17

Overview

Motivations

Objective: formal verification of properties

- Model the system S :
  - Discrete models: finite state automata, Petri nets, ... $\Rightarrow$ Lecture 1
  - Timed models:
    - Timed extensions of finite state automata: timed/hybrid automata $\Rightarrow$ Lecture 2
    - Timed extensions of Petri nets: time/stopwatch Petri nets $\Rightarrow$ Lecture 3
- Formalize the specification φ :
  - Observers
  - Temporal logics: LTL, CTL, ... $\Rightarrow$ Lecture 1
  - Timed extensions of temporals logics $\Rightarrow$ Lectures 2 & 3
- Does $S \models \varphi$?

Model-checking algorithms

$\Rightarrow$ State space exploration

Motivations

Objective: formal verification of properties

- Model the system S :
  - Discrete models: finite state automata, Petri nets, ... $\Rightarrow$ Lecture 1
  - Timed models:
    - Timed extensions of finite state automata: timed/hybrid automata $\Rightarrow$ Lecture 2
    - Timed extensions of Petri nets: time/stopwatch Petri nets $\Rightarrow$ Lecture 3
- Formalize the specification φ :
  - Observers
  - Temporal logics: LTL, CTL, ... $\Rightarrow$ Lecture 1
  - Timed extensions of temporals logics $\Rightarrow$ Lectures 2 & 3
- Does $S \models \varphi$?

Model-checking algorithms

$\Rightarrow$ State space exploration

Some major issues

Need for modeling tasks with suspending/resuming features

Expressivity/**Decidability** compromise to discuss ⇒ Lectures 2 & 3

State space combinatorial explosion

- Need for symbolic approaches ⇒ Lectures 2 & 3
- Need for new models and abstracted algorithms ⇒ **Lecture 4**

Some major issues

Need for modeling tasks with suspending/resuming features

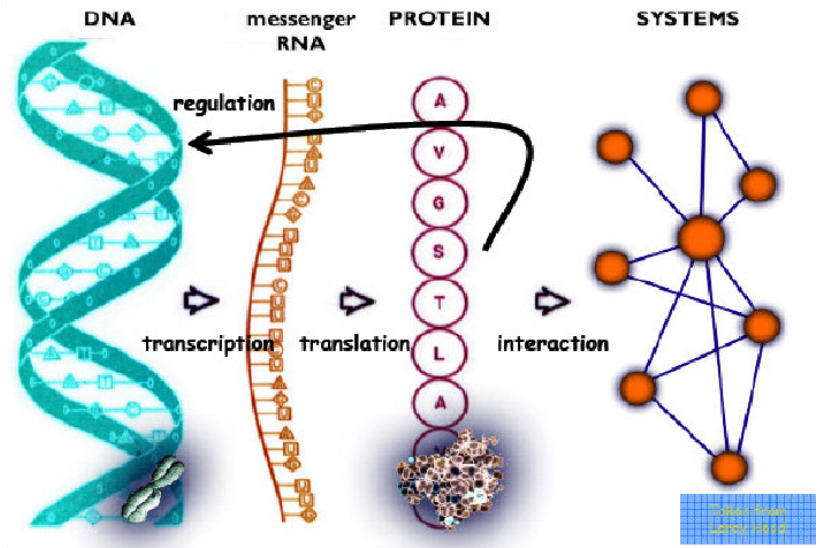
Expressivity/**Decidability** compromise to discuss ⇒ Lectures 2 & 3

State space combinatorial explosion

- Need for symbolic approaches ⇒ Lectures 2 & 3
- Need for new models and abstracted algorithms ⇒ **Lecture 4**

Context and Aims

MeForBio team: Algebraic modeling to study complex dynamical biological systems

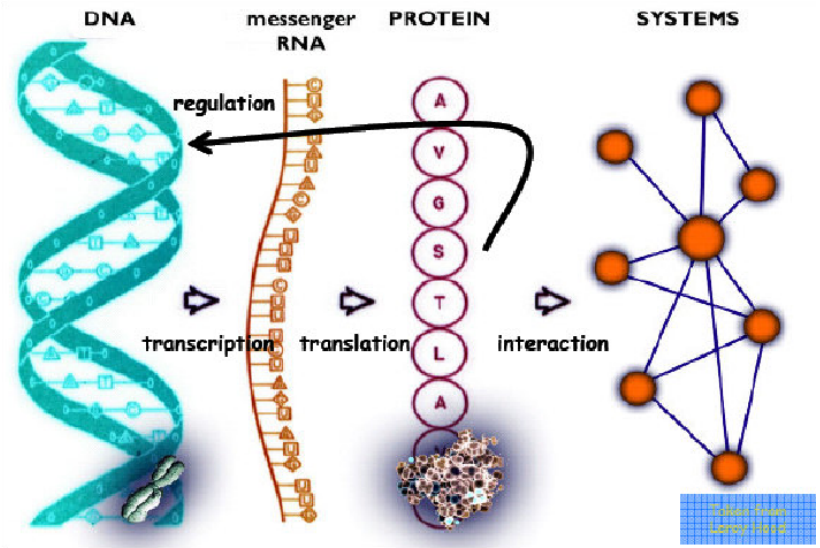


1) Two main models
   - Historical model: **Biological Regulatory Network (René Thomas)**
   - Recently designed model: **Process Hitting**

2) Allow efficient translation from one model to the other

Context and Aims

MeForBio team: Algebraic modeling to study complex dynamical biological systems



1) Two main models
   - Historical model: **Biological Regulatory Network (René Thomas)**
   - Recently designed model: **Process Hitting**

2) Allow efficient translation from one model to the other

Today's issue

Tricky question

How can we study complex dynamical biological systems, **involving up to 1.000 interacting components**?

Observation

- Classical model-checking approaches suffer from state space explosion
- Leads:
  - Taking profit for Process Algebra structure, based on a **compact representation of the interactions**
  - Develop **static analysis approaches** to verify some crucial properties, e.g. stable states, reachability, key processes, ...

Contribution

Scientific challenge

How can we cope with the analysis of **large-scale systems**, involving up to 1.000 interacting components?

Objectives of this talk

- Introduce a Process Algebra inspired framework based on a compact representation of the interactions
- Develop efficient **static analysis approaches** to answer most common problems
- Apply the methodology to large-scale biological regulatory networks

Joint work with

- L. Paulevé (ETH Zurich), M. Folschette, O. Roux (IRCCyN)
- K. Inoue (NII)

Overview

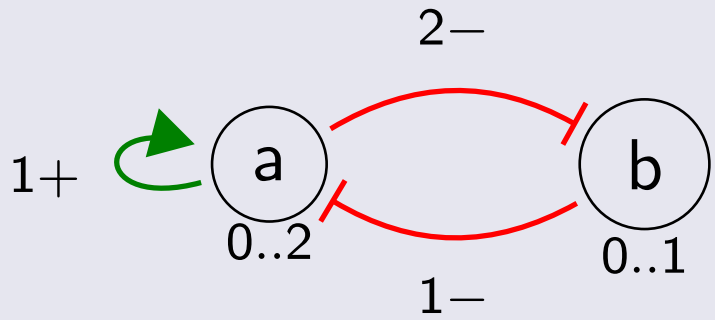
Short introduction to Biological Regulatory Networks

Principle of R. Thomas' discrete modeling [TGL76]

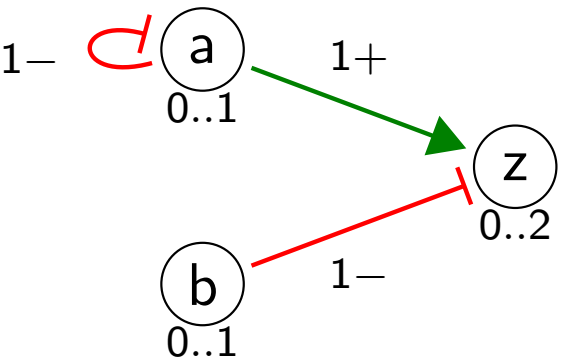
- **Activations** and **inhibitions** between genes
- Gene/protein couples
- Genes expression is associated to a set of discrete logic levels
- **Effective control** beyond a given threshold; opposite effect below.

Interaction graph

- Nodes = **Genes**
- Directed edges = **Interactions**
- But what is the **evolutionary tendency** of a when a is at level 1 and b at level 1? $\Rightarrow$ Need for **parametrization**



Biological Regulatory Network (Thomas' modeling)



ω		$k_{z,\omega}$
a	b	
$-$	$+$	1
$-$	$-$	0
$+$	$+$	2
$+$	$-$	1

ω	$k_{a,\omega}$
a	
$+$	1
$-$	0

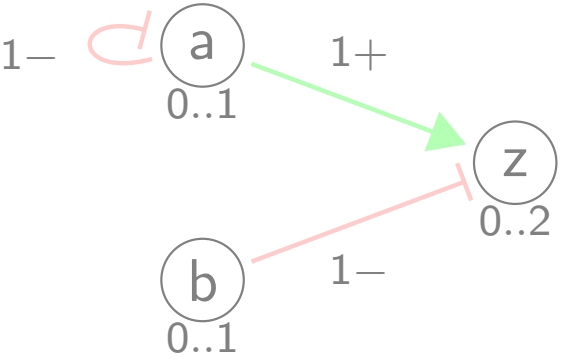
$k_{b,\omega}$
1

Proposed by René Thomas in 1973, several extensions since then

Historical bio-informatics model for studying genes interactions

Widely used and well-adapted to represent dynamic gene systems

Biological Regulatory Network (Thomas' modeling)



ω		$k_{z,\omega}$
a	b	
$-$	$+$	1
$-$	$-$	0
$+$	$+$	2
$+$	$-$	1

ω	$k_{a,\omega}$
a	
$+$	1
$-$	0

$k_{b,\omega}$
1

Interaction Graph: structure of the system (genes & interactions)

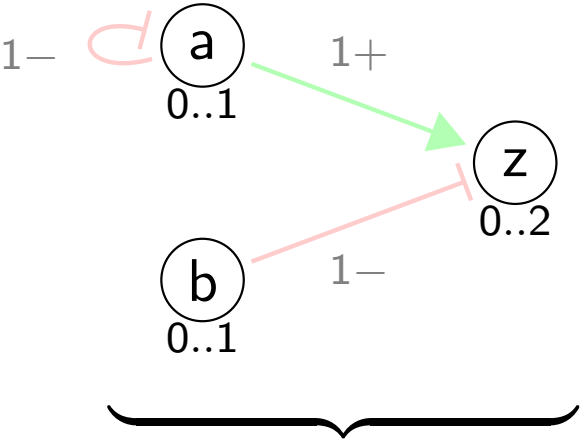
Nodes: genes

- Name a, b, z
- Possible values (levels of expression) 0..1, 0..2

Edges: interactions

- Threshold 1
- Type (activation or inhibition) $+$ / $-$

Biological Regulatory Network (Thomas' modeling)



ω		$k_{z,\omega}$
a	b	
$-$	$+$	1
$-$	$-$	0
$+$	$+$	2
$+$	$-$	1

ω	$k_{a,\omega}$
a	
$+$	1
$-$	0

$k_{b,\omega}$
1

Interaction Graph: structure of the system (genes & interactions)

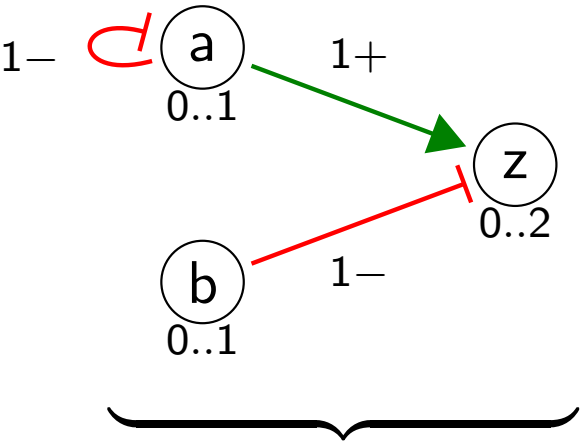
Nodes: genes

- Name a, b, z
- Possible values (levels of expression) 0..1, 0..2

Edges: interactions

- Threshold 1
- Type (activation or inhibition) $+$ / $-$

Biological Regulatory Network (Thomas' modeling)



ω		$k_{z,\omega}$
a	b	
$-$	$+$	1
$-$	$-$	0
$+$	$+$	2
$+$	$-$	1

ω	$k_{a,\omega}$
a	
$+$	1
$-$	0

$k_{b,\omega}$
1

Interaction Graph: structure of the system (genes & interactions)

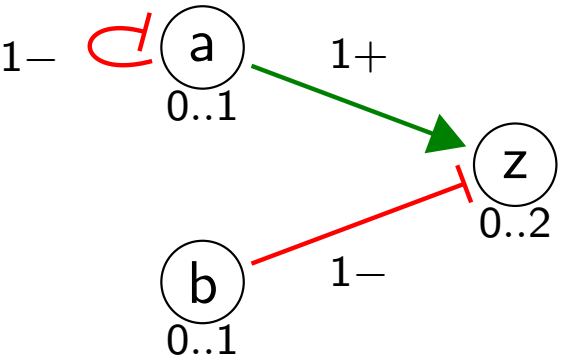
Nodes: genes

- Name a, b, z
- Possible values (levels of expression) 0..1, 0..2

Edges: interactions

- Threshold 1
- Type (activation or inhibition) $+$ / $-$

Biological Regulatory Network (Thomas' modeling)



ω		$k_{z,\omega}$
a	b	
$-$	$+$	1
$-$	$-$	0
$+$	$+$	2
$+$	$-$	1

ω	$k_{a,\omega}$
a	
$+$	1
$-$	0

$k_{b,\omega}$
1

Parametrization

Parametrization: strength of the influences (cooperations)

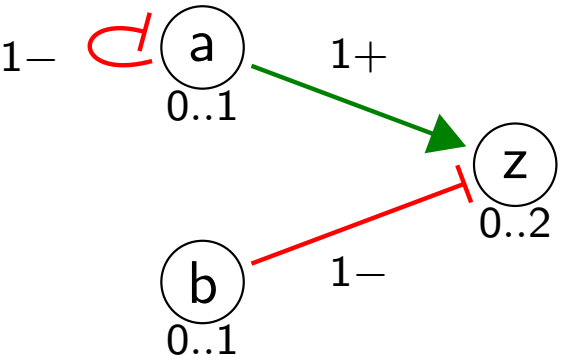
Maps of tendencies for each gene

→ To any **influences of predecessors** ω

→ Corresponds a **parameter** $k_{x,\omega}$

$k_{z,\{a^+,b^+\}} = 2$ means: z tends to 2 when $a \geq 1$ and $b < 1$

Biological Regulatory Network (Thomas' modeling)



ω		$k_{z,\omega}$
a	b	
$-$	$+$	1
$-$	$-$	0
$+$	$+$	2
$+$	$-$	1

ω	$k_{a,\omega}$
a	
$+$	1
$-$	0

$k_{b,\omega}$
1

Parametrization

Parametrization: strength of the influences (cooperations)

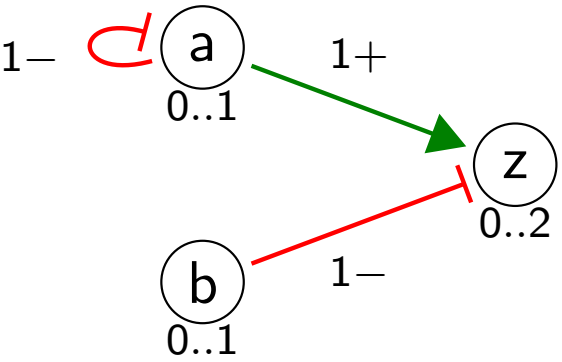
Maps of tendencies for each gene

→ To any **influences of predecessors** ω

→ Corresponds a **parameter** $k_{x,\omega}$

$k_{z,\{a^+,b^+\}} = 2$ means: z tends to 2 when $a \geq 1$ and $b < 1$

Biological Regulatory Network (Thomas' modeling)



ω		$k_{z,\omega}$
a	b	
$-$	$+$	1
$-$	$-$	0
$+$	$+$	2
$+$	$-$	1

ω	$k_{a,\omega}$
a	
$+$	1
$-$	0

$k_{b,\omega}$
1

Biological Regulatory Network

→ All needed information to run the model or study its dynamics:
- Build the State Graph
- Find reachability properties, fixed points, attractors
- Other properties...

→ **Strengths**: well adapted for the study of biological systems

→ **Drawbacks**: inherent complexity; needs the full specification of cooperations

Limits of discrete modeling

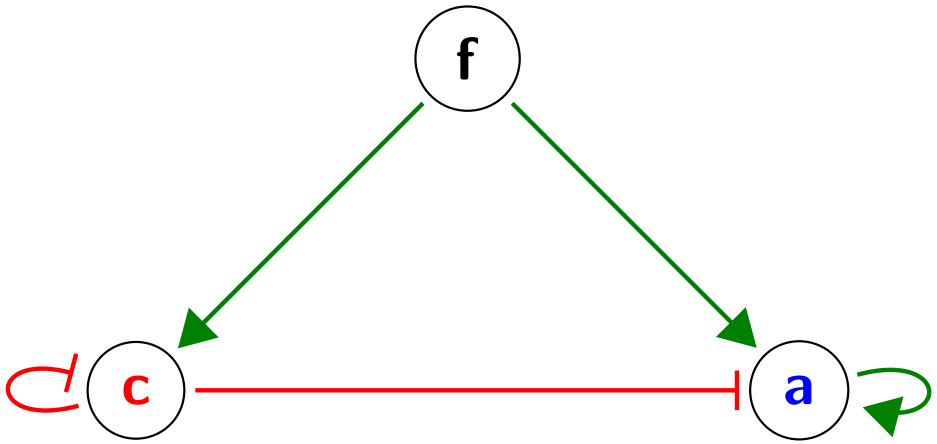
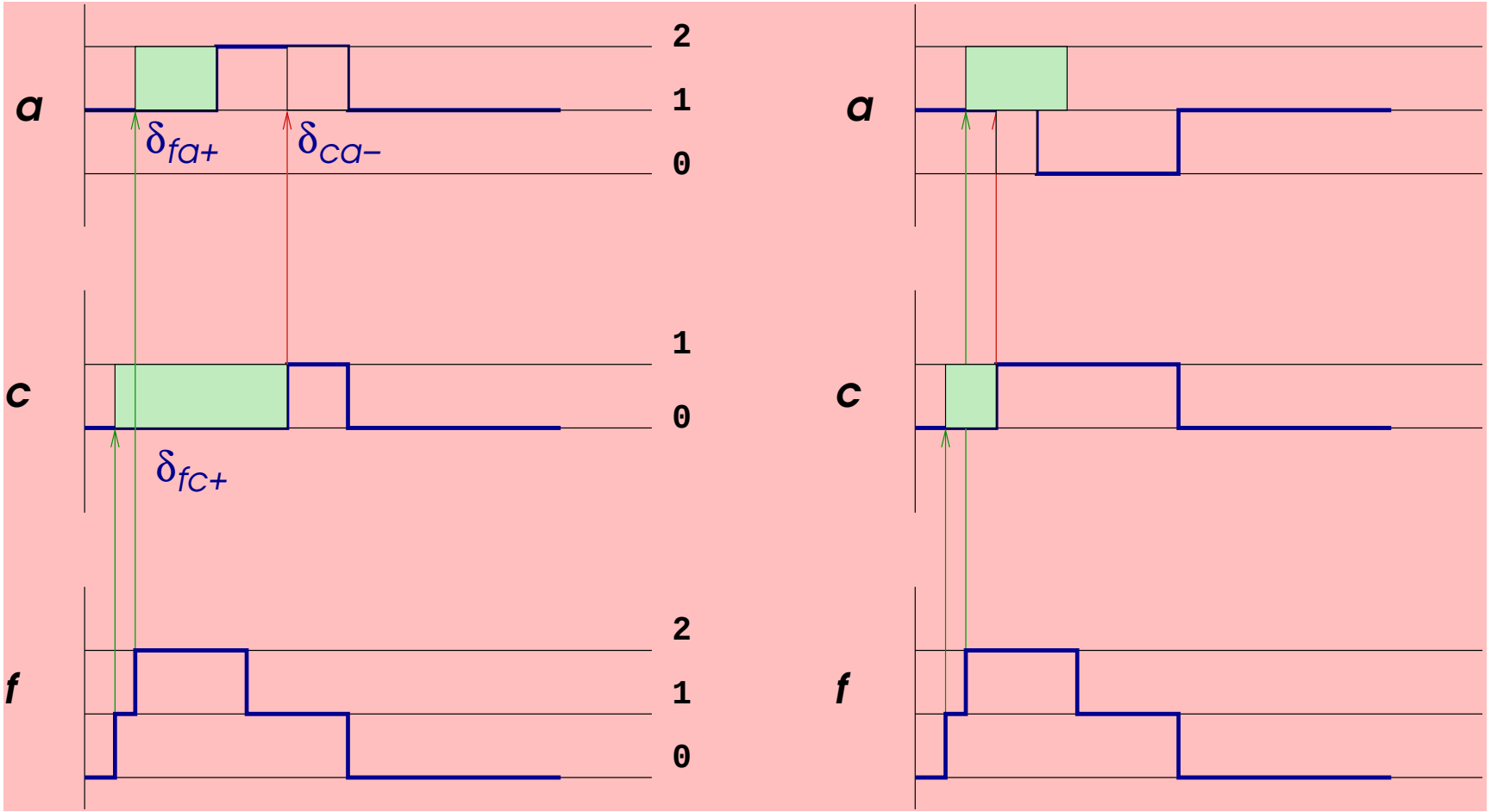


Figure: Motif inside biological segmentation networks (e.g. drosophila)

(Rmq.: boolean network)

Limits of discrete modeling



Issues related to the synthesis of timed parameters

Problems

- Infer the production and degradation rates
- Consider **accumulation** mechanisms (on/off oscillations)

Adaptation of the R. Thomas model

The logic interaction graph is enriched with two functions. These functions give, for any discrete state of the network and for any gene:

- The **production delay** of the gene, depending on its set of resources;
- The **degradation delay** of the gene, depending on its set of resources.

Increasing the models expressivity

Adapt the model to biological issues

- Introduction of delays $\Rightarrow$ timed transition systems
- Need for modeling tasks with suspend / resume $\Rightarrow$ introduce the notion of stopwatches
- Balance between **expressivity** and **decidability**

Problem

Choosing an appropriate time model for S

- Dense time?
- Discrete time?

Inference and verification of quantitative temporal properties

$\Rightarrow$ Efficient state-space exploration algorithms

$\Rightarrow$ Compact data structures for **storage** and **computation** of the state space

Problem

Choosing an appropriate time model for S

- Dense time?
- Discrete time?

Inference and verification of quantitative temporal properties

$\Rightarrow$ Efficient state-space exploration algorithms

$\Rightarrow$ Compact data structures for **storage** and **computation** of the state space

Analysis of Biological Regulatory Networks

Motivation: apply modal logics

- Compute state graph and check properties
- Translate the model into a **discrete-events** (resp. **timed**) model, e.g. (time) Petri nets, and check properties

Motivation: why Petri nets?

- Mathematical **and** graphic formalism
- Easy representation **concurrence and parallelism**
- **Structural** properties (P-invariants, T-invariants, ...)
- **Dynamical** properties (liveness, boundedness, reachability, ...)
- Mature tools : Snoopy, ginSIM, Roméo, etc.

Petri net - Reminder

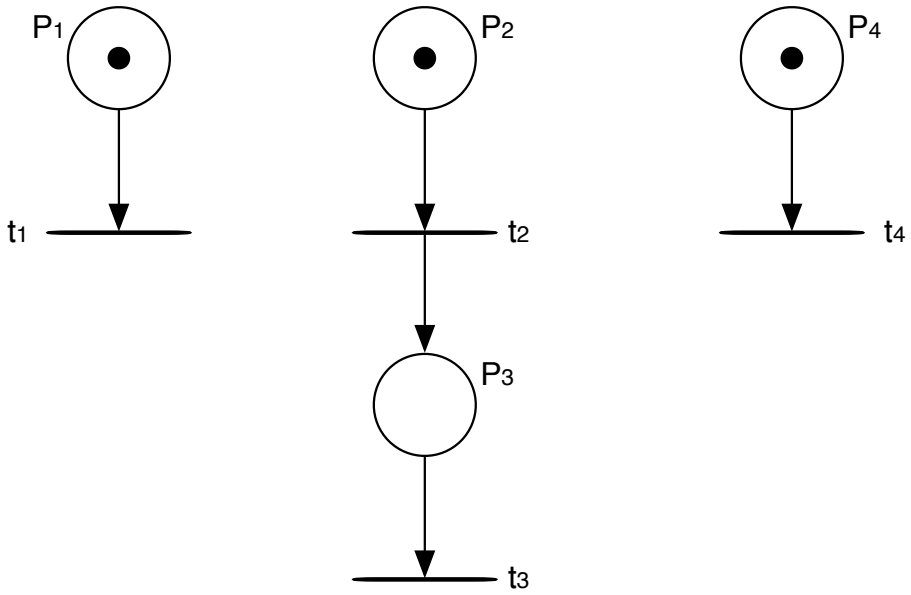


Figure: A Petri net

$$\{P_1, P_2, P_4\}$$

Petri net - Reminder

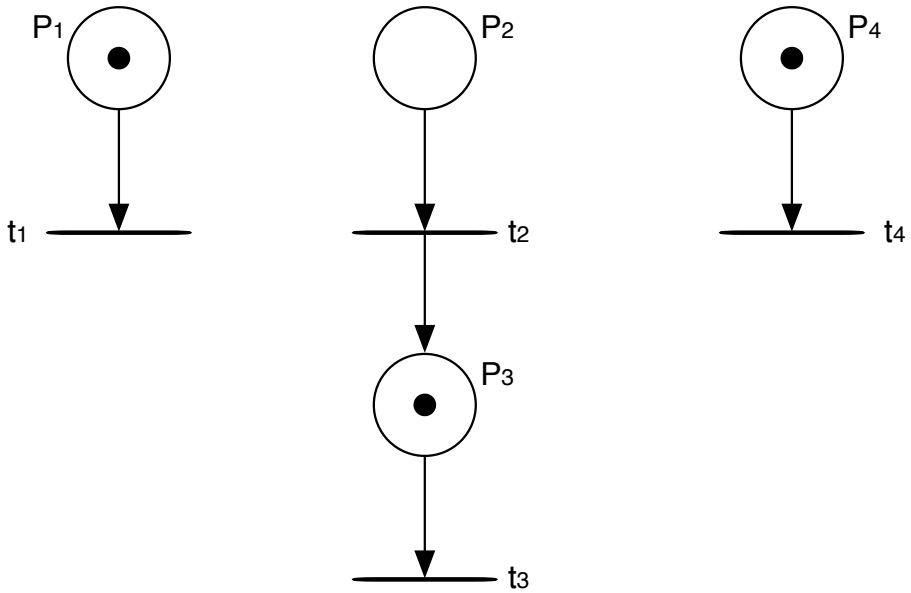


Figure: A Petri net

$$\{P_1, P_2, P_4\} \xrightarrow{t_2} \{P_1, P_3, P_4\} \xrightarrow{t_1} \dots$$

Petri net - Reminder

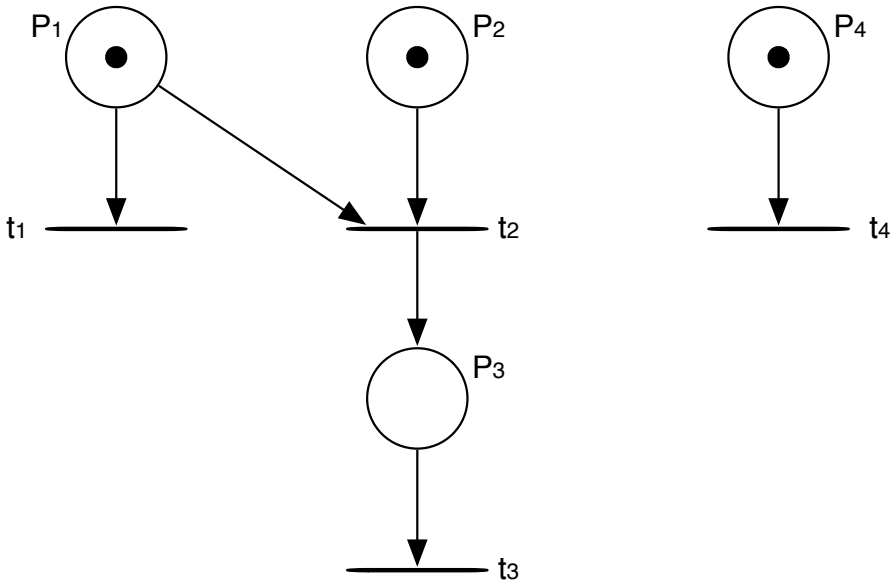


Figure: An other Petri net

$$\{P_1, P_2, P_4\}$$

Petri net - Reminder

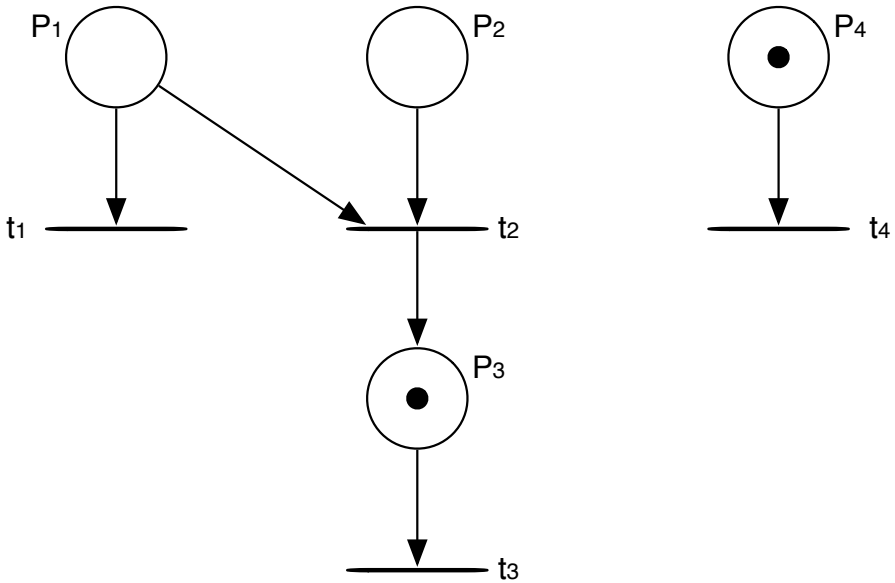


Figure: An other Petri net

$$\{P_1, P_2, P_4\} \xrightarrow{t_2} \{P_3, P_4\} \ldots$$

Petri net with reset arcs - Reminder

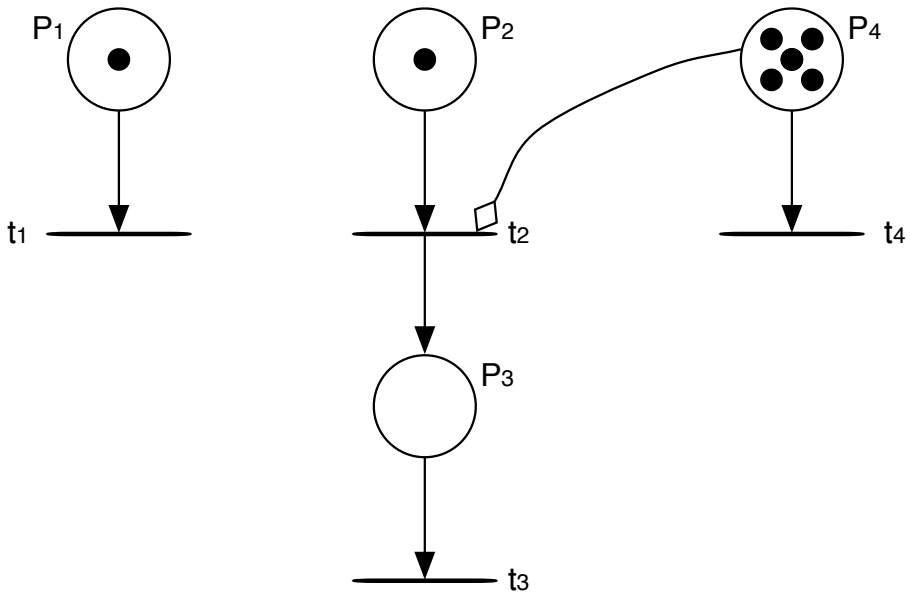


Figure: A Petri net with reset arcs

$$\{P_1, P_2, 5 \times P_4\} \xrightarrow{t_2} \dots$$

Petri net with reset arcs - Reminder

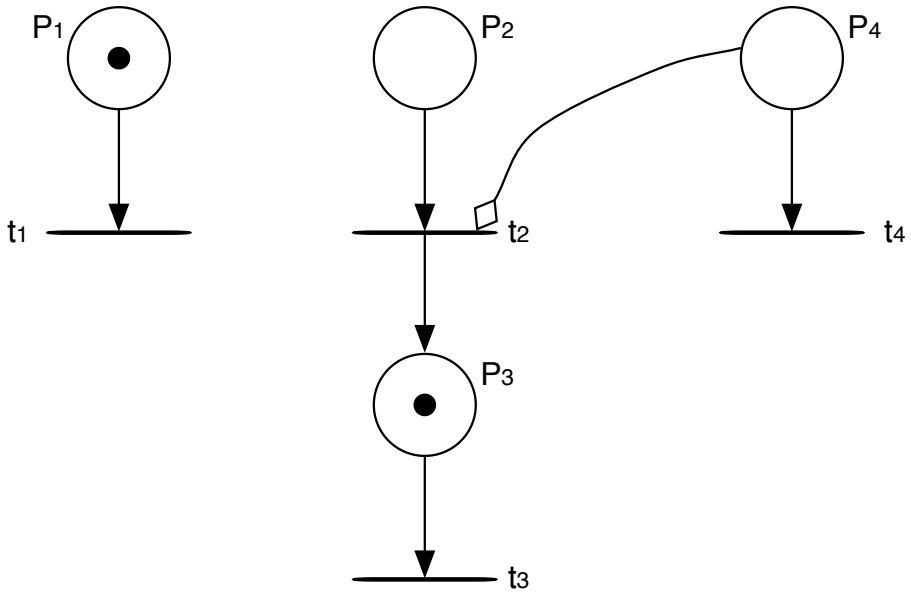


Figure: A Petri net with reset arcs

$$\{P_1, P_2, 5 \times P_4\} \xrightarrow{t_2} \{P_1, P_3\} \xrightarrow{t_1} \dots$$

Petri nets with read arcs

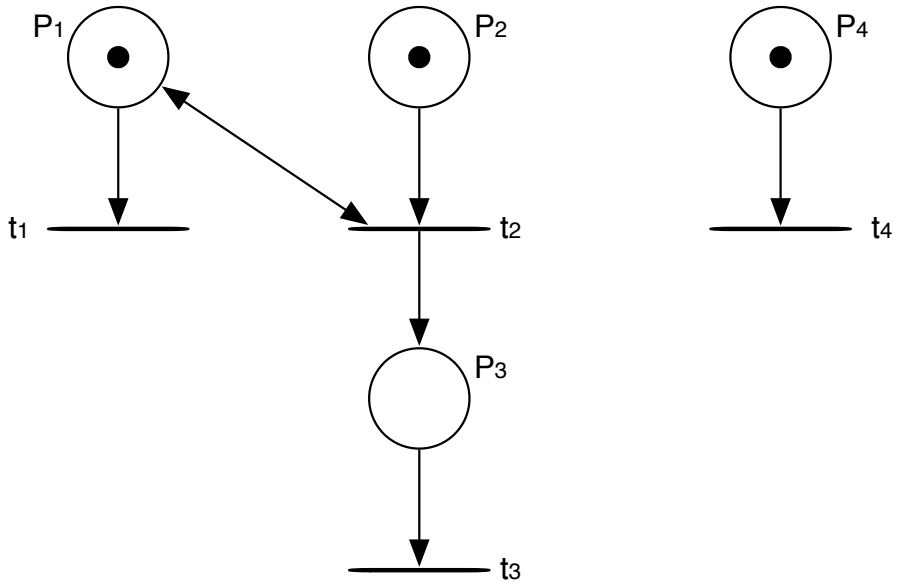


Figure: A Petri net with read arcs

$$\{P_1, P_2, P_4\}$$

Petri nets with read arcs

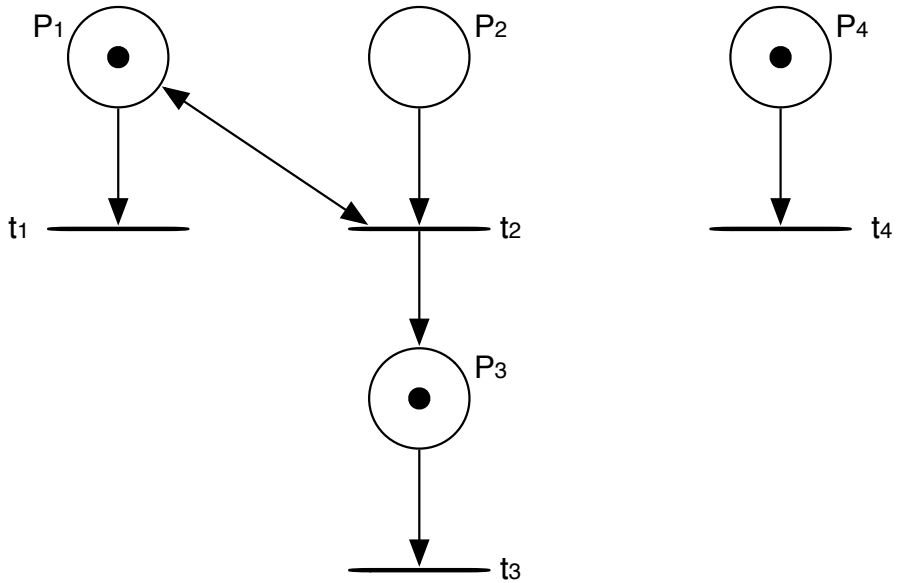


Figure: A Petri net with read arcs

$$\{P_1, P_2, P_4\} \xrightarrow{t_2} \{P_1, P_3, P_4\} \ldots$$

(Logic) Inhibitor Hyperarc Petri nets : AND inhibition

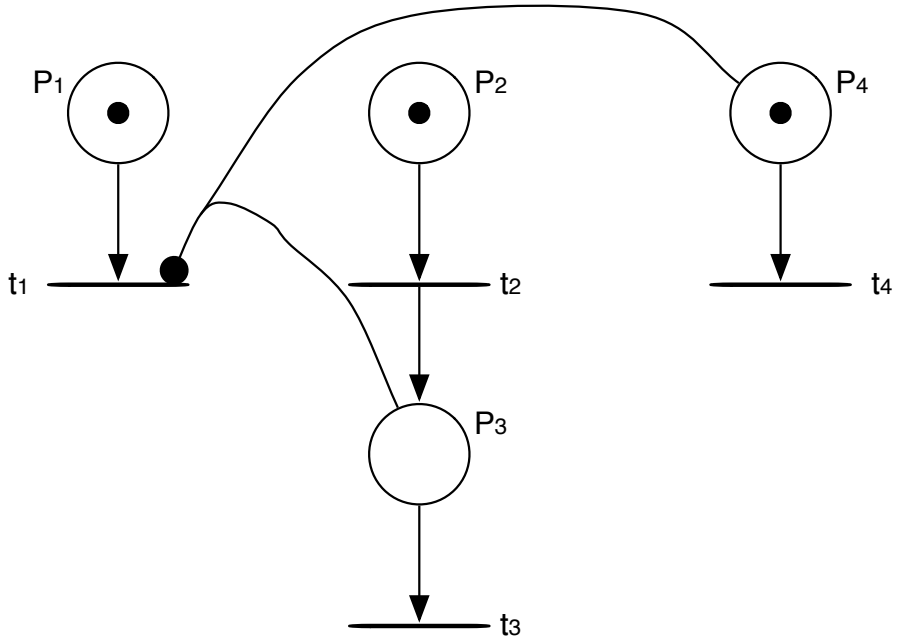


Figure: An inhibitor hyperarc Petri net

$$\{P_1, P_2, P_4\}$$

(Logic) Inhibitor Hyperarc Petri nets : AND inhibition

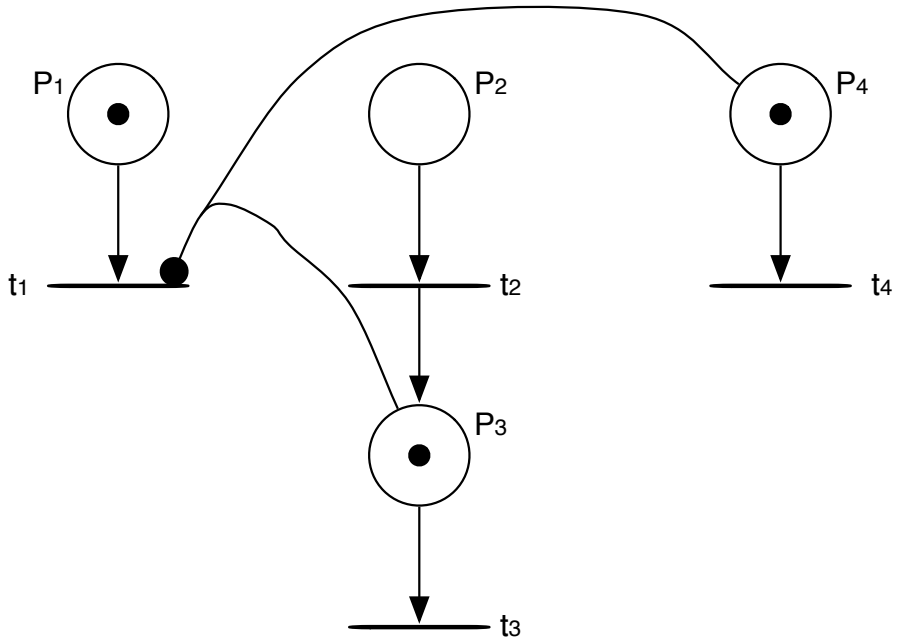


Figure: An inhibitor hyperarc Petri net : t_1 inhibited iff ($M(P_3) \geq 1$ **and** $M(P_4) \geq 1$)

$$\{P_1, P_2, P_4\} \xrightarrow{t_2} \{P_1, P_3, P_4\}$$

(Logic) Inhibitor Hyperarc Petri nets : AND inhibition

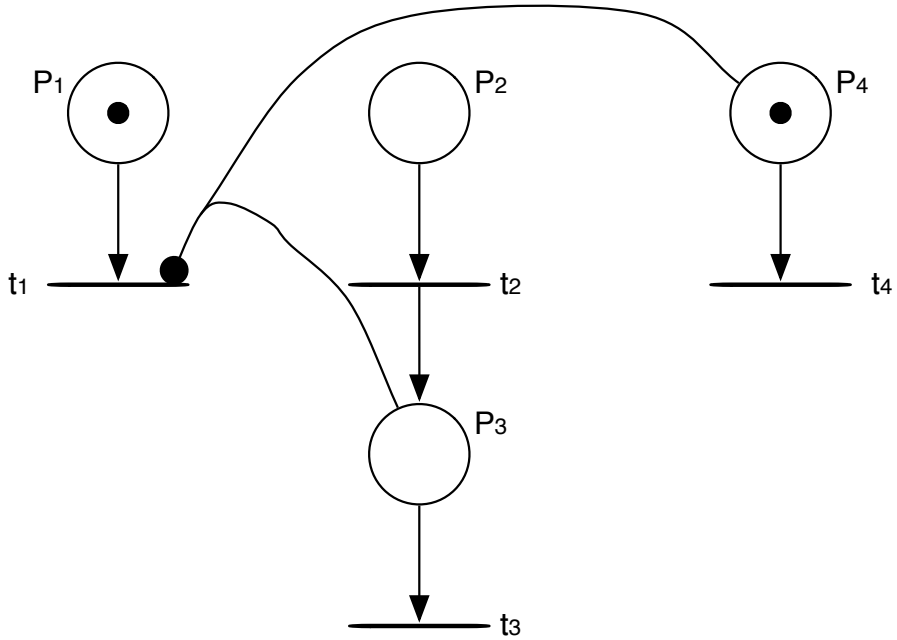


Figure: An inhibitor hyperarc Petri net

$$\{P_1, P_2, P_4\} \xrightarrow{t_2} \{P_1, P_3, P_4\} \xrightarrow{t_3} \{P_1, P_4\} \xrightarrow{t_1} \dots$$

From regulatory networks to Petri nets

Principle

- One place per gene
- Marking: discrete level of concentration

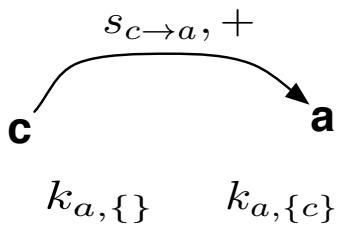


Figure: A simple regulatory network

Critical issues

- How to **test** the concentration level without decrementing it?
- How to model an action that takes place only **below** a given concentration?

$\rightarrow$ Use read and inhibitor (hyper)arcs

From regulatory networks to Petri nets

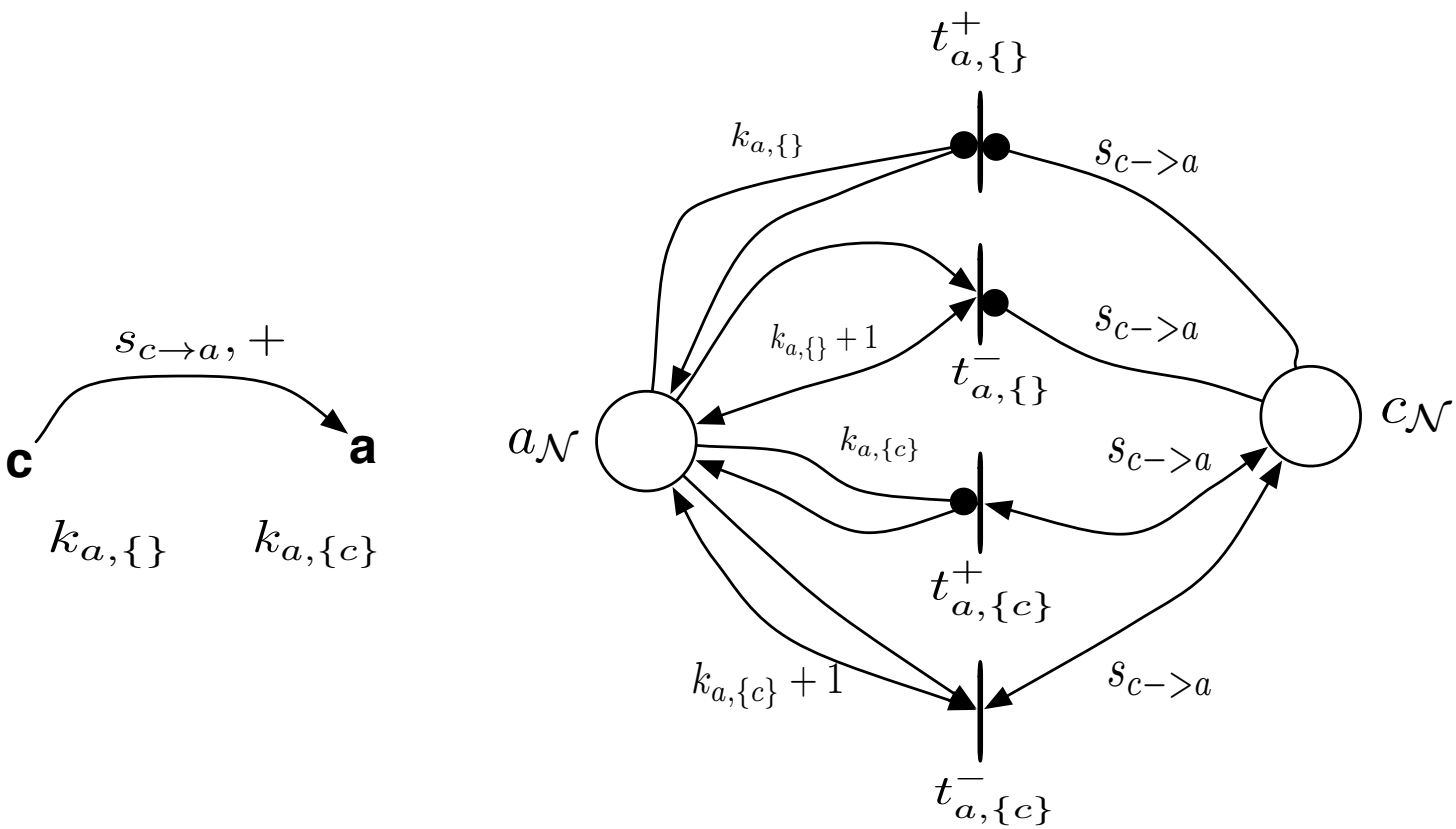


Figure: Translation towards Petri nets

From regulatory networks to Petri nets

Analysis

- Automated translation
- **Bounded** networks $\rightarrow$ reduced cost of read and logic inhibitor hyperarcs

From regulatory networks to time extensions of Petri nets

Principle

- Thinly **discretize** the concentration levels (thus thresholds) of each gene
- Associate the **production and degradation delays** to transitions resulting from the discrete translation

From regulatory networks to time extensions of Petri nets

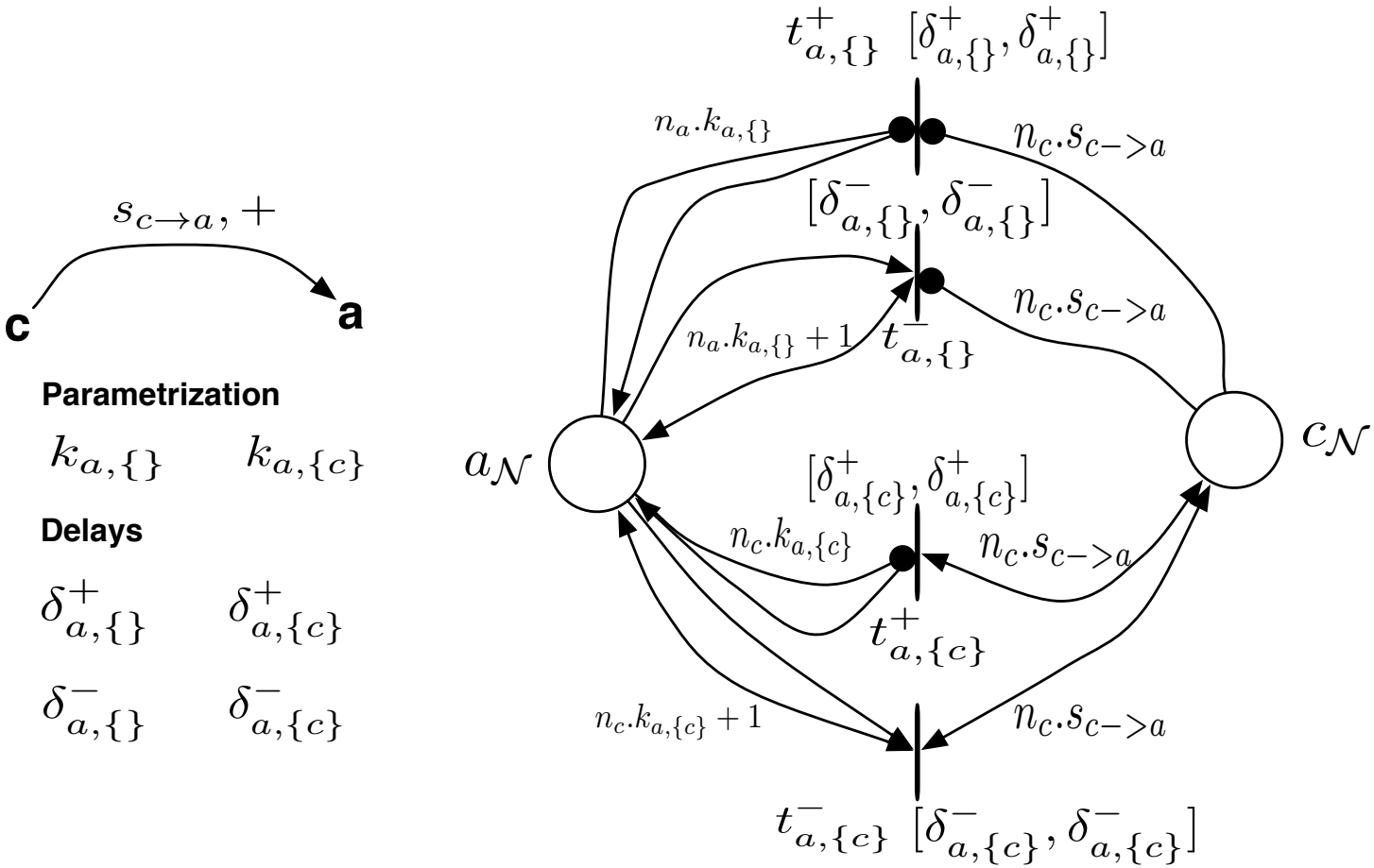


Figure: Translation towards time Petri nets

From regulatory networks to time extensions of Petri nets

Analysis

- Allows the model-checking of (parametric) TCTL formulae
- Possibility to infer the time parameters associated to a transition
- Automate the translation and export to ROMÉO software

Model validation

Objective : formal verification of a model properties

- Model the S system:
  $\rightarrow$ Petri nets, **Time Petri nets**, Stopwatch Petri nets, ...
- Formalize the specification φ :
  $\rightarrow$ observers, timed logics (LTL, CTL, **TCTL**),...
- Does $S \models \varphi$?

Algorithms implemented in ROMÉO in both dense time and discrete-time

Model validation

Objective : formal verification of a model properties

- Model the S system:
  $\rightarrow$ Petri nets, **Time Petri nets**, Stopwatch Petri nets, ...
- Formalize the specification φ :
  $\rightarrow$ observers, timed logics (LTL, CTL, **TCTL**),...
- Does $S \models \varphi$?

Algorithms implemented in ROMÉO in both dense time and discrete-time

Application : p53-MdM2 network

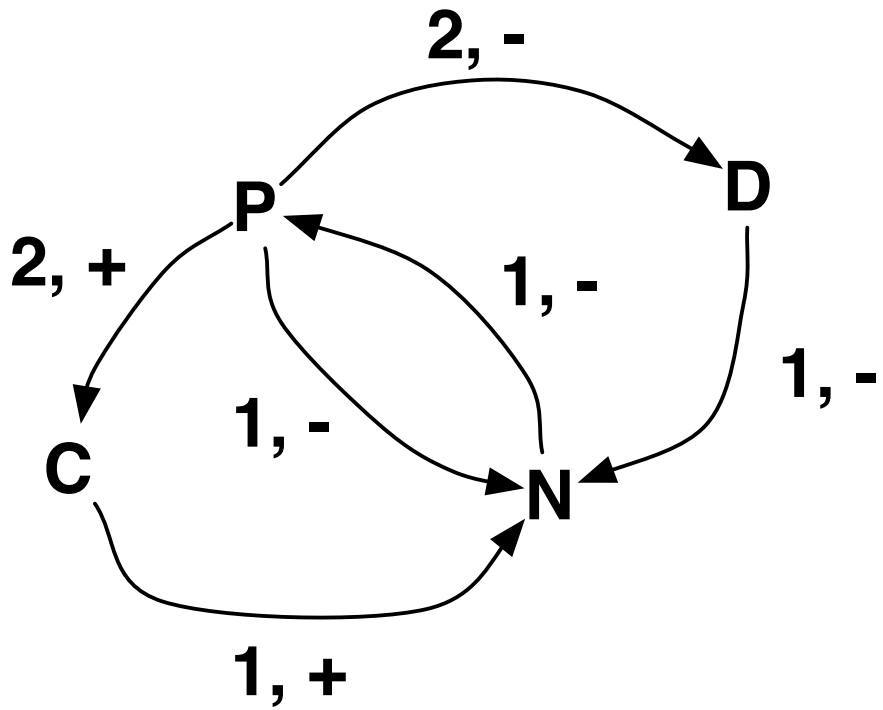


Figure: Network from [WAjK09]

Application : p53-MdM2 network

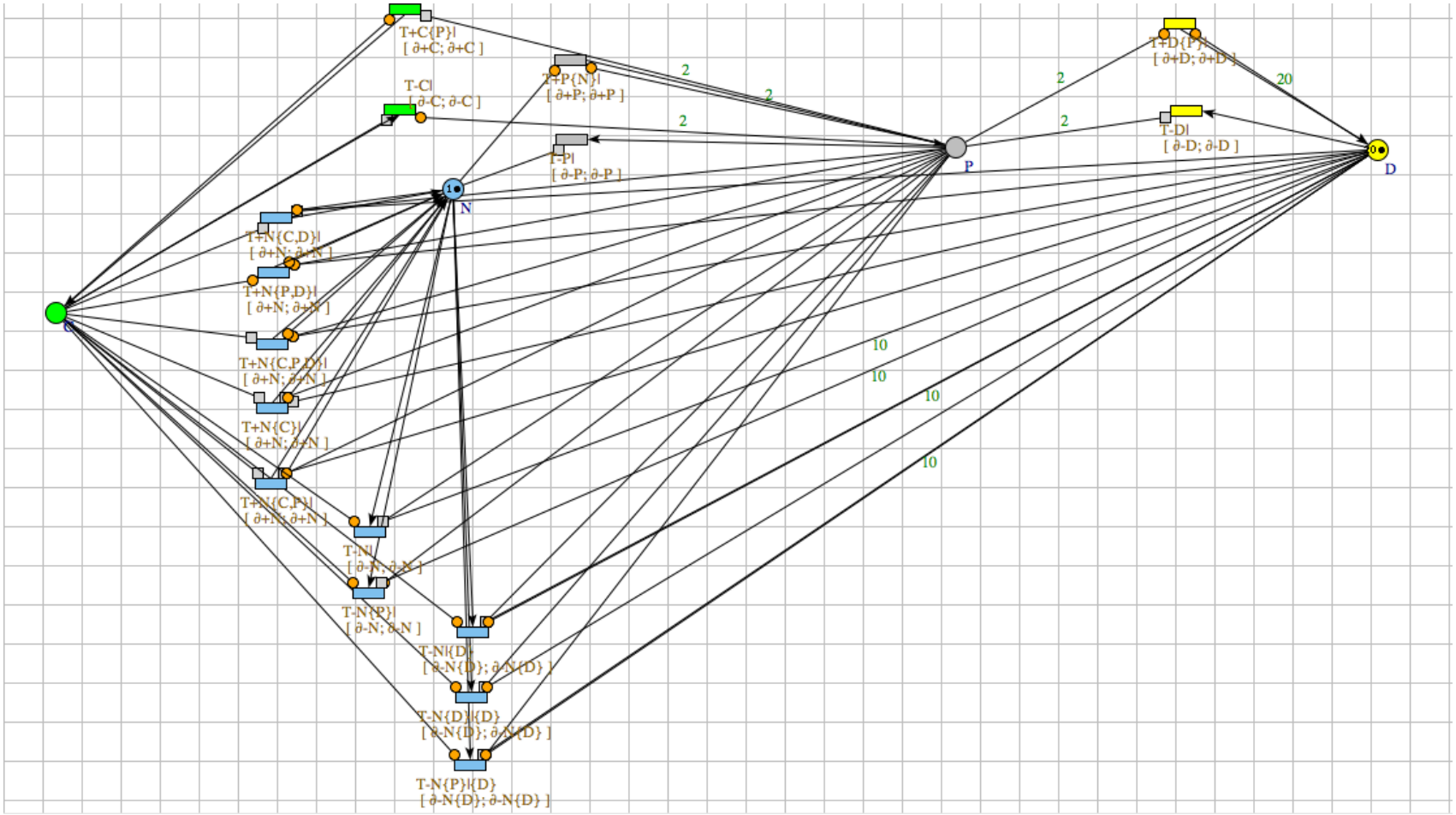


Figure: Translation into time Petri net

Biological analysis

Model validation

- Verify properties (sustained oscillations, damped oscillations, ...)
- Model-checking of TCTL formulae

Delays inference

Model-checking of **parametric** TCTL formulae

Biological analysis

Limits

- **Undecidability** of TCTL model-checking, even for bounded parametric TPN [TLR09]
- State space combinatorial **explosion**
- Limitation in the **size** of the nets and **number** of parameters

Methodology

- Identification of relevant sub-problems
- Progressive inference of time delays

Tools for Interaction Graphs Study

Interaction Graphs [RCB05]

- No positive circuit $\Rightarrow$ only 1 attractor
- No negative circuit $\Rightarrow$ no cyclic attractor
- Positive circuits $\Rightarrow$ criterion for max. number of attractors
- Temporal logics $\Rightarrow$ check properties (needs State Graph)
  - $\rightarrow$ SM-BIONET [KCRB09], ginSIM [CNT12], Biocham [CFS06]
  - $\rightarrow$ Translate models into discrete-event systems and run model-checkers
- Some recent works focus on boolean networks topological fixed points: [PR10]

Tools for Interaction Graphs Study

Interaction Graphs [RCB05]

- No positive circuit $\Rightarrow$ only 1 attractor
- No negative circuit $\Rightarrow$ no cyclic attractor
- Positive circuits $\Rightarrow$ criterion for max. number of attractors
- Temporal logics $\Rightarrow$ check properties (**needs State Graph**)
  - $\rightarrow$ SM-BIONET [KCRB09], ginSIM [CNT12], Biocham [CFS06]
  - $\rightarrow$ Translate models into discrete-event systems and run model-checkers
- Some recent works focus on boolean networks topological fixed points: [PR10]

Problem

Combinatorial explosion when computing the State Graph

$\rightarrow$ Need for static analysis $\rightarrow$ introduction of the Process Hitting

Overview

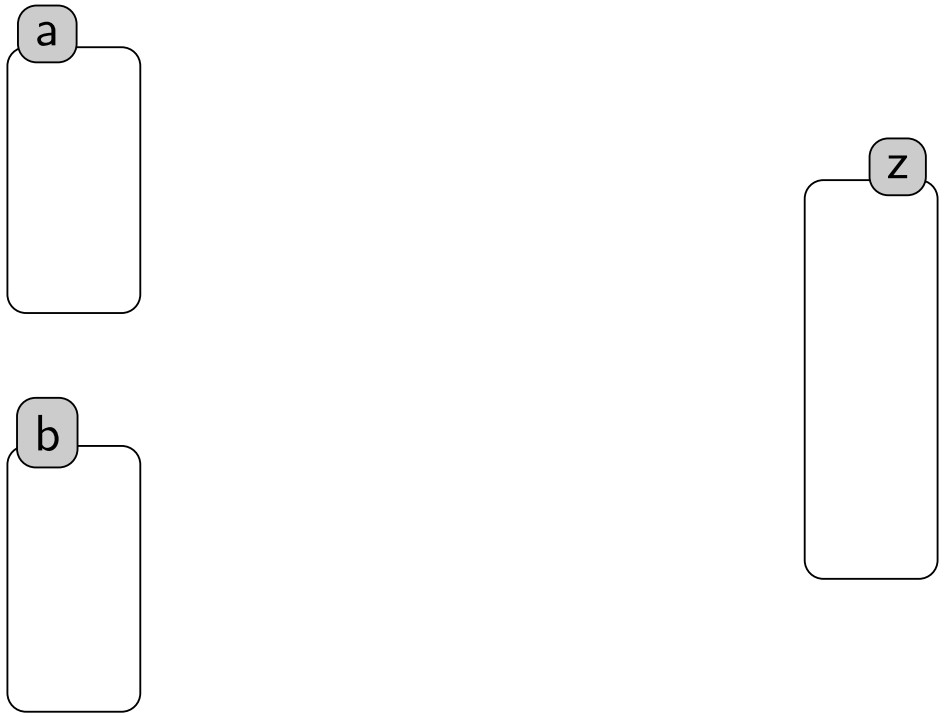
Intuitive principle of the Process Hitting framework

Process	=	component a at level i
Interaction	=	a at level i makes b at level j increase or decrease to level k
denoted		$a_i \rightarrow b_j \uparrow b_k$ (hit and bounce)

Definition (Interaction and Retroaction)

Interaction $(a_i \rightarrow b_j \uparrow b_k)$, where a_i is the level of a process a and $b_j \neq b_k$,
Retroaction $(a_i \rightarrow a_i \uparrow a_k)$: when $a_i = b_j$.

The Process Hitting modeling



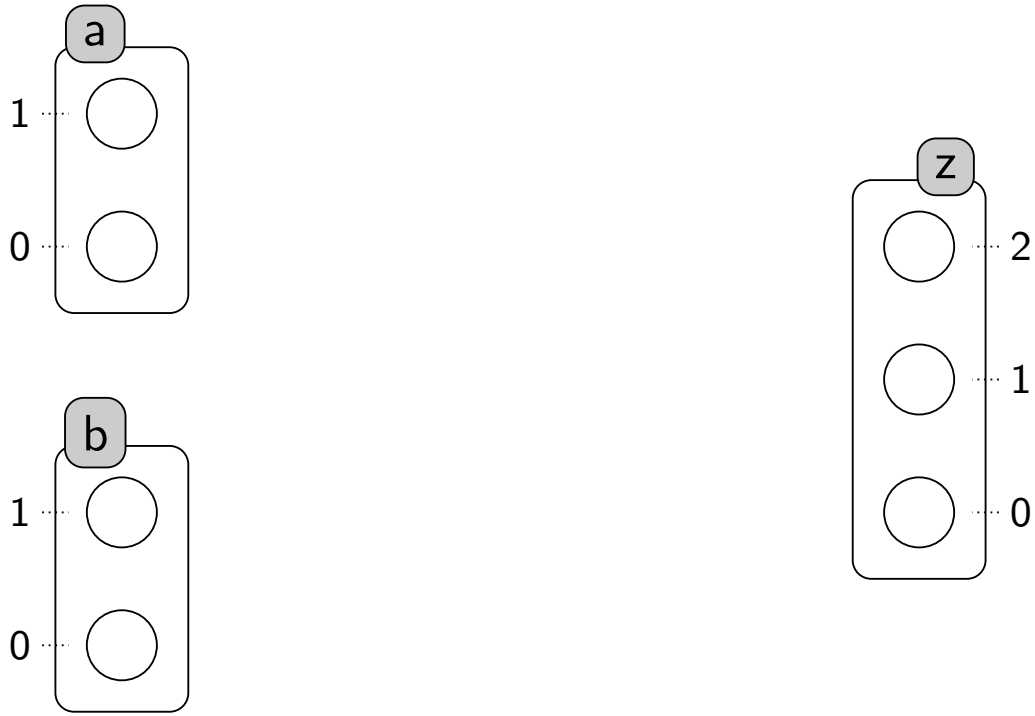
Sorts: components a, b, z

Processes: local states / levels of expression z_0, z_1, z_2

States: sets of active processes

Actions: dynamics $b_1 \longrightarrow z_0 \restriction z_1$, $a_0 \longrightarrow a_0 \restriction a_1$, $a_1 \longrightarrow z_1 \restriction z_2$

The Process Hitting modeling



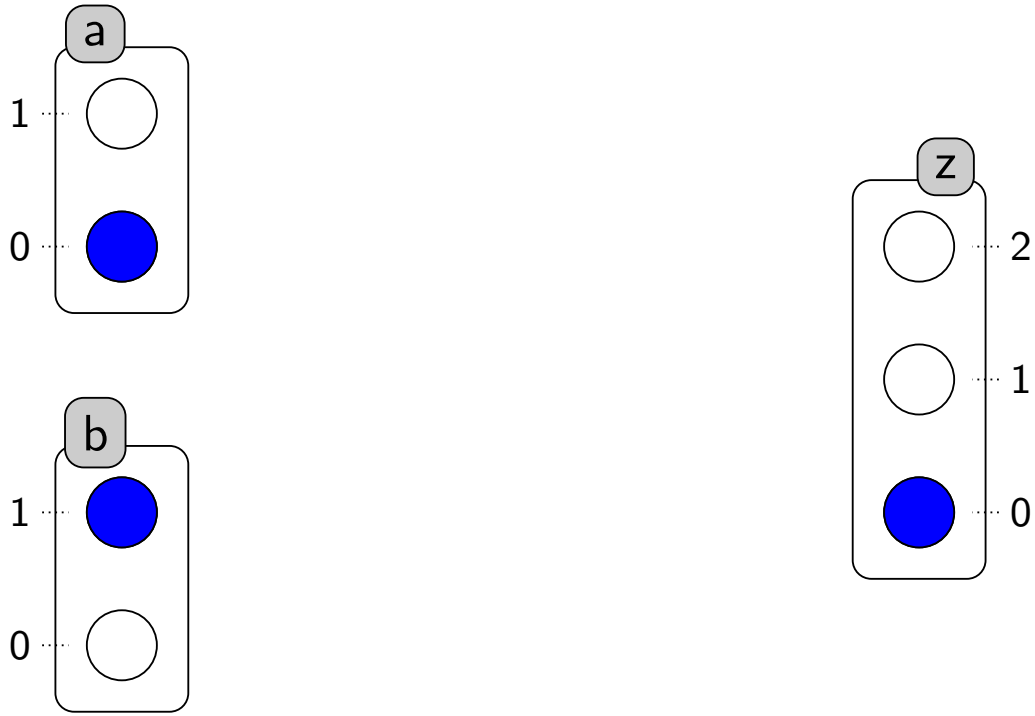
Sorts: components a, b, z

Processes: local states / levels of expression z_0, z_1, z_2

States: sets of active processes

Actions: dynamics $b_1 \to z_0 \Rsh z_1$, $a_0 \to a_0 \Rsh a_1$, $a_1 \to z_1 \Rsh z_2$

The Process Hitting modeling



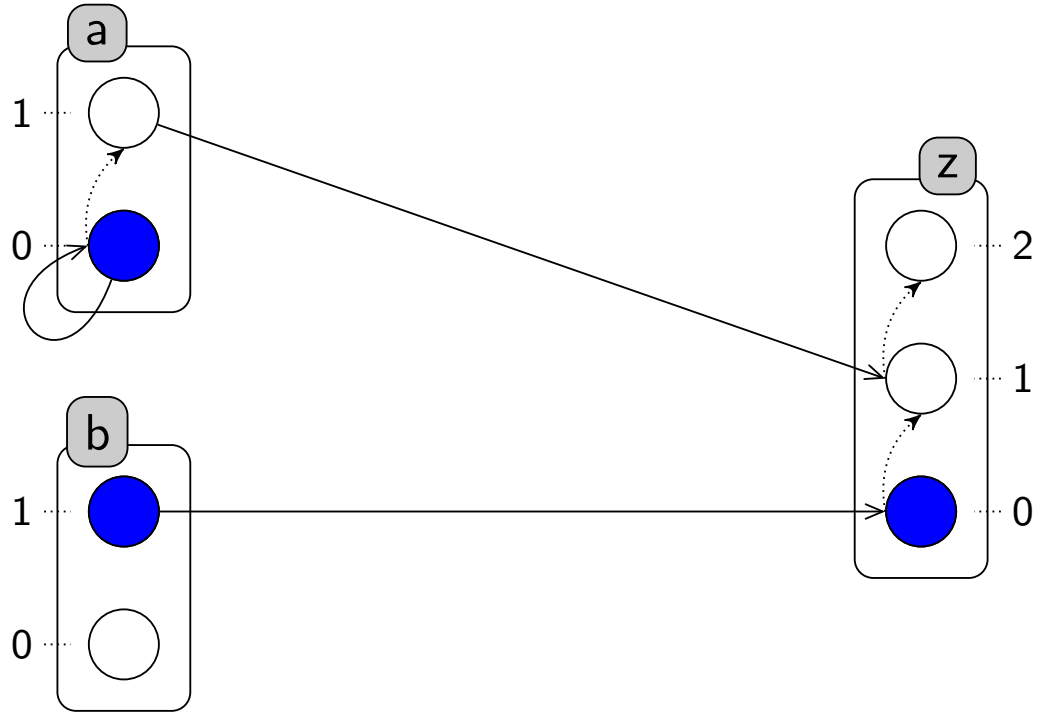
Sorts: components a, b, z

Processes: local states / levels of expression z_0, z_1, z_2

States: sets of active processes $\langle a_0, b_1, z_0 \rangle$

Actions: dynamics $b_1 \longrightarrow z_0 \restriction z_1$, $a_0 \longrightarrow a_0 \restriction a_1$, $a_1 \longrightarrow z_1 \restriction z_2$

The Process Hitting modeling



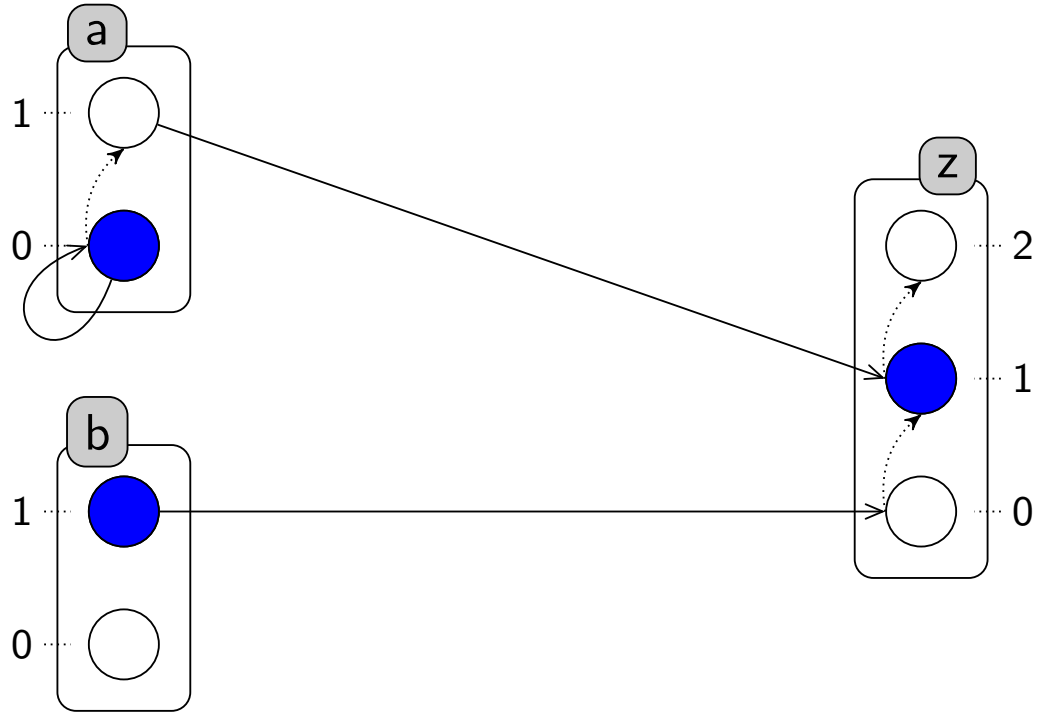
Sorts: components a, b, z

Processes: local states / levels of expression z_0, z_1, z_2

States: sets of active processes $\langle a_0, b_1, z_0 \rangle$

Actions: dynamics $\underline{b_1 \to z_0 \uparrow z_1}$, $\underline{a_0 \to a_0 \uparrow a_1}$, $a_1 \to z_1 \uparrow z_2$

The Process Hitting modeling



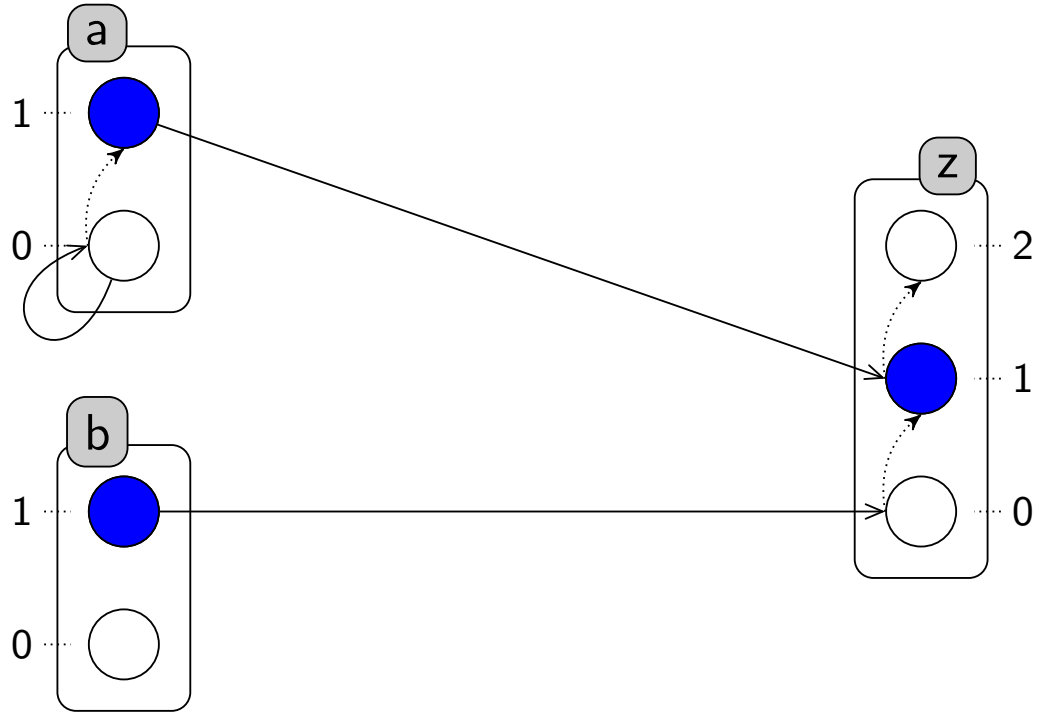
Sorts: components $a,\ b,\ z$

Processes: local states / levels of expression $z_0,\ z_1,\ z_2$

States: sets of active processes $\langle a_0, b_1, z_1 \rangle$

Actions: dynamics $b_1 \to z_0 \Rsh z_1,\ \underline{a_0 \to a_0 \Rsh a_1},\ a_1 \to z_1 \Rsh z_2$

The Process Hitting modeling



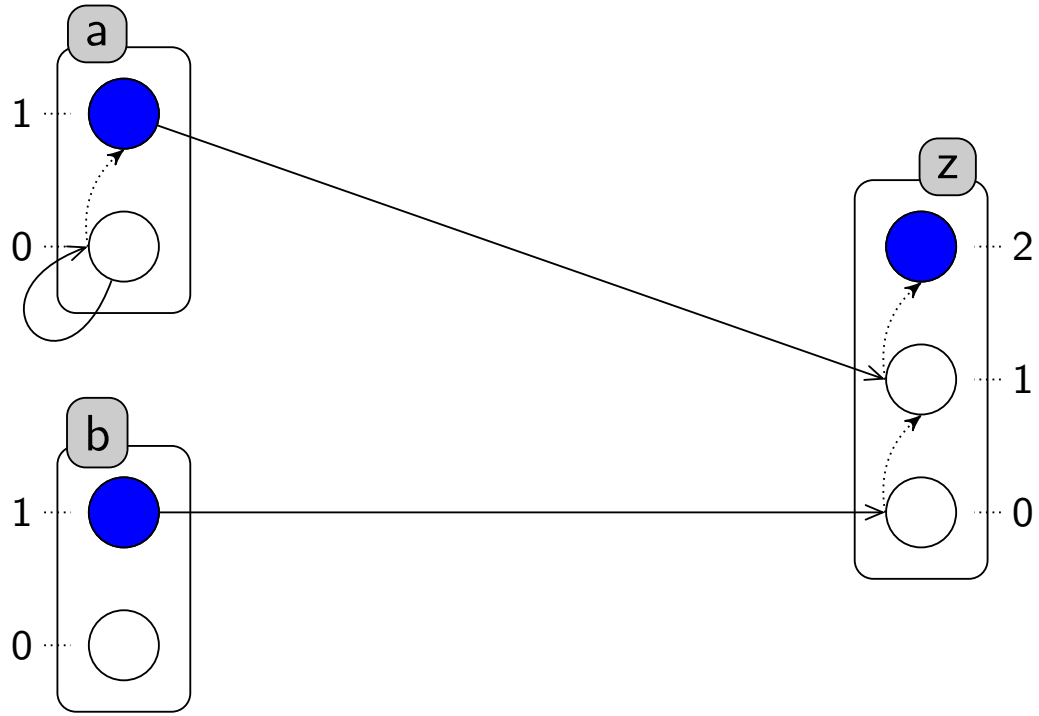
Sorts: components $a,\ b,\ z$

Processes: local states / levels of expression $z_0,\ z_1,\ z_2$

States: sets of active processes $\langle a_1, b_1, z_1 \rangle$

Actions: dynamics $b_1 \to z_0 \Rsh z_1,\ a_0 \to a_0 \Rsh a_1,\ a_1 \to z_1 \Rsh z_2$

The Process Hitting modeling



Sorts: components $a,\ b,\ z$

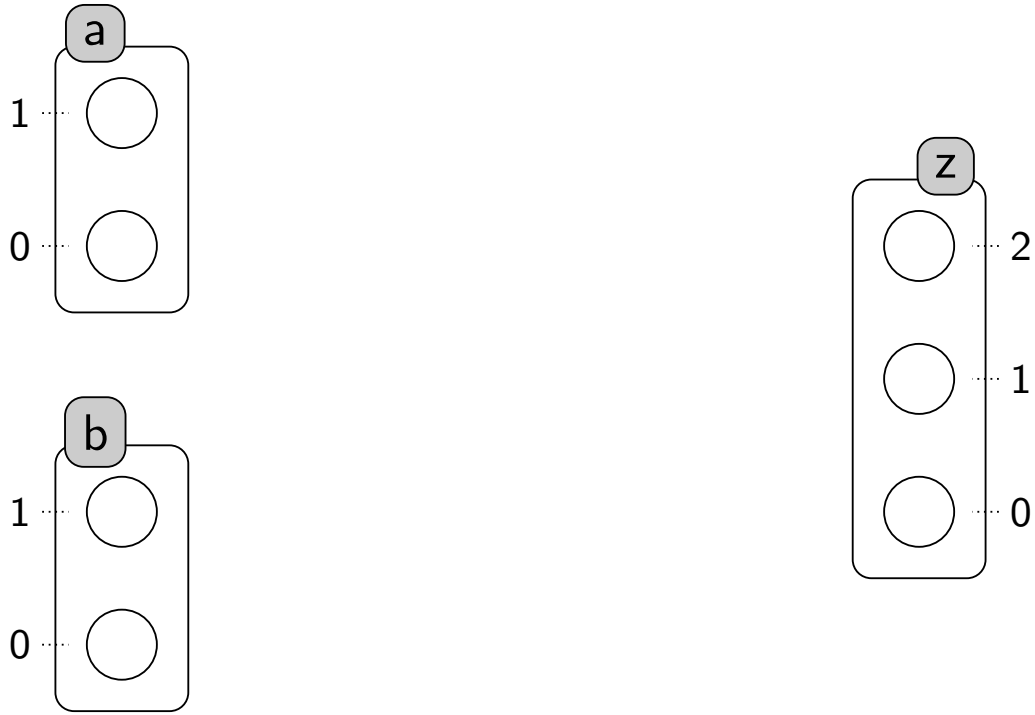
Processes: local states / levels of expression $z_0,\ z_1,\ z_2$

States: sets of active processes $\langle a_1, b_1, z_2 \rangle$

Actions: dynamics $b_1 \to z_0 \Rsh z_1,\ a_0 \to a_0 \Rsh a_1,\ a_1 \to z_1 \Rsh z_2$

Adding cooperations

[PMR12]



How to introduce some **cooperation** between sorts? $a_1 \wedge b_0 \rightarrow z_1 \Rsh z_2$

Solution: a **cooperative sort** ab

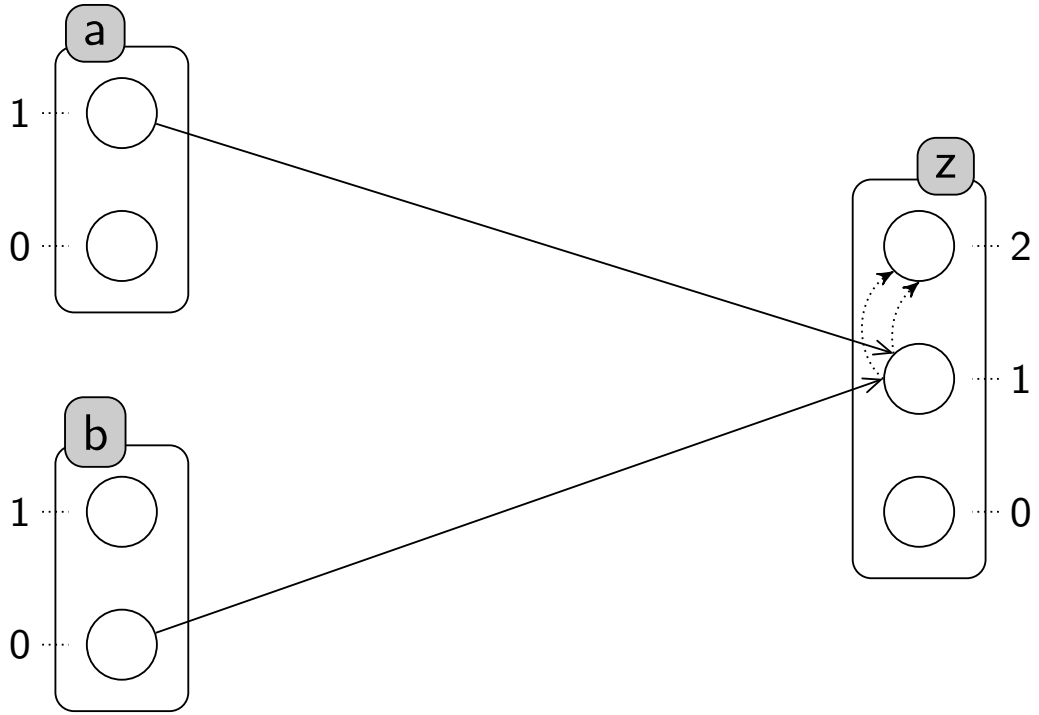
Constraint: each configuration is represented by one process

$\langle a_1, b_0 \rangle \Rightarrow ab_{10}$

Advantage: regular sort; drawbacks: complexity, temporal shift

Adding cooperations

[PMR12]



How to introduce some **cooperation** between sorts? $a_1 \wedge b_0 \rightarrow z_1 \Uparrow z_2$

Solution: a **cooperative sort** ab

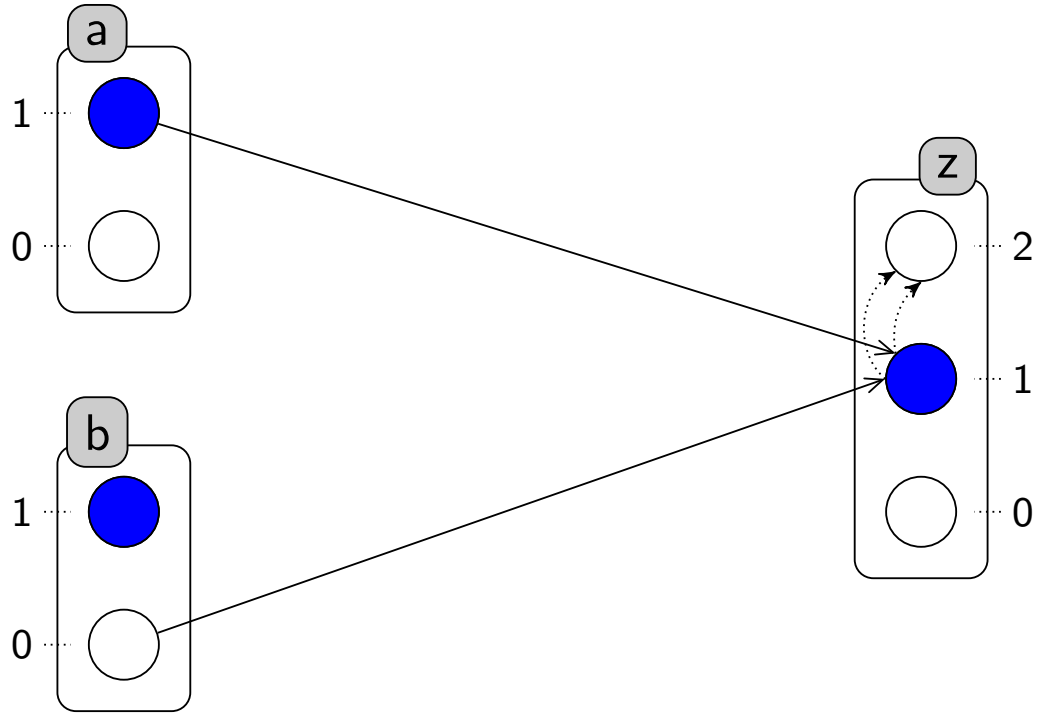
Constraint: each configuration is represented by one process

$\langle a_1, b_0 \rangle \Rightarrow ab_{10}$

Advantage: regular sort; drawbacks: complexity, temporal shift

Adding cooperations

[PMR12]



How to introduce some **cooperation** between sorts? $a_1 \wedge b_0 \rightarrow z_1 \uparrow z_2$

Solution: a **cooperative sort** ab

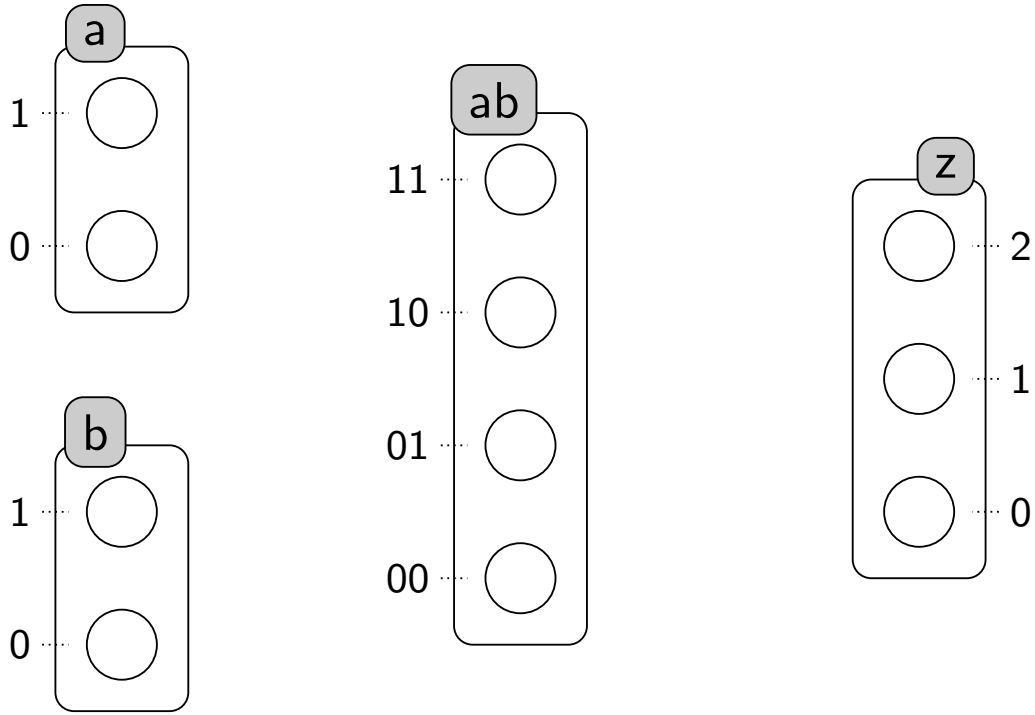
Constraint: each configuration is represented by one process

$\langle a_1, b_0 \rangle \Rightarrow ab_{10}$

Advantage: regular sort; drawbacks: complexity, temporal shift

Adding cooperations

[PMR12]



How to introduce some **cooperation** between sorts? $a_1 \wedge b_0 \rightarrow z_1 \uparrow z_2$

Solution: a **cooperative sort** ab

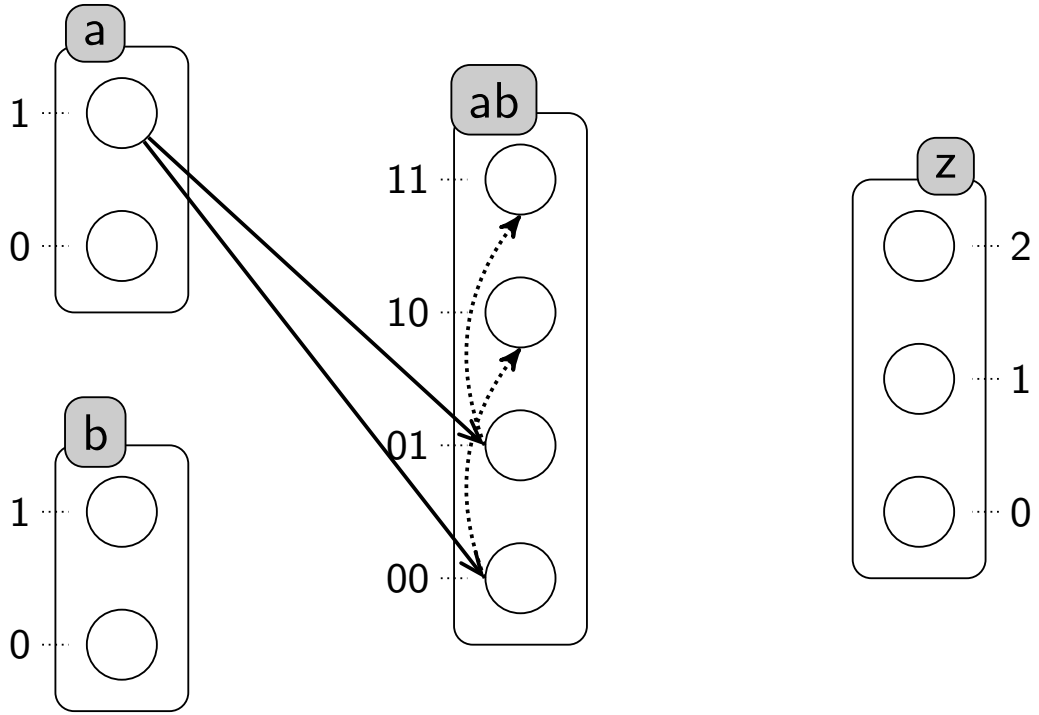
Constraint: each configuration is represented by one process

$\langle a_1, b_0 \rangle \Rightarrow ab_{10}$

Advantage: regular sort; drawbacks: complexity, temporal shift

Adding cooperations

[PMR12]



How to introduce some **cooperation** between sorts? $a_1 \wedge b_0 \rightarrow z_1 \Rsh z_2$

Solution: a **cooperative sort** ab

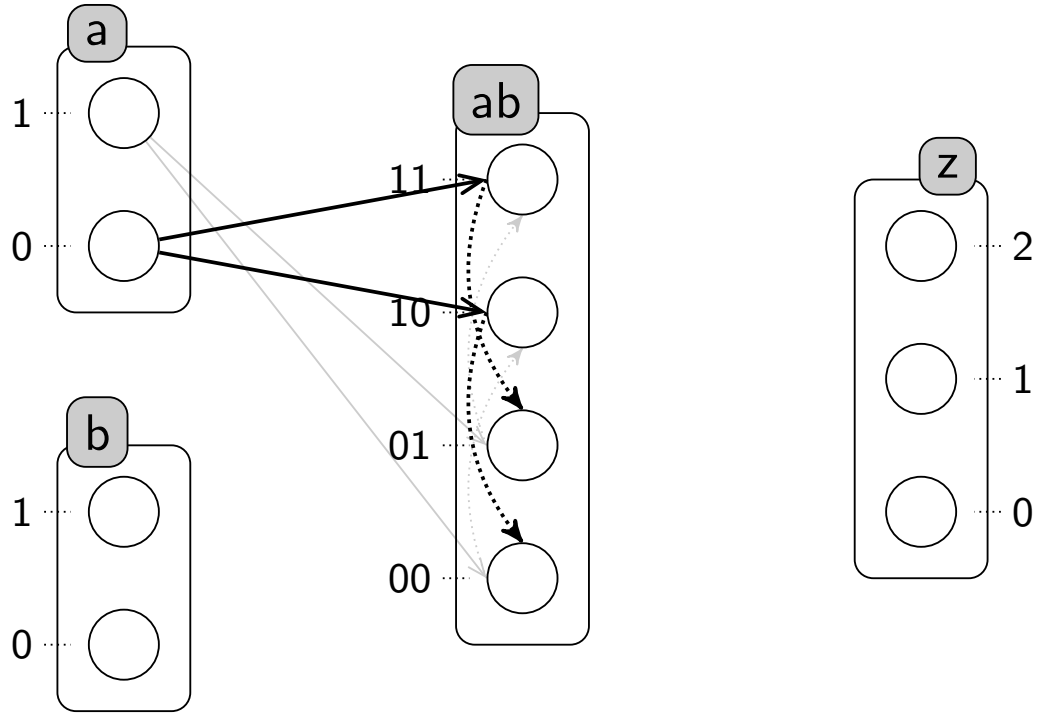
Constraint: each configuration is represented by one process

$\langle a_1, b_0 \rangle \Rightarrow ab_{10}$

Advantage: regular sort; drawbacks: complexity, temporal shift

Adding cooperations

[PMR12]



How to introduce some **cooperation** between sorts? $a_1 \wedge b_0 \rightarrow z_1 \Rsh z_2$

Solution: a **cooperative sort** ab

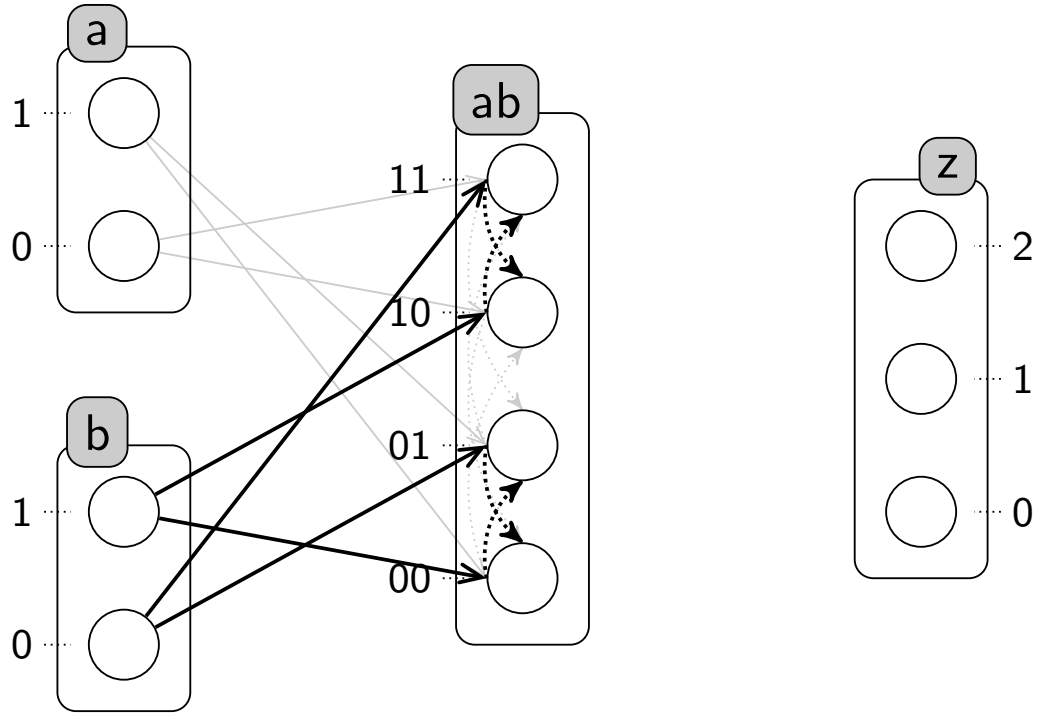
Constraint: each configuration is represented by one process

$\langle a_1, b_0 \rangle \Rightarrow ab_{10}$

Advantage: regular sort; drawbacks: complexity, temporal shift

Adding cooperations

[PMR12]



How to introduce some **cooperation** between sorts? $a_1 \wedge b_0 \rightarrow z_1 \Rsh z_2$

Solution: a **cooperative sort** ab

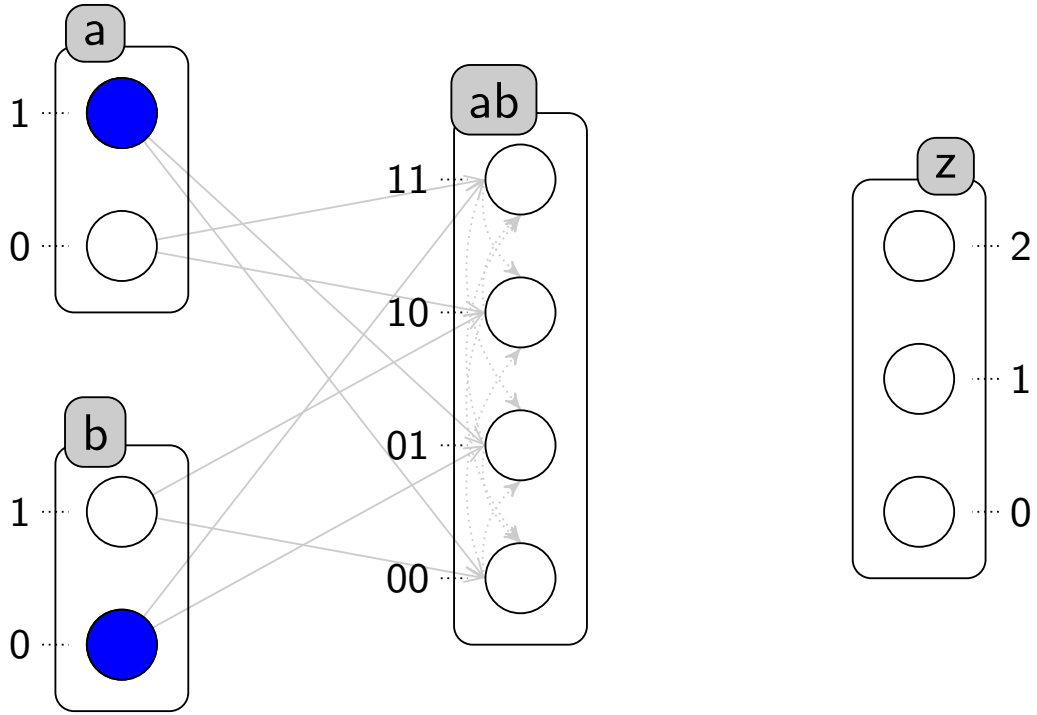
Constraint: each configuration is represented by one process

$\langle a_1, b_0 \rangle \Rightarrow ab_{10}$

Advantage: regular sort; drawbacks: complexity, temporal shift

Adding cooperations

[PMR12]



How to introduce some **cooperation** between sorts? $a_1 \wedge b_0 \rightarrow z_1 \uparrow z_2$

Solution: a **cooperative sort** ab

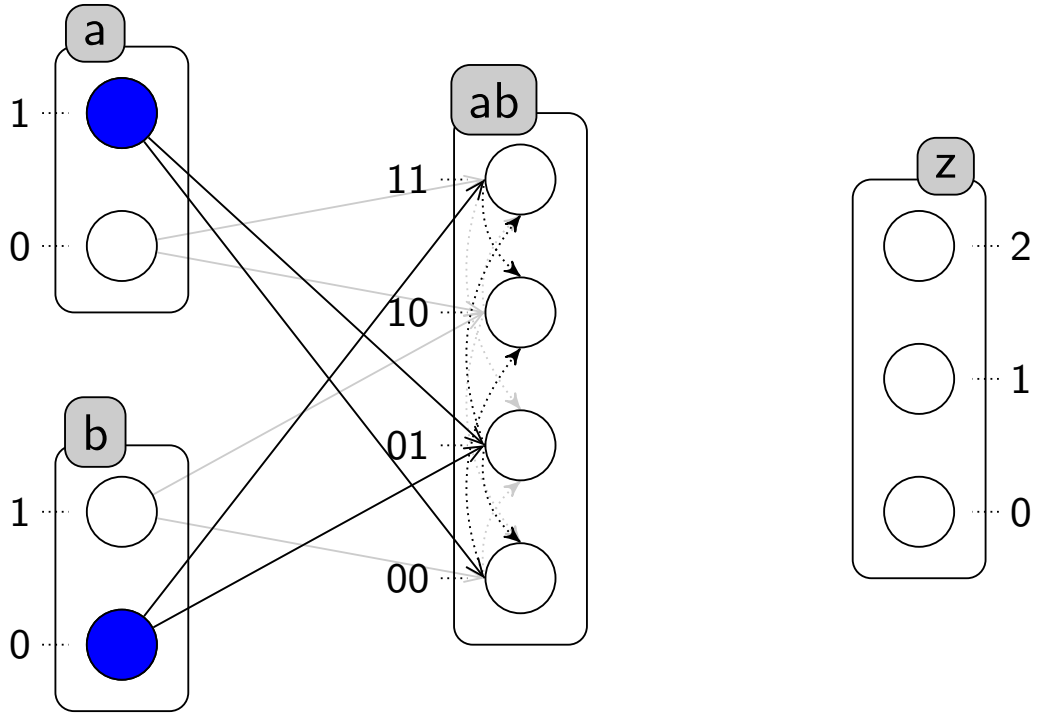
Constraint: each configuration is represented by one process

$\langle a_1, b_0 \rangle \Rightarrow ab_{10}$

Advantage: regular sort; drawbacks: complexity, temporal shift

Adding cooperations

[PMR12]



How to introduce some **cooperation** between sorts? $a_1 \wedge b_0 \rightarrow z_1 \Rsh z_2$

Solution: a **cooperative sort** ab

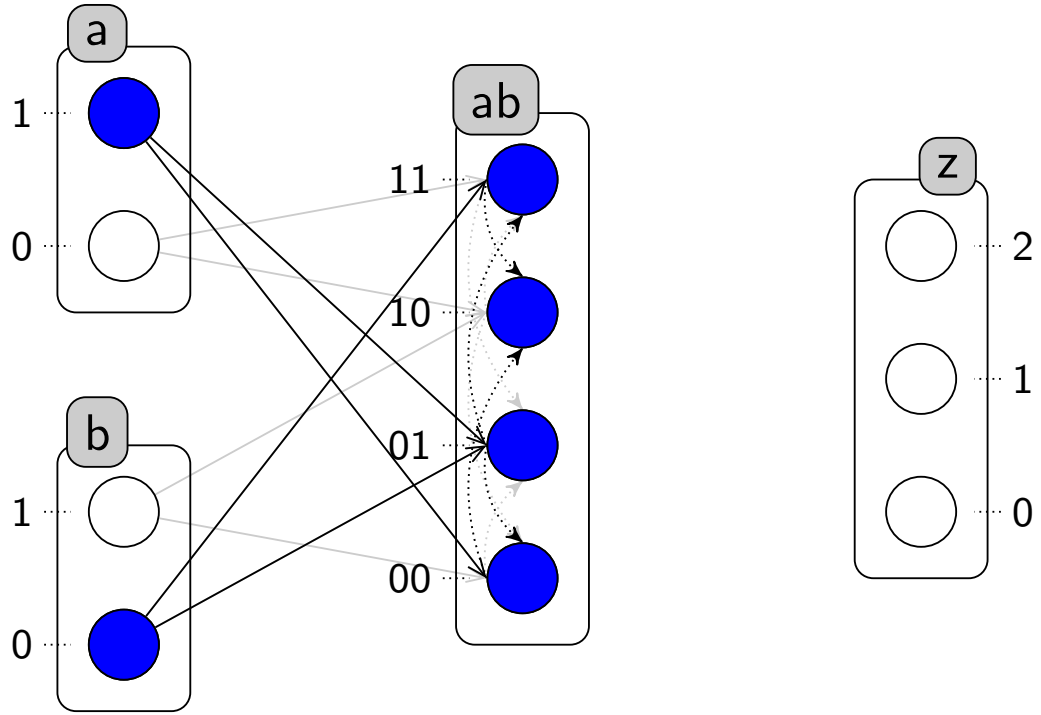
Constraint: each configuration is represented by one process

$\langle a_1, b_0 \rangle \Rightarrow ab_{10}$

Advantage: regular sort; drawbacks: complexity, temporal shift

Adding cooperations

[PMR12]



How to introduce some **cooperation** between sorts? $a_1 \wedge b_0 \rightarrow z_1 \Rsh z_2$

Solution: a **cooperative sort** ab

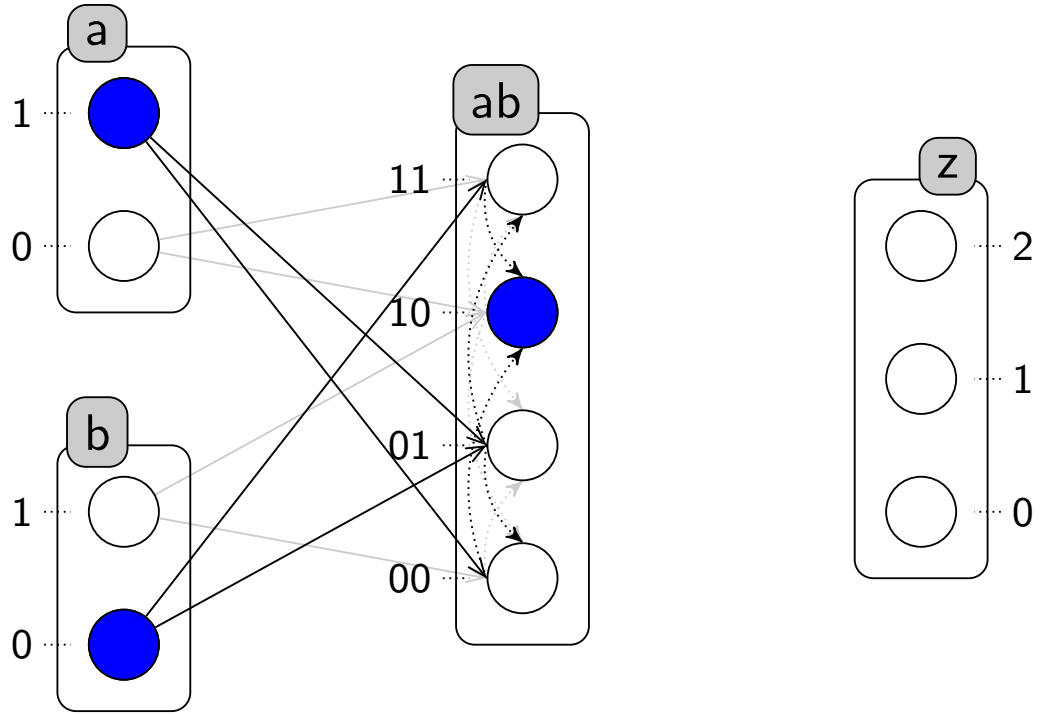
Constraint: each configuration is represented by one process

$\langle a_1, b_0 \rangle \Rightarrow ab_{10}$

Advantage: regular sort; drawbacks: complexity, temporal shift

Adding cooperations

[PMR12]



How to introduce some **cooperation** between sorts? $a_1 \wedge b_0 \rightarrow z_1 \uparrow z_2$

Solution: a **cooperative sort** ab

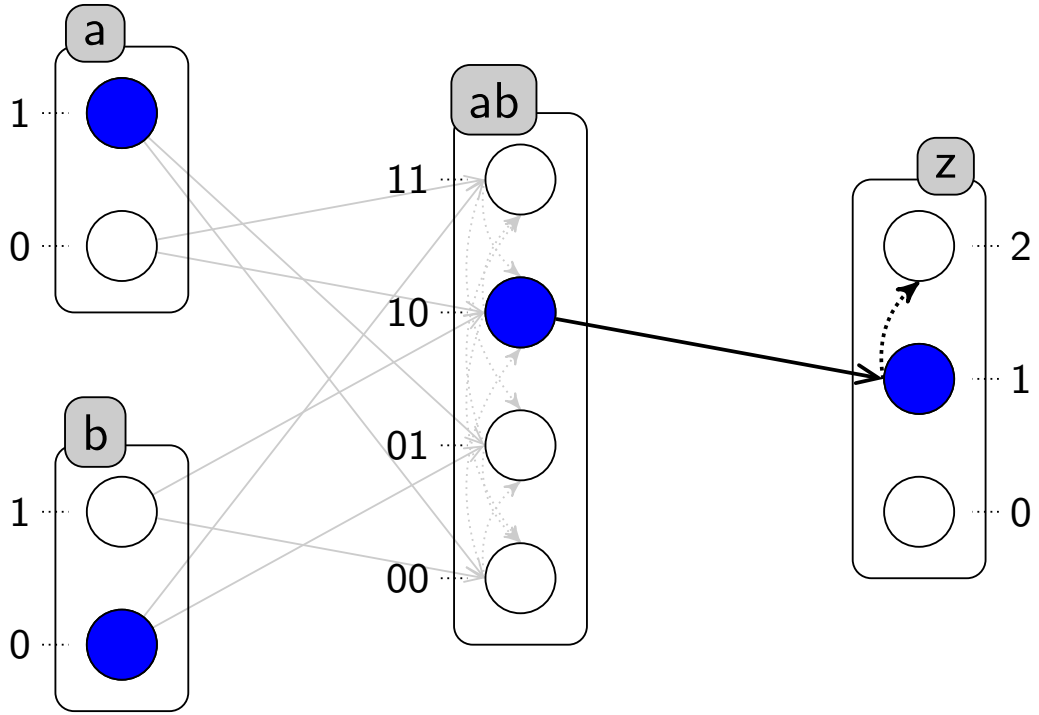
Constraint: each configuration is represented by one process

$\langle a_1, b_0 \rangle \Rightarrow ab_{10}$

Advantage: regular sort; drawbacks: complexity, temporal shift

Adding cooperations

[PMR12]



How to introduce some **cooperation** between sorts? $a_1 \wedge b_0 \rightarrow z_1 \uparrow z_2$

Solution: a **cooperative sort** ab to express $a_1 \wedge b_0$

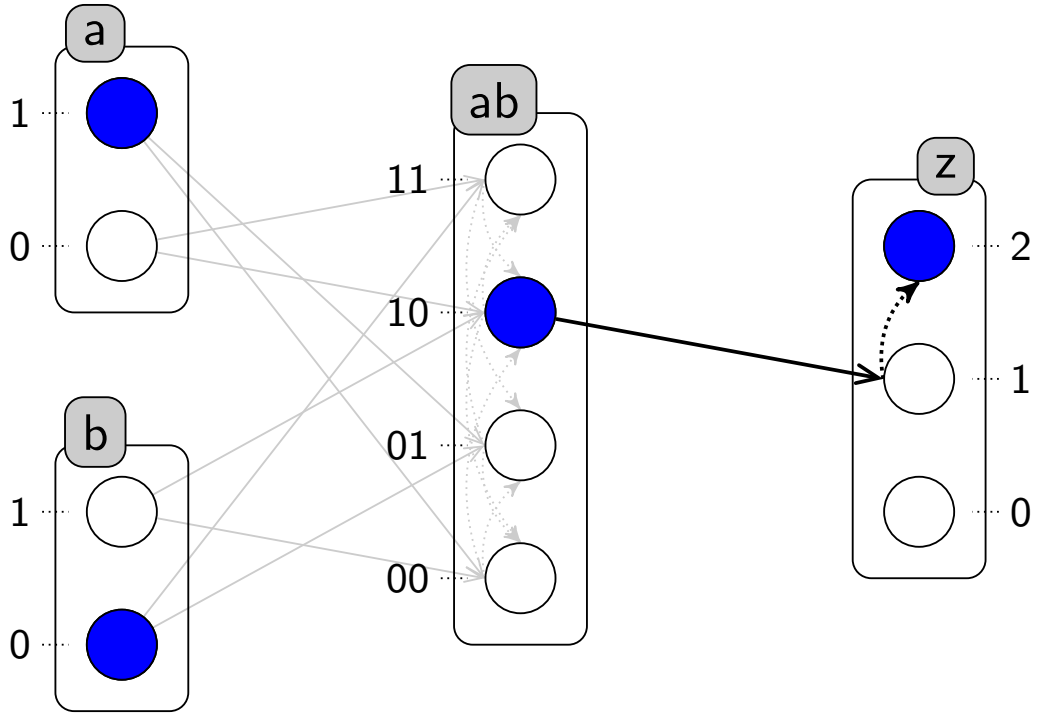
Constraint: each configuration is represented by one process

$\langle a_1, b_0 \rangle \Rightarrow ab_{10}$

Advantage: regular sort; drawbacks: complexity, temporal shift

Adding cooperations

[PMR12]



How to introduce some **cooperation** between sorts? $a_1 \wedge b_0 \rightarrow z_1 \uparrow z_2$

Solution: a **cooperative sort** ab to express $a_1 \wedge b_0$

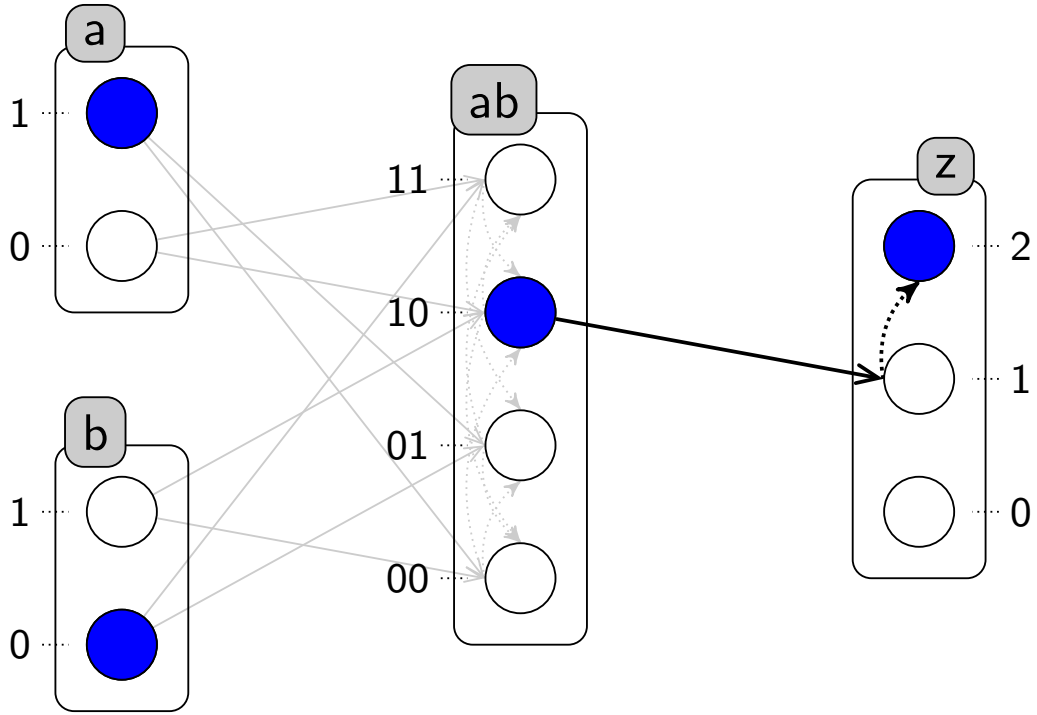
Constraint: each configuration is represented by one process

$\langle a_1, b_0 \rangle \Rightarrow ab_{10}$

Advantage: regular sort; drawbacks: complexity, temporal shift

Adding cooperations

[PMR12]



How to introduce some **cooperation** between sorts? $a_1 \wedge b_0 \rightarrow z_1 \uparrow z_2$

Solution: a **cooperative sort** ab to express $a_1 \wedge b_0$

Constraint: each configuration is represented by one process

$\langle a_1, b_0 \rangle \Rightarrow ab_{10}$

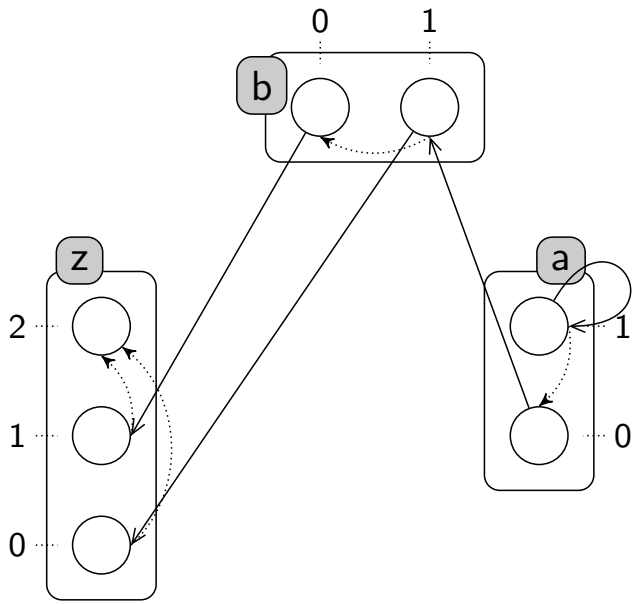
Advantage: regular sort; drawbacks: complexity, temporal shift

Static Analysis: Fixed Points

[PMR11a]

Fixed point = state where no action can be fired

→ avoid couples of processes bounded by an action



Exponential complexity w.r.t. the number of sorts

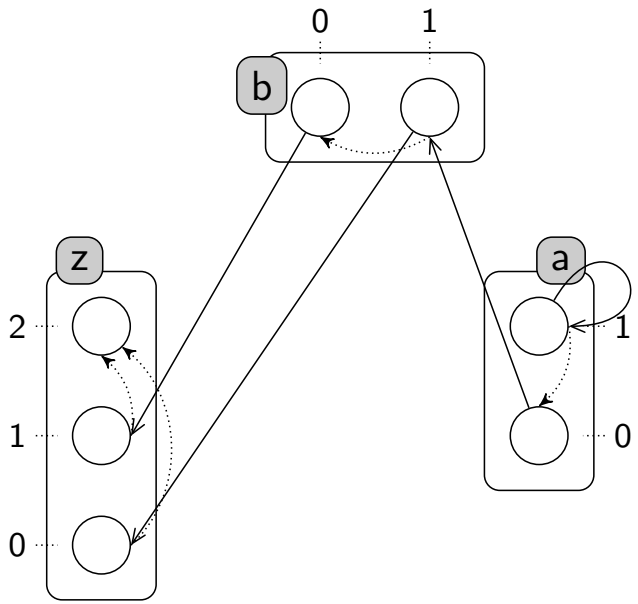
Static Analysis: Fixed Points

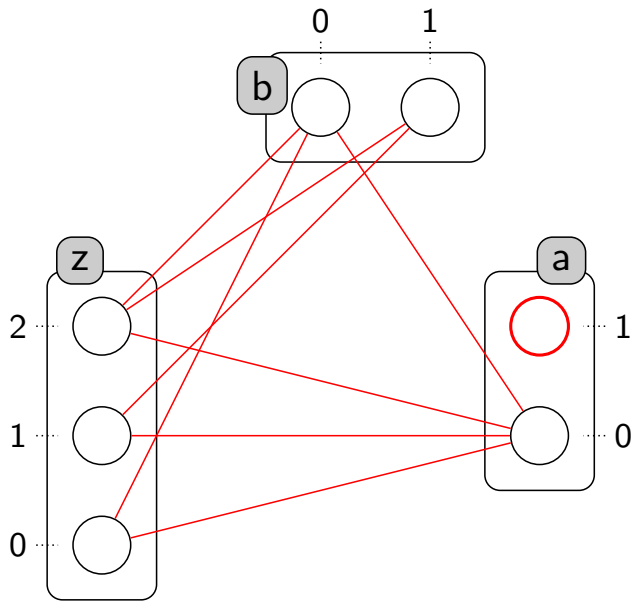
[PMR11a]

Fixed point = state where no action can be fired

→ avoid couples of processes bounded by an action

→ Hitless Graph





Exponential complexity w.r.t. the number of sorts

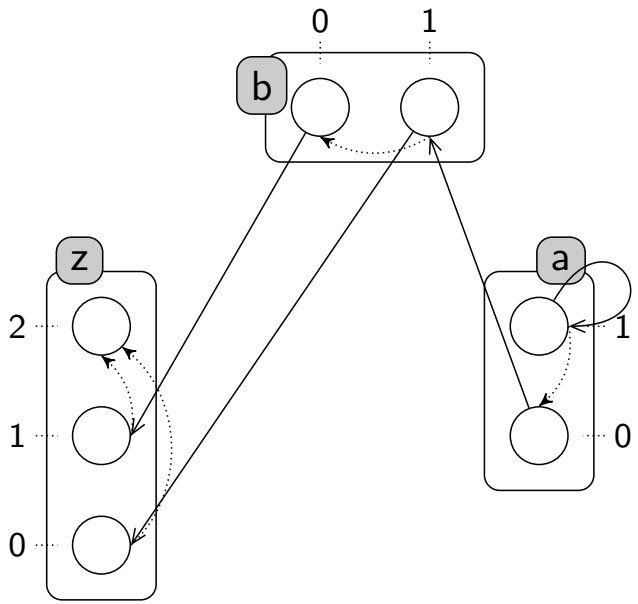
Static Analysis: Fixed Points

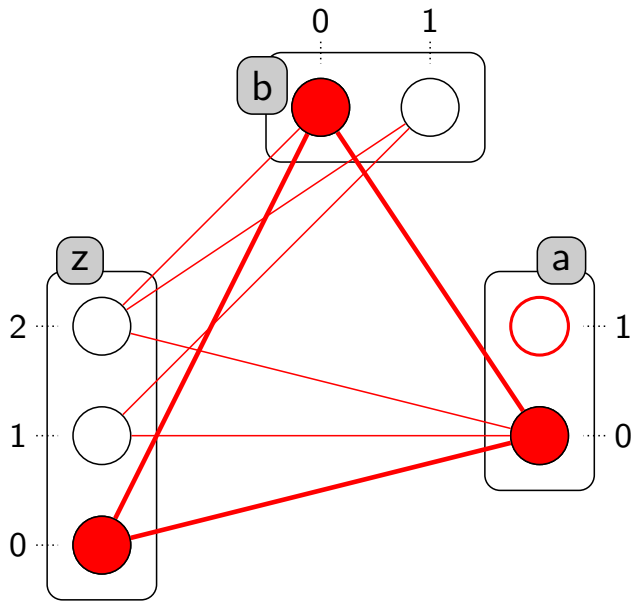
[PMR11a]

Fixed point = state where no action can be fired

→ avoid couples of processes bounded by an action

→ Hitless Graph → **n-cliques** = fixed points





Exponential complexity w.r.t. the number of sorts

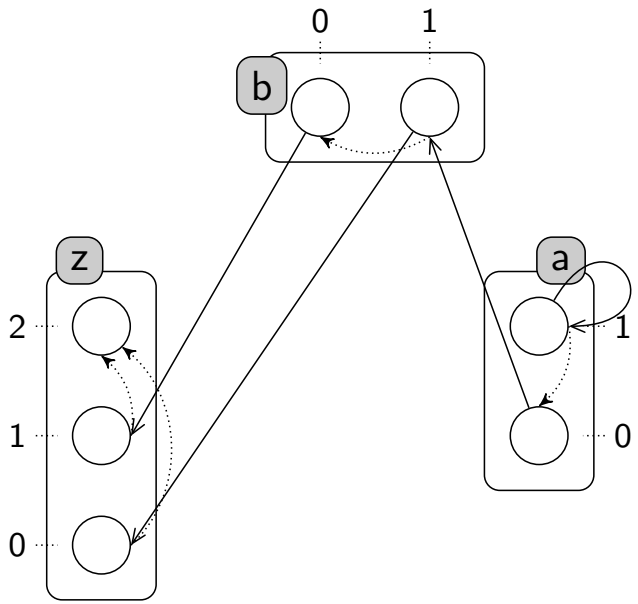
Static Analysis: Fixed Points

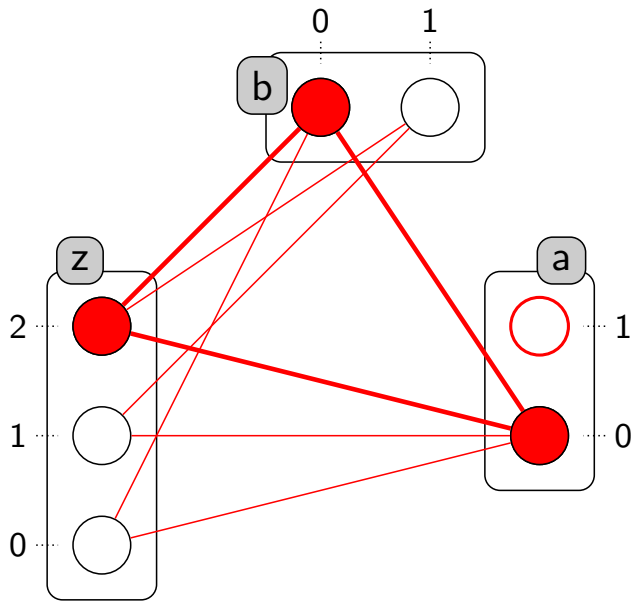
[PMR11a]

Fixed point = state where no action can be fired

→ avoid couples of processes bounded by an action

→ Hitless Graph → **n-cliques** = fixed points





Exponential complexity w.r.t. the number of sorts

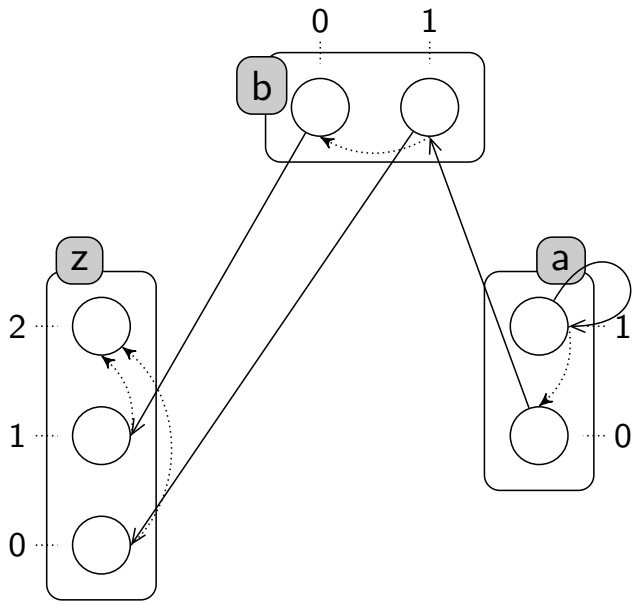
Static Analysis: Fixed Points

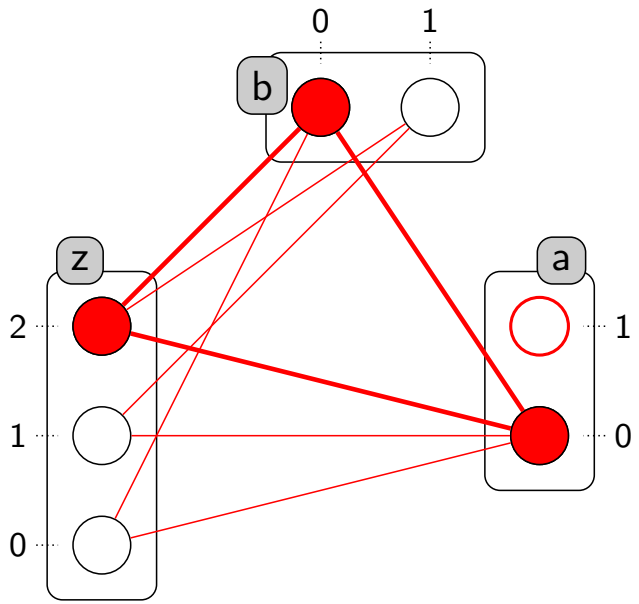
[PMR11a]

Fixed point = state where no action can be fired

→ avoid couples of processes bounded by an action

→ Hitless Graph → **n-cliques** = fixed points





Exponential complexity w.r.t. the number of sorts

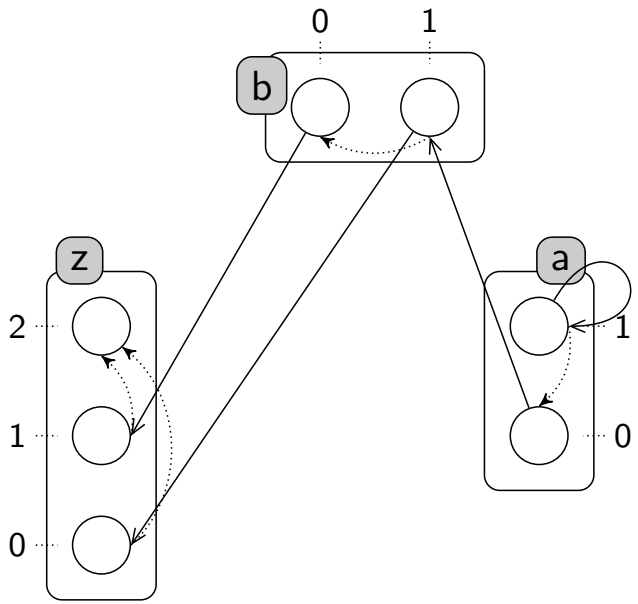
Static Analysis: Fixed Points

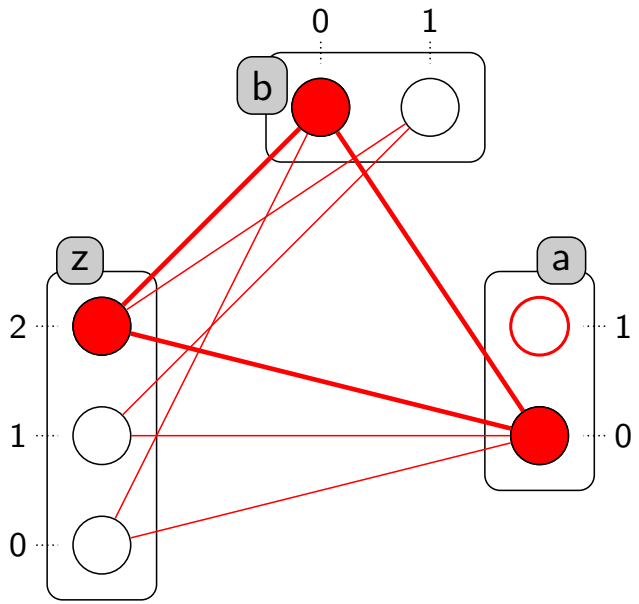
[PMR11a]

Fixed point = state where no action can be fired

→ avoid couples of processes bounded by an action

→ Hitless Graph → **n-cliques** = fixed points





Exponential complexity w.r.t. the number of sorts

Static analysis: successive reachability

[PMR12]

Problem

Given an initial state of a Process Hitting, is it possible to reach successively a_i, then b_j, then a_k, then c_l, ...?

$\Rightarrow$ Combinatorial explosion of the dynamics to explore

Key idea

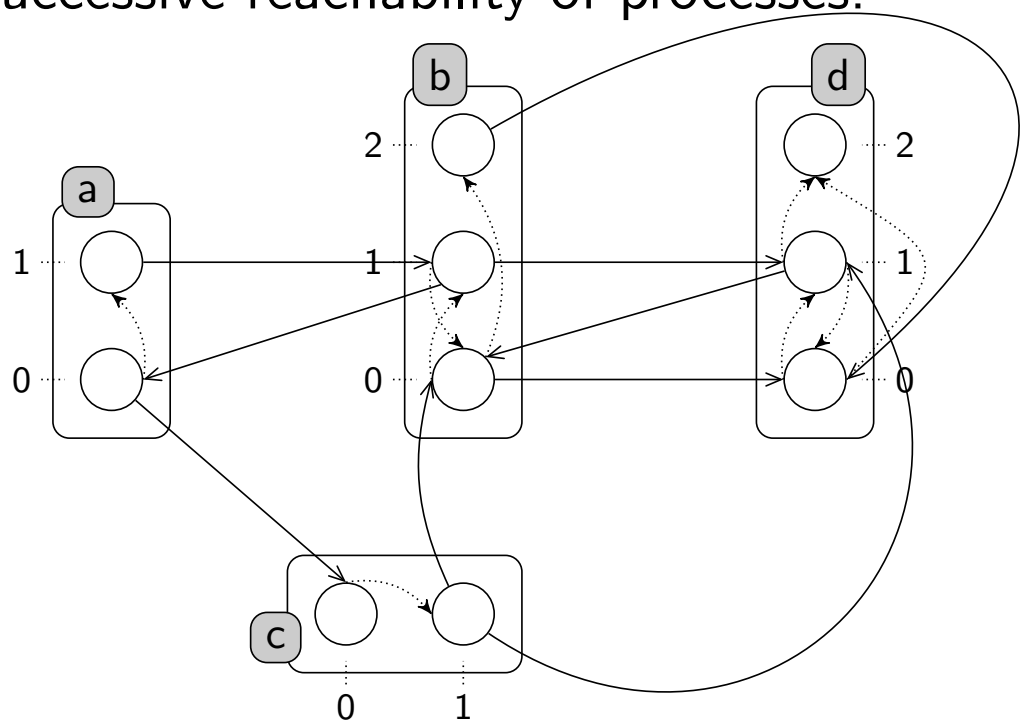
Instead of checking the successive reachability $\mathcal{R}$, which is complex, we will check:

- an **under-approximation** $\mathcal{P}$: if $\mathcal{P}$ is not satisfied, then $\mathcal{R}$ neither
- an **over-approximation** $\mathcal{Q}$: if $\mathcal{Q}$ is satisfied, then $\mathcal{R}$ too.

Static analysis: successive reachability

[PMR12]

Successive reachability of processes:



- Initial context

$$\langle a_1, \{b_0, b_1\}, c_0, d_0 \rangle$$

- Objectives

$$[\ \lceil\!\!\lceil\ d_1 :: \lceil\!\!\lceil\ b_1 :: \lceil\!\!\lceil\ d_2\ ]$$

$$[\ \lceil\!\!\lceil\ d_2\ ]$$

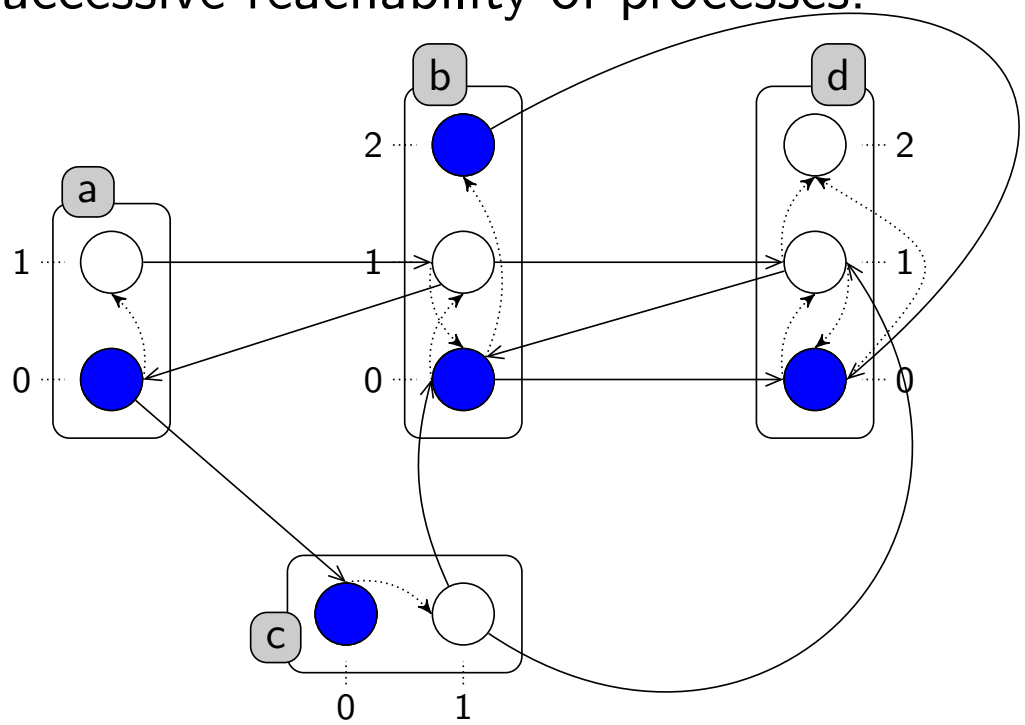
$\rightarrow$ Concretization of the objective = scenario

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

Static analysis: successive reachability

[PMR12]

Successive reachability of processes:



- Initial context

$$\langle a_1, \{b_0, b_1\}, c_0, d_0 \rangle$$

- Objectives

$$[\ \lceil\!\!\lceil\ d_1 :: \lceil\!\!\lceil\ b_1 :: \lceil\!\!\lceil\ d_2\ ]$$

$$[\ \lceil\!\!\lceil\ d_2\ ]$$

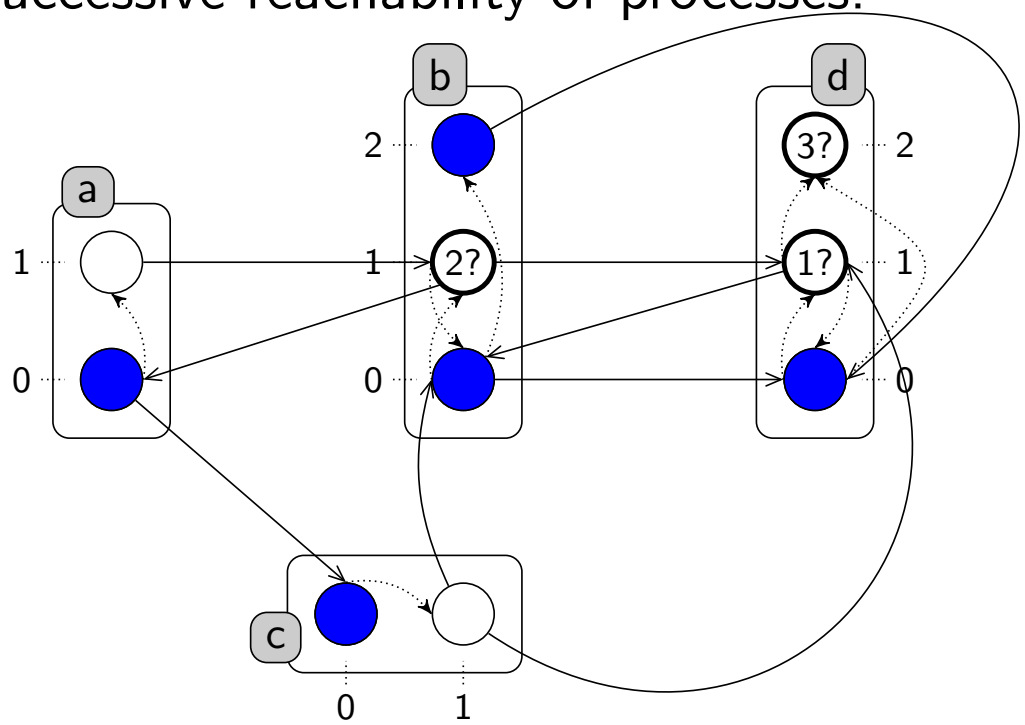
$\rightarrow$ Concretization of the objective = scenario

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

Static analysis: successive reachability

[PMR12]

Successive reachability of processes:



- Initial context

$$\langle a_1, \{b_0, b_1\}, c_0, d_0 \rangle$$

- Objectives

$$[\ \lceil\!\!\lceil\ d_1 :: \lceil\!\!\lceil\ b_1 :: \lceil\!\!\lceil\ d_2\ ]$$

$$[\ \lceil\!\!\lceil\ d_2\ ]$$

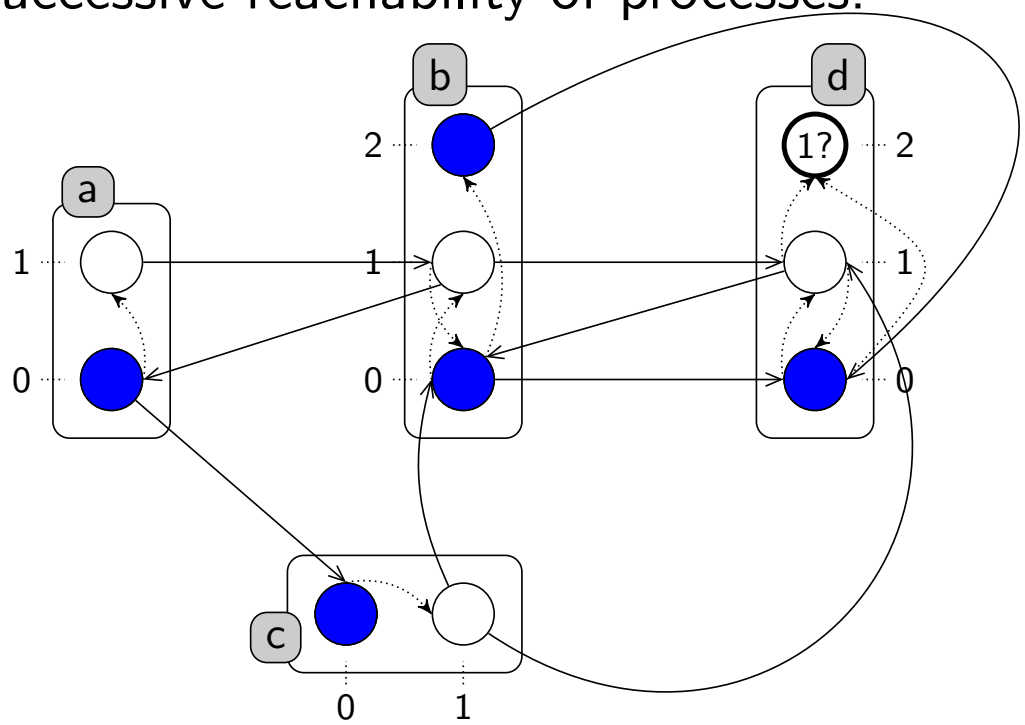
$\rightarrow$ Concretization of the objective = scenario

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

Static analysis: successive reachability

[PMR12]

Successive reachability of processes:



- Initial context

$$\langle a_1, \{b_0, b_1\}, c_0, d_0 \rangle$$

- Objectives

$$[\ \lceil\!\rceil\ d_1 :: \lceil\!\rceil\ b_1 :: \lceil\!\rceil\ d_2\ ]$$

$$[\ \lceil\!\rceil\ d_2\ ]$$

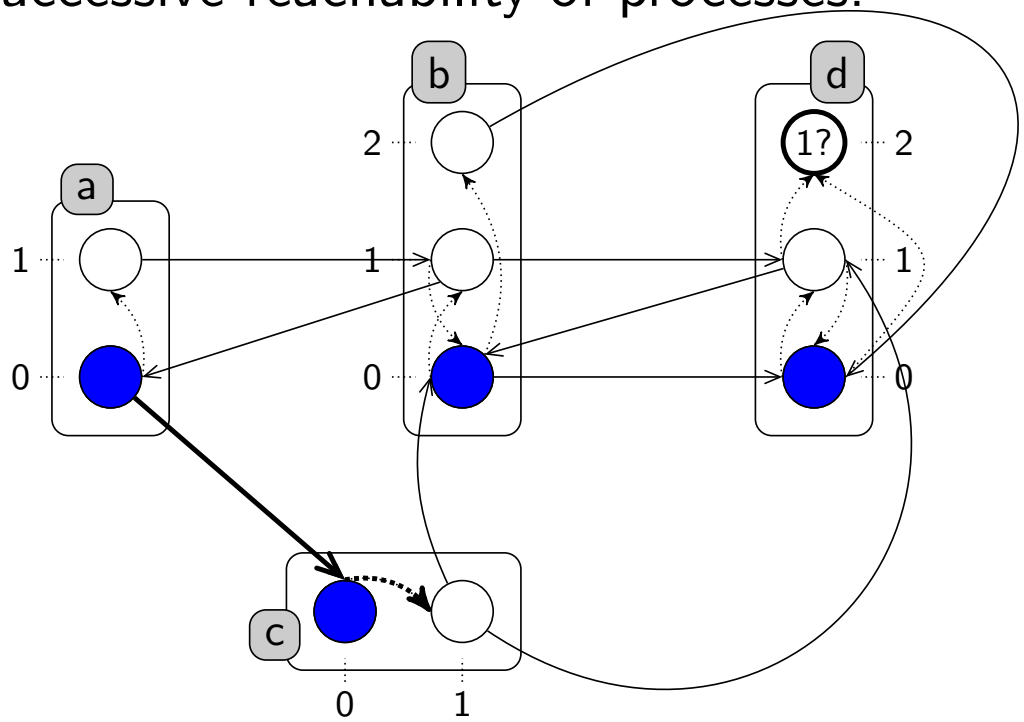
$\rightarrow$ Concretization of the objective = scenario

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

Static analysis: successive reachability

[PMR12]

Successive reachability of processes:



- Initial context

$$\langle a_1, \{b_0, b_1\}, c_0, d_0 \rangle$$

- Objectives

$$[\ \lceil\!\!\lceil\ d_1 :: \lceil\!\!\lceil\ b_1 :: \lceil\!\!\lceil\ d_2\ ]$$

$$[\ \lceil\!\!\lceil\ d_2\ ]$$

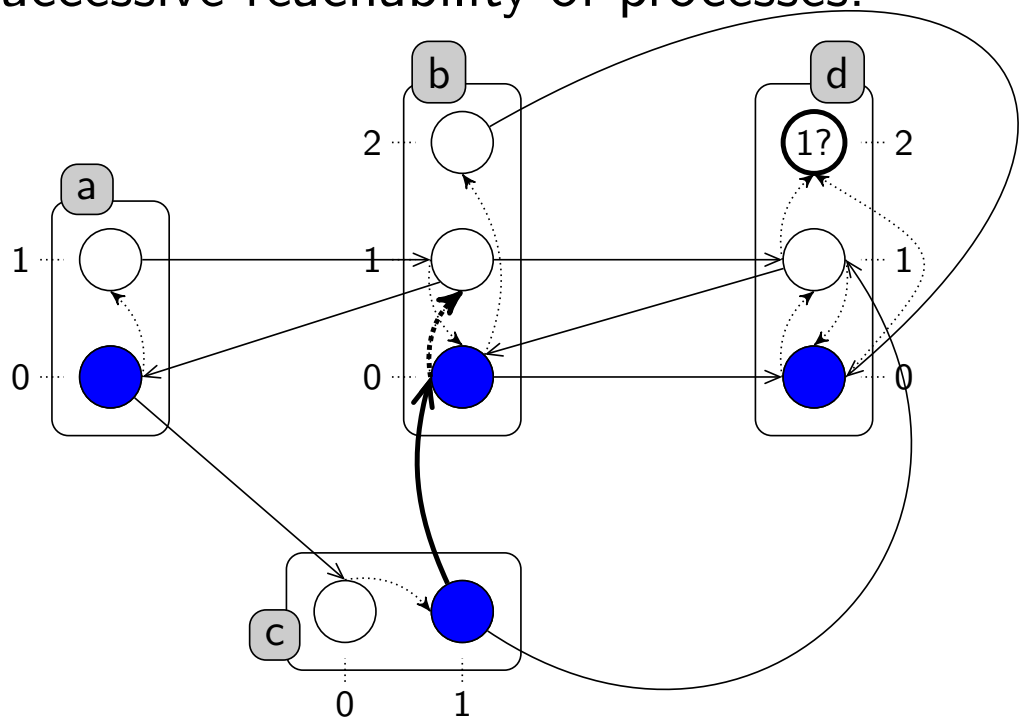
$\rightarrow$ Concretization of the objective = scenario

$$\underline{a_0 \rightarrow c_0 \lceil\!\!\lceil\ c_1} :: \quad b_0 \rightarrow d_0 \lceil\!\!\lceil\ d_1 :: \quad c_1 \rightarrow b_0 \lceil\!\!\lceil\ b_1 :: \quad b_1 \rightarrow d_1 \lceil\!\!\lceil\ d_2$$

Static analysis: successive reachability

[PMR12]

Successive reachability of processes:



- Initial context

$$\langle a_1, \{b_0, b_1\}, c_0, d_0 \rangle$$

- Objectives

$$[\ \lceil^{\uparrow} d_1 :: \lceil^{\uparrow} b_1 :: \lceil^{\uparrow} d_2\ ]$$

$$[\ \lceil^{\uparrow} d_2\ ]$$

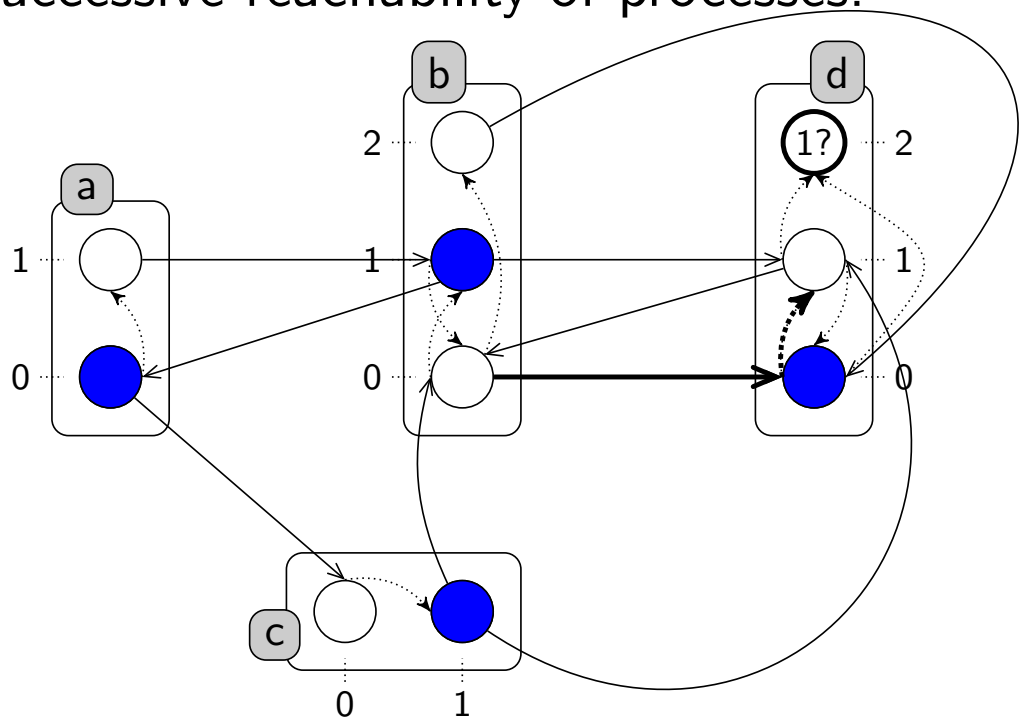
$\rightarrow$ Concretization of the objective = scenario

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

Static analysis: successive reachability

[PMR12]

Successive reachability of processes:



- Initial context

$$\langle a_1, \{b_0, b_1\}, c_0, d_0 \rangle$$

- Objectives

$$[\ \Rsh\ d_1 :: \Rsh\ b_1 :: \Rsh\ d_2\ ]$$

$$[\ \Rsh\ d_2\ ]$$

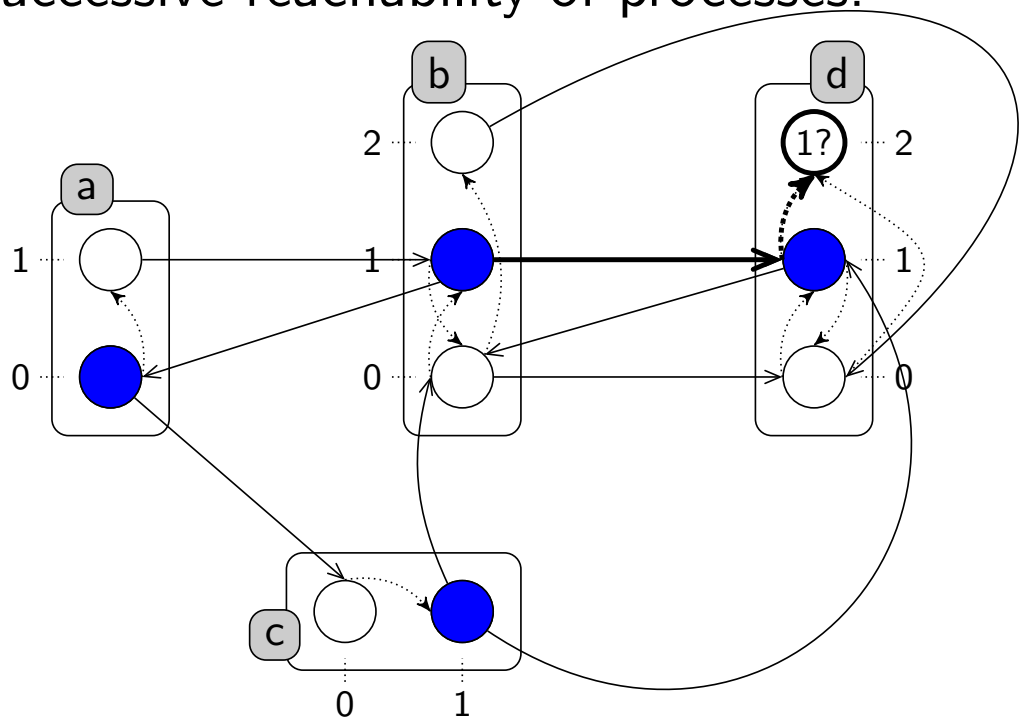
$\rightarrow$ Concretization of the objective = scenario

$$a_0 \rightarrow c_0 \Rsh c_1 :: \quad b_0 \rightarrow d_0 \Rsh d_1 :: \quad \underline{c_1 \rightarrow b_0 \Rsh b_1} :: \quad b_1 \rightarrow d_1 \Rsh d_2$$

Static analysis: successive reachability

[PMR12]

Successive reachability of processes:



- Initial context

$$\langle a_1, \{b_0, b_1\}, c_0, d_0 \rangle$$

- Objectives

$$[\ \lceil' d_1 :: \lceil' b_1 :: \lceil' d_2\ ]$$

$$[\ \lceil' d_2\ ]$$

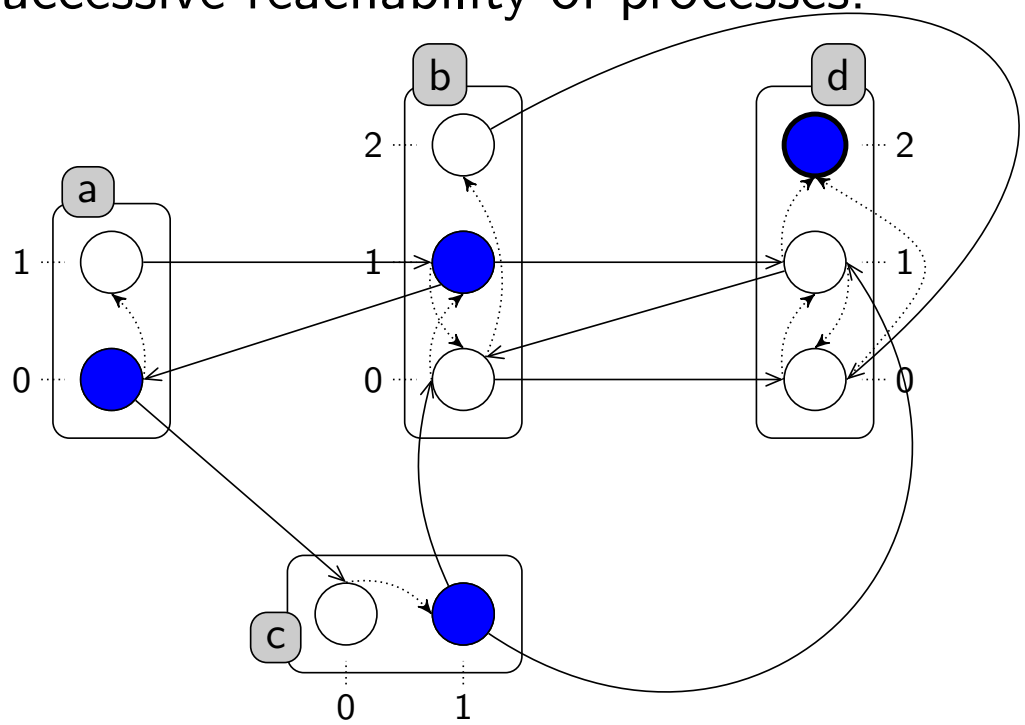
$\rightarrow$ Concretization of the objective = scenario

$$a_0 \rightarrow c_0 \lceil' c_1 :: \quad b_0 \rightarrow d_0 \lceil' d_1 :: \quad c_1 \rightarrow b_0 \lceil' b_1 :: \quad \underline{b_1 \rightarrow d_1 \lceil' d_2}$$

Static analysis: successive reachability

[PMR12]

Successive reachability of processes:



- Initial context

$$\langle a_1, \{b_0, b_1\}, c_0, d_0 \rangle$$

- Objectives

$$[\ \lceil^{\uparrow}\ d_1 :: \lceil^{\uparrow}\ b_1 :: \lceil^{\uparrow}\ d_2\ ]$$

$$[\ \lceil^{\uparrow}\ d_2\ ]$$

$\rightarrow$ Concretization of the objective = scenario

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

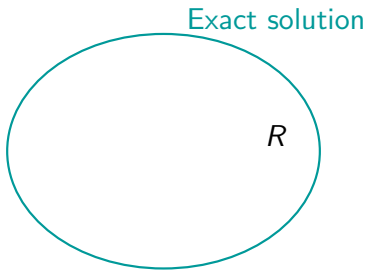
Over- and Under-approximations

[PMR12]

Static analysis by abstractions:

→ Directly checking an objective sequence R is hard

→ Rather check the approximations P and Q, where $P \Rightarrow R \Rightarrow Q$:

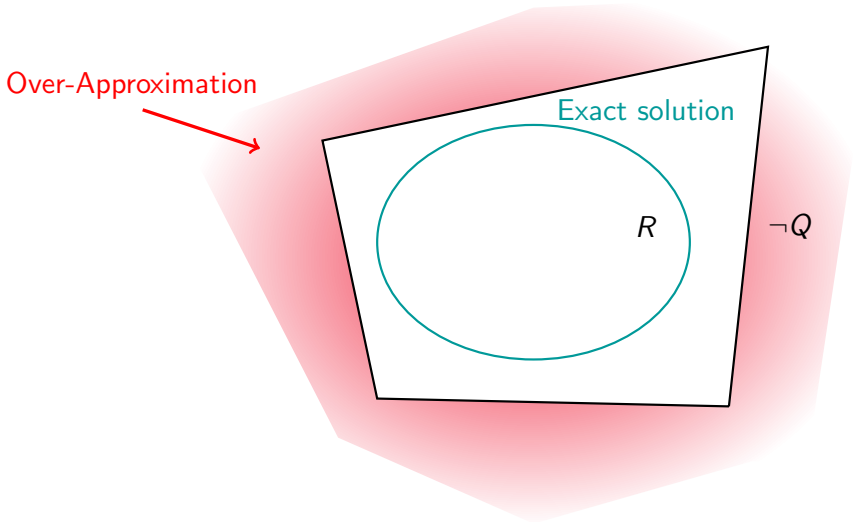


Over- and Under-approximations

[PMR12]

Static analysis by abstractions:

- $\rightarrow$ Directly checking an objective sequence R is hard
- $\rightarrow$ Rather check the approximations P and Q, where $P \Rightarrow R \Rightarrow Q$:

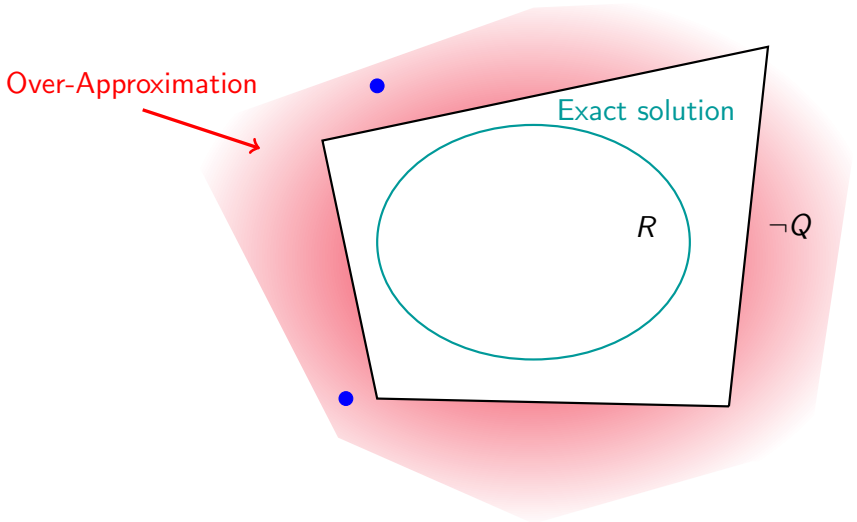


Over- and Under-approximations

[PMR12]

Static analysis by abstractions:

- $\rightarrow$ Directly checking an objective sequence R is hard
- $\rightarrow$ Rather check the approximations P and Q, where $P \Rightarrow R \Rightarrow Q$:

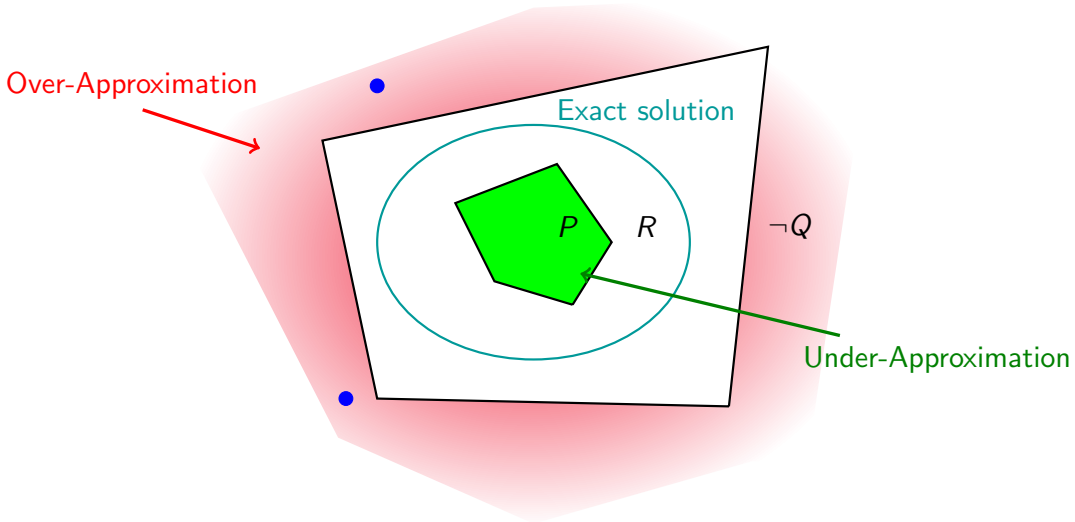


Over- and Under-approximations

[PMR12]

Static analysis by abstractions:

- $\rightarrow$ Directly checking an objective sequence R is hard
- $\rightarrow$ Rather check the approximations P and Q, where $P \Rightarrow R \Rightarrow Q$:

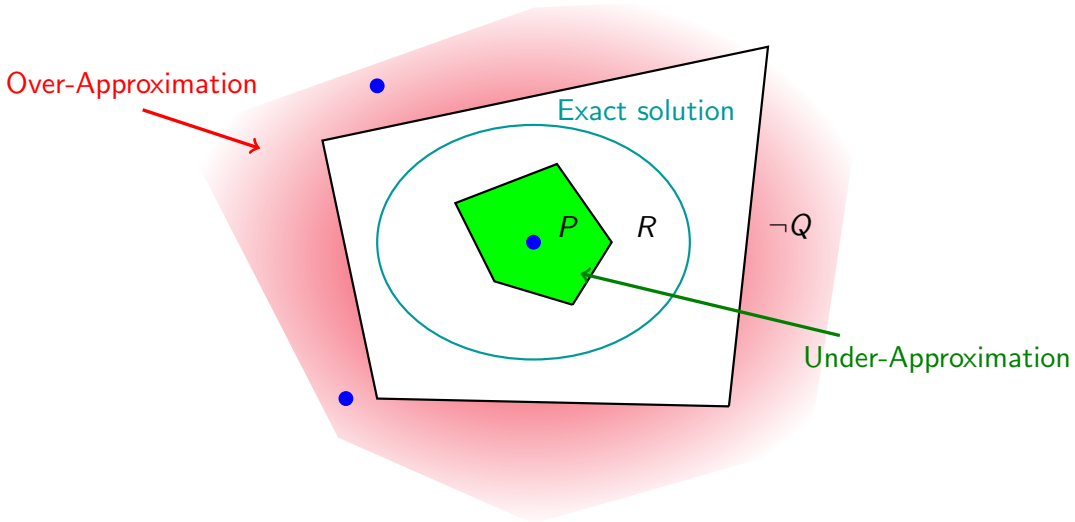


Over- and Under-approximations

[PMR12]

Static analysis by abstractions:

- $\rightarrow$ Directly checking an objective sequence R is hard
- $\rightarrow$ Rather check the approximations P and Q, where $P \Rightarrow R \Rightarrow Q$:

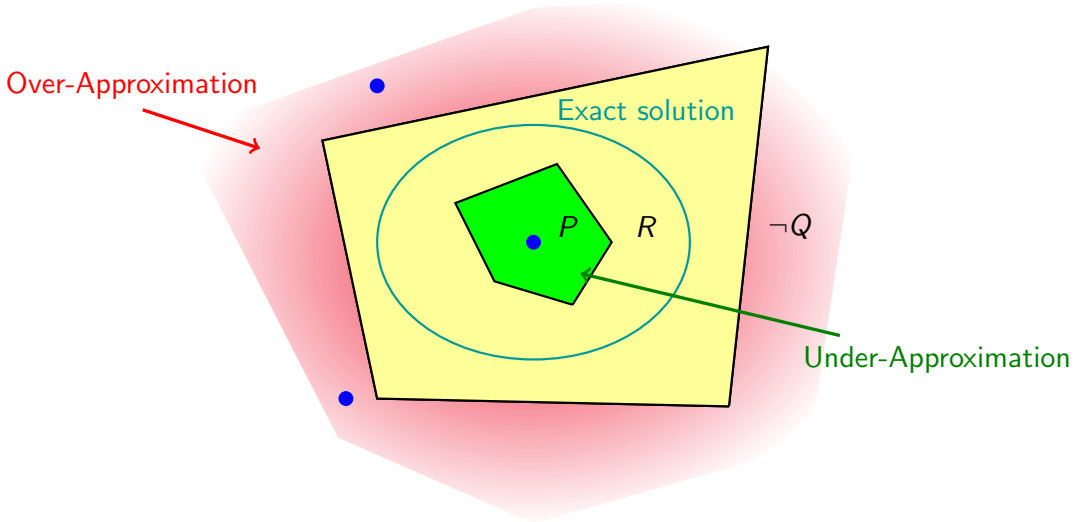


Over- and Under-approximations

[PMR12]

Static analysis by abstractions:

- $\rightarrow$ Directly checking an objective sequence R is hard
- $\rightarrow$ Rather check the approximations P and Q, where $P \Rightarrow R \Rightarrow Q$:

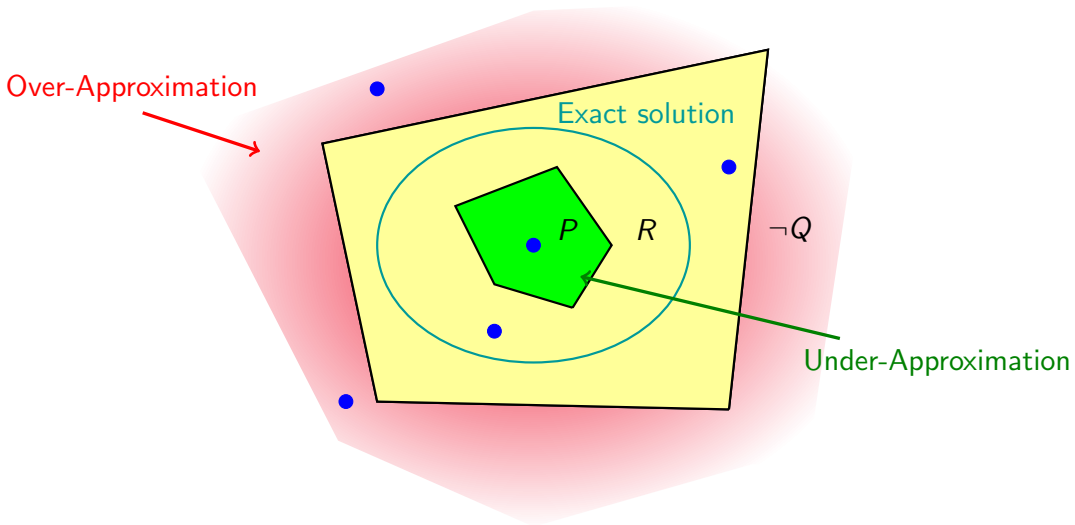


Over- and Under-approximations

[PMR12]

Static analysis by abstractions:

- $\rightarrow$ Directly checking an objective sequence R is hard
- $\rightarrow$ Rather check the approximations P and Q, where $P \Rightarrow R \Rightarrow Q$:



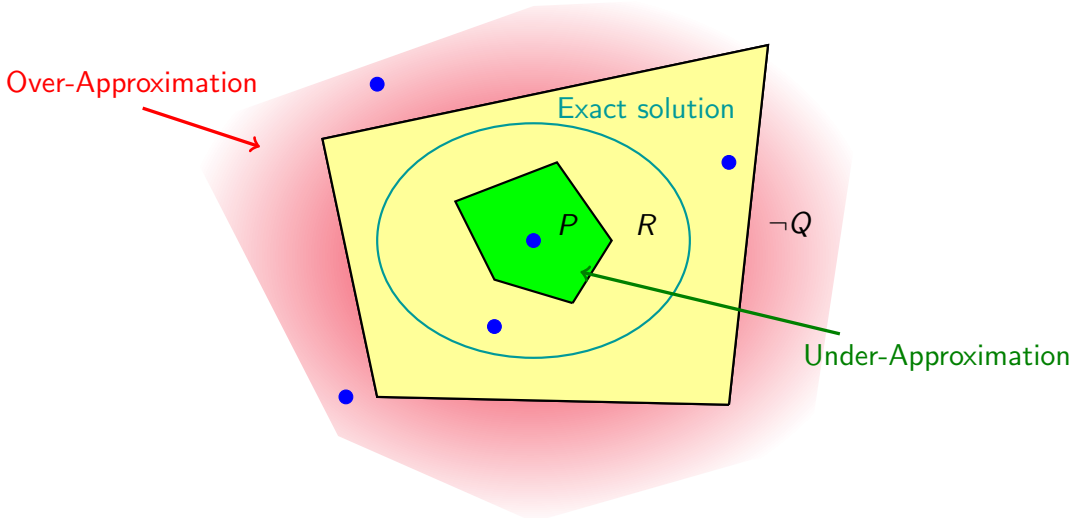
Over- and Under-approximations

[PMR12]

Static analysis by abstractions:

→ Directly checking an objective sequence R is hard

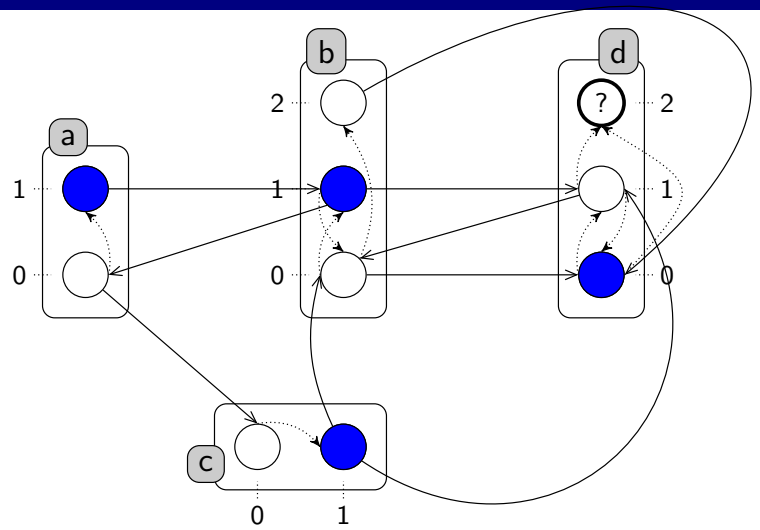
→ Rather check the approximations P and Q, where $P \Rightarrow R \Rightarrow Q$:



Linear w.r.t. the number of sorts and
exponential w.r.t. the number of processes in each sort

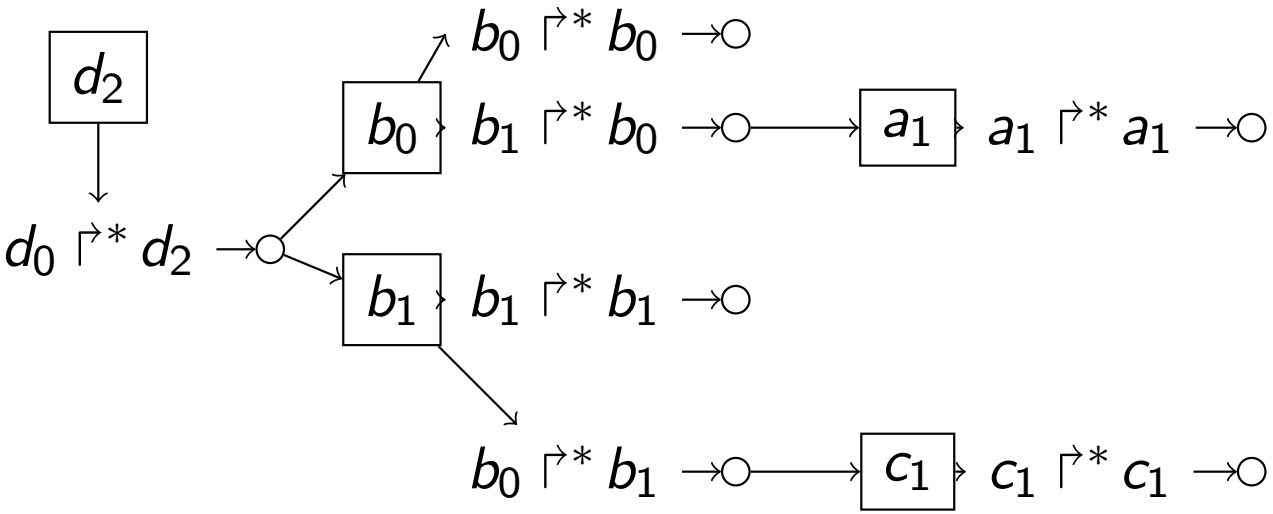
→ Efficient for big models with few levels of expression

Under-approximation

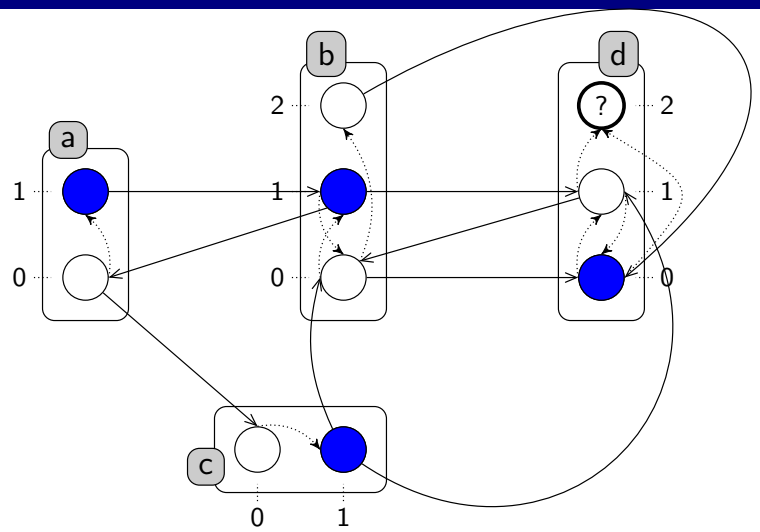


→ New abstract structure

Sufficient condition:



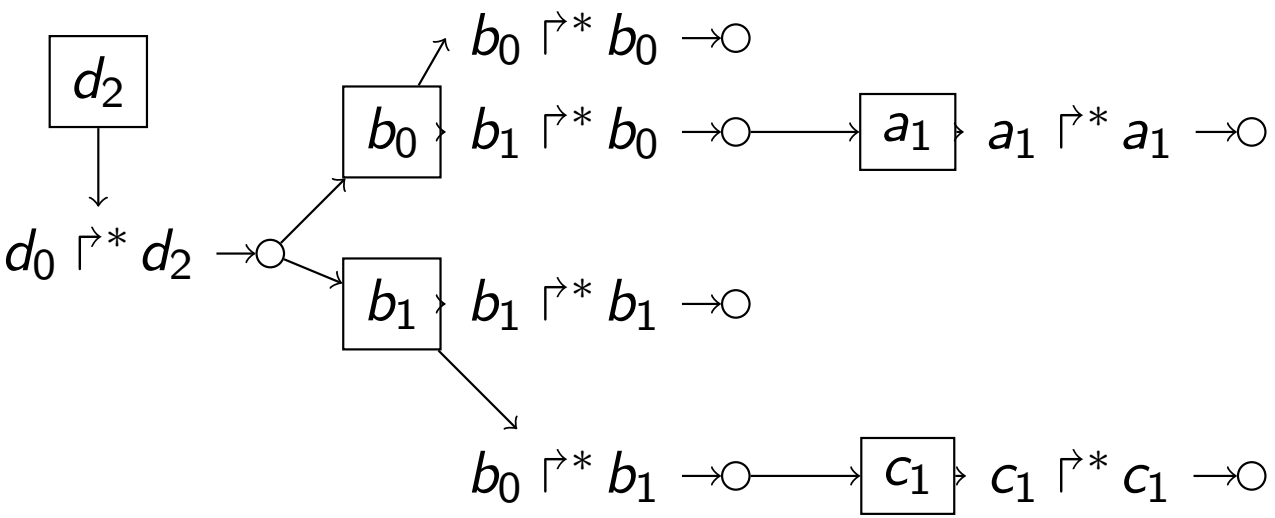
Under-approximation



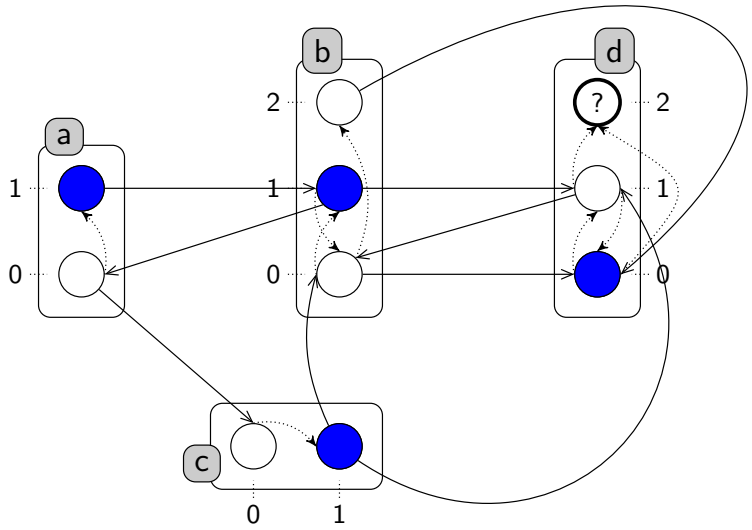
→ New abstract structure

Sufficient condition:

- no cycle
- each objective has a solution



Under-approximation

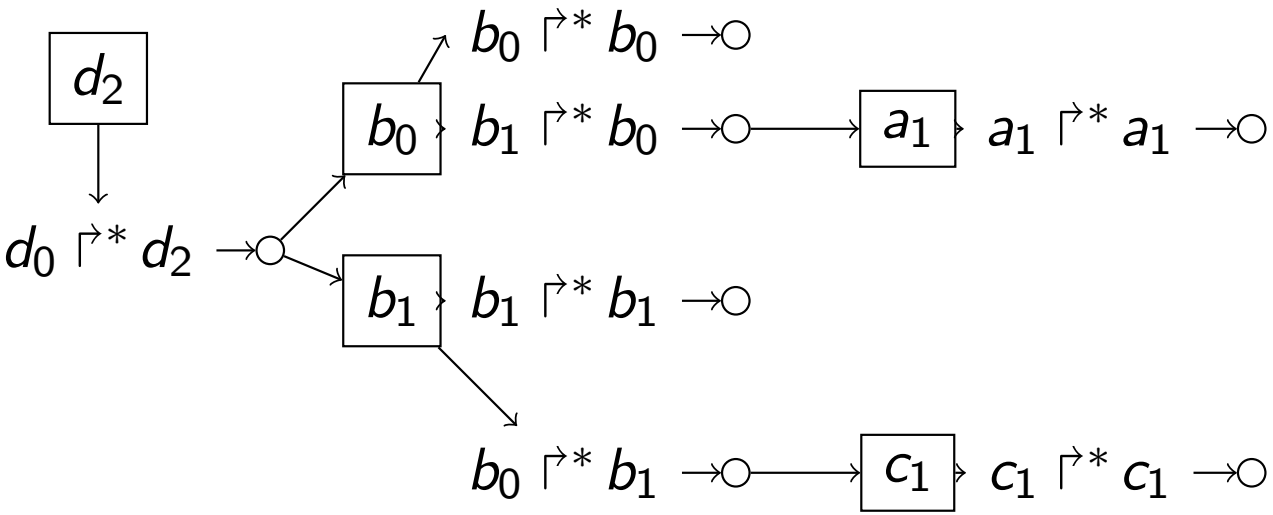


$\rightarrow$ New abstract structure

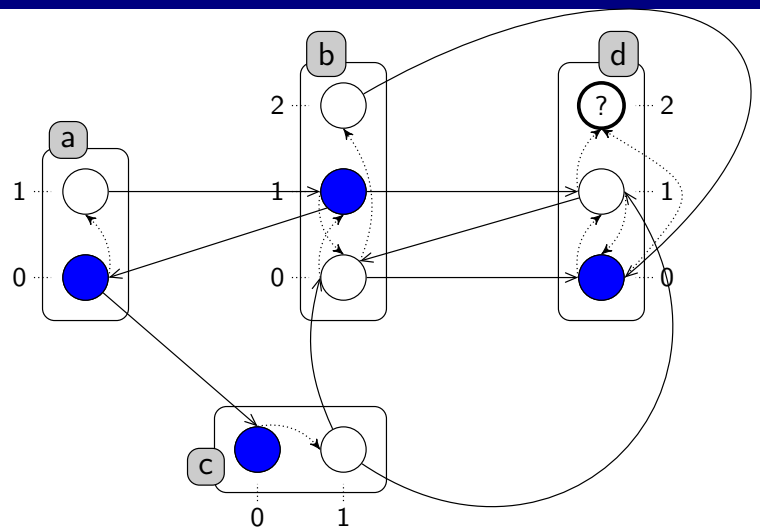
Sufficient condition:

- no cycle
- each objective has a solution

R is **true**



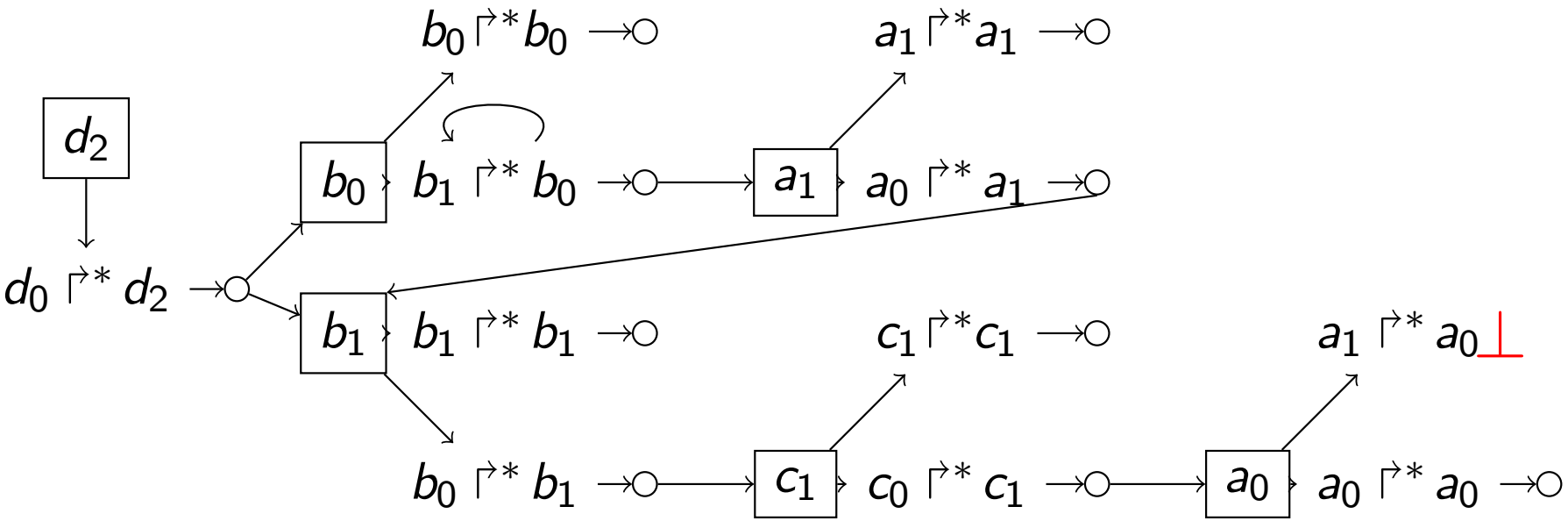
Under-approximation



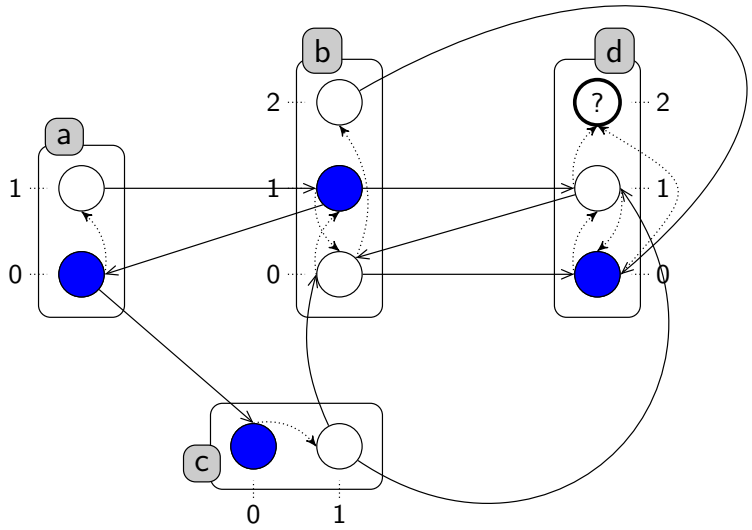
→ New abstract structure

Sufficient condition:

- ~~no cycle~~
- ~~each objective has a solution~~



Under-approximation

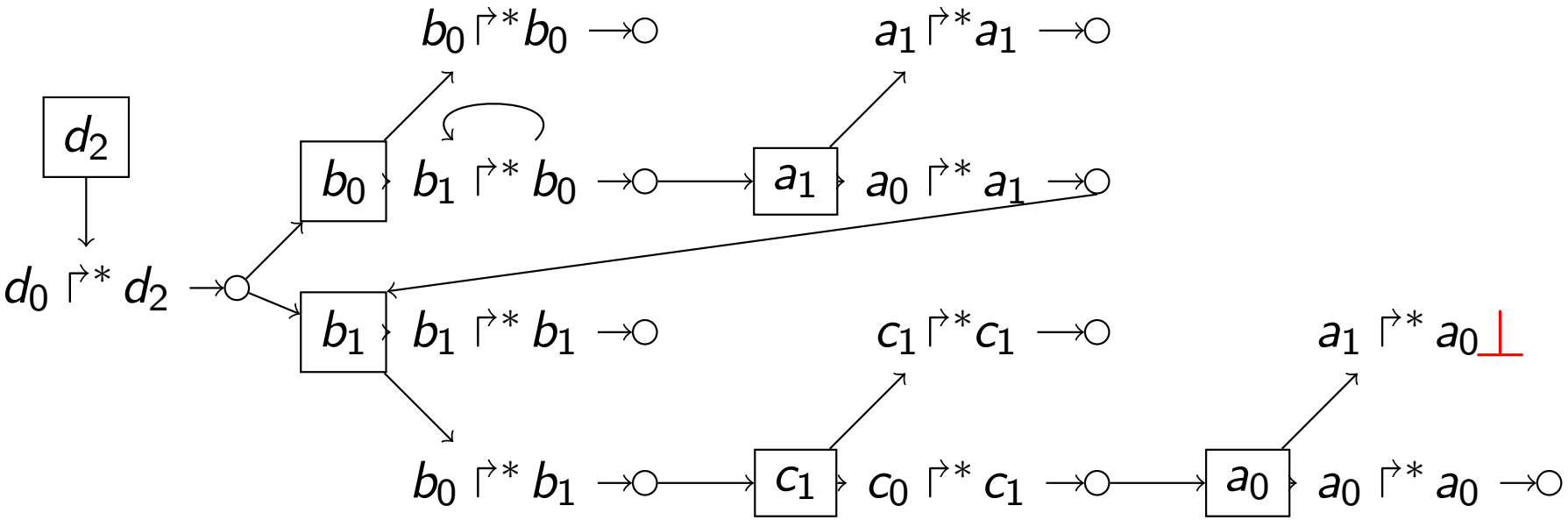


→ New abstract structure

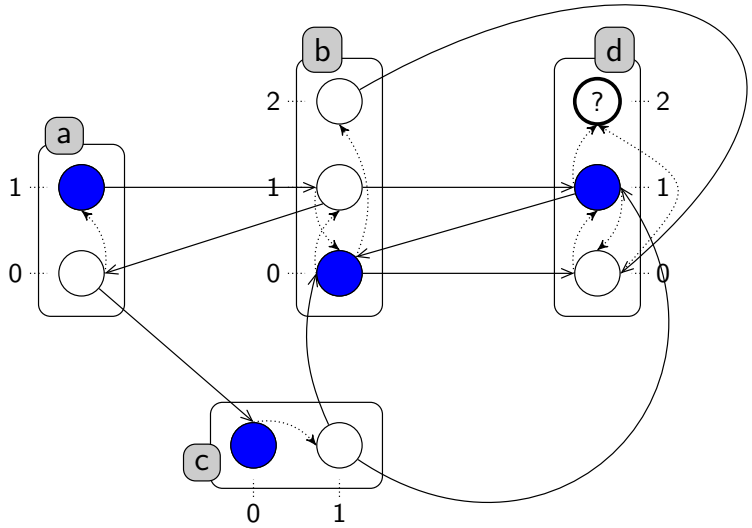
Sufficient condition:

- ~~no cycle~~
- ~~each objective has a solution~~

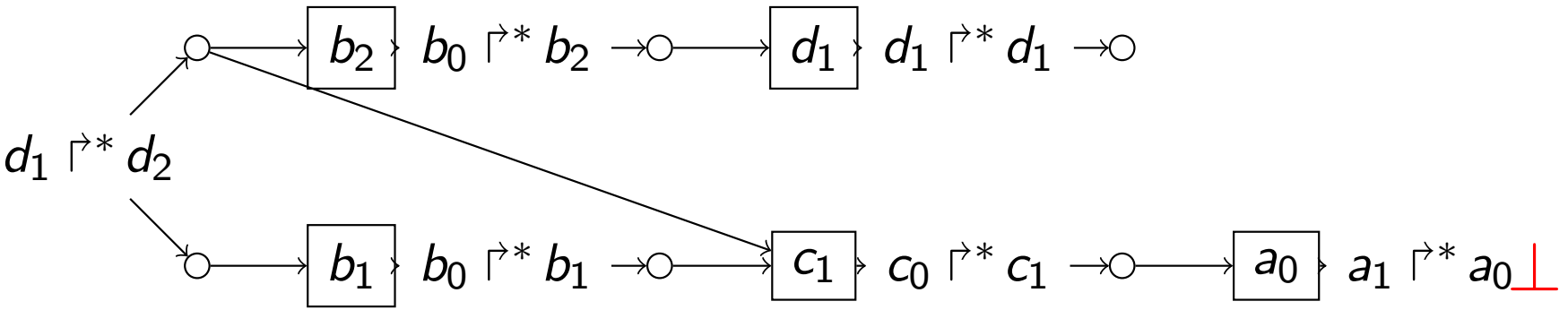
Inconclusive



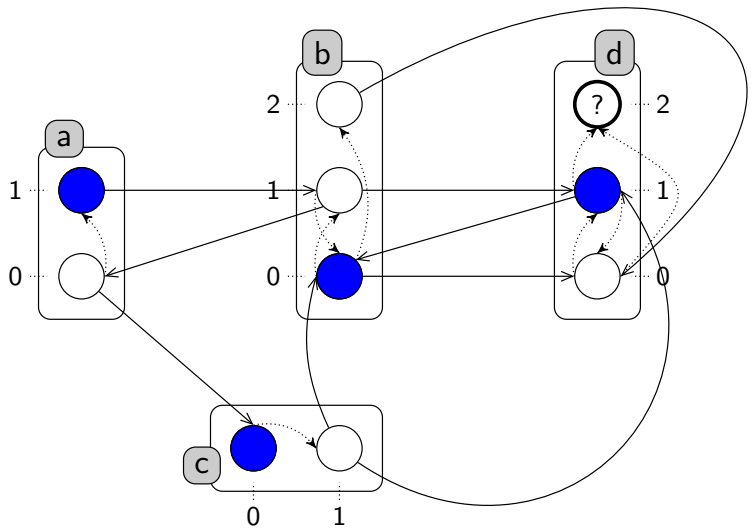
Over-approximation



Necessary condition:



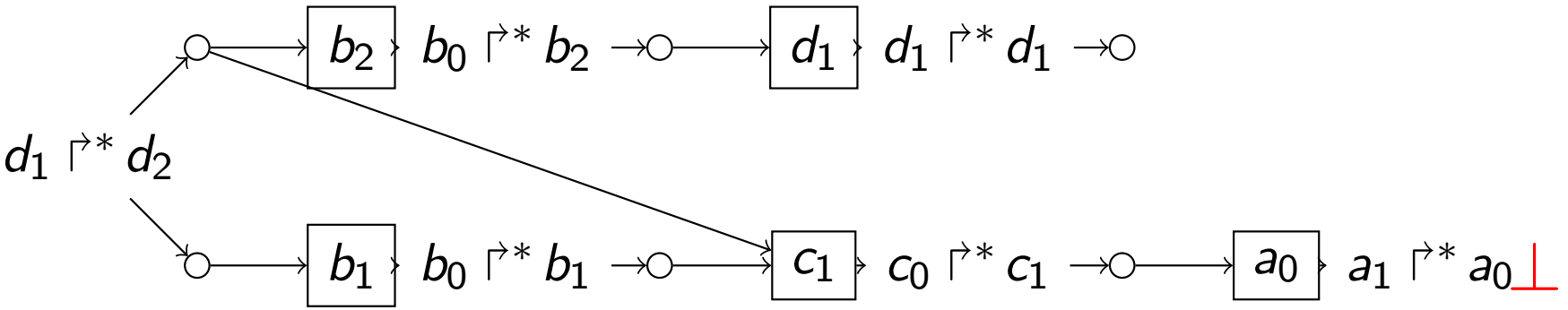
Over-approximation



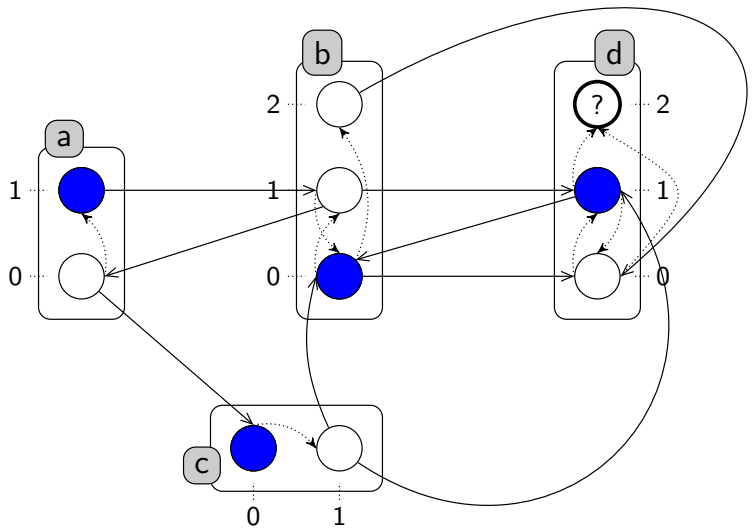
Necessary condition:

There exists a traversal with no cycle

- objective $\rightarrow$ follow one solution
- solution $\rightarrow$ follow all processes
- process $\rightarrow$ follow all objectives



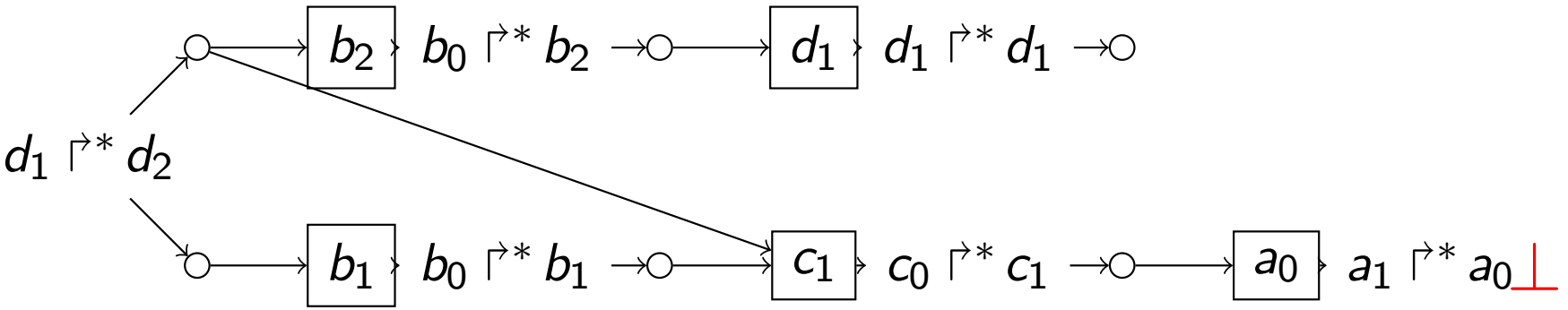
Over-approximation



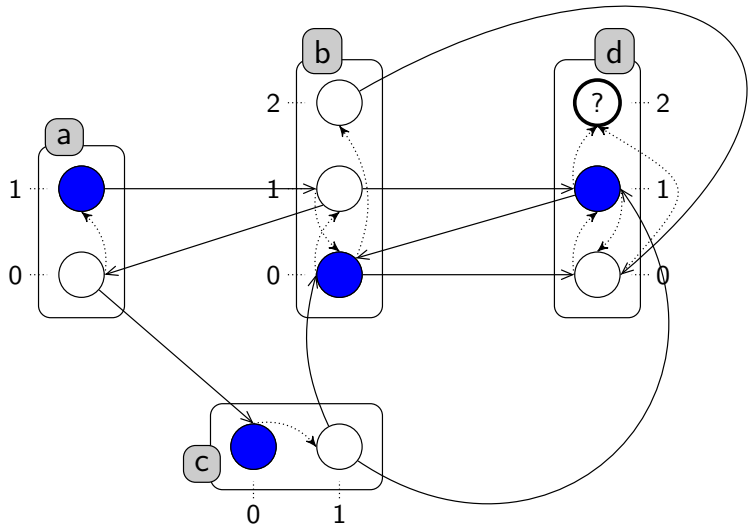
Necessary condition:

~~There exists a traversal~~ with no cycle

- ~~objective → follow one solution~~
- solution → follow all processes
- process → follow all objectives



Over-approximation

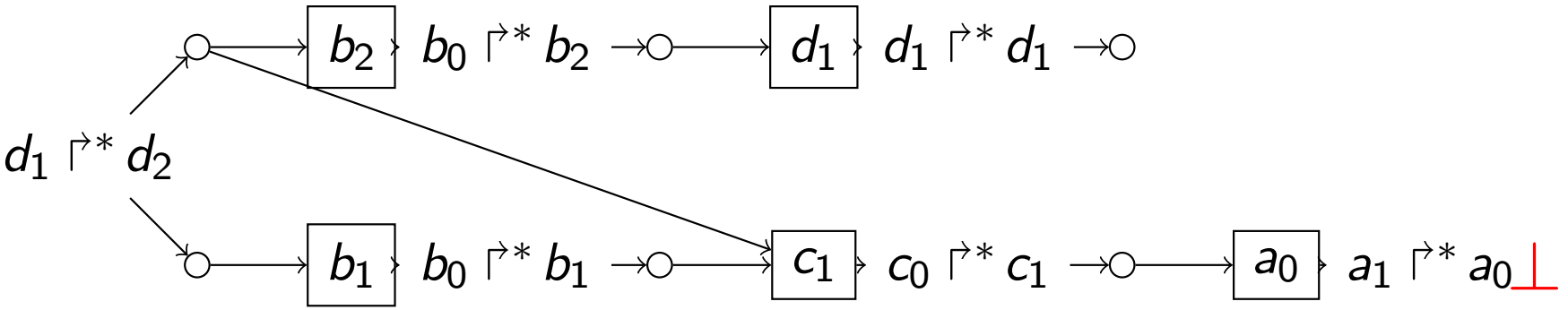


Necessary condition:

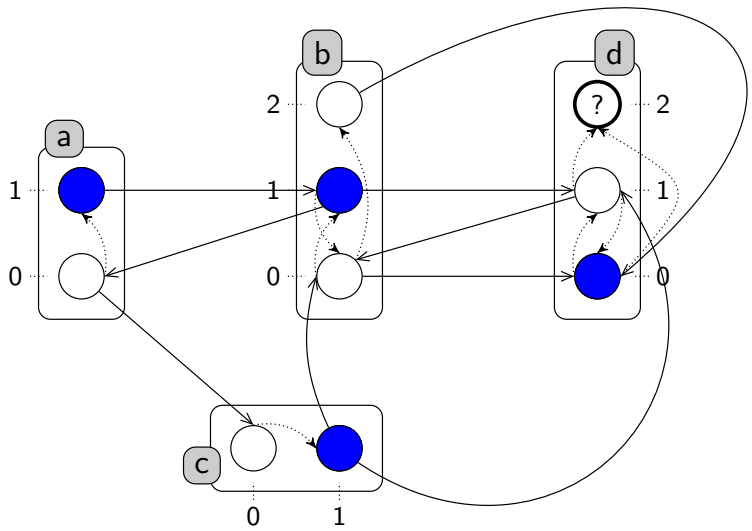
~~There exists a traversal~~ with no cycle

- ~~objective → follow one solution~~
- solution → follow all processes
- process → follow all objectives

R is **false**



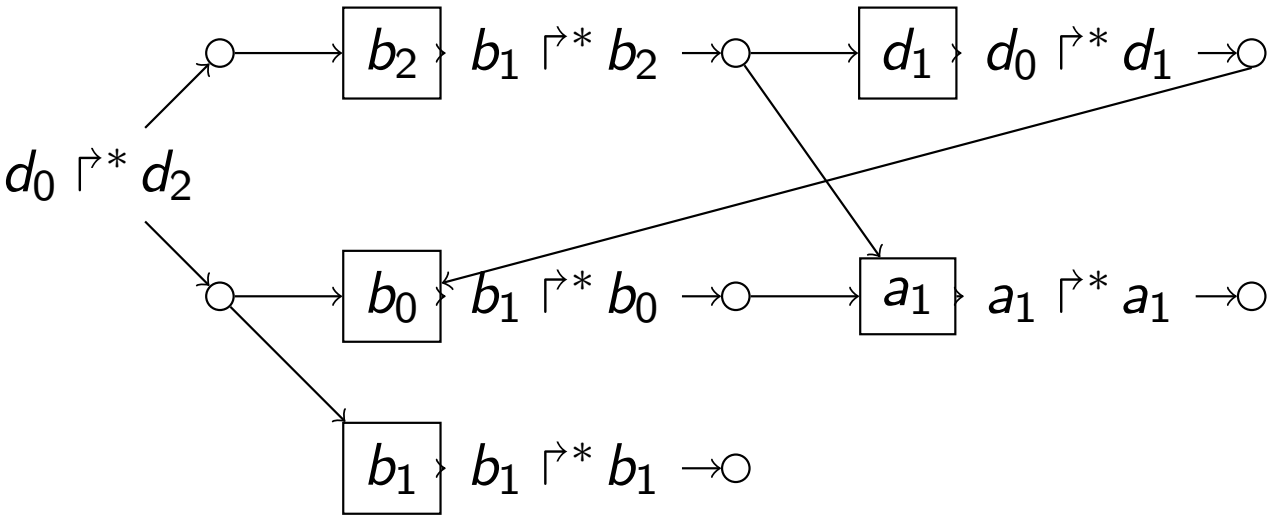
Over-approximation



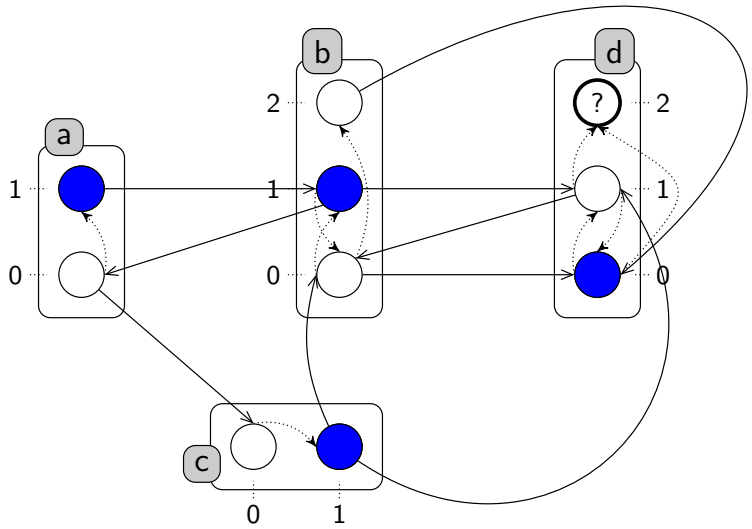
Necessary condition:

There exists a traversal with no cycle

- objective $\rightarrow$ follow one solution
- solution $\rightarrow$ follow all processes
- process $\rightarrow$ follow all objectives



Over-approximation

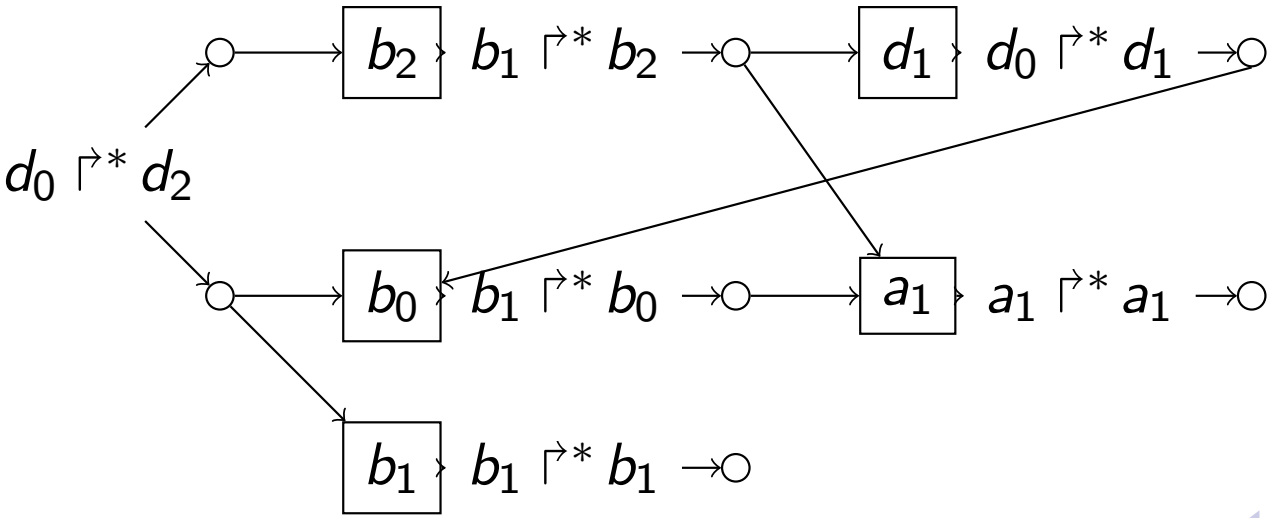


Necessary condition:

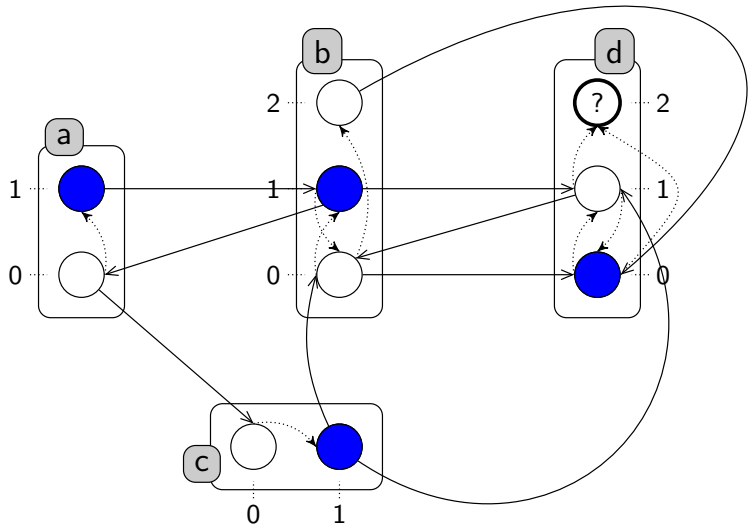
There exists a traversal with no cycle

- objective → follow one solution
- solution → follow all processes
- process → follow all objectives

Inconclusive



Over-approximation

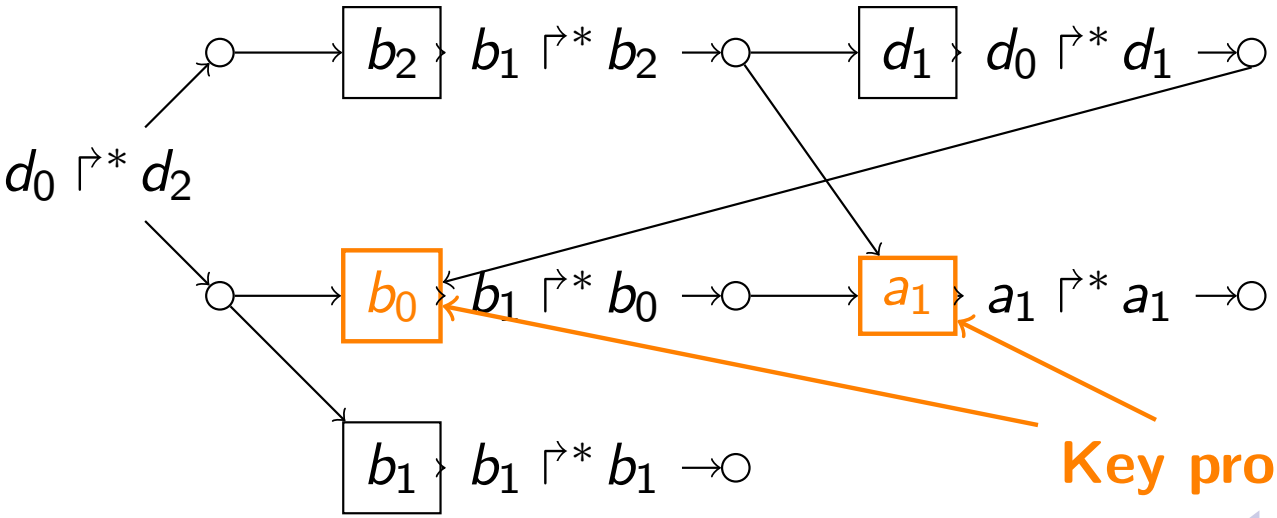


Necessary condition:

There exists a traversal with no cycle

- objective → follow one solution
- solution → follow all processes
- process → follow all objectives

Inconclusive



Stochastic Features

[PMR11b]

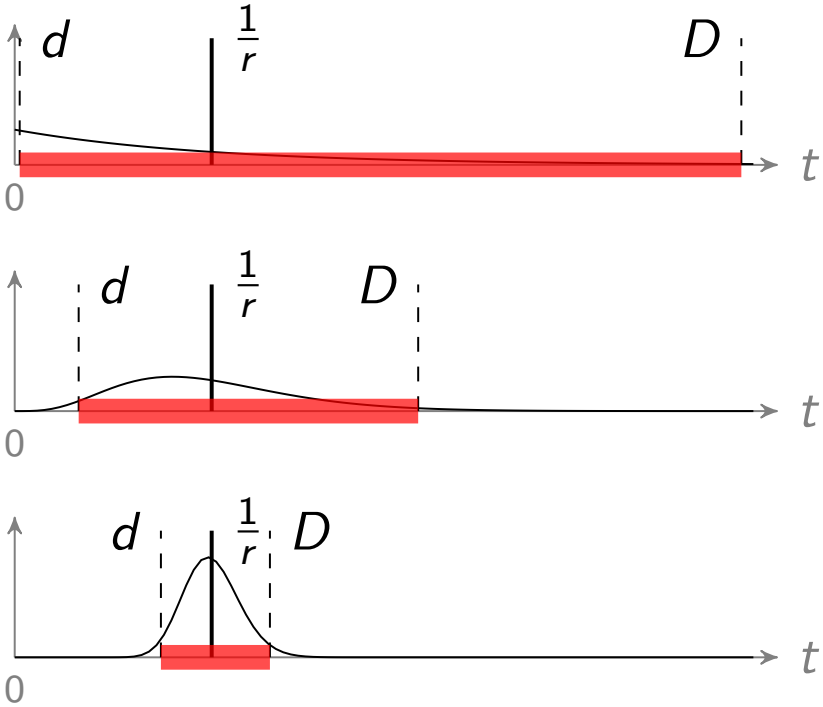
- Introduces time features
- Parameters: either (r, sa), or the **firing interval** $[d; D]$.

$\rightarrow$ Tests by **simulation** and **model-checking**

Stochastic Features

[PMR11b]

- Introduces time features
- Parameters: either (r, sa), or the **firing interval** $[d; D]$.



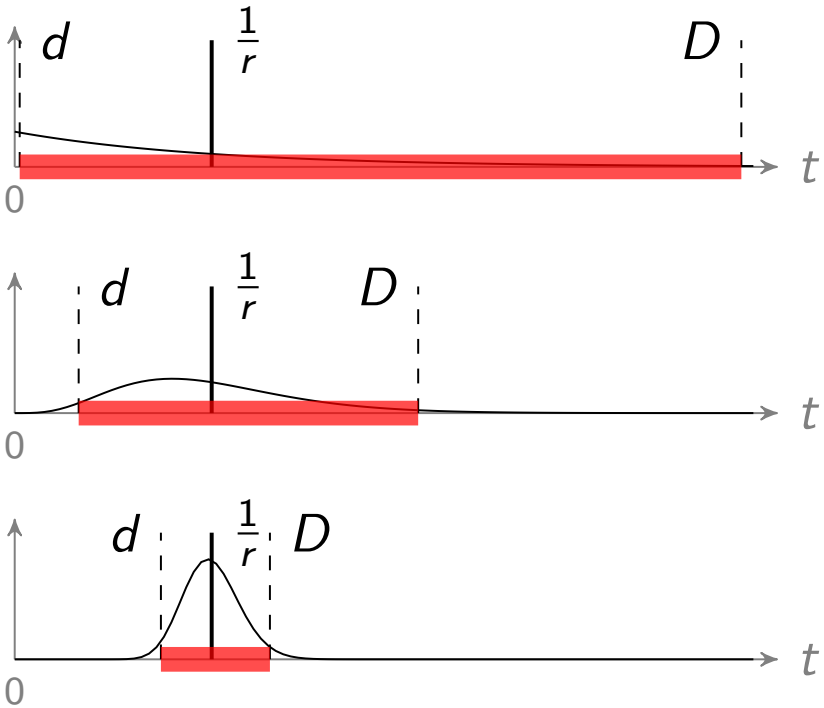
action duration

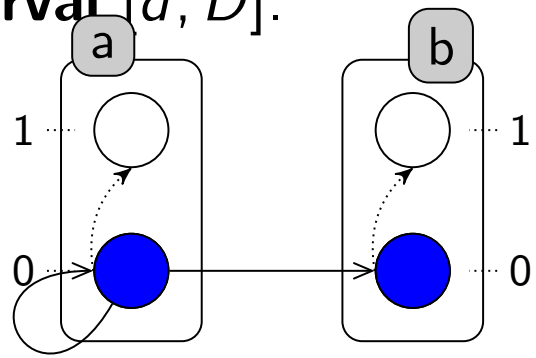
→ Tests by **simulation** and **model-checking**

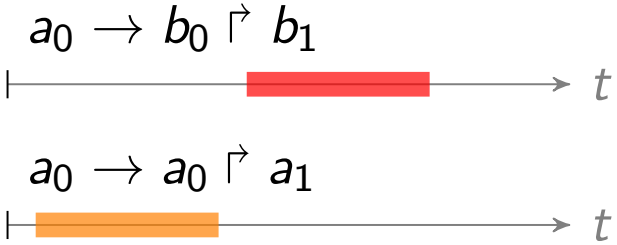
Stochastic Features

[PMR11b]

- Introduces time features
- Parameters: either (r, sa), or the **firing interval** $[d; D]$.







$\to$ b_1 reached with a **very low probability**.

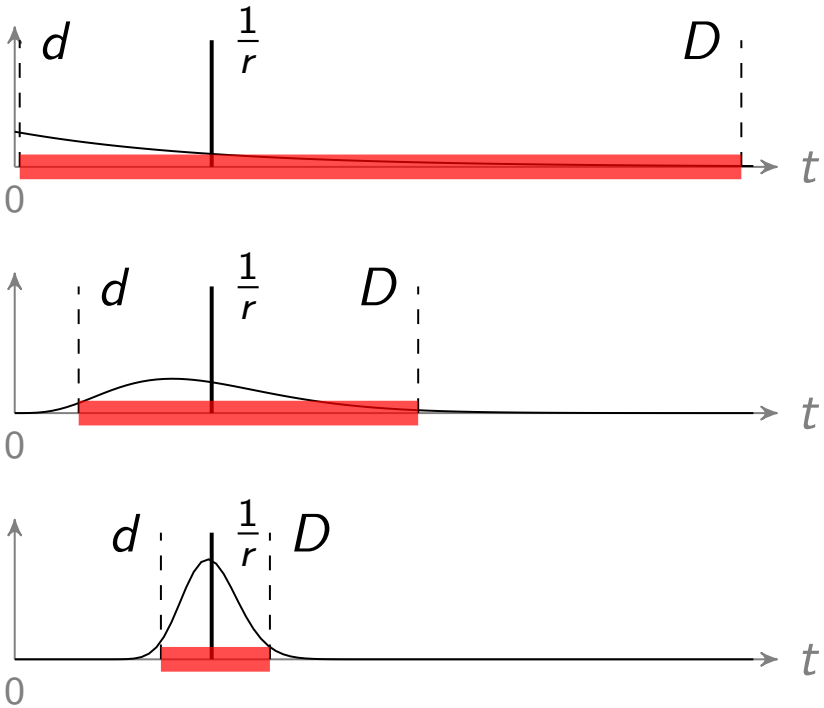
action duration

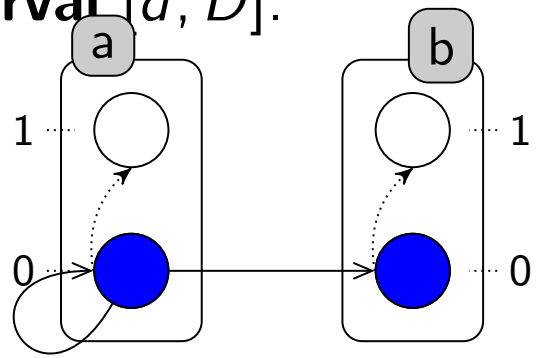
$\to$ Tests by **simulation** and **model-checking**

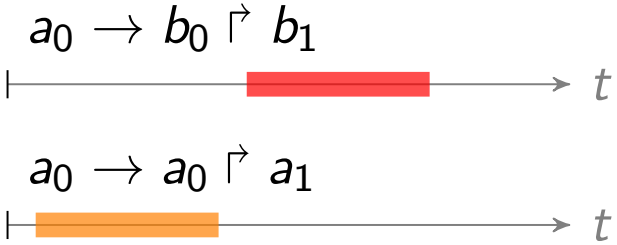
Stochastic Features

[PMR11b]

- Introduces time features
- Parameters: either (r, sa), or the **firing interval** $[d; D]$.







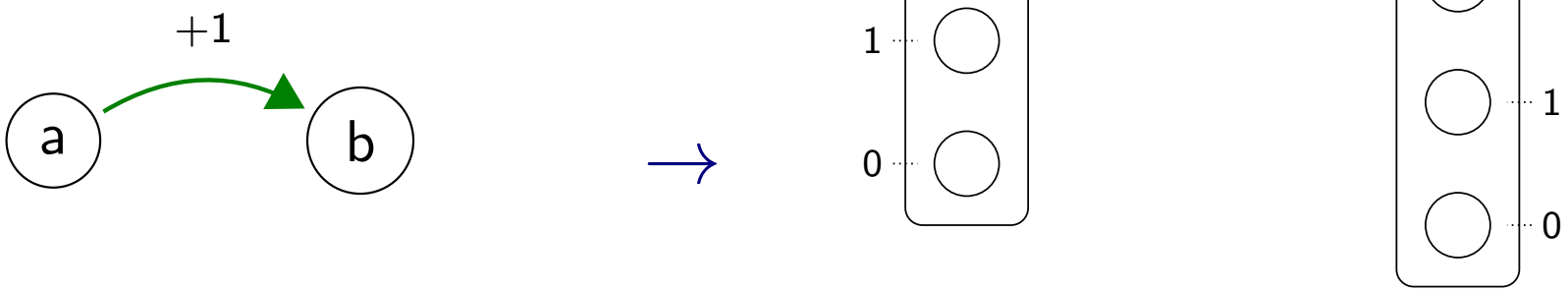
$\to$ b_1 reached with a **very low probability**.

action duration

$\to$ Tests by **simulation** and **model-checking**

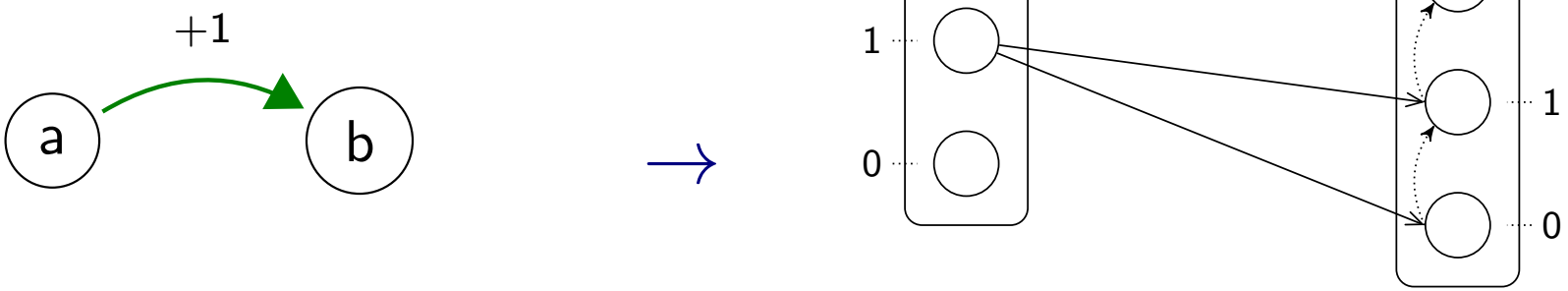
Translation of the Generalized Dynamics

Positive interaction:



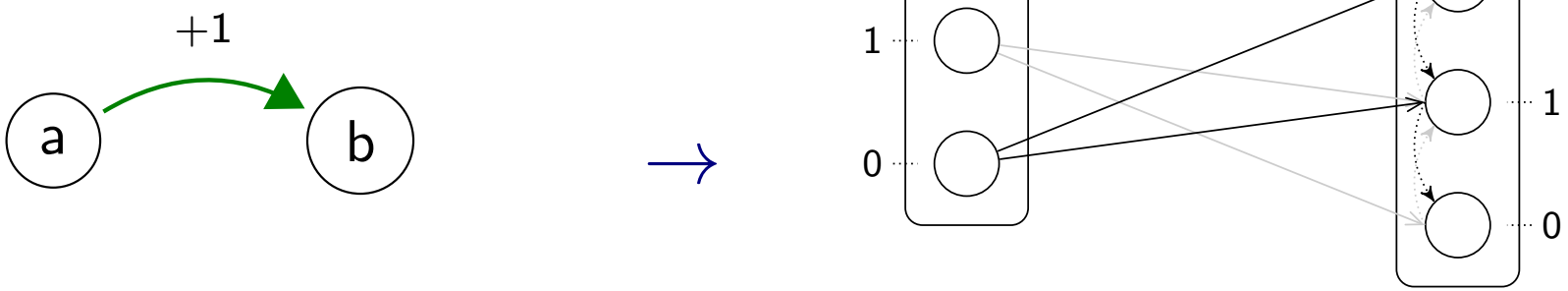
Translation of the Generalized Dynamics

Positive interaction:



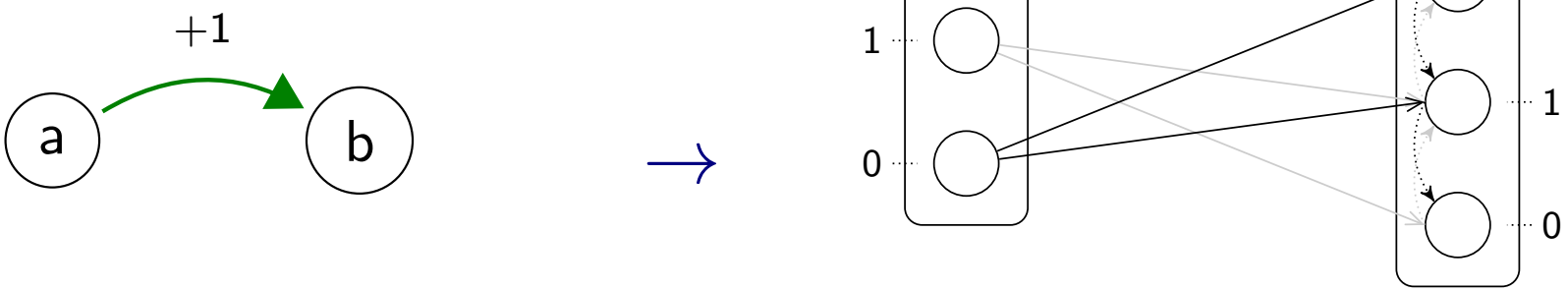
Translation of the Generalized Dynamics

Positive interaction:

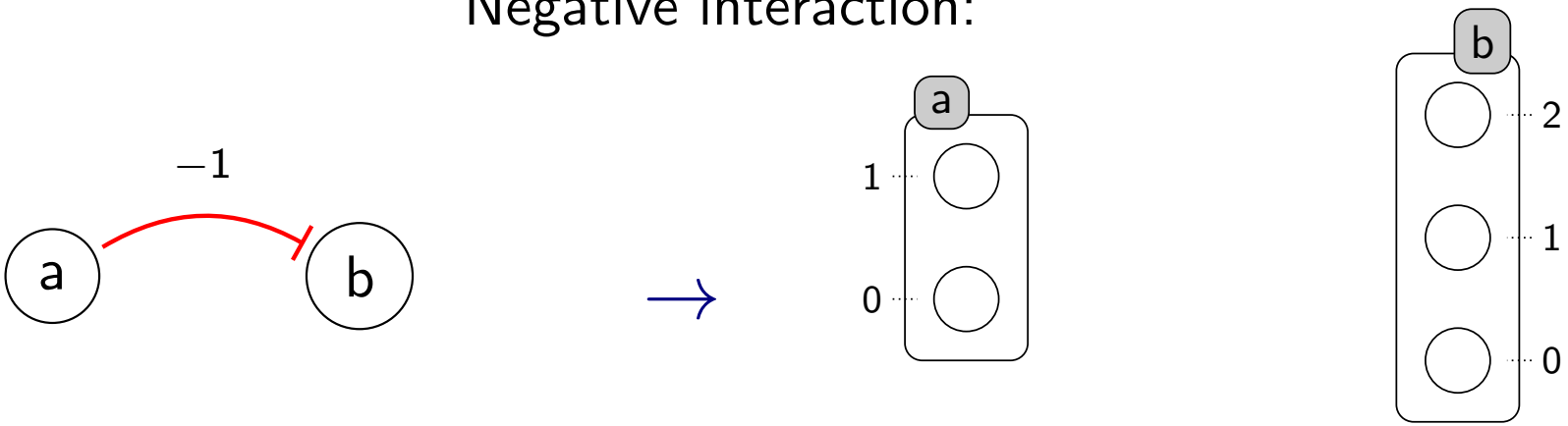


Translation of the Generalized Dynamics

Positive interaction:

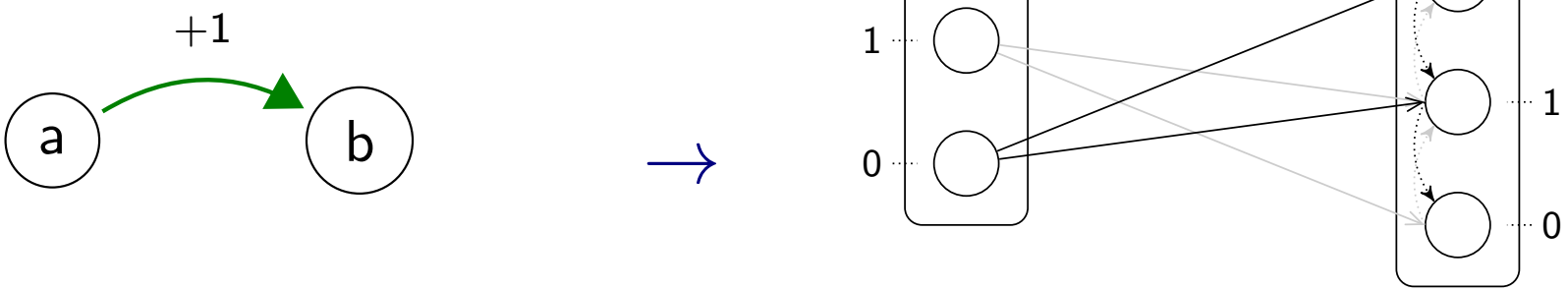


Negative interaction:

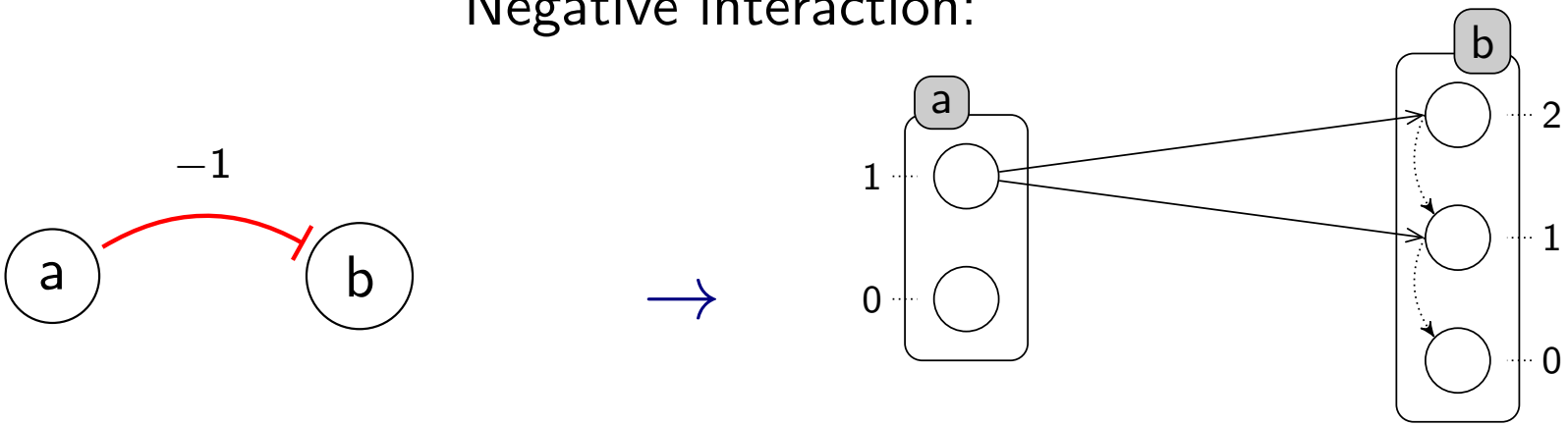


Translation of the Generalized Dynamics

Positive interaction:

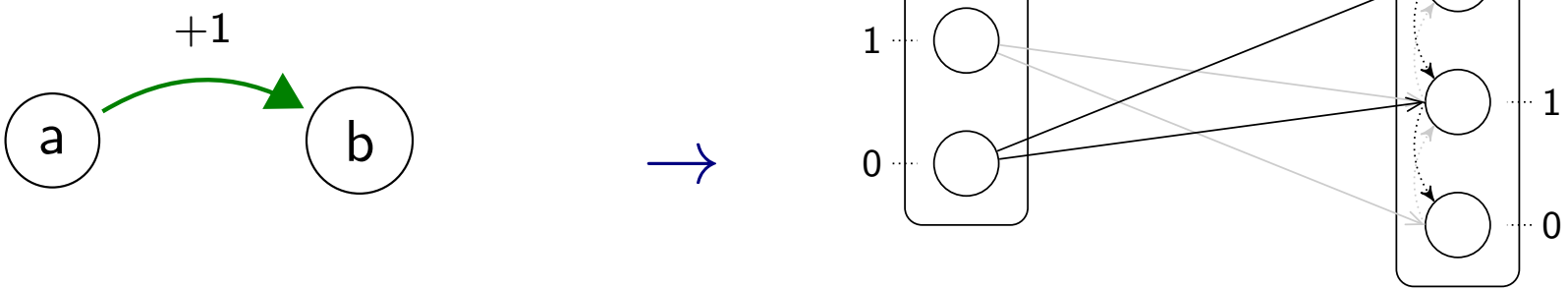


Negative interaction:

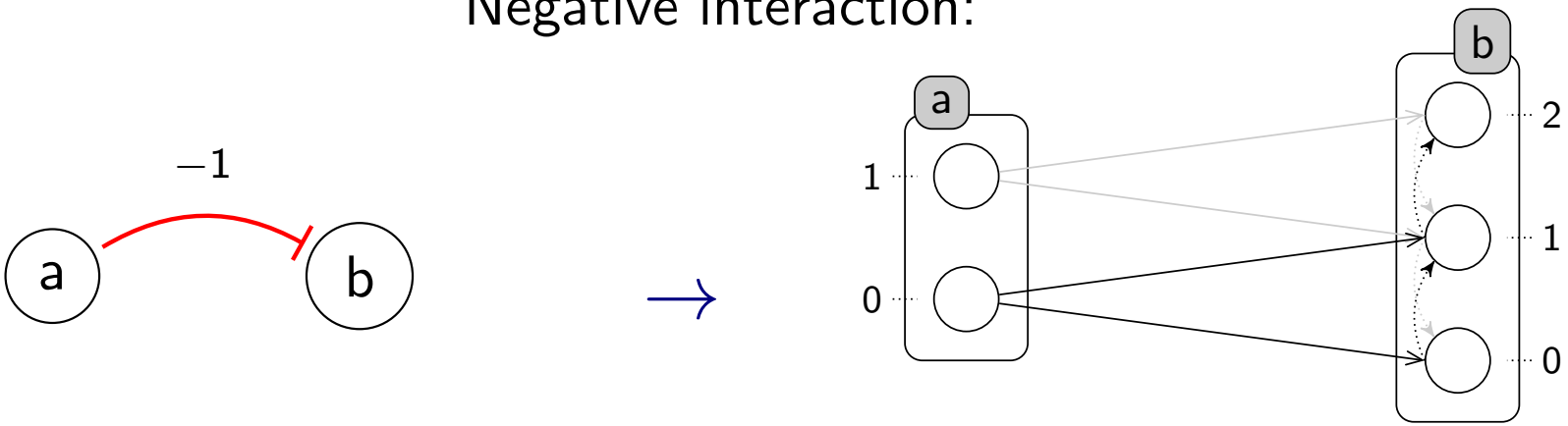


Translation of the Generalized Dynamics

Positive interaction:

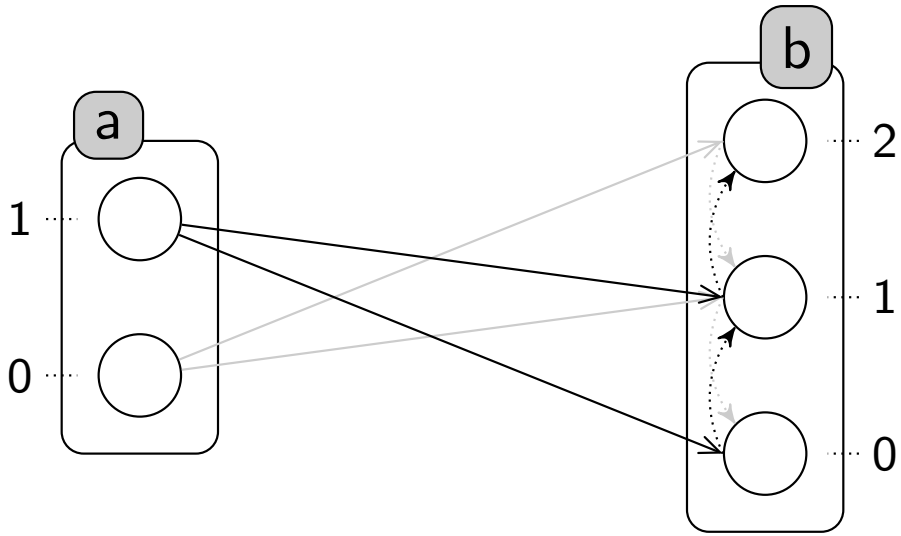


Negative interaction:



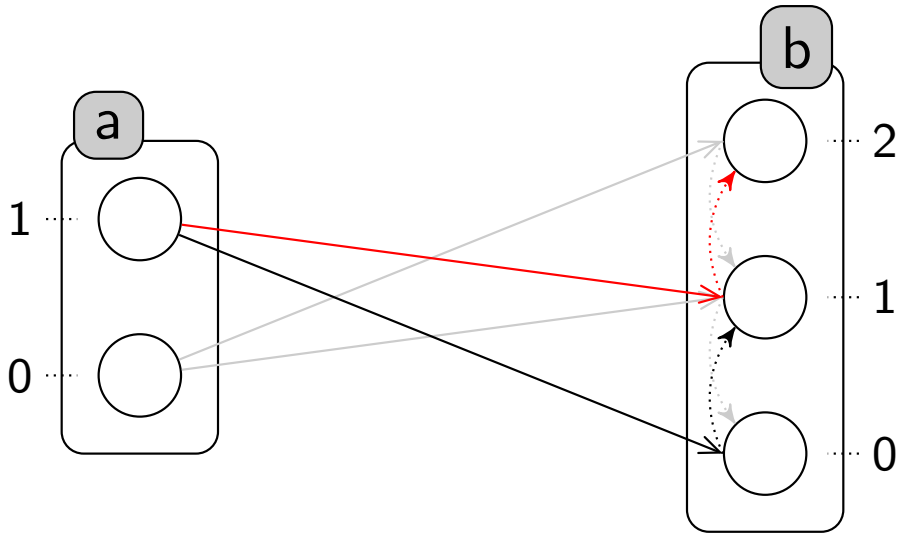
Refining with Actions Removal

Prevent behaviors by deleting **unrealistic actions**



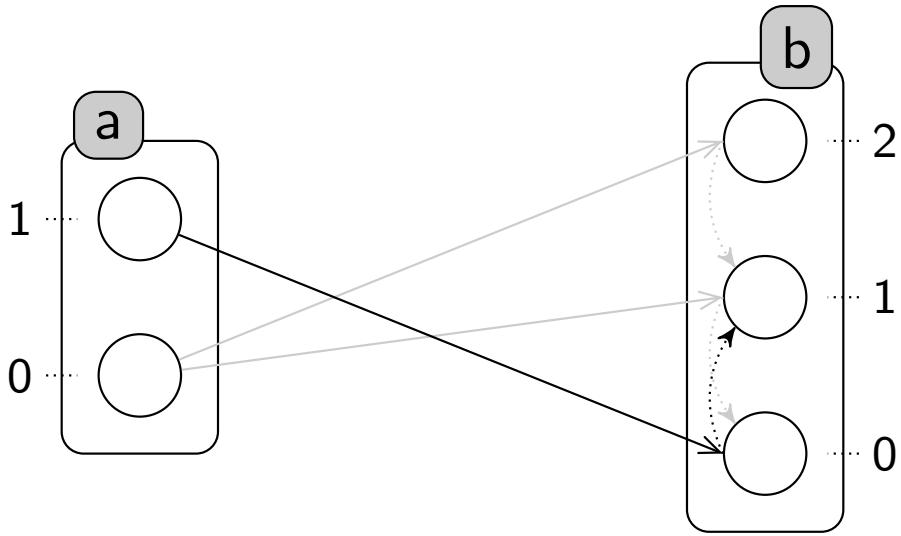
Refining with Actions Removal

Prevent behaviors by deleting **unrealistic actions**



Refining with Actions Removal

Prevent behaviors by deleting **unrealistic actions**

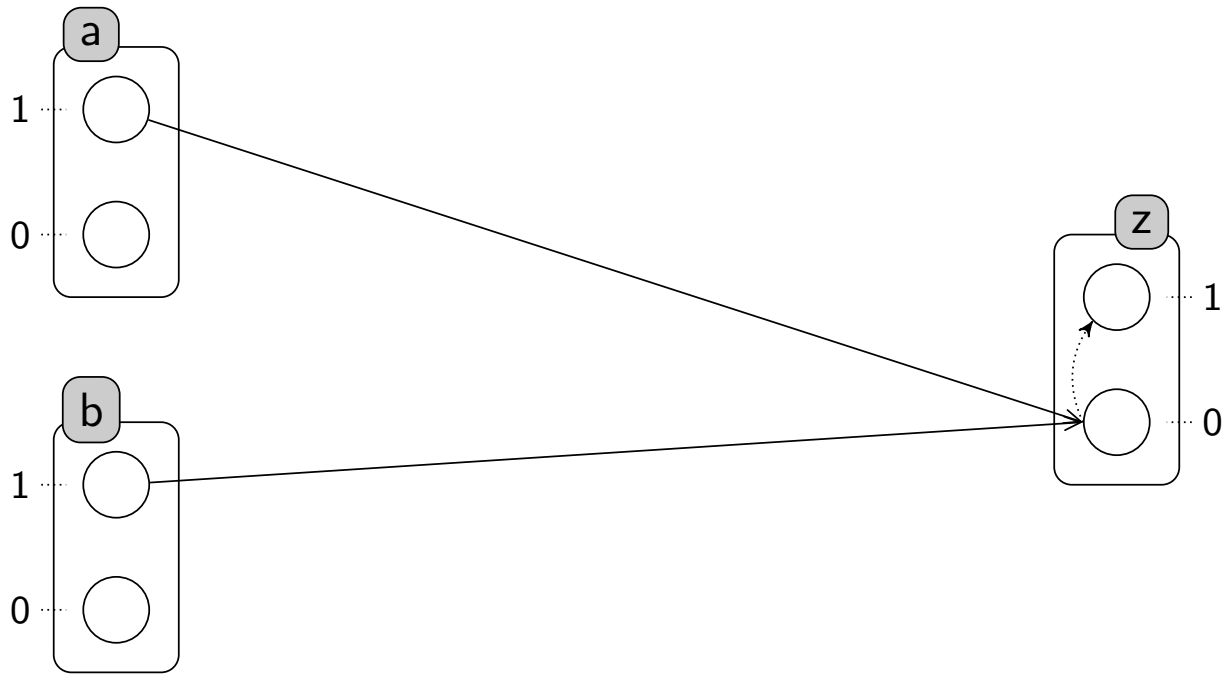


Refining with Cooperation

Allow **cooperation** between two genes

- How to express $(a_1 \wedge b_1) \rightarrow z_0 \uparrow z_1$?

$\rightarrow$ Add a **cooperative sort** reflecting the state of a and b

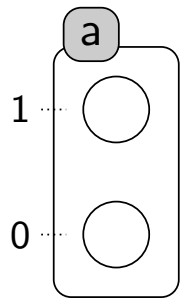


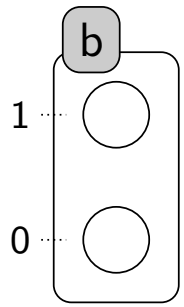
Refining with Cooperation

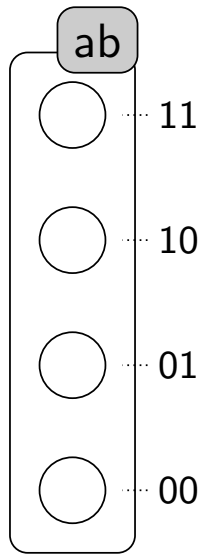
Allow **cooperation** between two genes

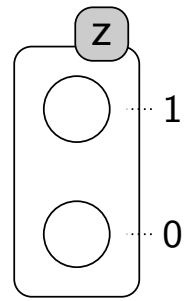
- How to express $(a_1 \wedge b_1) \rightarrow z_0 \overset{\uparrow}{\uparrow} z_1$?

$\rightarrow$ Add a **cooperative sort** reflecting the state of a and b







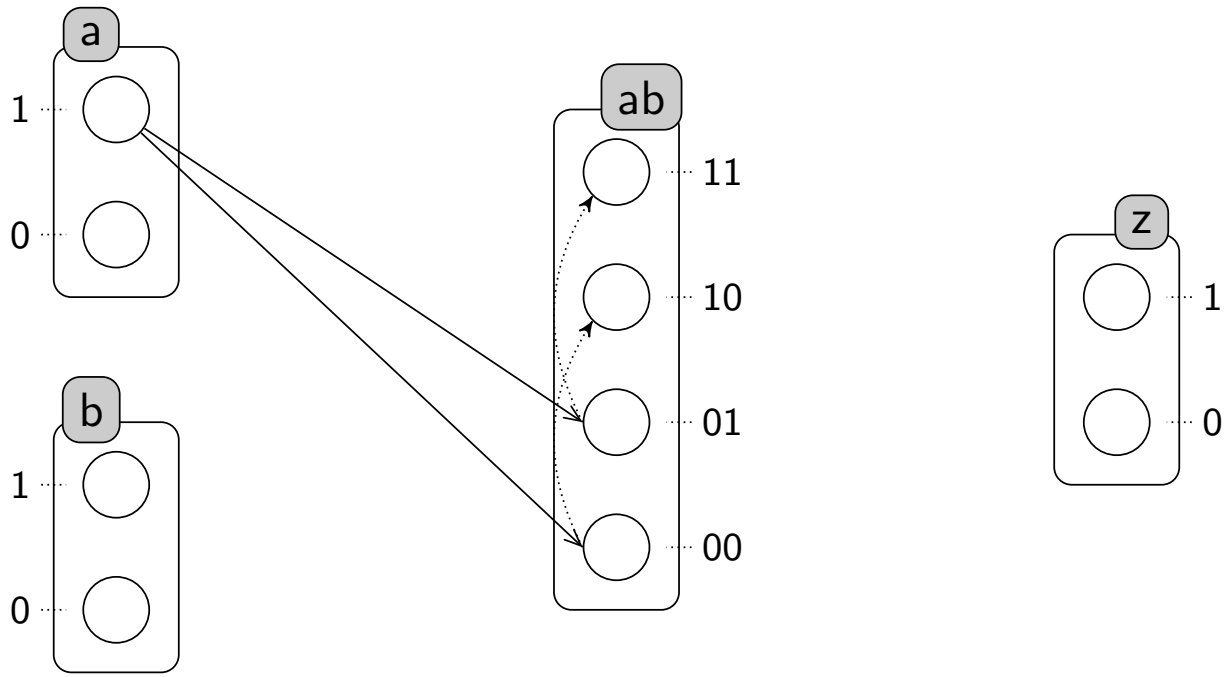


Refining with Cooperation

Allow **cooperation** between two genes

- How to express $(a_1 \wedge b_1) \rightarrow z_0 \uparrow z_1$?

$\rightarrow$ Add a **cooperative sort** reflecting the state of a and b

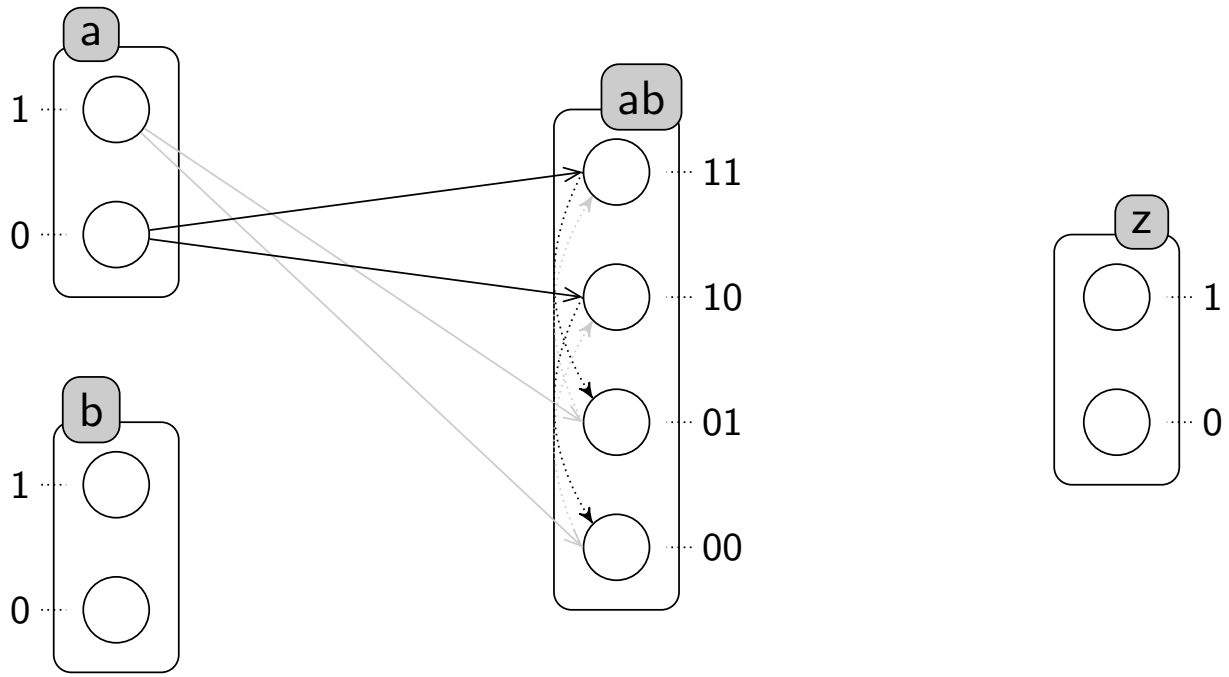


Refining with Cooperation

Allow **cooperation** between two genes

- How to express $(a_1 \wedge b_1) \rightarrow z_0 \overset{\uparrow}{\uparrow} z_1$?

$\rightarrow$ Add a **cooperative sort** reflecting the state of a and b

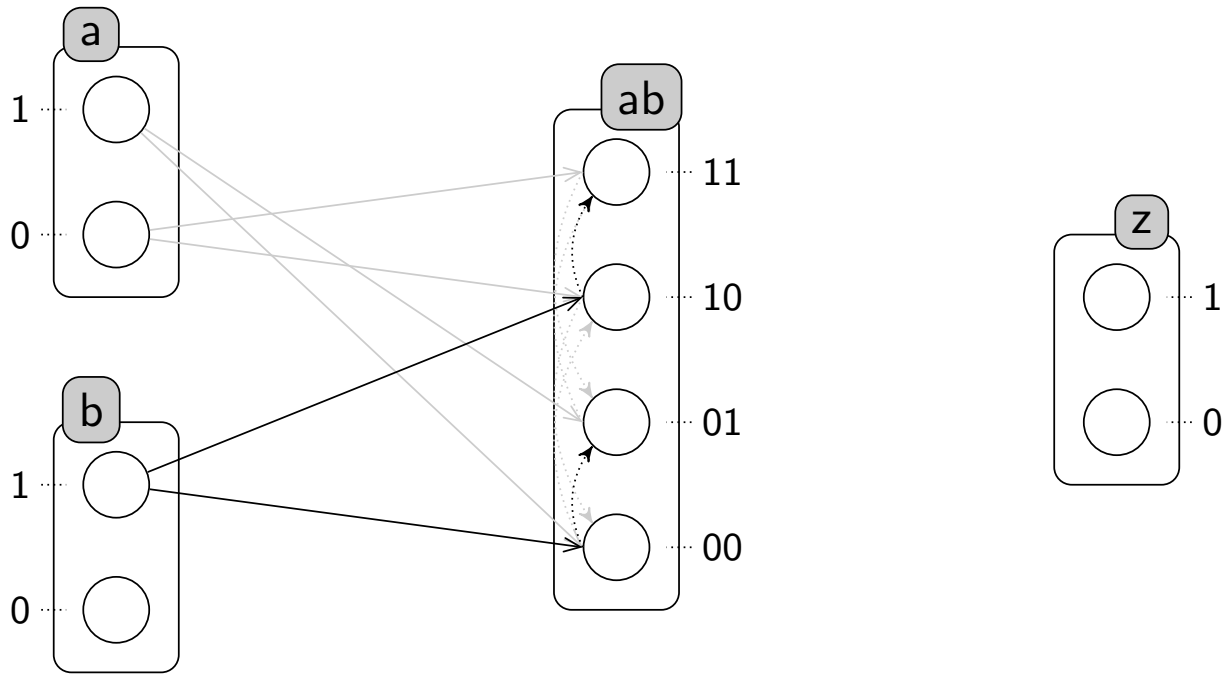


Refining with Cooperation

Allow **cooperation** between two genes

- How to express $(a_1 \wedge b_1) \rightarrow z_0 \overset{\uparrow}{} z_1$?

$\rightarrow$ Add a **cooperative sort** reflecting the state of a and b

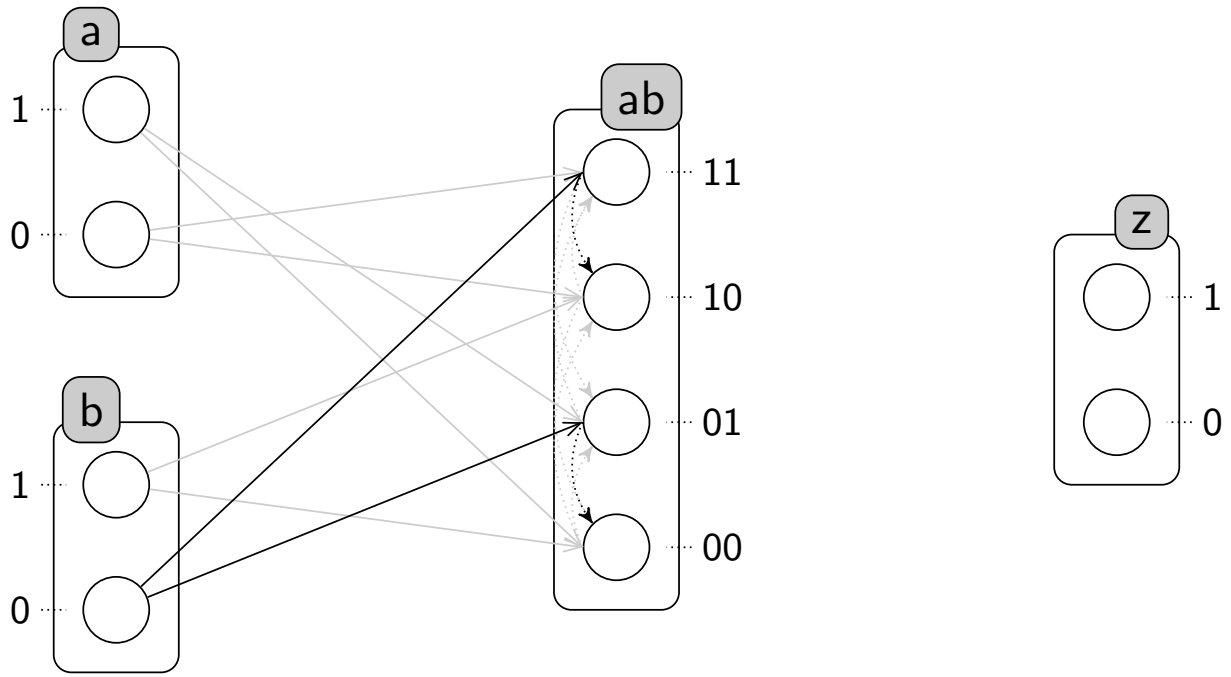


Refining with Cooperation

Allow **cooperation** between two genes

- How to express $(a_1 \wedge b_1) \rightarrow z_0 \uparrow z_1$?

$\rightarrow$ Add a **cooperative sort** reflecting the state of a and b

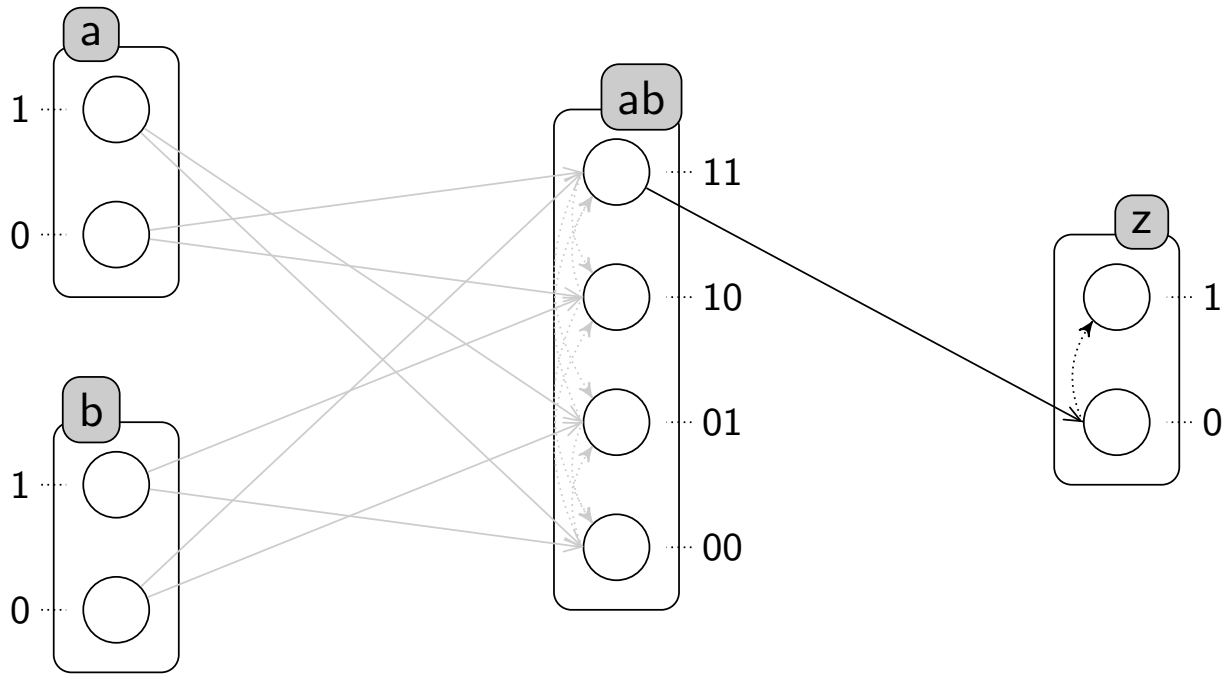


Refining with Cooperation

Allow **cooperation** between two genes

- How to express $(a_1 \wedge b_1) \rightarrow z_0 \overset{\uparrow}{\uparrow} z_1$?

$\rightarrow$ Add a **cooperative sort** reflecting the state of a and b

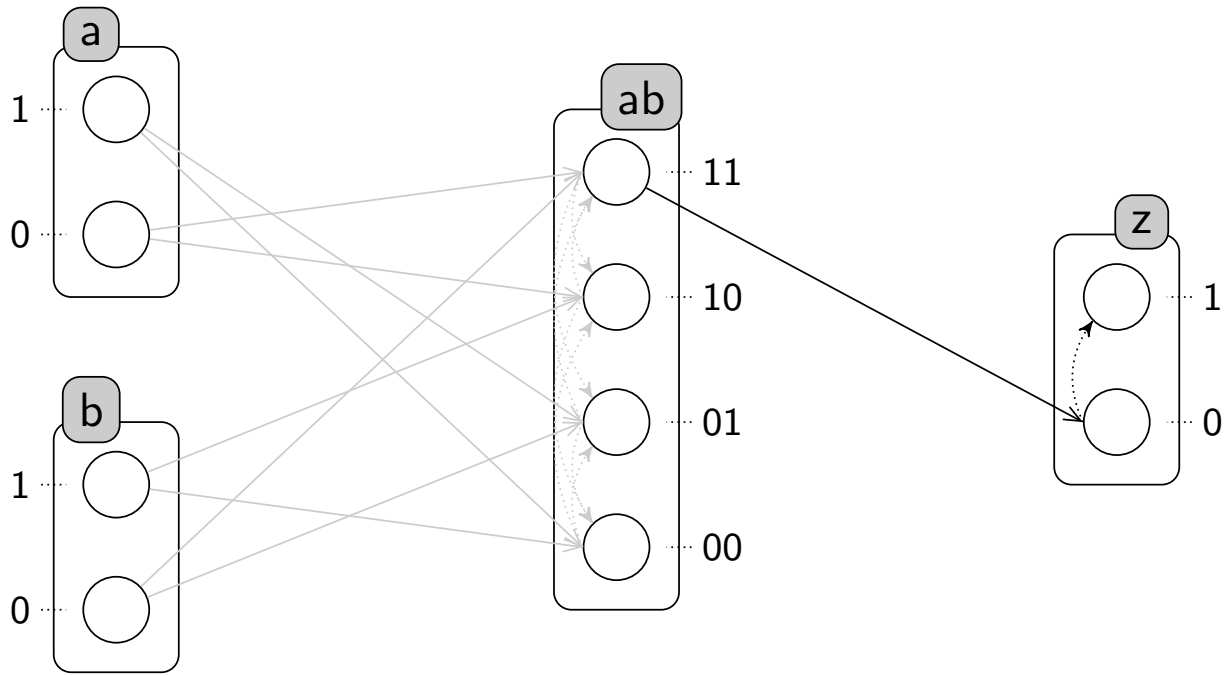


Refining with Cooperation

Allow **cooperation** between two genes

- How to express $(a_1 \wedge b_1) \rightarrow z_0 \Uparrow z_1$?

$\rightarrow$ Add a **cooperative sort** reflecting the state of a and b



$\rightarrow$ Introduces a temporal shift (over-approximation)

Using Process Hitting for Interaction Graphs Study

Motivation

- Interaction Graph is the **historical discrete model** (suitable and widespread in biological research)
- Several tools exist of the analysis of interaction graphs, but the **state graph** is needed for some results $\Rightarrow$ **combinatorial explosion**

Contribution: Process Hitting to study large Biological Regulatory Networks

- Translation from Interaction Graphs + Refining
- **Efficient static analysis**

The Process Hitting modeling

Key features

- **Dynamic** modeling with an **atomistic** point of view
  - → Independent actions
  - → Cooperation modeled with cooperative sorts
- Efficient **static analysis**
  - → Reachability of a process can be computed in **linear time** in the number of sorts
- Useful for the study of **large biological models**
  - → Up to hundreds of sorts

(Future) extensions

- Actions with stochasticity
- Actions with priorities
- Continuous time with clocks?

The Pint Tool

[http://processhitting.wordpress.com]

Features

- Free software (API available for future developments)
- **Textual language** to describe a Process Hitting (GUI currently under development)
- **Implemented tools**:
  - Translations from and to various other models
  - Fixed points research
  - Stochastic simulation
  - Reachability checker

The Pint Tool

[http://processhitting.wordpress.com]

Results and performance (reachability analysis):

Model	sorts	procs	actions	states	Biocham[1]	libddd[2]	PINT
egfr20	35	196	670	2^{64}	[3s-KO]	[1s-150s]	**0.007s**
tcrsig40	54	156	301	2^{73}	[1s-KO]	[0.6s-KO]	**0.004s**
tcrsig94	133	448	1124	2^{194}	KO	KO	**0.030s**
egfr104	193	748	2356	2^{320}	KO	KO	**0.050s**

[1] [Inria Paris-Rocquencourt/Contraintes]

[2] [LIP6/Move]

The Mobyle portal

[http://cardioserve.nantes.inserm.fr/cgi-bin/mobyle/portal.py]

Presentation

- **Web application unifying** tools for systems biology analysis
- Powered by the Mobyle framework
- Project led in the context of the French ANR "BIOTempo" project

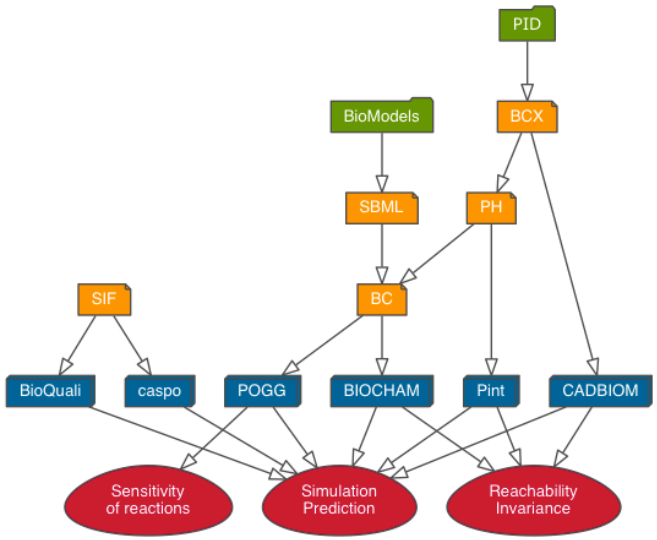


Figure: General architecture of the BIOtempo Mobyle server

The Mobyle portal

[http://cardioserve.nantes.inserm.fr/cgi-bin/mobyle/portal.py]

Presentation

- **Web application unifying** tools for systems biology analysis
- Powered by the Mobyle framework
- Project led in the context of the French ANR "BIOTempo" project

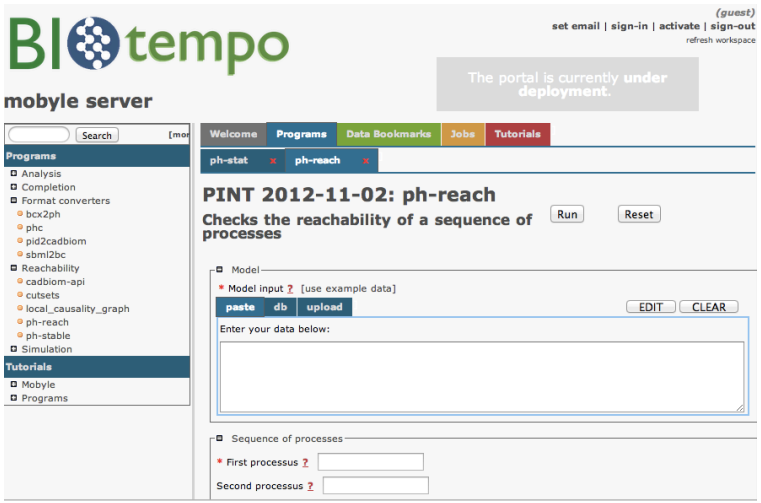
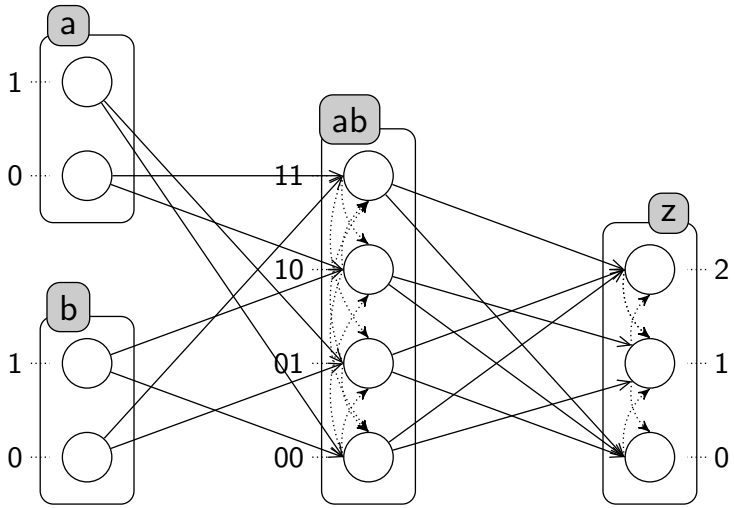


Figure: Screenshot from the BIOtempo Mobyle server: cardioserve.nantes.inserm.fr/cgi-bin/mobyle/portal.py

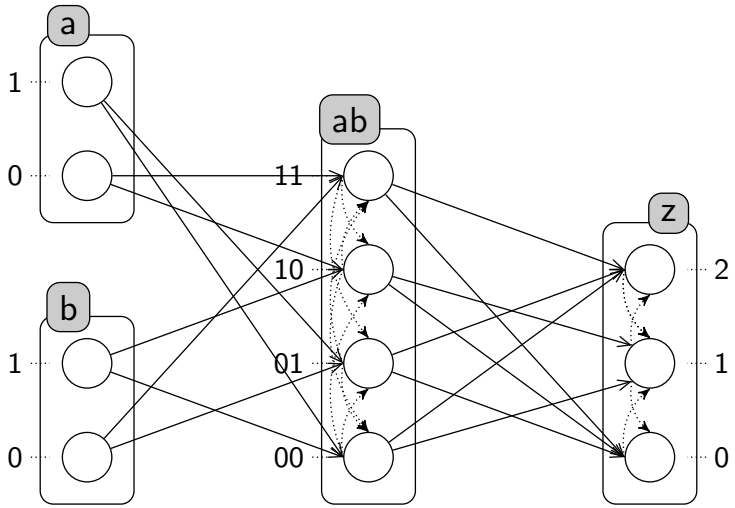
Overview

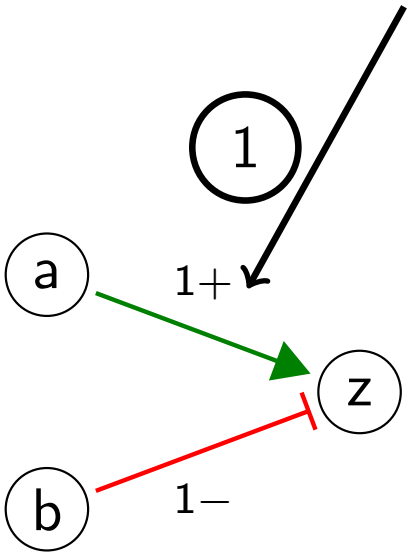
Inferring a BRN with Thomas' parameters



ω	$k_{z,\omega}$
$\varnothing$	1
$\{b\}$	0
$\{a\}$	2
$\{a; b\}$	1

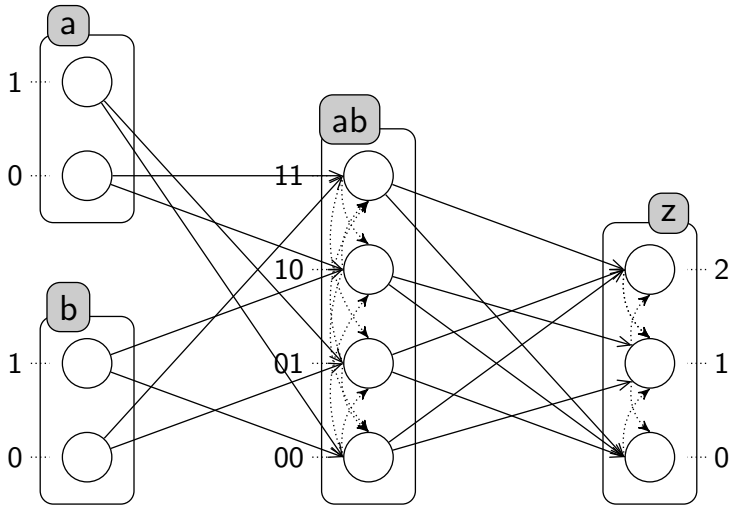
Inferring a BRN with Thomas' parameters

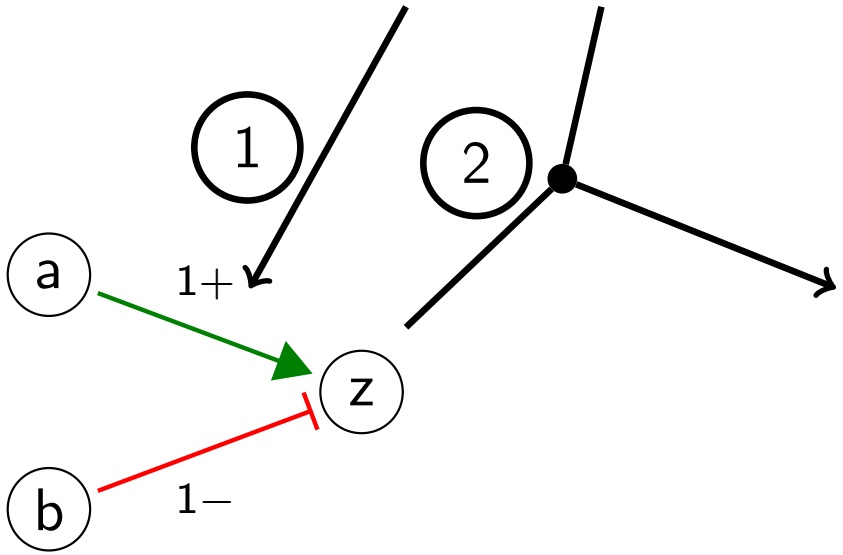




ω	$k_{z,\omega}$
$\varnothing$	1
$\{b\}$	0
$\{a\}$	2
$\{a; b\}$	1

Inferring a BRN with Thomas' parameters

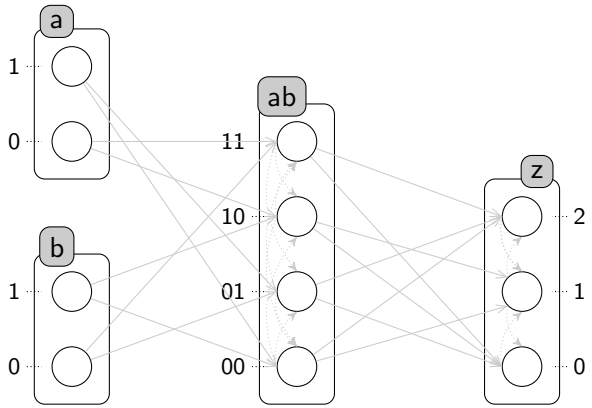


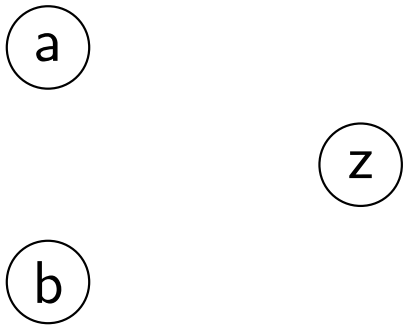


ω	$k_{z,\omega}$
$\varnothing$	1
$\{b\}$	0
$\{a\}$	2
$\{a; b\}$	1

Inferring the Interaction Graph

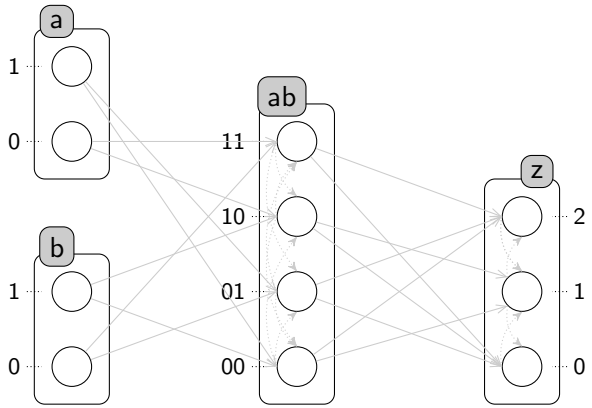
[FPI+12]

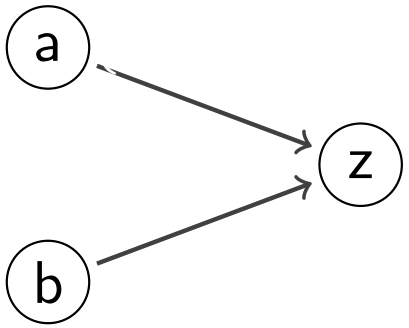




Inferring the Interaction Graph

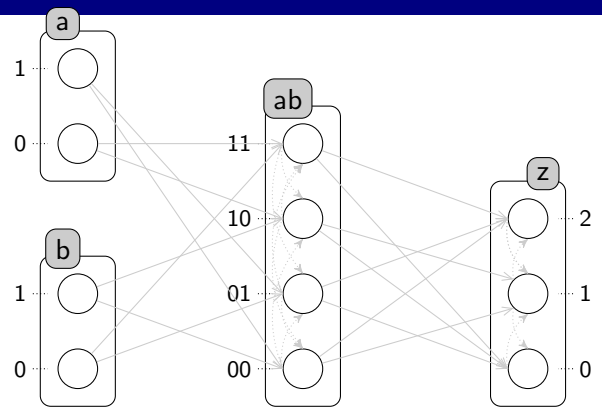
[FPI+12]

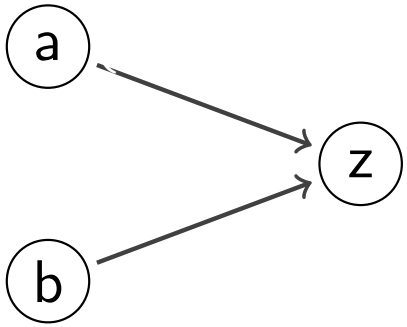




Inferring the Interaction Graph

[FPI+12]



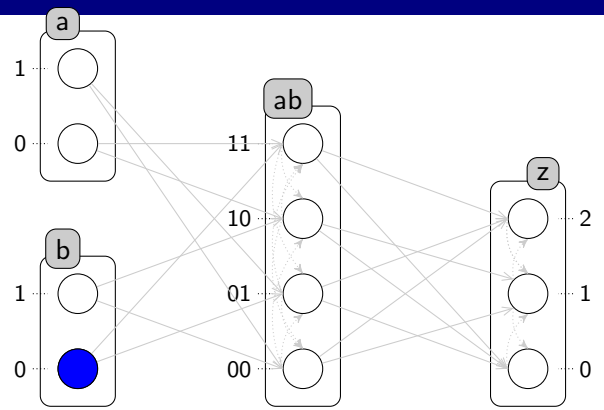


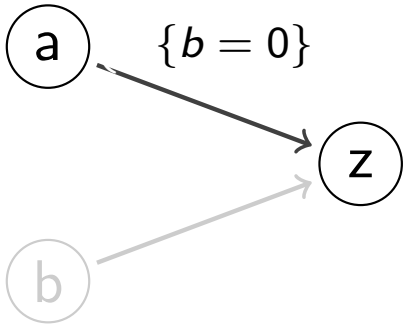
→ **Exhaustive search in all possible configurations**

1. Pick one regulator [a], and choose an active process for all the others [b_0].
2. Change the active process of this regulator [a_0, a_1] and watch the **focal processes**.
3. Conclude locally: $(a_0 \not\uparrow a_1 \Rightarrow z_0 \not\uparrow z_2)$
   $\Rightarrow$ activation $(+)$ & threshold $= 1$.
4. Iterate

Inferring the Interaction Graph

[FPI+12]



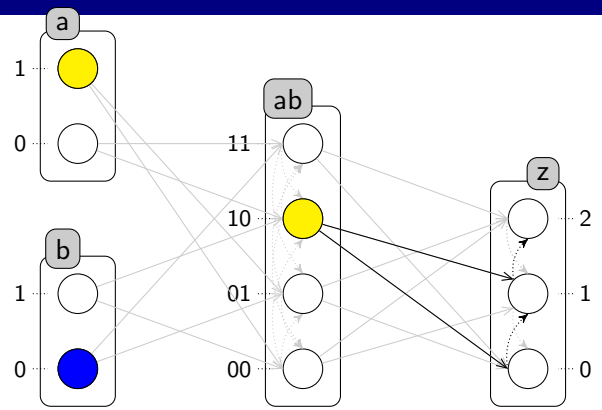


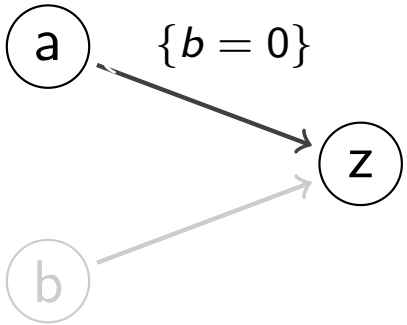
→ **Exhaustive search in all possible configurations**

1. Pick one regulator [a], and choose an active process for all the others [b_0].
2. Change the active process of this regulator [a_0, a_1] and watch the **focal processes**.
3. Conclude locally: $(a_0 \Rsh a_1 \Rightarrow z_0 \Rsh z_2)$
   $\Rightarrow$ activation $(+)$ & threshold $= 1$.
4. Iterate

Inferring the Interaction Graph

[FPI+12]



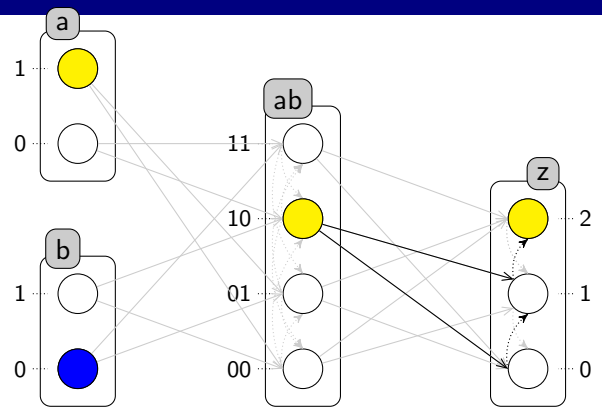


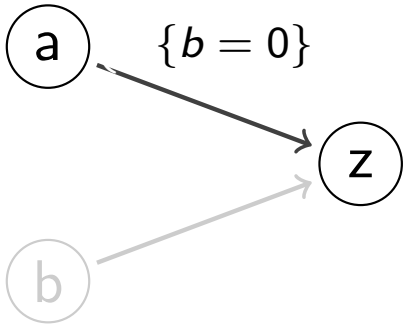
→ **Exhaustive search in all possible configurations**

1. Pick one regulator [a], and choose an active process for all the others [b_0].
2. Change the active process of this regulator [a_0, a_1] and watch the **focal processes**.
3. Conclude locally: $(a_0 \not\vdash a_1 \Rightarrow z_0 \not\vdash z_2)$ $\Rightarrow$ activation $(+)$ & threshold $= 1$.
4. Iterate

Inferring the Interaction Graph

[FPI+12]



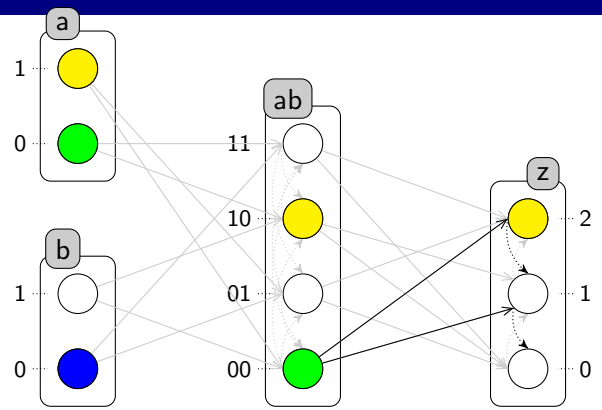


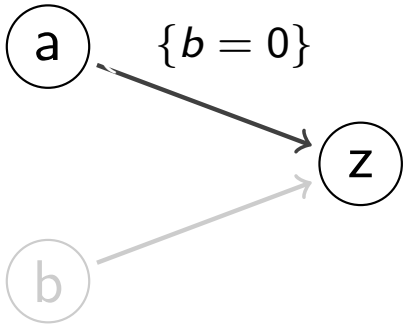
→ **Exhaustive search in all possible configurations**

1. Pick one regulator [a], and choose an active process for all the others [b_0].
2. Change the active process of this regulator [a_0, a_1] and watch the **focal processes**.
3. Conclude locally: $(a_0 \Rsh a_1 \Rightarrow z_0 \Rsh z_2)$ $\Rightarrow$ activation $(+)$ & threshold $= 1$.
4. Iterate

Inferring the Interaction Graph

[FPI+12]



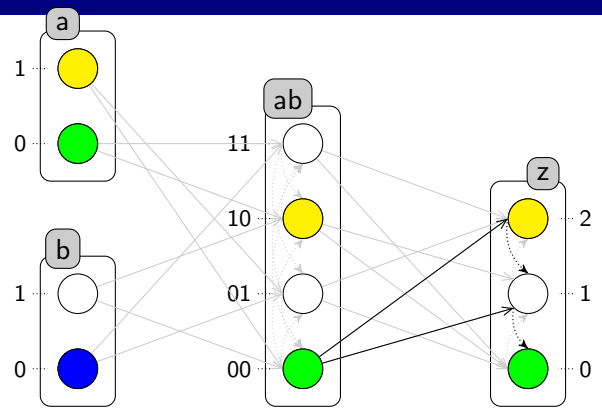


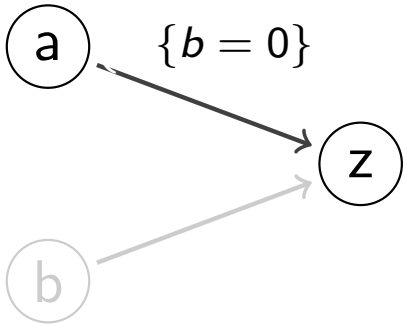
→ **Exhaustive search in all possible configurations**

1. Pick one regulator [a], and choose an active process for all the others [b_0].
2. Change the active process of this regulator [a_0, a_1] and watch the **focal processes**.
3. Conclude locally: $(a_0 \Rsh a_1 \Rightarrow z_0 \Rsh z_2)$ $\Rightarrow$ activation $(+)$ & threshold $= 1$.
4. Iterate

Inferring the Interaction Graph

[FPI+12]



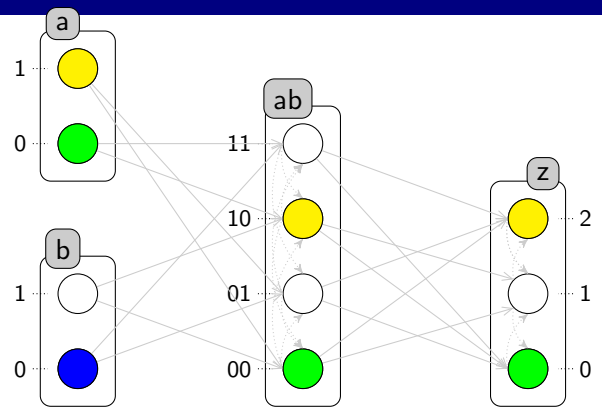


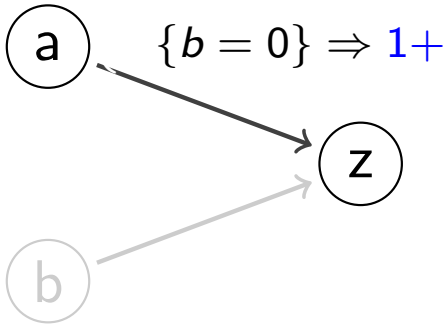
→ **Exhaustive search in all possible configurations**

1. Pick one regulator [a], and choose an active process for all the others [b_0].
2. Change the active process of this regulator [a_0, a_1] and watch the **focal processes**.
3. Conclude locally: $(a_0 \not\vdash a_1 \Rightarrow z_0 \not\vdash z_2)$ $\Rightarrow$ activation $(+)$ & threshold $= 1$.
4. Iterate

Inferring the Interaction Graph

[FPI+12]



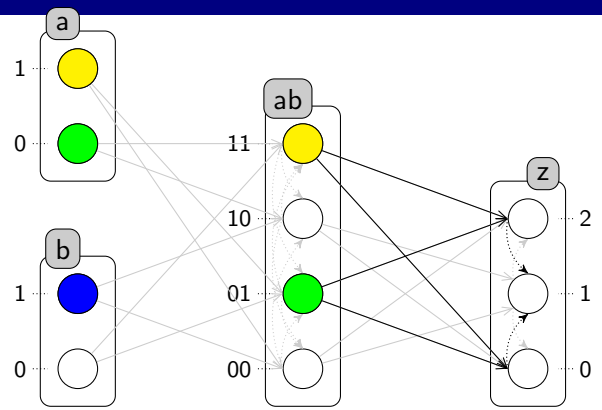


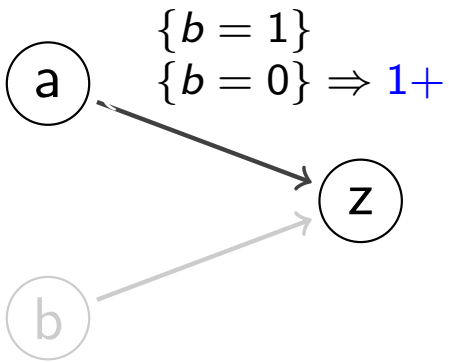
→ **Exhaustive search in all possible configurations**

1. Pick one regulator [a], and choose an active process for all the others [b_0].
2. Change the active process of this regulator [a_0, a_1] and watch the **focal processes**.
3. Conclude locally: $(a_0 \Rsh a_1 \Rightarrow z_0 \Rsh z_2)$
   $\Rightarrow$ activation $(+)$ & threshold $= 1$.
4. Iterate

Inferring the Interaction Graph

[FPI+12]



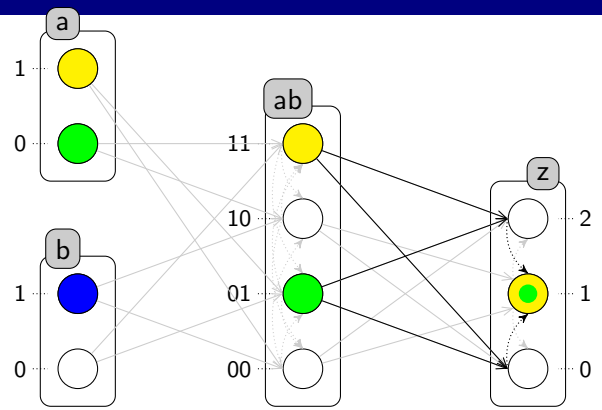


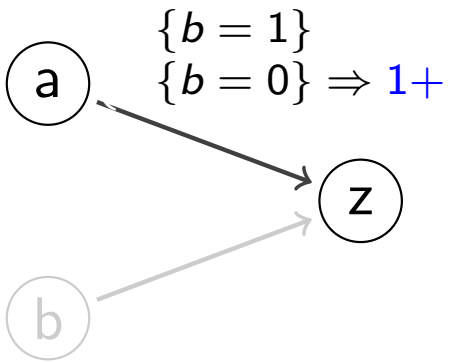
→ **Exhaustive search in all possible configurations**

1. Pick one regulator [a], and choose an active process for all the others [b_0].
2. Change the active process of this regulator [a_0, a_1] and watch the **focal processes**.
3. Conclude locally: $(a_0 \upharpoonright a_1 \Rightarrow z_0 \upharpoonright z_2)$
   $\Rightarrow$ activation $(+)$ & threshold $= 1$.
4. Iterate

Inferring the Interaction Graph

[FPI+12]



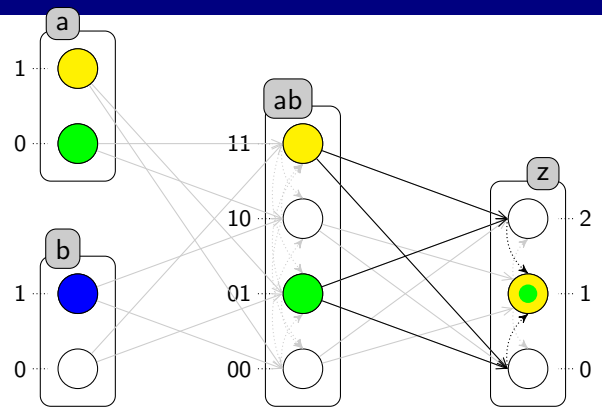


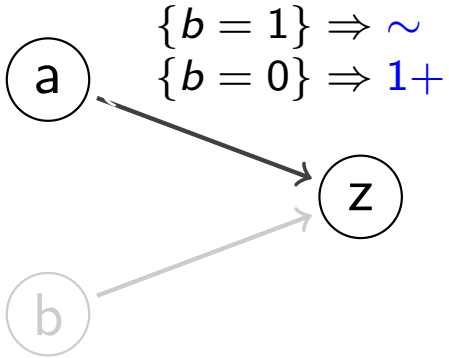
→ **Exhaustive search in all possible configurations**

1. Pick one regulator [a], and choose an active process for all the others [b_0].
2. Change the active process of this regulator [a_0, a_1] and watch the **focal processes**.
3. Conclude locally: $(a_0 \Rsh a_1 \Rightarrow z_0 \Rsh z_2)$
   $\Rightarrow$ activation $(+)$ & threshold $= 1$.
4. Iterate

Inferring the Interaction Graph

[FPI+12]



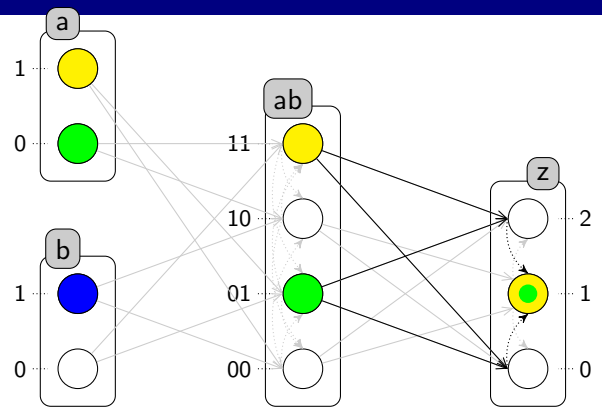


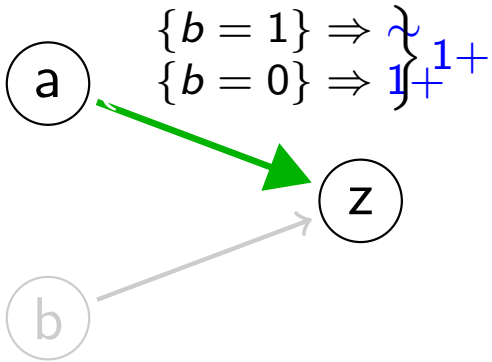
→ **Exhaustive search in all possible configurations**

1. Pick one regulator [a], and choose an active process for all the others [b_0].
2. Change the active process of this regulator [a_0, a_1] and watch the **focal processes**.
3. Conclude locally: $(a_0 \Rsh a_1 \Rightarrow z_0 \Rsh z_2)$
   $\Rightarrow$ activation $(+)$ & threshold $= 1$.
4. Iterate

Inferring the Interaction Graph

[FPI+12]



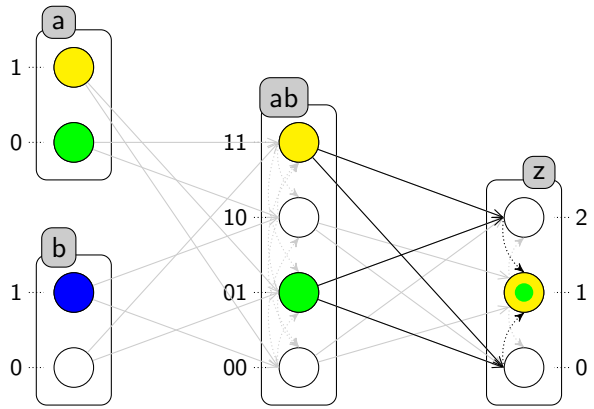


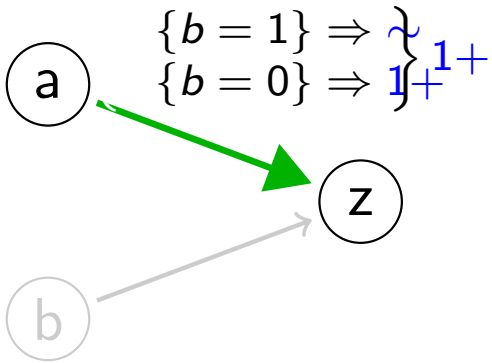
→ **Exhaustive search in all possible configurations**

1. Pick one regulator [a], and choose an active process for all the others [b_0].
2. Change the active process of this regulator [a_0, a_1] and watch the **focal processes**.
3. Conclude locally: $(a_0 \nearrow a_1 \Rightarrow z_0 \nearrow z_2)$
   $\Rightarrow$ activation $(+)$ & threshold $= 1$.
4. Iterate and conclude globally.

Inferring the Interaction Graph

[FPI+12]





Problematic cases:

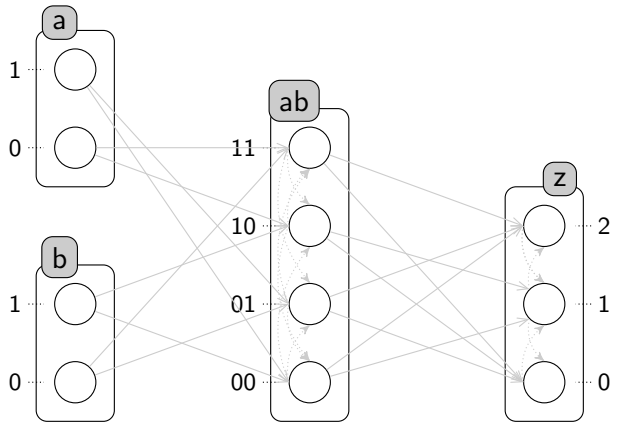
→ No focal processes (**cycle**)

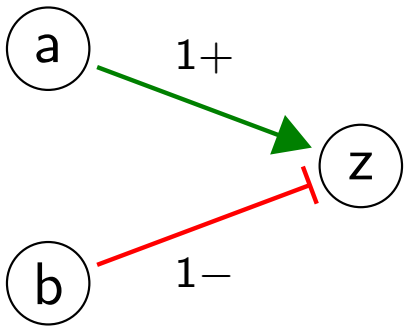
→ Opposite influences ($+$ & $-$)

$\Rightarrow$ **Unsigned edge**

Inferring Parameters

[FPI+12]





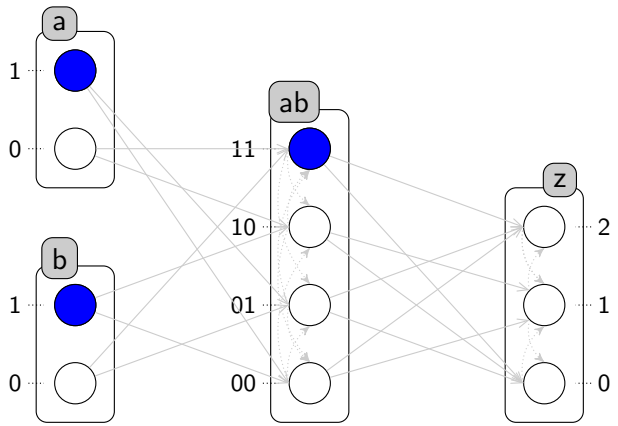
ω		$k_{z,\omega}$
a	b	
$-$	$+$	
$-$	$-$	
$+$	$+$	
$+$	$-$	1

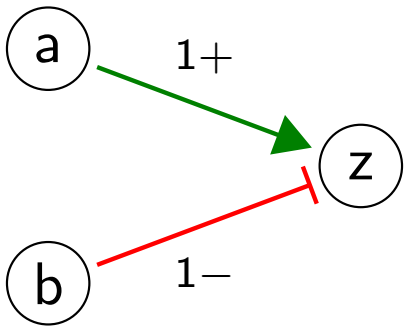
1. For each configuration of resources $[\omega = \{a^+, b^-\}]$ find the **focal processes**. If possible, conclude. $[k_{z,\{a^+,b^-\}} = 1]$

   Inconclusive cases:
   - Behavior cannot be represented as a BRN
   - Lack of cooperation (no focal processes)

Inferring Parameters

[FPI+12]



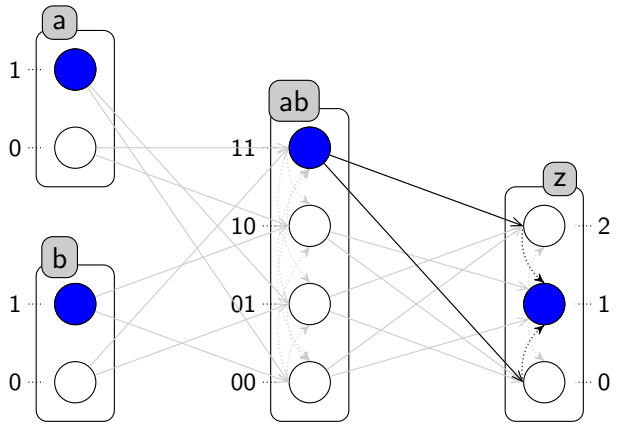


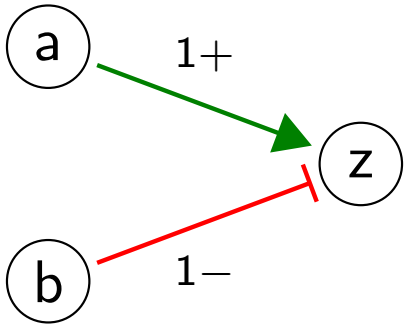
ω		$k_{z,\omega}$
a	b	
−	+	
−	−	
+	+	
+	−	1

1. For each configuration of resources $[\omega = \{a^+, b^-\}]$
   find the **focal processes**. If possible, conclude. $[k_{z,\{a^+,b^-\}} = 1]$
   Inconclusive cases:
   - Behavior cannot be represented as a BRN
   - Lack of cooperation (no focal processes)

Inferring Parameters

[FPI+12]





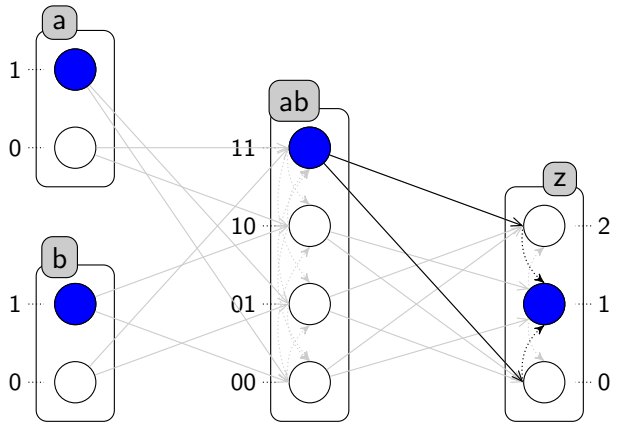
ω		$k_{z,\omega}$
a	b	
−	+	
−	−	
+	+	
+	−	1

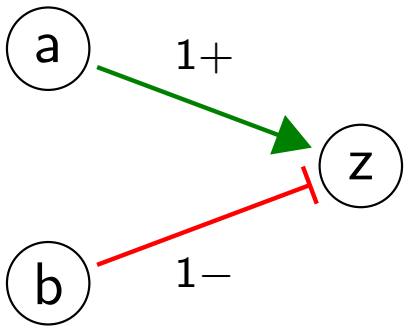
1. For each configuration of resources $[\omega = \{a^+, b^-\}]$ find the **focal processes**. If possible, conclude. $[k_{z,\{a^+,b^-\}} = 1]$

   Inconclusive cases:
   - Behavior cannot be represented as a BRN
   - Lack of cooperation (no focal processes)

Inferring Parameters

[FPI+12]





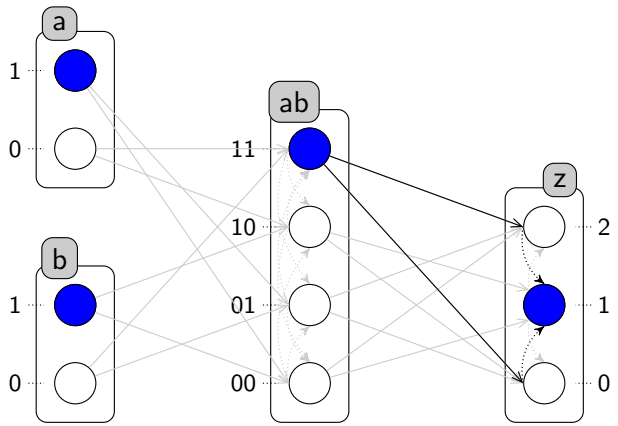
ω		$k_{z,\omega}$
a	b	
−	+	
−	−	
+	+	
+	−	1

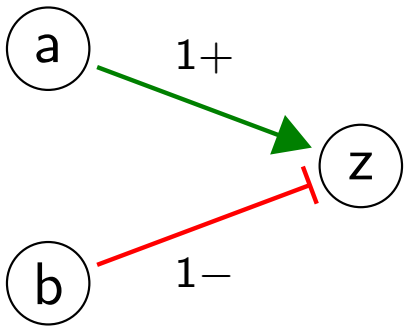
1. For each configuration of resources $[\omega = \{a^+, b^-\}]$
   find the **focal processes**. If possible, conclude. $[k_{z,\{a^+,b^-\}} = 1]$

   Inconclusive cases:
   - Behavior cannot be represented as a BRN
   - Lack of cooperation (no focal processes)

Inferring Parameters

[FPI+12]



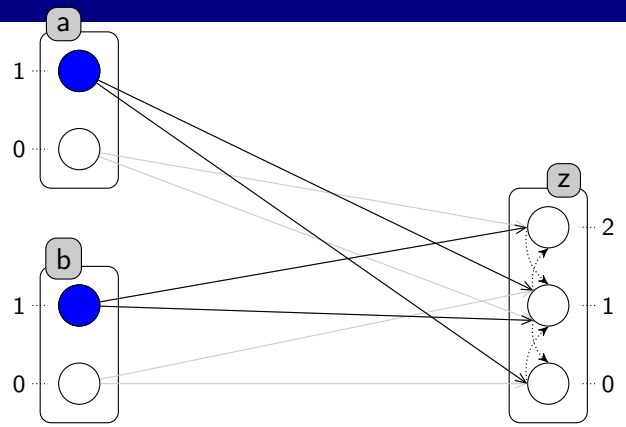


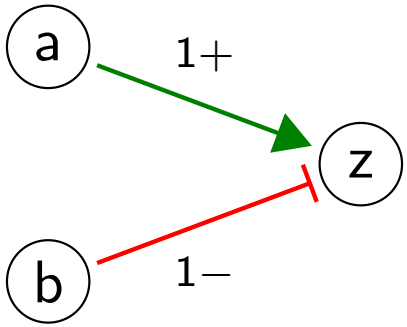
ω		$k_{z,\omega}$
a	b	
−	+	
−	−	
+	+	
+	−	1

1. For each configuration of resources $[\omega = \{a^+, b^-\}]$ find the **focal processes**. If possible, conclude. $[k_{z,\{a^+,b^-\}} = 1]$

   Inconclusive cases:
   - Behavior cannot be represented as a BRN
   - Lack of cooperation (no focal processes)

Inferring Parameters





ω		$k_{z,\omega}$
a	b	
$-$	$+$	?
$-$	$-$	0
$+$	$+$	2
$+$	$-$	?

1. For each configuration of resources $[\omega = \{a^+, b^-\}]$ find the **focal processes**. If possible, conclude. $[k_{z,\{a^+,b^-\}} = 1]$
   Inconclusive cases:
   - Behavior cannot be represented as a BRN
   - Lack of cooperation (no focal processes)

2. If some parameters could not be inferred, enumerate all admissible parametrizations, regarding **biological constraints** and **the dynamics of the Process Hitting** $\Rightarrow k_{z,\{a^+,b^-\}} \in \{0;1;2\}$; $k_{z,\{a^-,b^+\}} \in \{0;1;2\}$

Overview

Implementation

Workflow:

- Read and translate the models with **OCaml**
  - → Uses the existing free library **Pint**
  - → Documentation + examples:

  `http://processhitting.wordpress.com/`
- Express the problem in **ASP** (logic programming)
  - → Solve with **Clingo** (**Gringo** + **Clasp**)

Model specifications				IG inference		Parameters inferenc	
Name	S+CS	P	A	Δt	Edges	Δt	Paramet
[EGFR20]	**20**+22	152	399	**1s**	50	**1s**	191
[TCRSIG40]	**40**+14	156	301	**1s**	54	**1s**	143
[TCRSIG94]	**94**+39	448	1124	**13s**	169	∞	2.10^9
[EGFR104]	**104**+89	748	2356	**4min**	241	**1min 30s**	$1.10^6/2.1$

S = Sorts CS = Cooperative sorts P = Processes A = Actions

[EGFR20]: Epidermal Growth Factor Receptor, by Özgür Sahin et al.

[EGFR104]: Epidermal Growth Factor Receptor, by Regina Samaga et al.

Implementation

Workflow:

- Read and translate the models with **OCaml**
  - → Uses the existing free library **Pint**
  - → Documentation + examples:

  `http://processhitting.wordpress.com/`
- Express the problem in **ASP** (logic programming)
  - → Solve with **Clingo** (**Gringo** + **Clasp**)

Model specifications				IG inference		Parameters inferenc	
Name	S+CS	P	A	Δt	Edges	Δt	Paramet
[EGFR20]	**20**+22	152	399	**1s**	50	**1s**	191
[TCRSIG40]	**40**+14	156	301	**1s**	54	**1s**	143
[TCRSIG94]	**94**+39	448	1124	**13s**	169	∞	2.10^9
[EGFR104]	**104**+89	748	2356	**4min**	241	**1min 30s**	$1.10^6/2.1$

S = Sorts CS = Cooperative sorts P = Processes A = Actions

[EGFR20]: Epidermal Growth Factor Receptor, by Özgür Sahin et al.
[EGFR104]: Epidermal Growth Factor Receptor, by Regina Samaga et al.

Summary

Process Hitting and ASP

- Inference of the **complete Interaction Graph**
- Inference of the **possibly partial Parametrization**
- Enumerate all full & **admissible Parametrizations**

$\rightarrow$ Exhaustive approaches

Complexity: linear in the number of genes, exponential in the number of regulators of one gene

Summary

Contribution: new translation Process Hitting $\rightsquigarrow$ René Thomas

- $\rightarrow$ New **formal link** between the two models
- $\rightarrow$ More **visibility** to the Process Hitting
- $\rightarrow$ Inference approach that takes benefit from both the Process Hitting compact structure and the power of ASP

Further work

Models and algorithms

- Add **priorities** in the Process Hitting framework and adapt the static analyses approaches for this enriched model ($\Rightarrow$ paper currently submitted at CS2Bio'13)
- From priorities to **quantitative timing information**
- Connect Process Hitting compact structure with **decomposition techniques** in continuous approaches [ACC12] ($\Rightarrow$ paper currently submitted at CMSB'13)

Application

- Use the approach for the analysis of **larger** biological networks
- Contribute to the **discovery** of biological regulatory networks based on biological data
- Study key properties (e.g. concept of **resilience**)


Amine Ammar, Elias Cueto, and Francisco Chinesta.
Reduction of the chemical master equation for gene regulatory networks using proper generalized decompositions.
International Journal for Numerical Methods in Biomedical Engineering, 28(9):960–973, 2012.


Laurence Calzone, François Fages, and Sylvain Soliman.
Biocham: an environment for modeling biological systems and formalizing experimental knowledge.
Bioinformatics, 22(14):1805–1807, 2006.


Claudine Chaouiya, Aurélien Naldi, and Denis Thieffry.
Logical Modelling of Gene Regulatory Networks with GINsim., volume 804, pages 463–79.
2012 2012.


Maxime Folschette, Loïc Paulevé, Katsumi Inoue, Morgan Magnin, and Olivier Roux.
Concretizing the process hitting into biological regulatory networks.

In Proceedings of the 10th international conference on Computational Methods in Systems Biology, CMSB'12, pages 166–186, Berlin, Heidelberg, 2012. Springer-Verlag.


Z. Khalis, J.-P. Comet, A. Richard, and G. Bernot.
The SMBioNet method for discovering models of gene regulatory networks.
Genes, Genomes and Genomics, 3(special issue 1):15–22, 2009.


Loïc Paulevé, Morgan Magnin, and Olivier Roux.
Refining dynamics of gene regulatory networks in a stochastic π-calculus framework.
In Transactions on Computational Systems Biology XIII, volume 6575 of Lecture Notes in Comp Sci, pages 171–191. Springer, 2011.


Loïc Paulevé, Morgan Magnin, and Olivier Roux.
Tuning Temporal Features within the Stochastic π-Calculus.
IEEE Transactions on Software Engineering, 37(6):858–871, 2011.


Loïc Paulevé, Morgan Magnin, and Olivier Roux.

Static analysis of biological regulatory networks dynamics using abstract interpretation.
Mathematical Structures in Computer Science, 22(04):651–685, 2012.


Loïc Paulevé and Adrien Richard.
Topological Fixed Points in Boolean Networks.
Comptes Rendus de l'Académie des Sciences - Series I - Mathematics, 348(15-16):825 – 828, 2010.


Adrien Richard, Jean-Paul Comet, and Gilles Bernot.
R. thomas' modeling of biological regulatory networks: Introduction of singular states in the qualitative dynamics.
Fundam. Inform., 65(4):373–392, 2005.


R. Thomas, A.M. Gathoye, and L. Lambert.
A complex control circuit. regulation of immunity in temperate bacteriophages.
Eur J Biochem, 71(1):211–27, 1976.


Louis-Marie Traonouez, Didier Lime, and Olivier (H.) Roux.

Parametric model-checking of stopwatch petri nets.
Journal of Universal Computer Science, 15(17):3273–3304, December 2009.

D. A. Ouattara W. Abou-jaoudé and M. Kaufman.
From structure to dynamics : Frequency tuning in the p53-mdm2 network. i. logical approach.
Journal of Theoretical Biology, 258:561–577, 2009.