Towards the use of Process Hitting to tackle biological observations inconsistent with background knowledge

Joint work: Yoshitaka Yamamoto (1), Morgan Magnin (2,3,4), Maxime Folschette (2), Katsumi Inoue (3)

(1) University of Yamanashi, Yamanashi, Japan
(2) Ecole Centrale de Nantes, IRCCyN, Nantes, France
(3) National Institute of Informatics, Tôkyô, Japan
(4) JSPS Fellow

NII-Yamanashi-LRI Workshop
2014/10/07

Motivations

- Given an existing network (**background knowledge**) and a (**new**) **observation** that is **inconsistent** with this network, how can we automatically design **the minimal set of missing actions that can mimic the observation**?

- Process Hitting is an efficient framework to cope with **large** networks (~ 100 components)

Motivations

- Our proposition: design a method taking advantage of the Process Hitting methods to address the **completion of networks with inconsistent observations**

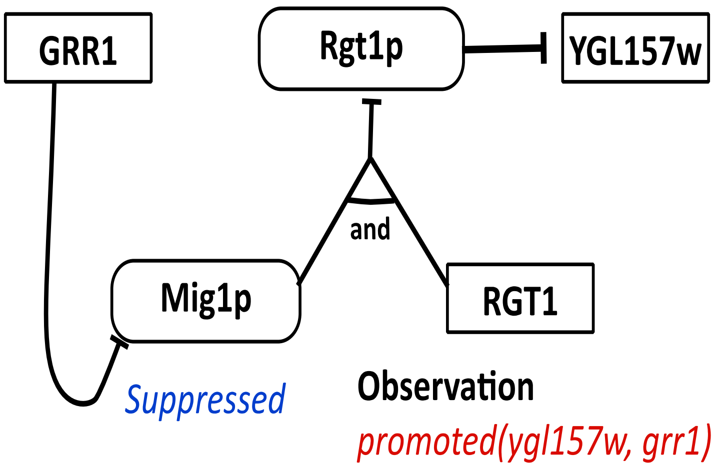
- Restrictions w.r.t. current work:
  - Consider **only addition of actions**, not removal of actions
  - Modeling of the evolution of a gene expression in case of ko w.r.t. wild type, under **steady state assumption**

Overview

- Motivating example
- Reminder about the Process Hitting framework
- 4-level based logics and associated truth tables
- Translating 4-level based models into Process Hitting
- Further discussions

Motivating example

- Background theory ***B***: Boolean network consisting of the three Boolean functions
  - Mig1p = not GRR1
  - Rgt1p = not (Mig1p & RGT1)
  - YGL157w = not Rgt1p
- Observation ***O***:



When GRR1 is ko, then the gene expression of YGL157w decreases, i.e.:

When the gene expression of GRR1 decreases, the gene expression of YGL157w also decreases.

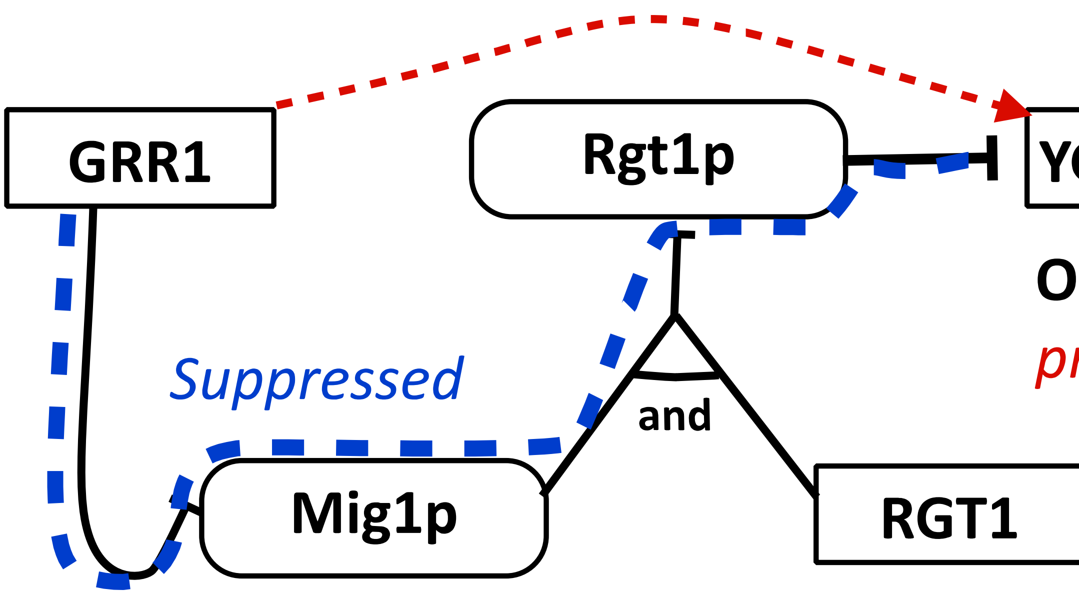
(we write it by ***promoted(ygl157w, grr1)***)

Inconsistency between B and O

- Given the following initial state, we meet the fact that the gene expression of YGL157w decreases

< **GRR1 = -1**, Mig1p = 0, Rgt1p = 0, RGT1 = 0, YGL157w = 0 >

$\Rightarrow$ This is *inconsistent* with the observation…



Observation

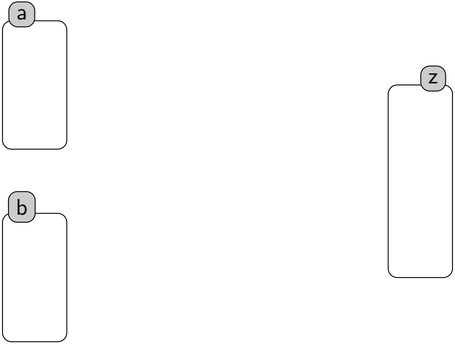
promoted(ygl157w, grr1)

Overview

- Motivating example
- Reminder about the Process Hitting framework
- 4-level based logics and associated truth tables
- Translating 4-level based models into Process Hitting
- Further discussions

The Process Hitting modelling

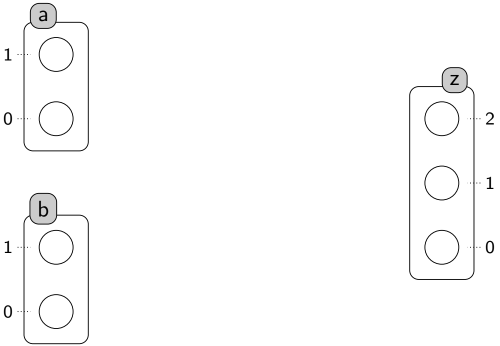
[Paulevé *et al.*, *Transactions on Computational Systems Biology*, 2011]



Sorts: components a, b, z

The Process Hitting modelling

[Paulevé *et al.*, *Transactions on Computational Systems Biology*, 2011]

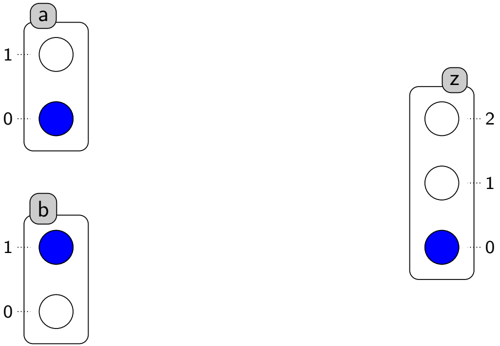


Sorts: components a, b, z

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

The Process Hitting modelling

[Paulevé *et al.*, *Transactions on Computational Systems Biology*, 2011]



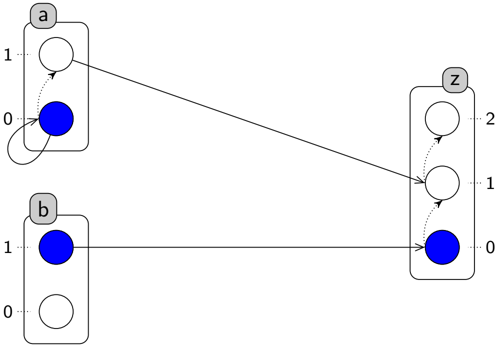
Sorts: components a, b, z

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

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

The Process Hitting modelling

[Paulevé *et al.*, *Transactions on Computational Systems Biology*, 2011]



Sorts: components a, b, z

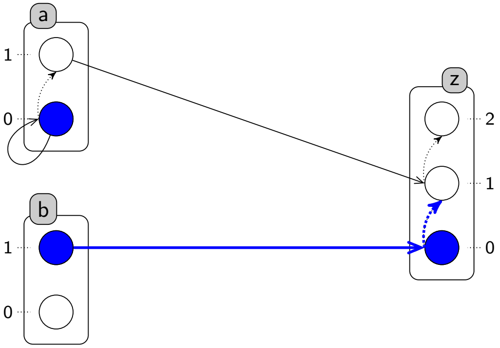
Processes: local states / levels of expression z_0, z_1, z_2

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

Actions: dynamics $b_1 \rightarrow z_0 \upharpoonright z_1,\ a_0 \rightarrow a_0 \upharpoonright a_1,\ a_1 \rightarrow z_1 \upharpoonright z_2$

The Process Hitting modelling

[Paulevé *et al.*, *Transactions on Computational Systems Biology*, 2011]



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

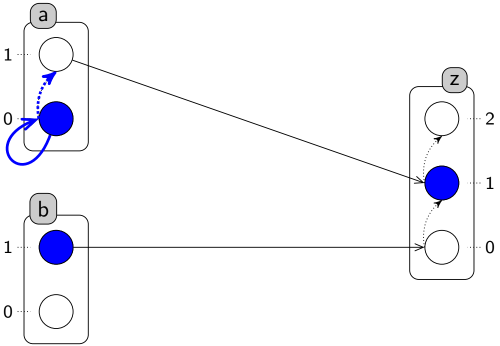
Processes: local states / levels of expression $z_0,\ z_1,\ z_2$

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

Actions: dynamics $\underline{b_1 \rightarrow z_0 \upharpoonright z_1},\ a_0 \rightarrow a_0 \upharpoonright a_1,\ a_1 \rightarrow z_1 \upharpoonright z_2$

The Process Hitting modelling

[Paulevé *et al.*, *Transactions on Computational Systems Biology*, 2011]



Sorts: components a, b, z

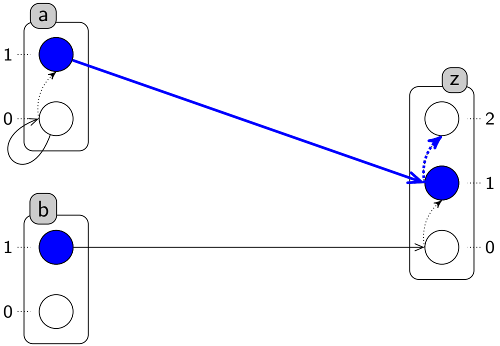
Processes: local states / levels of expression z_0, z_1, z_2

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

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

The Process Hitting modelling

[Paulevé *et al.*, *Transactions on Computational Systems Biology*, 2011]



Sorts: components a, b, z

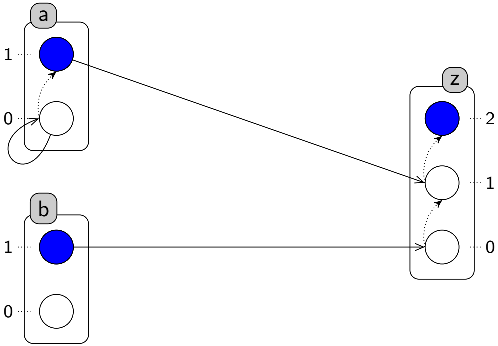
Processes: local states / levels of expression z_0, z_1, z_2

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

Actions: dynamics $b_1 \rightarrow z_0 \upharpoonright z_1$, $a_0 \rightarrow a_0 \upharpoonright a_1$, $\underline{a_1 \rightarrow z_1 \upharpoonright z_2}$

The Process Hitting modelling

[Paulevé *et al.*, *Transactions on Computational Systems Biology*, 2011]



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

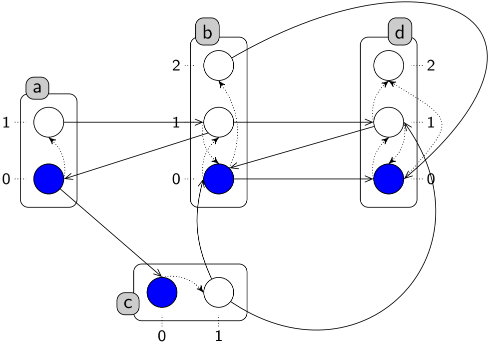
Processes: local states / levels of expression $z_0, \; z_1, \; z_2$

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

Actions: dynamics $b_1 \rightarrow z_0 \upharpoonright z_1, \; a_0 \rightarrow a_0 \upharpoonright a_1, \; a_1 \rightarrow z_1 \upharpoonright z_2$

Static analysis: successive reachability

[Paulevé *et al.*, *Mathematical Structures in Computer Science*, 2012]

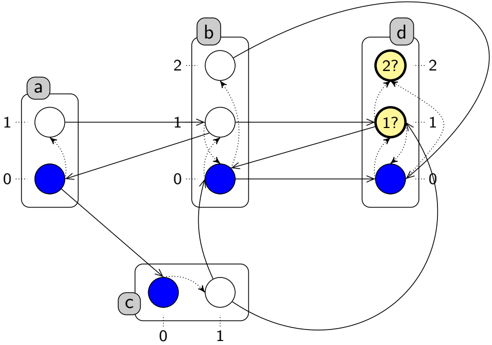


- Initial state

$\langle a_1, b_0, c_0, d_0 \rangle$

Static analysis: successive reachability

[Paulevé *et al.*, *Mathematical Structures in Computer Science*, 2012]



- Initial state

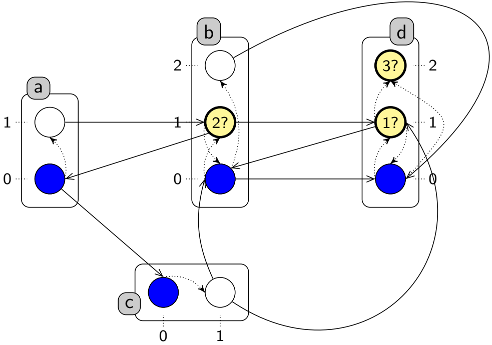
$$\langle a_1, b_0, c_0, d_0 \rangle$$

- Objectives

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

Static analysis: successive reachability

[Paulevé *et al.*, *Mathematical Structures in Computer Science*, 2012]



- Initial state

$$\langle a_1, b_0, c_0, d_0 \rangle$$

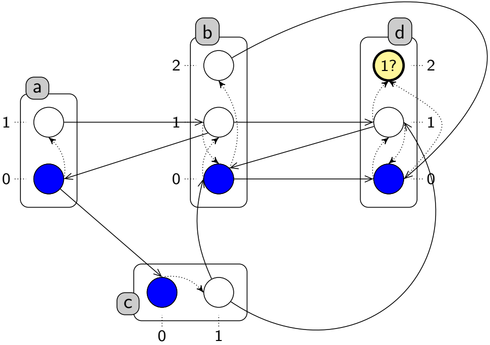
- Objectives

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

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

Static analysis: successive reachability

[Paulevé *et al.*, *Mathematical Structures in Computer Science*, 2012]



- Initial state

  $\langle a_1, b_0, c_0, d_0 \rangle$

- Objectives

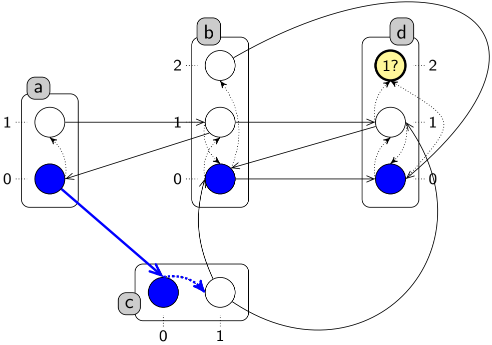
  $[\ \Rsh d_1 :: \Rsh d_2\ ]$

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

  $[\ \Rsh d_2\ ]$

Static analysis: successive reachability

[Paulevé *et al.*, *Mathematical Structures in Computer Science*, 2012]



- Initial state

$$\langle a_1, b_0, c_0, d_0 \rangle$$

- Objectives

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

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

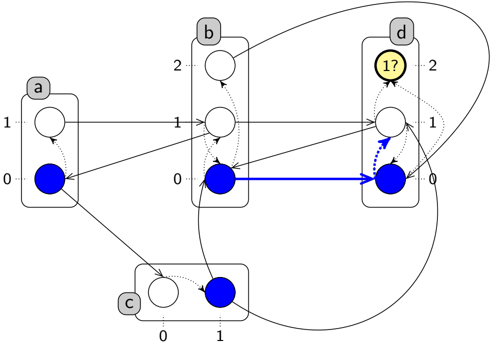
$$[\ \Rsh d_2\ ]$$

$\rightarrow$ Concretization of the objective = scenario

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

Static analysis: successive reachability

[Paulevé *et al.*, *Mathematical Structures in Computer Science*, 2012]



- Initial state

$$\langle a_1, b_0, c_0, d_0 \rangle$$

- Objectives

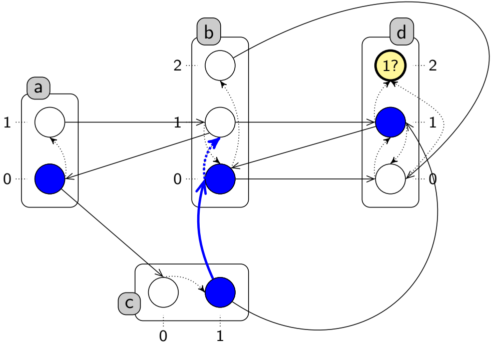
$$[\ \Rsh d_1 \ :: \Rsh d_2\ ]$$
$$[\ \Rsh d_1 :: \Rsh b_1 :: \Rsh d_2\ ]$$
$$[\ \Rsh d_2\ ]$$

$\rightarrow$ Concretization of the objective = scenario

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

Static analysis: successive reachability

[Paulevé *et al.*, *Mathematical Structures in Computer Science*, 2012]



- Initial state

$$\langle a_1, b_0, c_0, d_0 \rangle$$

- Objectives

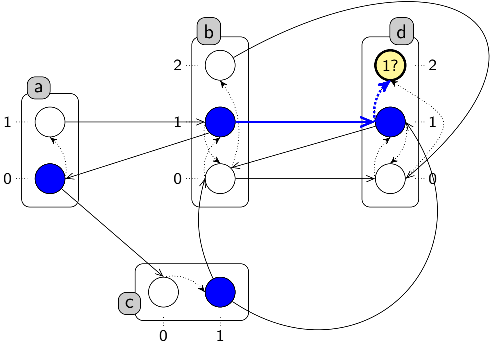
$$[\ \Rsh d_1\ ::\ \Rsh d_2\ ]$$
$$[\ \Rsh d_1 :: \Rsh b_1 :: \Rsh d_2\ ]$$
$$[\ \Rsh d_2\ ]$$

$\rightarrow$ Concretization of the objective = scenario

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

Static analysis: successive reachability

[Paulevé *et al.*, *Mathematical Structures in Computer Science*, 2012]



- Initial state

$$\langle a_1, b_0, c_0, d_0 \rangle$$

- Objectives

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

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

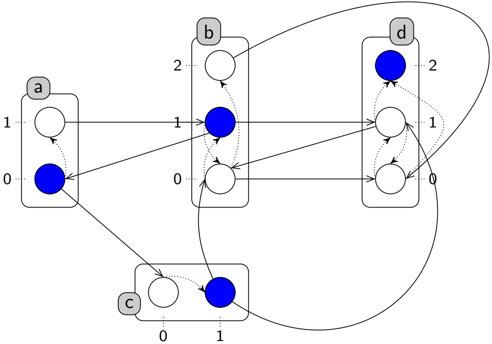
$$[\ \Rsh d_2\ ]$$

$\rightarrow$ Concretization of the objective = scenario

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

Static analysis: successive reachability

[Paulevé *et al.*, *Mathematical Structures in Computer Science*, 2012]



- Initial state

$$\langle a_1, b_0, c_0, d_0 \rangle$$

- Objectives

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

$\rightarrow$ Concretization of the objective = scenario

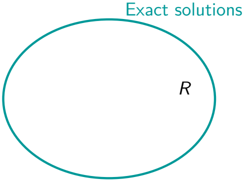
$$a_0 \rightarrow c_0 \Rsh c_1 :: b_0 \rightarrow d_0 \Rsh d_1 :: c_1 \rightarrow b_0 \Rsh b_1 :: b_1 \rightarrow d_1 \Rsh d_2$$

Over- and Under-approximations

[Paulevé *et al.*, *Mathematical Structures in Computer Science*, 2012]

$\rightarrow$ Directly checking R is hard (**exponential**)

$\rightarrow$ Rather check **approximations** P and Q so that: $\underline{P \Rightarrow R \Rightarrow Q}$

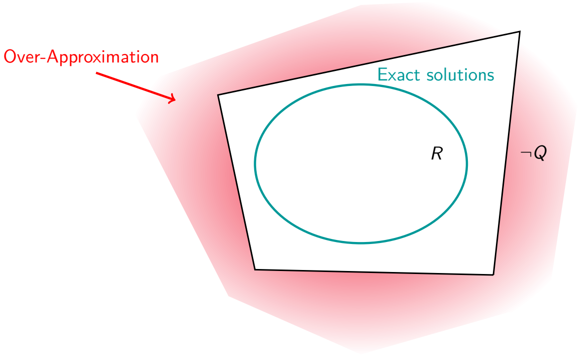


Over- and Under-approximations

[Paulevé *et al.*, *Mathematical Structures in Computer Science*, 2012]

→ Directly checking R is hard (**exponential**)

→ Rather check **approximations** P and Q so that: $\underline{P \Rightarrow R \Rightarrow Q}$

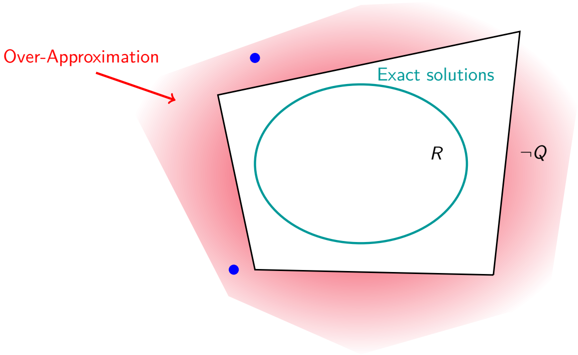


Over- and Under-approximations

[Paulevé *et al.*, *Mathematical Structures in Computer Science*, 2012]

→ Directly checking R is hard (**exponential**)

→ Rather check **approximations** P and Q so that: $\underline{P \Rightarrow R \Rightarrow Q}$

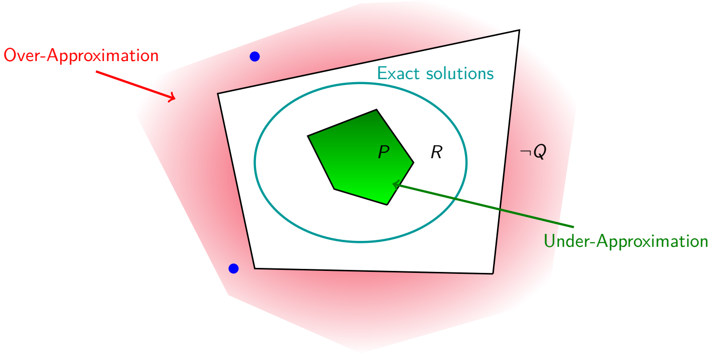


Over- and Under-approximations

[Paulevé *et al.*, *Mathematical Structures in Computer Science*, 2012]

$\rightarrow$ Directly checking R is hard (**exponential**)

$\rightarrow$ Rather check **approximations** P and Q so that: $\underline{P \Rightarrow R \Rightarrow Q}$

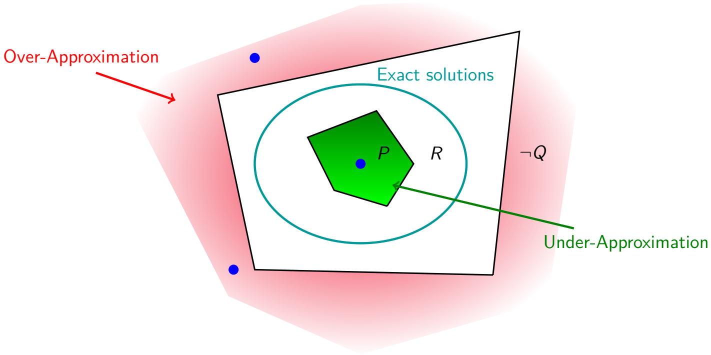


Over- and Under-approximations

[Paulevé *et al.*, *Mathematical Structures in Computer Science*, 2012]

$\rightarrow$ Directly checking R is hard (**exponential**)

$\rightarrow$ Rather check **approximations** P and Q so that: $\underline{P \Rightarrow R \Rightarrow Q}$

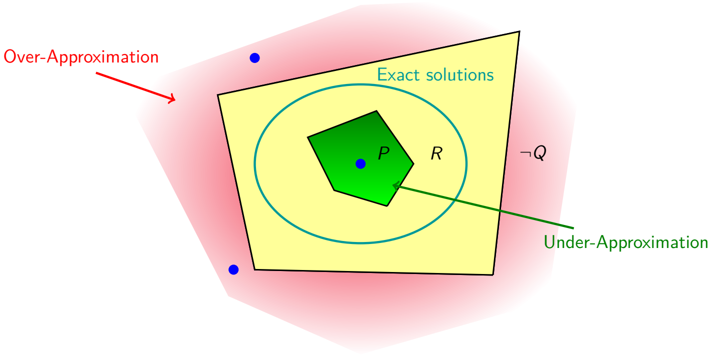


Over- and Under-approximations

[Paulevé *et al.*, *Mathematical Structures in Computer Science*, 2012]

→ Directly checking R is hard (**exponential**)

→ Rather check **approximations** P and Q so that: $\underline{P \Rightarrow R \Rightarrow Q}$

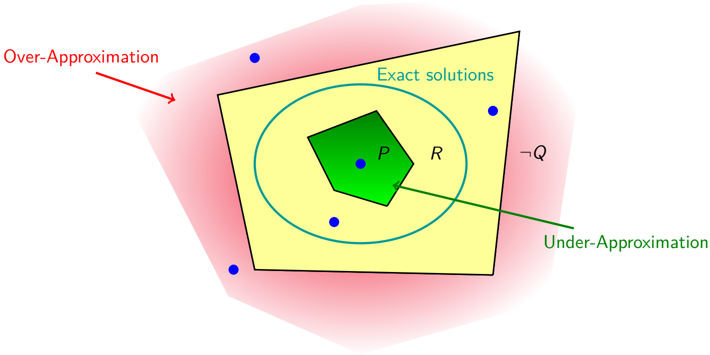


Over- and Under-approximations

[Paulevé *et al.*, *Mathematical Structures in Computer Science*, 2012]

$\rightarrow$ Directly checking R is hard (**exponential**)

$\rightarrow$ Rather check **approximations** P and Q so that: $\underline{P \Rightarrow R \Rightarrow Q}$

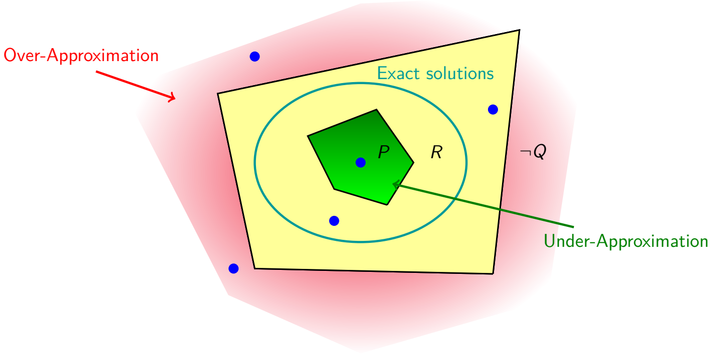


Over- and Under-approximations

[Paulevé *et al.*, *Mathematical Structures in Computer Science*, 2012]

$\rightarrow$ Directly checking R is hard (**exponential**)

$\rightarrow$ Rather check **approximations** P and Q so that: $\underline{P \Rightarrow R \Rightarrow Q}$



Computing P or Q is much simpler (roughly **polynomial**)

$\rightarrow$ Efficient for big models $\rightarrow$ **Hundredths of seconds**

Enrichment of the Process Hitting

[Folschette *et al.*, *CS2Bio'13*, 2013]

Several additions to improve the expressiveness of the Process Hitting:

- Priorities
  - Groups of actions with similar temporal/probabilistic parameters
- Neutralizing edges
  - Atomistic delay relations between actions
- Synchronous actions
  - Multiple reactants and products $\rightarrow$ Biochemical reactions

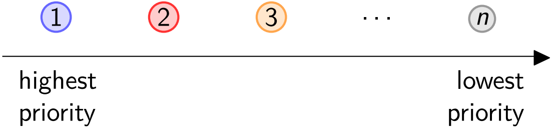
All these formalisms can be translated to a canonical form

A new static analysis has been developed to check reachability properties
$\rightarrow$ Efficient dynamic analysis on large models

Priorities

[Folschette *et al.*, *CS2Bio'13*, 2013]

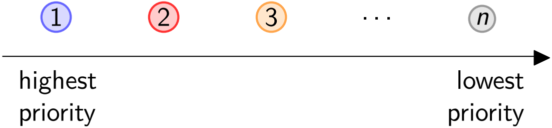
- Each action is linked to a class of priority
- An action is playable only if no action with a higher priority is playable

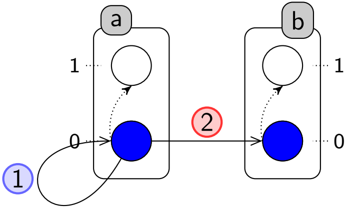


Priorities

[Folschette *et al.*, *CS2Bio'13*, 2013]

- Each action is linked to a class of priority
- An action is playable only if no action with a higher priority is playable



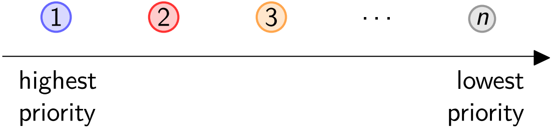


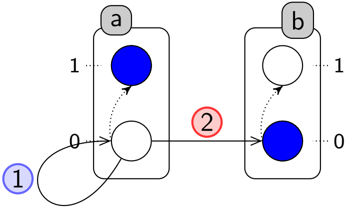
$\rightarrow$ b_1 **can never** be reached

Priorities

[Folschette *et al.*, *CS2Bio'13*, 2013]

- Each action is linked to a class of priority
- An action is playable only if no action with a higher priority is playable



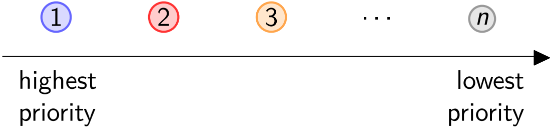


$\rightarrow$ b_1 **can never** be reached

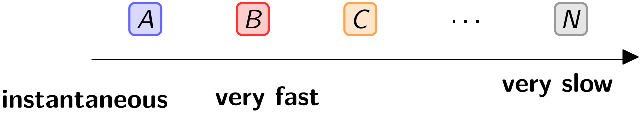
Priorities

[Folschette *et al.*, *CS2Bio'13*, 2013]

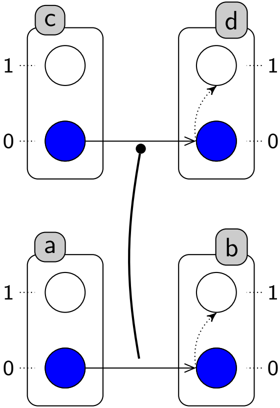
- Each action is linked to a class of priority
- An action is playable only if no action with a higher priority is playable



- Allows to model classes of actions with similar temporal/stochastic parameters



Neutralizing edges



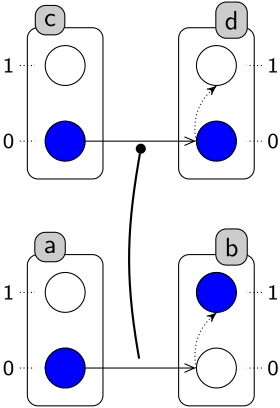
- Allows to integrate temporal data about relative reaction delays
- Atomistic preemptions

$c_0 \rightarrow d_0 \upharpoonright d_1$ cannot be played **while**

$a_0 \rightarrow b_0 \upharpoonright b_1$ is playable

$\rightarrow d_1$ is **always** reached after b_1

Neutralizing edges



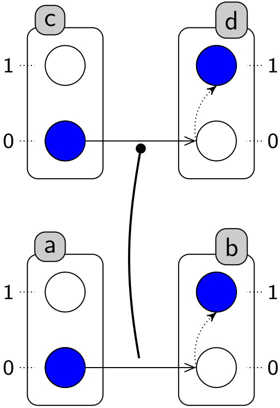
- Allows to integrate temporal data about relative reaction delays
- Atomistic preemptions

$c_0 \rightarrow d_0 \Rsh d_1$ cannot be played **while**

$a_0 \rightarrow b_0 \Rsh b_1$ is playable

$\rightarrow$ d_1 is **always** reached after b_1

Neutralizing edges



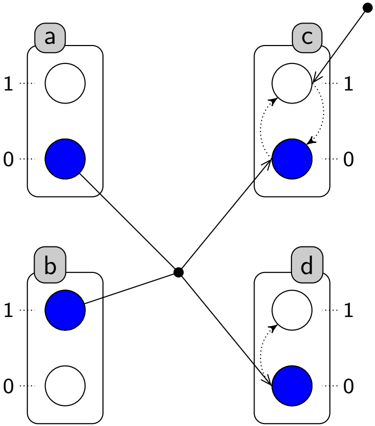
- Allows to integrate temporal data about relative reaction delays
- Atomistic preemptions

$c_0 \rightarrow d_0 \Rsh d_1$ cannot be played **while**

$a_0 \rightarrow b_0 \Rsh b_1$ is playable

$\rightarrow$ d_1 is **always** reached after b_1

Synchronous actions



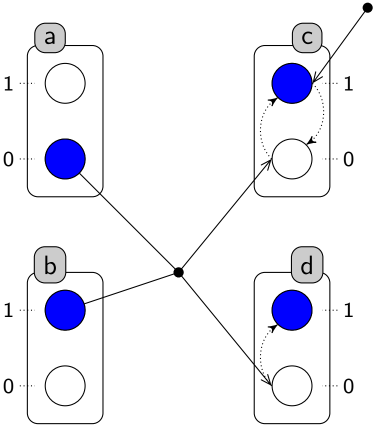
- Synchronization between actions:
  - Presence of catalysts
  - Consumption of reactants
  - Creation of products
- Convenient for biochemical equations: $X \xrightarrow{Y} Z$
  in the following form:
  $$\{X_1, Y_1, Z_0\} \rightarrowtail \{X_0, Z_1\}$$

$$h_1 = \{c_1\} \rightarrowtail \{c_0\}$$
$$h_2 = \{a_0, b_1, c_0, d_0\} \rightarrowtail \{c_1, d_1\}$$

All processes of A
must be present to play $A \rightarrowtail B$

After playing $A \rightarrowtail B$,
all processes of B are active

Synchronous actions

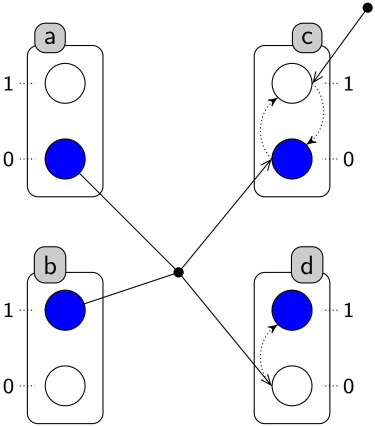


- Synchronization between actions:
  - Presence of catalysts
  - Consumption of reactants
  - Creation of products
- Convenient for biochemical equations: $X \xrightarrow{Y} Z$ in the following form:
  $$\{X_1, Y_1, Z_0\} \rightarrowtail \{X_0, Z_1\}$$

$$h_1 = \{c_1\} \rightarrowtail \{c_0\}$$
$$h_2 = \{a_0, b_1, c_0, d_0\} \rightarrowtail \{c_1, d_1\}$$

All processes of A must be present to play $A \rightarrowtail B$

After playing $A \rightarrowtail B$, all processes of B are active



Synchronous actions

- Synchronization between actions:
  - Presence of catalysts
  - Consumption of reactants
  - Creation of products
- Convenient for biochemical equations: $X \xrightarrow{Y} Z$
  in the following form:
  $$\{X_1, Y_1, Z_0\} \rightarrowtail \{X_0, Z_1\}$$

$$h_1 = \{c_1\} \rightarrowtail \{c_0\}$$
$$h_2 = \{a_0, b_1, c_0, d_0\} \rightarrowtail \{c_1, d_1\}$$

All processes of A
must be present to play $A \rightarrowtail B$

After playing $A \rightarrowtail B$,
all processes of B are active

Overview

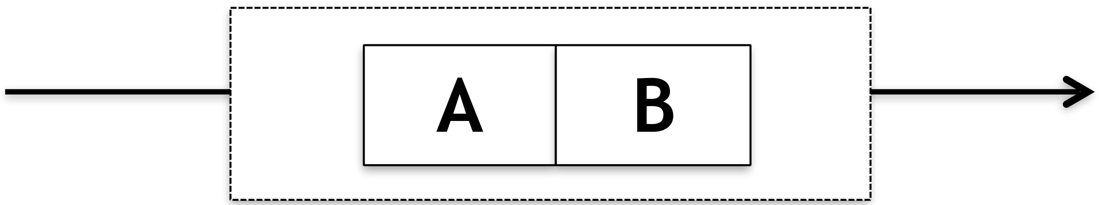
- Motivating example
- Reminder about the Process Hitting framework
- 4-level based logics and associated truth tables
- Translating 4-level based models into Process Hitting
- Further discussions

Modeling ideas

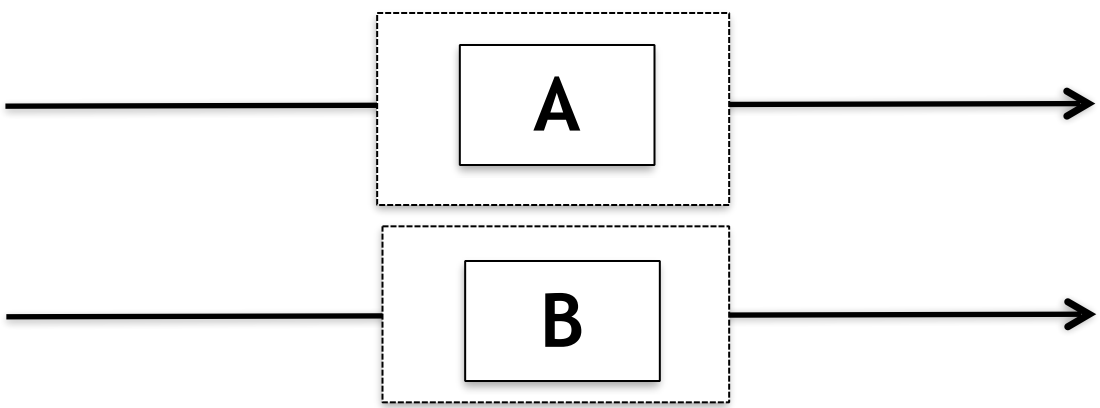
- 4 cases to consider:
  - The concentration of a component c in ko of a given gene g is **higher** than its concentration in Wild Type (which will be denoted ↑)
  - The concentration of a component c in case of ko of a given gene g is **lower** than its concentration in Wild Type (which then will be denoted ↓)
  - The concentration of a component c in case of ko of a given gene g is **stable** compared to Wild Type (which then will be denoted -)
  - When the evolution of the concentration of a component c between ko and wild type is **unknown**: add a fourth level « unknown" in the logical framework, but not necessary in the Process Hitting final representation.

Our stoichiometric modeling

- ❑ **A and B**: denoting the effect by *the complex* of A and B
  - ⇒ Strength: depending on the amount of the complex



- ❑ **A or B**: denoting the (individual) effects by A and B
  - ⇒ Strength: depending on the amount of both A and B



Truth table in 4 valued logic (1/2)

↑: increase. ↓: decrease. -: unchanged.

A	B	A and B	A or B
↑	↑	↑	↑
↑	↓	↓	*unknown*
↑	-	-	↑
↑	*unknown*	*unknown*	*unknown*
↓	↓	↓	↓
↓	-	↓	↓
↓	*unknown*	↓	*unknown*
-	-	-	-
-	*unknown*	*unknown*	*unknown*
unknown	*unknown*	*unknown*	*unknown*

Truth table in 4 values logic (2/2)

↑: increase. ↓: decrease. –: unchanged.

A	¬A
↑	↓
↓	↑
-	-
unknown	*unknown*

Overview

- Motivating example
- Reminder about the Process Hitting framework
- 4-level based logics and associated truth tables
- Translating 4-level based models into Process Hitting
- Further discussions

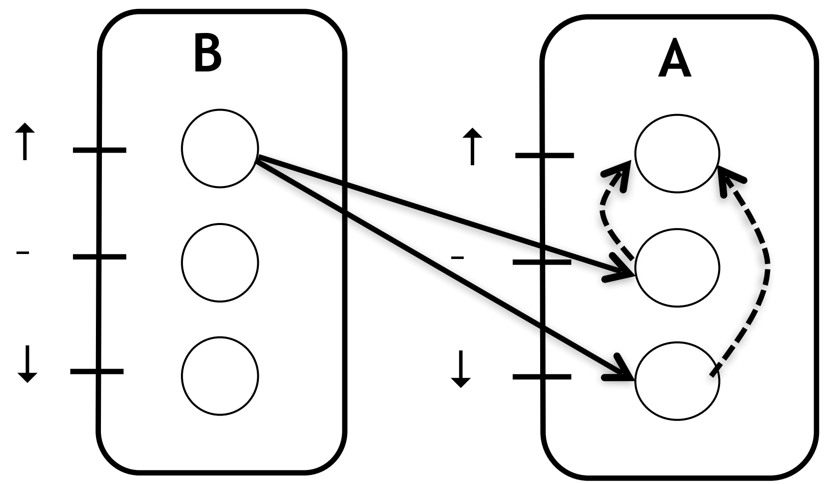
Principle of the translation of « 4 valued logics » into PH

- When A has more than one regulator, use a **cooperative sort** to update A according to the state of regulators —> need to use **priorities** in PH

- « unknown » is modeled by modeling every potential underlying behavior

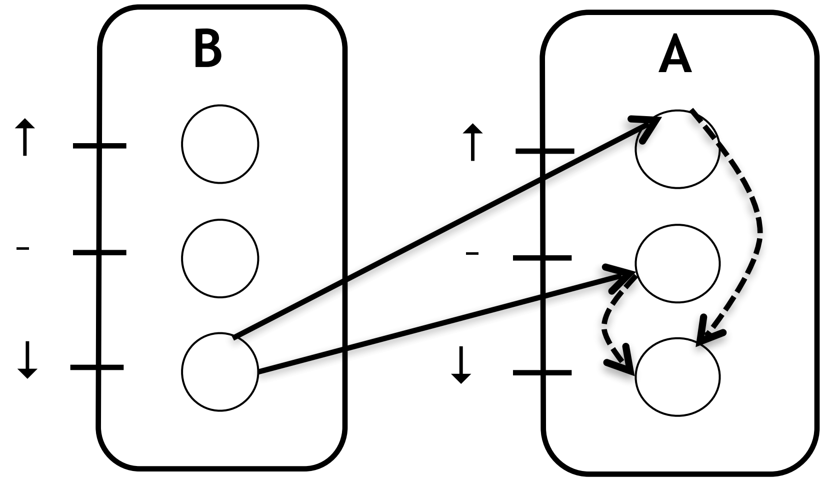
Translating 4 valued logics into Process Hitting: A=B

1. A = B
2. A = not B
3. A = B and C
4. A = B or C



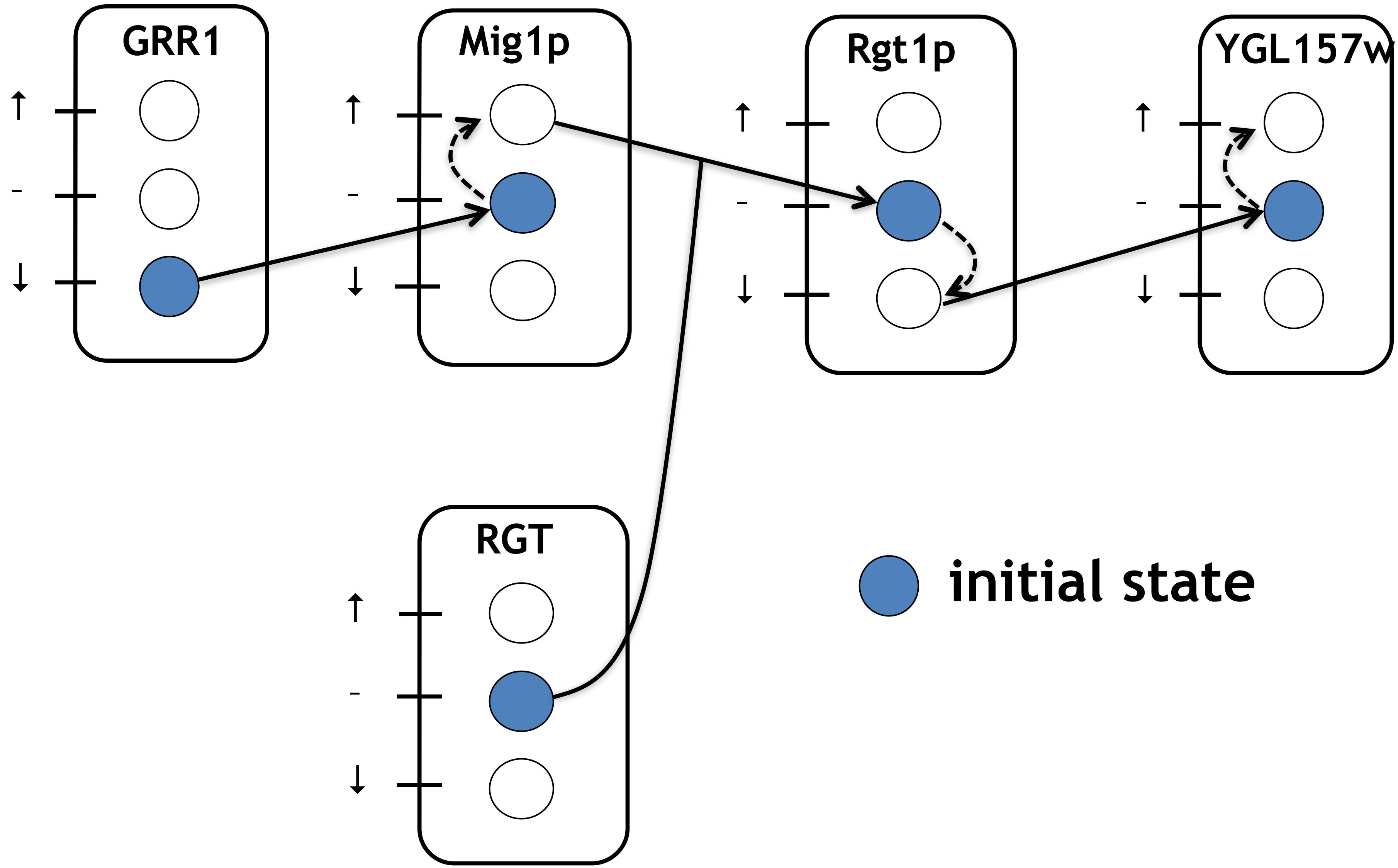
Translating 4 valued logics into Process Hitting

1. A = B
2. A = not B
3. A = B and C
4. A = B or C

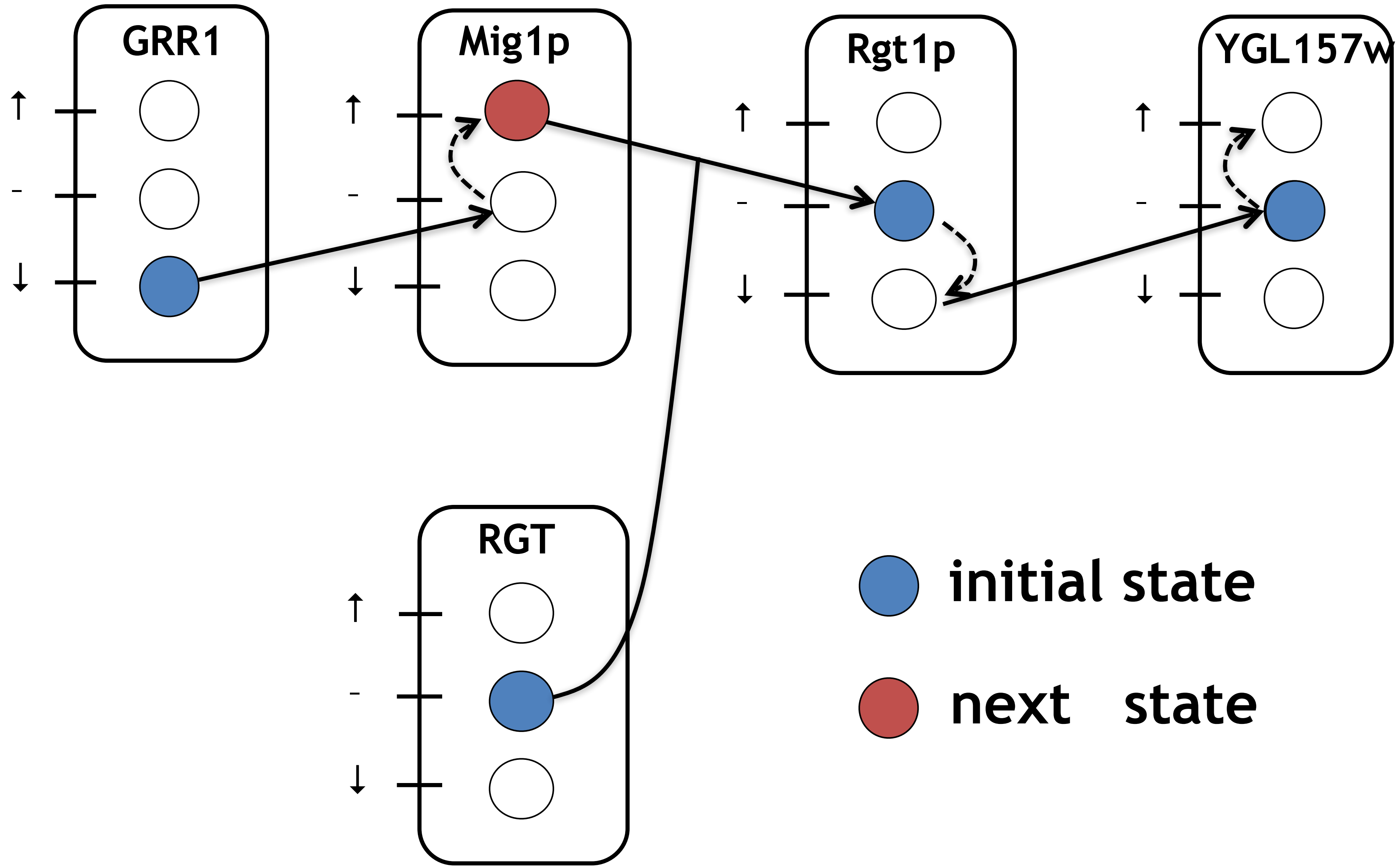


- Maybe add a slide with the translation of A = B and C, but the resulting PH is quite complex ?

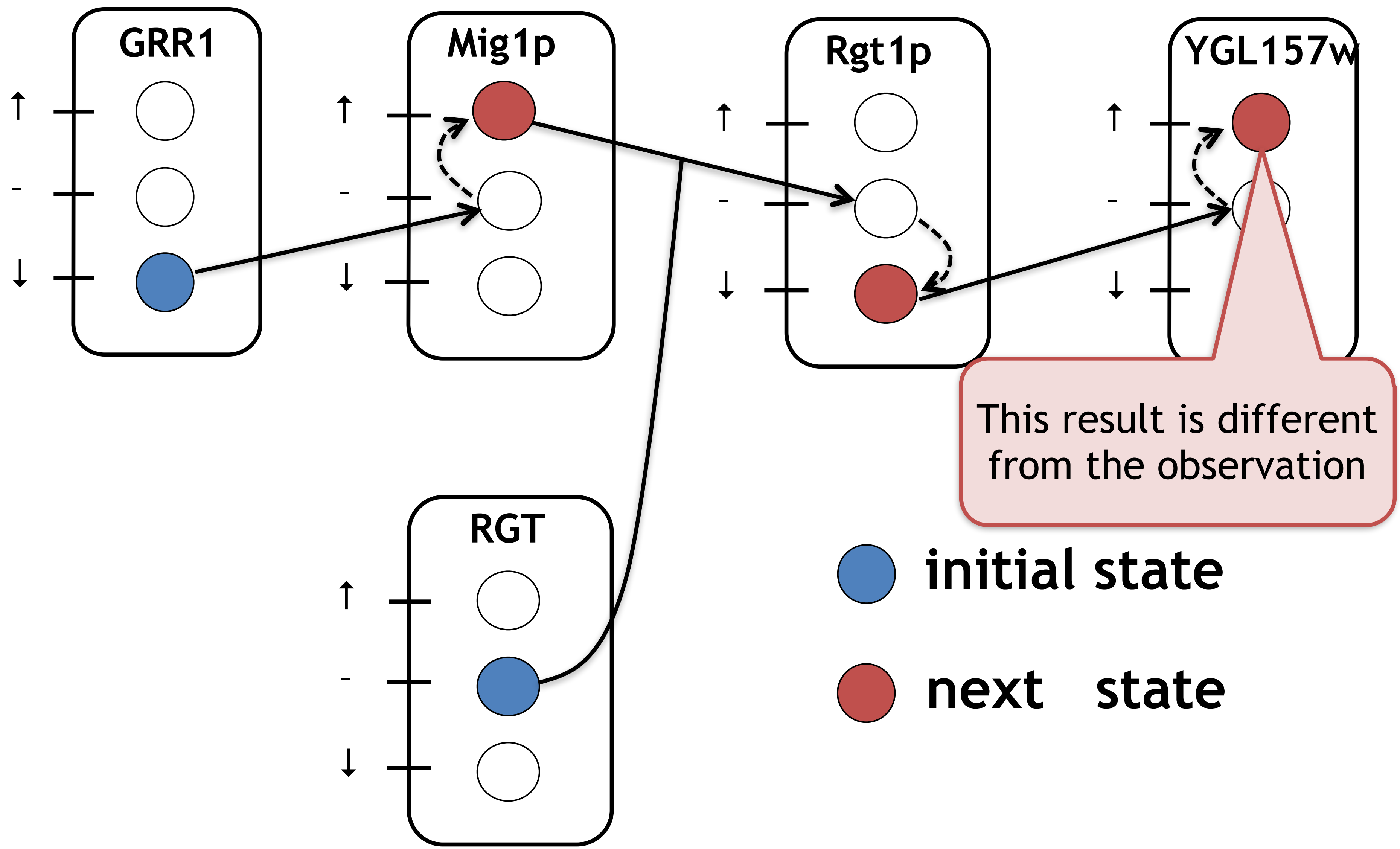
Back to the example



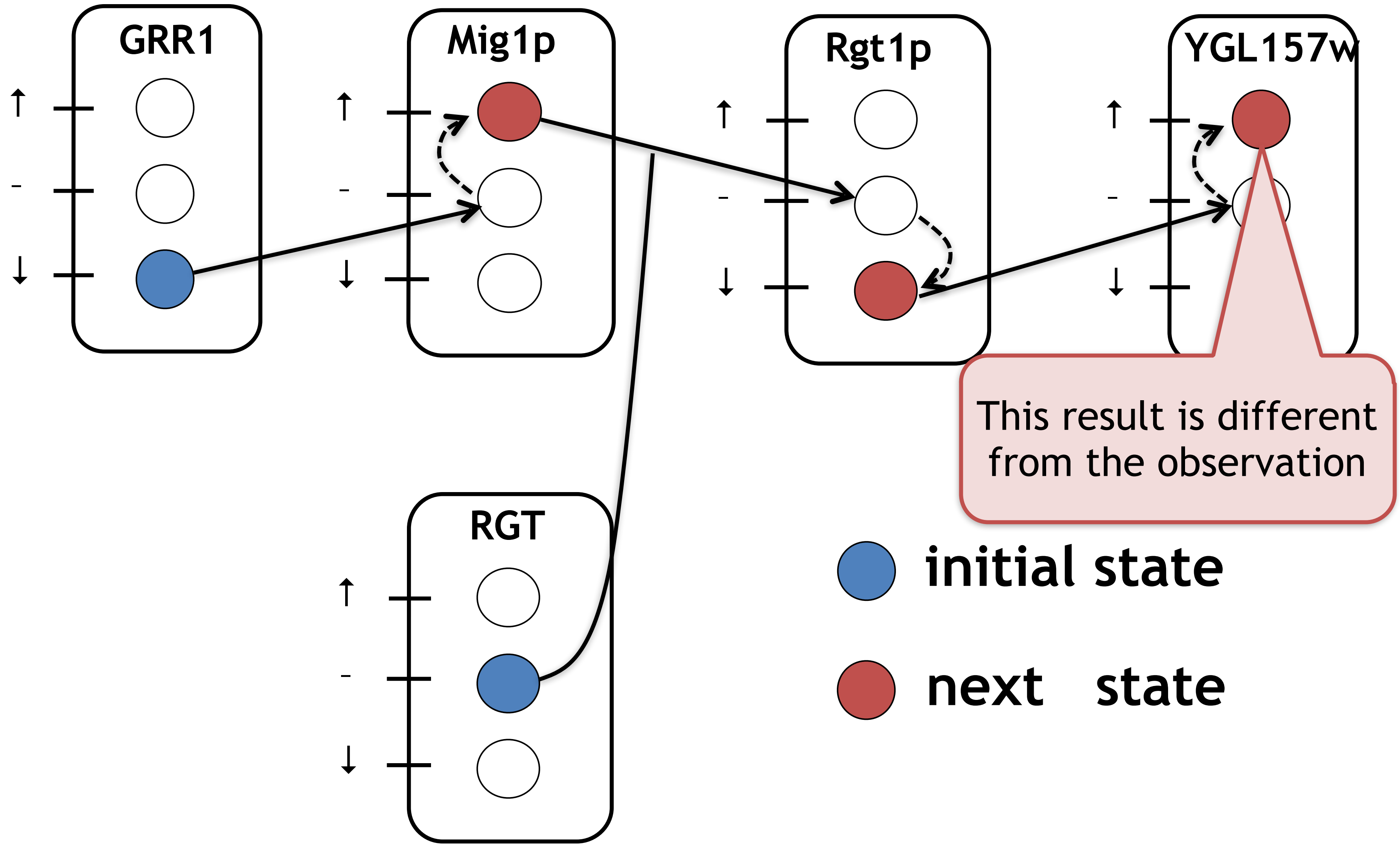
Back to the example: one execution



Back to the example: one execution



And with synchronous semantics?



Our question

In case that the dynamics of the model does not encompass the observation into

any playable scenario of actions… how to **detect missing actions as few as possible** that can lead the goal state?

Related discussions

- **Asynchronous** versus **synchronous** semantics, w.r.t. the addition of **priorities**

- Compare **4-valued logics** with existing approaches with ODEs

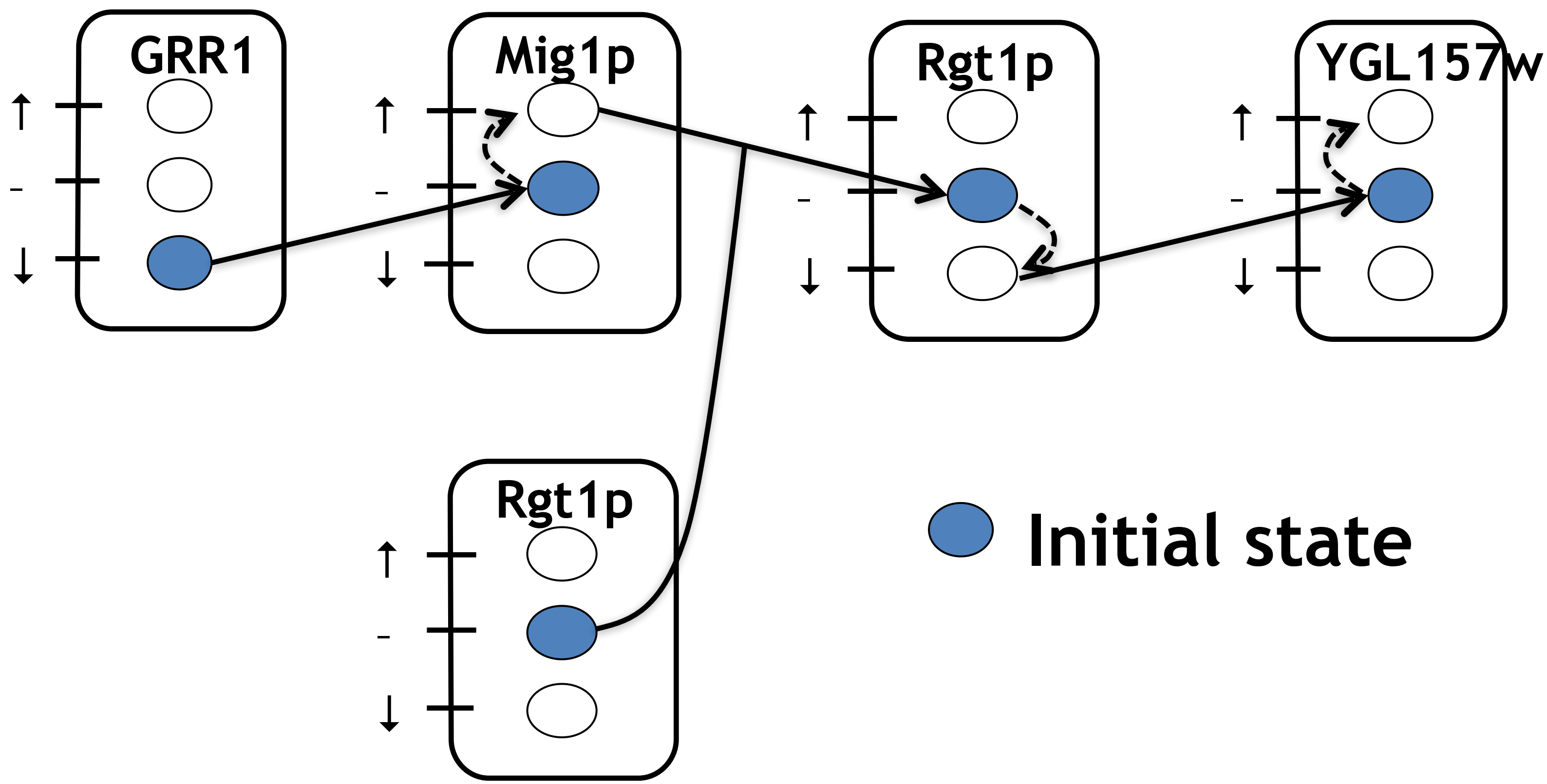
- Interest for a **cut-sets based approach**

Cut-sets in Process Hitting

- **Sets of necessary processes** that, should they be disabled, would prevent the considered reachability
- Useful to **refute a model**: if a cut set computed from the model does not prevent the reachability in the concrete (modeled) system, then it is a proof that there exists concrete behaviors that are not reproducible by the model.
- See (Paulevé et al., 2014) and Loïc’s talk last year

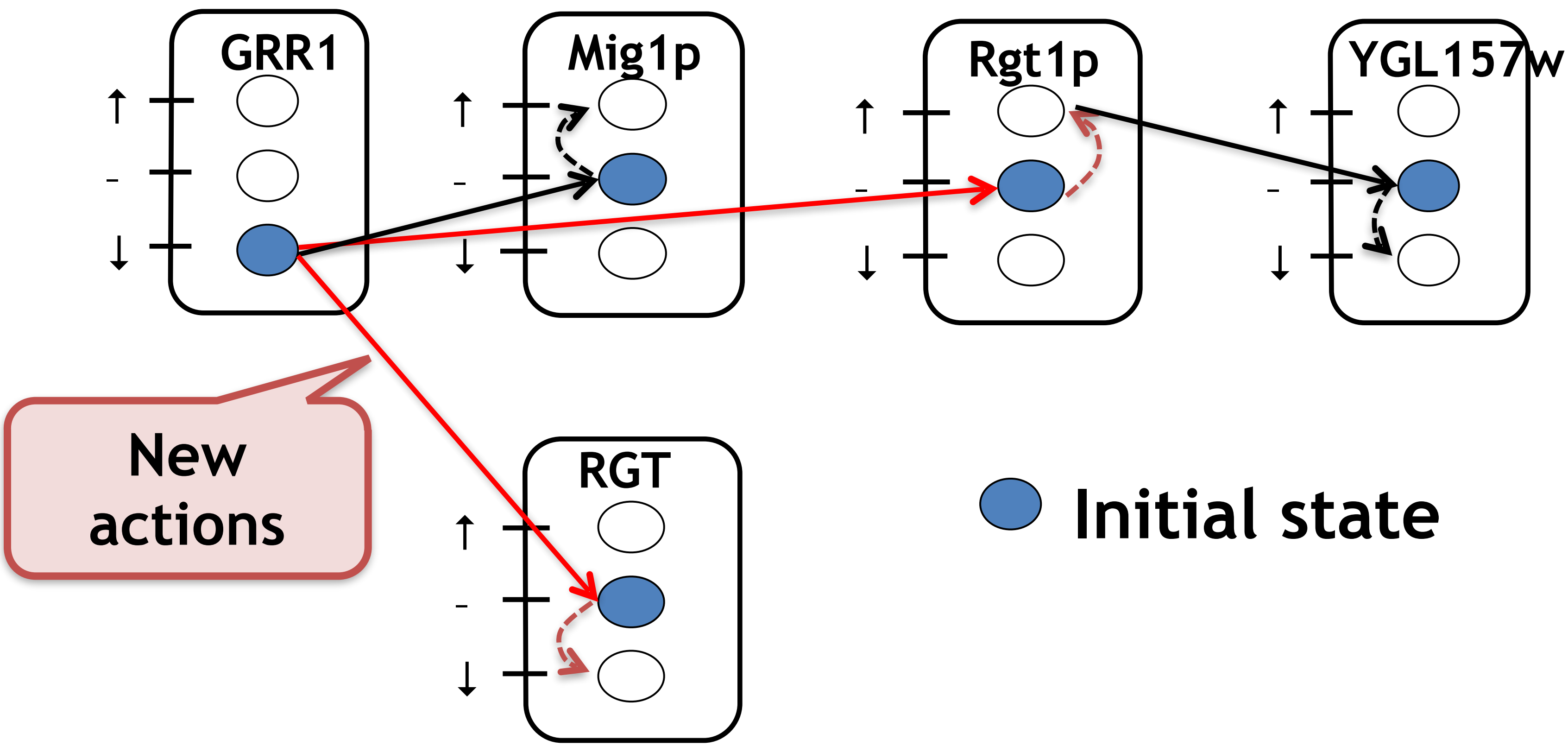
Problem setting
(for abduction in process hitting)

- Finding actions for explaining the observation with the background theory (Boolean network)



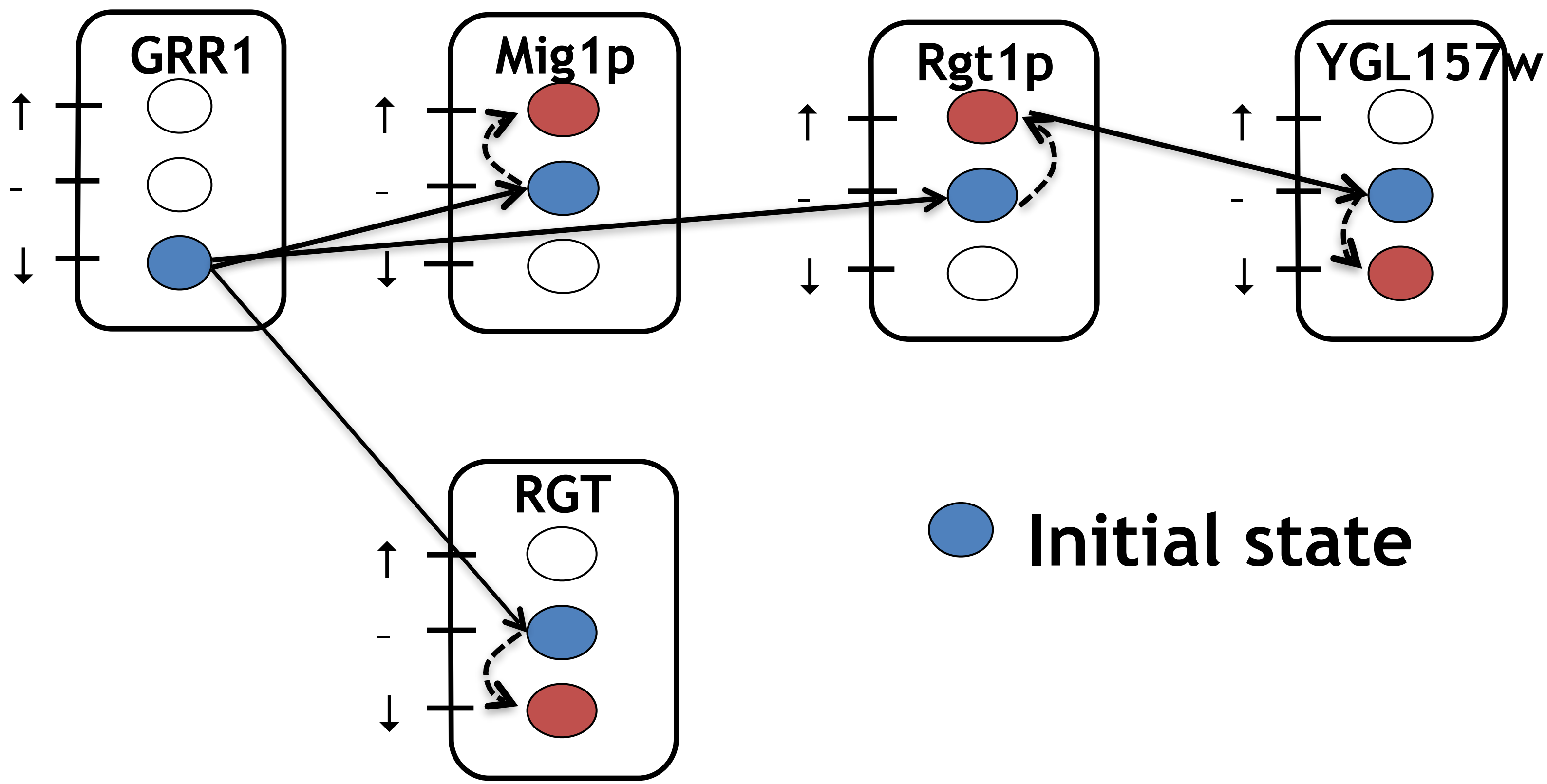
Problem setting (for abduction in process hitting)

- Finding actions for explaining the observation with the background theory (Boolean network)



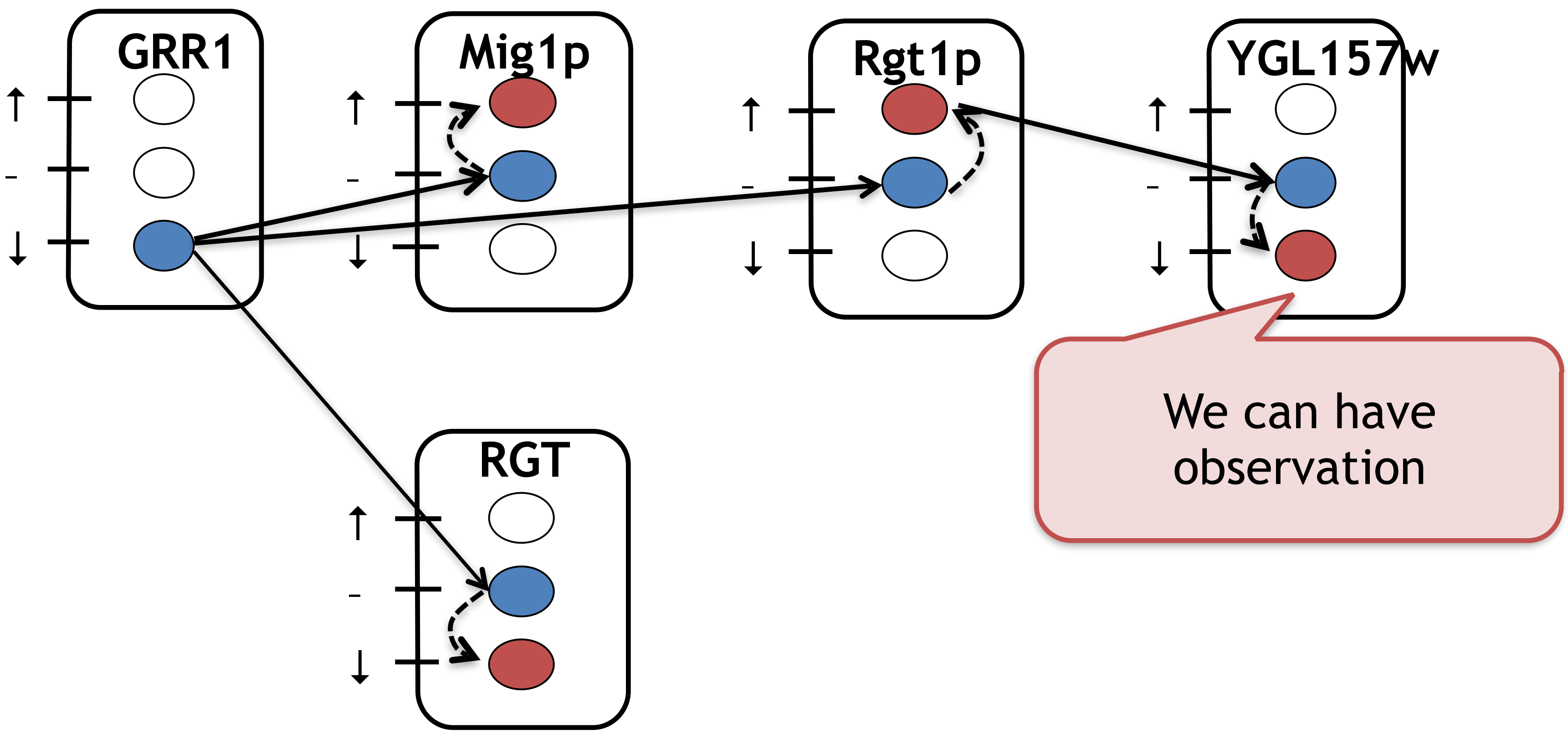
Problem setting (for abduction in process hitting)

- Finding actions for explaining the observation with the background theory (Boolean network)



Problem setting (for abduction in process hitting)

- Finding actions for explaining the observation with the background theory (Boolean network)



Overview

- Motivating example
- Reminder about the Process Hitting framework
- 4-level based logics and associated truth tables
- Translating 4-level based models into Process Hitting
- Further discussions

Research plan and future work

- **Formalize** an algorithmic approach to tackle this completion problem

- Study models with **feedback loops** and extend the principle of 4-valued logics

- Tackle models with **time series data** as input