

# Identification of logical models for signaling pathways: towards a systems biology loop

Anne Siegel  
IRISA/CNRS, Rennes, France



UMR

IRISA

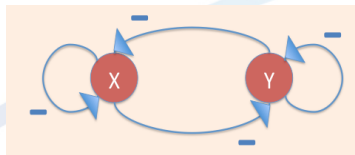
DyLISS

CIRM, Janvier 2017

# Dynamical systems

## Historical motivation

Modeling the evolution of a set of components  $\mathbb{A}$  of a system over time over a domain  $\mathbb{T}$ .



$$\frac{dX}{dt} = \frac{k}{K + Y^n} - aX$$
$$\frac{dY}{dt} = \frac{l}{L + X^n} - bY$$

Parameterized  
numerical system

$$f(X) \leftarrow 1 - Y$$

$$f(Y) \leftarrow 1 - X$$

Boolean model with  
asynchronous update  
scheme

## Mathematical framework

$$F : \begin{array}{ccc} \mathbb{T} & \times & \mathbb{S} \\ (t & , & \mathbf{z}) \\ \text{(time} & , & \text{state)} \end{array} \begin{array}{c} \rightarrow \mathbb{S} \\ \mapsto F(t, \mathbf{z}) \\ \text{new state at time } t \end{array}$$

# Identification

**Model identification?** Find the most suitable function  $F$  which explains and depicts the observed responses of a system

# Identification

**Model identification?** Find the most suitable function  $F$  which explains and depicts the observed responses of a system

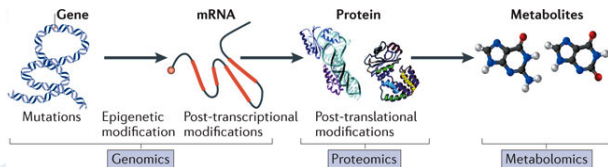
## What makes easier the model identification task?

- **A priori knowledge** → predetermined "shapes" for the function  $F$ .
- **A very limited number of components** → reduce the search space.
- **A wide panel of perturbations and sensors** → discriminate the models.

## Where is the complexity?

The search space exponentially grows with the number of measured components

## Experimental *omics* data



Patti et al. (2012). Metabolomics: the apogee of the omics trilogy. Nature

- Large-scale
- Knowledge incompleteness
- Noise

→ **Most biomolecular systems are not uniquely identifiable from large-scale datasets**

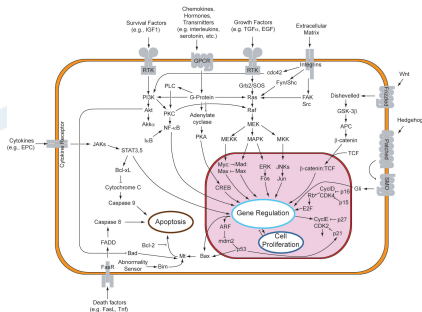
# How to analyse biomolecular networks in the complex of omics data?

**Strategy:** develop methods to reason over a **complete family of feasible models** instead of selecting one model

- **Discrete Dynamical Systems** → Reduce the space of feasible models.
- **Knowledge reasoning** → Precisely describe the search space.
- **Solve combinatorial problem** → Extract robust information common to all models in the search space.

# Signaling networks

They dictate the cell response to diverse signals in its environment



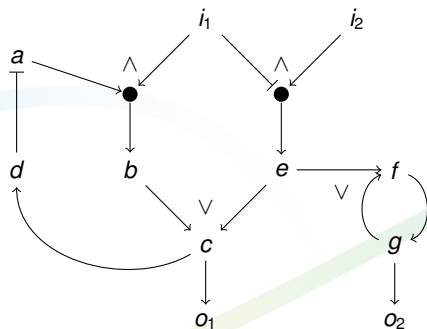
## Highlights

- Lack of kinetic information
- **Fast and slow reactions** can often be discriminated
- **ON/OFF switch-like behavior** at the protein level

# Modeling signaling networks

## Logical signaling networks

- Boolean networks
- species  $\rightarrow$  discrete variables
- interactions  $\rightarrow$  logic formulas or gates
- state  $\rightarrow$  updated over time steps



$$\phi = \left\{ \begin{array}{lll} a \leftarrow \neg d & d \leftarrow c & g \leftarrow f \\ b \leftarrow a \wedge i_1 & e \leftarrow \neg i_1 \wedge i_2 & o_1 \leftarrow c \\ c \leftarrow b \vee e & f \leftarrow e \vee g & o_2 \leftarrow g \end{array} \right\}$$

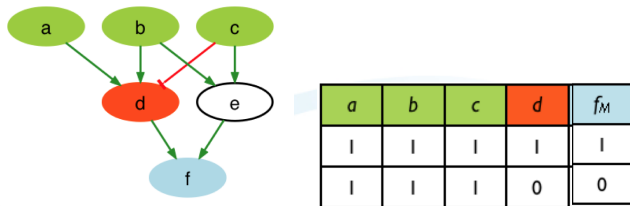


# Updating scheme

	synchronous [Kauffman'69]	asynchronous [Thomas'73]
Updates	all at the same time	one at a time
Time-scales	similar	various
Simulation	Tractable	Demanding
Training	Demanding	–

**Assumption: synchronous updates** are **rough** but **reasonable** models of the **(early) response** in signaling networks

# Phospho-proteomics data ... in theory



## Experimental assay

- Green nodes can be forced to be activated.
- Red nodes can be forced to be inhibited
- Blue nodes can be measured after a lapse-time.

## Response to perturbations

- Measure the system response after a certain number of perturbations.

**Several hundreds of different perturbations can be tested on a same sample.**

# (Exact) learning issue

## Inputs

- An interaction graph based on prior knowledge
- The results of several combinations of activators and inhibitors over readout

## Search space

All logical models compatible with the interaction graph

→ for the previous example, the search space contains  $2^{13}$  models.

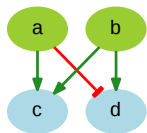
## Output

One or several logical models

- Compatible with the interaction graph
- Whose logical response is compatible with experimentations
- With minimal size (parsimony assumption)

**Identify the most simple models that can explain the observed responses.**

# Example of the learning procedure

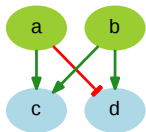


#	Stimul		Readouts	
	a	b	c	d
1	0	1	0	1
2	1	0	1	0
3	0	0	0	0
4	1	1	1	0

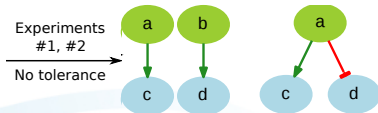
Experiments  
#1, #2  
→  
No tolerance

Figures

# Example of the learning procedure



#	Stimul		Readouts	
	a	b	c	d
1	0	1	0	1
2	1	0	1	0
3	0	0	0	0
4	1	1	1	0



Experiments  
#1, #2

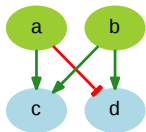
No tolerance

$$(c \equiv a) \wedge (d \equiv b)$$

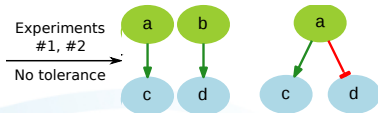
$$(c \equiv a) \wedge (d \equiv \neg a)$$

Figures

# Example of the learning procedure



#	Stimul		Readouts	
	a	b	c	d
1	0	1	0	1
2	1	0	1	0
3	0	0	0	0
4	1	1	1	0



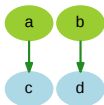
$$(c \equiv a) \wedge (d \equiv b)$$

$$(c \equiv a) \wedge (d \equiv \neg a)$$

Figures

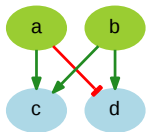
Experiments #1, #2, #3

No tolerance

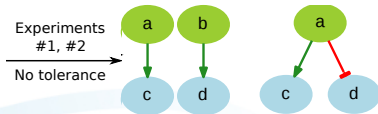


$$(c \equiv a) \wedge (d \equiv b)$$

# Example of the learning procedure



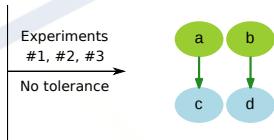
#	Stimul		Readouts	
	a	b	c	d
1	0	1	0	1
2	1	0	1	0
3	0	0	0	0
4	1	1	1	0



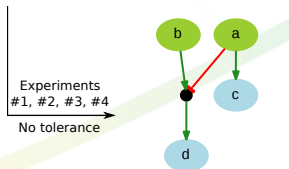
$$(c \equiv a) \wedge (d \equiv b)$$

$$(c \equiv a) \wedge (d \equiv \neg a)$$

Figures



$$(c \equiv a) \wedge (d \equiv b)$$

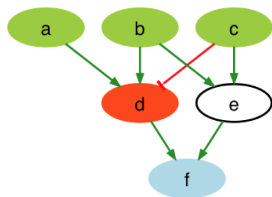


$$(c \equiv a) \wedge (d \equiv (b \wedge \neg a))$$

When  $a$  and  $b$  are activated, an additional effect over  $d$  emerges.

**The output of the learning problem is not monotone**  
Increasing the set of observed responses drastically changes the family of models which are solution to the identification problem.

## Phospho-proteomics data ... in practice



a	b	c	d	f	$f_M$
1	1	1	1	0,7	1
1	1	1	0	0,2	0

Phosphorylation activity is an average-value.

→ **Introduce a fitness score between boolean values and numerical experimental measurements**

$$\text{Residual score} = (0.2 - 0)^2 + (1 - 0.7)^2 = \mathbf{0.13}$$



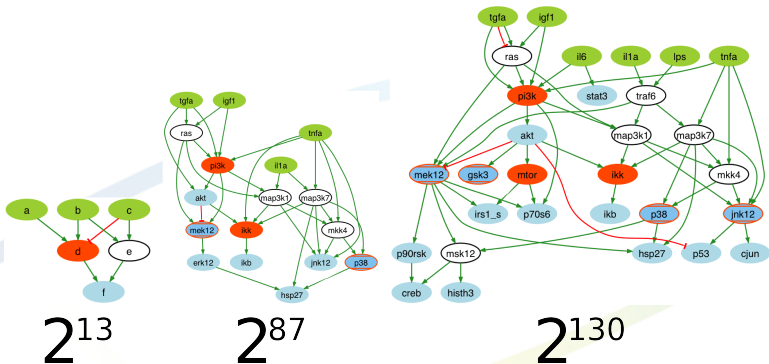
# Learning as an optimization problem

**Combinatorial issue. Find logical signaling networks which satisfy the following conditions:**

- **Structural condition** : networks supported by interaction graph.
- **Parsimonious assumption**: minimize model complexity.
- **Fitting condition**: minimize the distance between measured observations and predictions of the logical network.

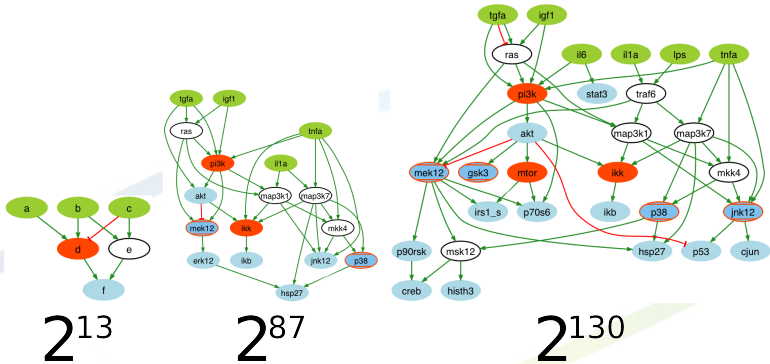
$$\arg \min_{(V, \phi) \in \mathbb{M}_{(V, E, \sigma)}} \underbrace{\text{Score}_{\text{rss}}((V, \phi), (P_1, \dots, P_n))}_{\text{residual sum of squares}}, \underbrace{\text{Score}_{\text{size}}((V, \phi))}_{\text{complexity}}$$

# Phospho-proteomics data ... in very practice (1)



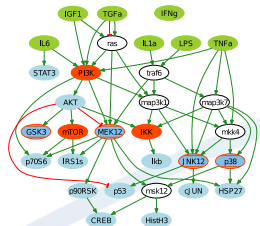
→ The search space grows exponentially

# Phospho-proteomics data ... in very practice (2)



Several non-observable species (white nodes)  
→ **uncertainty at the level of internal mechanisms**

# Phospho-proteomics data ... in very practice (3)



- TGFa Stimulus
- PI3K Inhibitor
- ras Non-observable/Non-controllable
- STAT3 Readout
- GSK3 Inhibitor/Readout

Stimuli	Inhibitors	Readouts
0 0 1 0 0 0	0 0 0 0 0 0	0.12 0.95 0.02 0.21 0.10
0 0 0 0 1 0	0 0 0 0 0 1	0.32 0.01 0.25 0.05 0.92
1 0 0 0 0 0	0 1 0 0 0 0	0.09 0.17 0.86 0.43 0.78

⋮  
Combinatorial  
perturbations

⋮  
Phosphorylation  
activity in [0,1]

## Experimental data are highly noisy

- Numerical value have to be considered up to 10% of noise
- This may have a strong impact on the residual score.

# Learning as a RELAXED optimization problem

Find logical signaling networks such that:

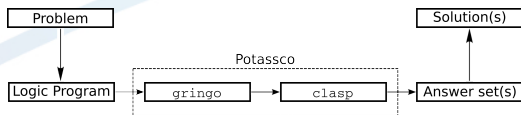
$$\arg \min_{(V, \phi) \in \mathbb{M}(V, E, \sigma)} \underbrace{(\text{Score}_{\text{RSS}}((V, \phi), (P_1, \dots, P_n)), \text{Score}_{\text{size}}((V, \phi)))}_{\text{residual sum of squares}} \underbrace{\hspace{10em}}_{\text{complexity}}$$

- **Structural condition** : networks supported by interaction graph
- **Parsimonious assumption**: minimize model complexity.
- **Fitting condition**: minimize the distance between measured observations and predictions
- **Noise tolerance condition**: Find all models whose MSE are at most 10% higher than the minimal MSE

**Data-noise** → new sub-optimal combinatorial problem

# Answer Set Programming: *what?* instead of *how?*

- Knowledge representation and reasoning problems
- Logical paradigm
- **NP combinatorial problems** → *Constraint satisfaction, diagnosis...*



Potassco: **Potsdam** Answer Set Solving Collection  
<http://potassco.sourceforge.net>

Modeling language → *gringo*

- **Propositional logics**

Solver → *clasp*

- **Boolean constraints** resolution technics

# Added value

## High-level modeling language

**expressivity: ASP  $\simeq$  Prolog**

- **PROPOSITIONAL LOGICS**

→ *ASP program can only consider a finite number of atoms*

- **NEGATION** : smart semantic.

→ *A predicate is false until any fact can predict it is true.*

# Added value

## High-level modeling language

**expressivity: ASP  $\simeq$  Prolog**

- **PROPOSITIONAL LOGICS**

→ *ASP program can only consider a finite number of atoms*

- **NEGATION** : smart semantic.

→ *A predicate is false until any fact can predict it is true.*

## High level solving capability

**ASP  $\simeq$  SAT, ILP**

- Combination of SAT and deductive databases resolution techniques.

→ *No program rewriting*

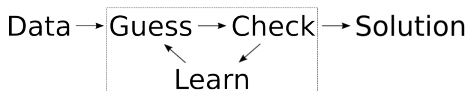
→ The order of clauses has (nearly) no impact

→ **NO INFINITE LOOPS** in the problem resolution

- **OPTIMISATION** is possible with preferences.



# Application to solve the learning issue



**Data:** PKN and phospho-proteomics dataset (facts)

```
node(tnfa). node(p38). edge(tnfa,p38,1). exp(1,tnfa,1). obs(1,p38,0).
```

**Guess:** Generate candidates models (non-deterministic)

```
{clause(A,N)} :- hyperedge(A,N).
```

**Check:** Eliminate invalid models (integrity constraints)

```
:- clause(A,N), clause(B,M), A!=B, redundant(A,B).
```

**Learn:** Loop between "guess" and "check"

**Optimize:** Minimize cost function (weighted sum of atoms)

```
#minimize[mismatch(E,R,W) = W, clause(A,N) : param(P) = N*P].
```

**ASP (answer set programming) methodologies are suitable to solve such combinatorial issues**

# Implementation

caspo software toolbox: <http://bioasp.github.io/caspo/>

## Python package

```
pip install caspo
```

- command-line interface (for end-users)
- python interface (for developers)
- several dependencies
- included in CellNOpt software.

All inclusive distribution: docker container (prototype)

```
docker pull svidela/caspo
```

**caspo**

Reasoning on the response of logical signaling networks with Answer Set Programming.

[View the Project on GitHub](#)  
[bioasp/caspo](#)

[Download ZIP File](#) [Download TAR Ball](#) [Clone via GitHub](#)

### Reasoning on the response of logical signaling networks

The manual identification of logic rules underlying a biological system is often hard, error-prone and time-consuming. Further, it has been shown that, if the inherent experimental noise is considered, many different logical networks can be compatible with a set of experimental observations. Thus, automated **inference of logical networks from experimental data** would allow for identifying admissible large-scale logic models saving a lot of efforts and without any prior knowledge. Instead, once a family of logical networks has been identified, one can suggest or **design new experiments** in order to reduce the uncertainty provided by this family. Finally, one can **look for intervention strategies** (i.e. inclusion/minimal sets of knock-out and knock-outs) that force a set of target species or compounds into a desired steady state. Altogether, this constitutes a pipeline for automated reasoning on logical signaling networks. Hence, the aim of **caspo** is to implement such a pipeline providing a powerful and easy-to-use software tool for systems biologists.

### Installation

If you are already using [Python with NumPy](#), you should be able to install **caspo** from [pip](#) simply by running:

```
$ pip install caspo
```

If you are not using Python and/or NumPy, please visit the [wiki](#) for detailed instructions.

### Usage

Ask for help by running:

```
$ caspo --help
usage: caspo [-h] [--quiet] [--out 0] [--version]
             [control, visualize, design, learn, test, analyze]
```

Reasoning on the response of logical signaling networks w/ optional arguments:

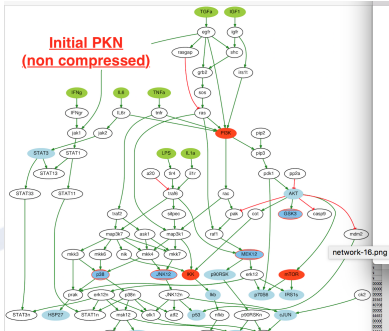
```
-h, --help          show this help message and exit
--quiet            do not print anything to standard
--out 0           output directory path (Default to
--version         show program's version number and
```

**caspo subcommands:**  
for specific help on each subcommand use: `caspo [cmd] -- [control, visualize, design, learn, test, analyze]`

This project is maintained by [bioasp](#)  
Hosted on GitHub Pages — Theme by [ordoval](#)

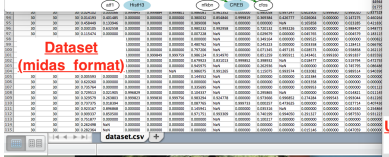
# Few optimal models (0% data noise)

**Initial PKN  
(non compressed)**



A complex biological network diagram with numerous nodes and edges. Nodes are color-coded: green for ligands, red for kinases, and blue for phosphatases. The network shows a dense web of interactions between various proteins.

**Dataset  
(midas format)**



A table showing the dataset in midas format. It contains columns for node IDs and interaction parameters. The table is partially visible, showing rows for various nodes and their interactions.

network-16.png

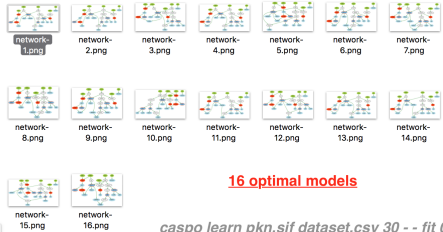
network-1.png network-2.png network-3.png network-4.png network-5.png network-6.png network-7.png

network-8.png network-9.png network-10.png network-11.png network-12.png network-13.png network-14.png

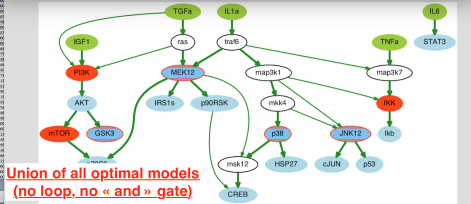
network-15.png network-16.png

**16 optimal models**

*caspo learn pkn.sif dataset.csv 30 -- fit 0*



A grid of 16 smaller network diagrams, each labeled network-1.png through network-16.png. These diagrams show different simplified versions of the initial PKN, representing optimal models. They are arranged in three rows: the first row has 7 models, the second row has 7 models, and the third row has 2 models. Each diagram shows a subset of nodes and edges from the initial network, with varying degrees of complexity and connectivity.



**Union of all optimal models  
(no loop, no « and » gate)**

A network diagram representing the union of all 16 optimal models. It shows a simplified network with 16 nodes: IGF1, IGF1R, PI3K, AKT, mTOR, GSK3, IRS1s, MEK12, p90RSK, p38, msk12, HSP27, cJUN, p53, CREB, IL1a, TrkB, map3k1, mkk4, p38, msk12, HSP27, cJUN, p53, CREB, TNF-a, and STAT3. The nodes are color-coded: green for ligands, red for kinases, and blue for phosphatases. The network shows a central core of interactions between PI3K, MEK12, and AKT, with other nodes branching off.

# More sub-optimal models (2% data noise)

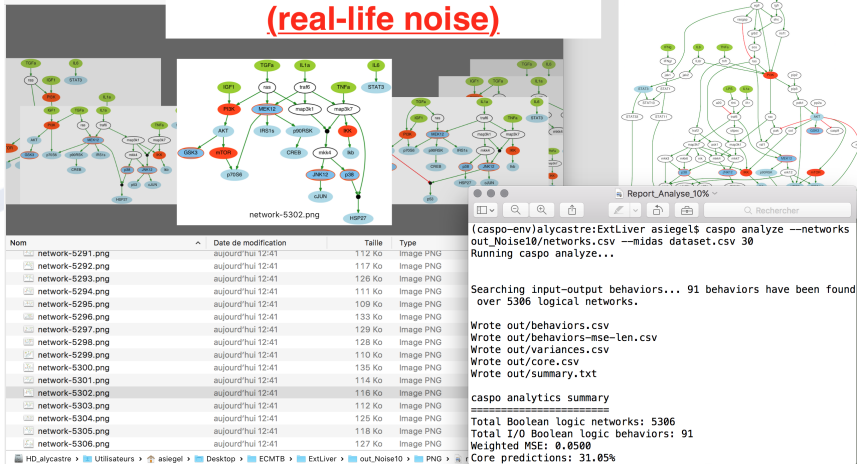
Union of all networks at 2% noise.  
« and » gates appear !

144 networks at 2% noise

*caspo learn pkn.sif dataset.csv 30 -- fit 0.02*

# But quite many sub-optimal models

**5306 networks at 10% noise  
(real-life noise)**



The screenshot displays a file explorer window containing a list of network files. The files are named 'network-5291.png' through 'network-5306.png'. A terminal window titled 'Report Analyse\_10%' is open in the foreground, showing the output of a Caspio analysis. The terminal output includes the command used, the search for input-output behaviors, and a summary of the results.

Nom	Date de modification	Taille	Type
network-5291.png	aujourd'hui 12:41	112 Ko	Image PNG
network-5292.png	aujourd'hui 12:41	117 Ko	Image PNG
network-5293.png	aujourd'hui 12:41	126 Ko	Image PNG
network-5294.png	aujourd'hui 12:41	111 Ko	Image PNG
network-5295.png	aujourd'hui 12:41	109 Ko	Image PNG
network-5296.png	aujourd'hui 12:41	133 Ko	Image PNG
network-5297.png	aujourd'hui 12:41	129 Ko	Image PNG
network-5298.png	aujourd'hui 12:41	128 Ko	Image PNG
network-5299.png	aujourd'hui 12:41	110 Ko	Image PNG
network-5300.png	aujourd'hui 12:41	135 Ko	Image PNG
network-5301.png	aujourd'hui 12:41	114 Ko	Image PNG
network-5302.png	aujourd'hui 12:41	116 Ko	Image PNG
network-5303.png	aujourd'hui 12:41	112 Ko	Image PNG
network-5304.png	aujourd'hui 12:41	125 Ko	Image PNG
network-5305.png	aujourd'hui 12:41	118 Ko	Image PNG
network-5306.png	aujourd'hui 12:41	127 Ko	Image PNG

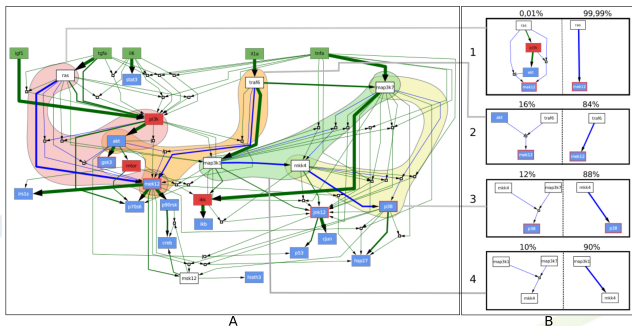
```
(caspo-env)alycastre:ExtLiver asiegel$ caspo analyze --networks out_Noise10/networks.csv --midas dataset.csv 30
Running caspo analyze...

Searching input-output behaviors... 91 behaviors have been found over 5306 logical networks.

Wrote out/behaviors.csv
Wrote out/behaviors-mse-len.csv
Wrote out/variances.csv
Wrote out/core.csv
Wrote out/summary.txt

caspo analytics summary
=====
Total Boolean logic networks: 5306
Total I/O Boolean logic behaviors: 91
Weighted MSE: 0.0500
Core predictions: 31.05%
```

## But quite many sub-optimal models

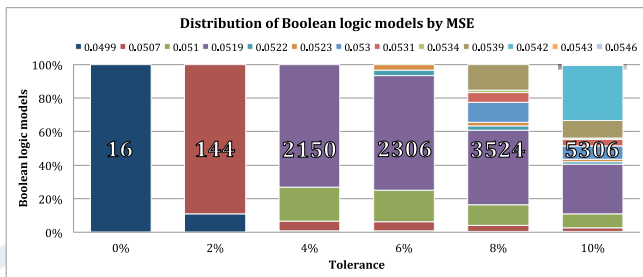


The combinatorics of white nodes introduces a huge disorder

**16 optimal models and 5306 admissible logical networks within 10% of noise tolerance**

*[Guziolowski, ..., Saez-Rodriguez et al., Bioinformatics'13]*

# Dependance to noise tolerance



Non-uniform distribution of logical networks among behaviors

[Guziolowski et al, Bioinformatics'13]

- Half of sub-optimal models were found by 1000 executions of celln-opt
- 4% of tolerance already yields  $\simeq$  2300 different models.

**An exhaustive search of models is mandatory to have a complete view of the variability**

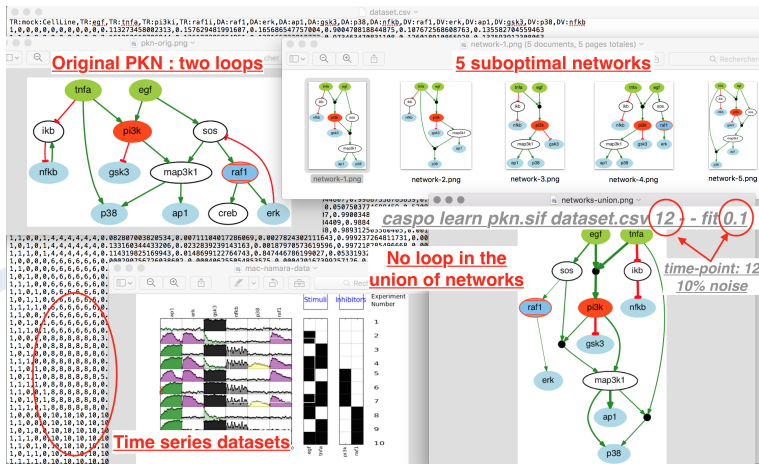
## Possible causes to variability?

- Not enough observations.
- **Parsimony principle** → loops are not learnt → how to learn more complex models ?
- **Early steady-state assumption** → use time-series data?
- **synchronous update** → impact over trajectories and accessibility?

→ **How can we take time-series data into account?**



# A novel dataset: introducing time-series data



Data and example from [mac namara and al; 2014]

Loops are recovered in none of the networks based on the early steady state assumption

# From static to time-series : main issues

## Main issues of the early steady-state problem

Learning problem Test all possible networks in a **large search space**.

Interpretation problem Networks must be mapped to an **information which can be confronted to observation data**.

Early steady-state interpretation Optimize according to **steady states**.

→ loops are naturally removed by the optimization procedure (makes things much simpler).

## Additional issues for time-series data interpretation?

Time-series data interpretation?

**Computing and verifying all dynamical traces is not possible !**

# From static to time-series learning procedure: strategy

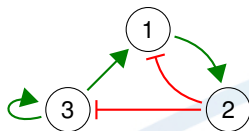
**Abstract the dynamical traces** so that they **reach a fixed point** within a bounded number of steps.

**Leverage the effect of the over-approximation**

*[Paulevé et al, CMSB 2015, Biosystems 2016]*

## Consider a general updating scheme

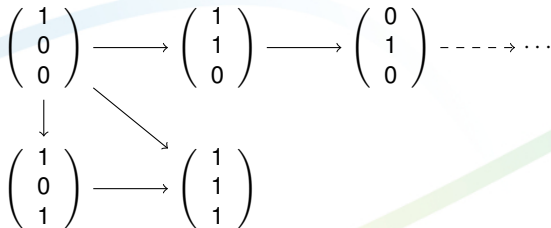
$$\forall x, x' \in \mathbb{B}^n, x \neq x', \quad x \rightarrow x' \Leftrightarrow \forall i \in \{1, \dots, n\}, x_i \neq x'_i \Rightarrow x'_i = f_i(x)$$



$$f_1(x) = \neg x_2 \vee x_3$$

$$f_2(x) = x_1$$

$$f_3(x) = \neg x_2 \vee x_3$$



Non-deterministic dynamics; possibility of loops

Verifying if  $x \rightarrow^* x'$  is hard (exact model-checking; NP-complete)

⇒ **check a weaker condition first.**

# Over-approximating trajectories with meta-states

## Meta-states

- Each node has its value in  $\mathbb{M} = \{0, 1, 0\ 1\}$ .
- If  $u \in \mathbb{M}^n$ ,  $S(u) = \{x \in \mathbb{B}^n \mid \forall i \in \{1, \dots, n\}, x_i \in u_i\}$

$$u = \begin{pmatrix} 0 \\ 0\ 1 \\ 1 \\ 0\ 1 \end{pmatrix} \quad S(u) = \left\{ \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \end{pmatrix} \right\}$$

Mixing 0 and 1 in a meta-state  $0\ 1$  if necessary

## Meta-states dynamics

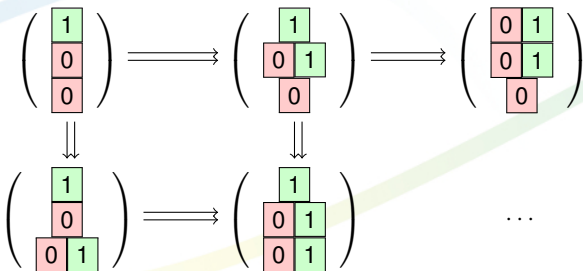
$$\left( \begin{array}{c} u_{1..i-1} \\ \boxed{a} \\ u_{i+1..n} \end{array} \right) \Rightarrow \left( \begin{array}{c} u_{1..i-1} \\ \boxed{0 \ 1} \\ u_{i+1..n} \end{array} \right) \quad \text{if } \exists x \in u : f_i(x) \neq a$$

### Example

$$f_1(x) = \neg x_2 \vee x_3$$

$$f_2(x) = x_1$$

$$f_3(x) = \neg x_2 \vee x_3$$



Verifying  $u \Rightarrow^* v$  is easier than  $x \rightarrow^* y$ :

- $\Rightarrow$  is **strictly monotonous** ( $S(u) \subsetneq S(v)$ );
- no cycles;
- traces have **at most  $n$  steps** (until fixed point).

# Handling time-series data and asynchronous processes?

We want **all models** (Logical Networks)

- compatible with the prior knowledge network (topology);
- that can reproduce the **time series data**.

Necessary conditions for reproducing time series data

- **Quickly invalidate models** with the over-approximation criteria.
- False positives can be filtered out: **a posteriori use of model-checking**.

Distance between Logical Networks and time series data

- When no valid models exist, find close ones (optimization of MSE).
- **"On-The-Fly" linear-like computation the mse**
- Several options wrt parsimony: minimal size, subset-minimal, complete enumeration.

Implementation using Answer-Set Programming (ASP)

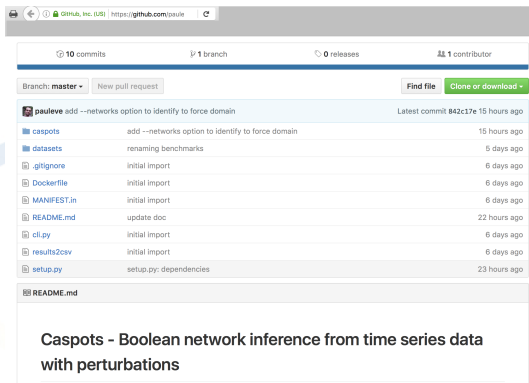
- Declarative approach.
- **Efficient solver for solution enumeration** and optimization.

work with Paulevé, M. Ostrowski, T. Schaub and C. Guziolowski [CMSB 2015]

# Implementation: caspo time-series

Python package

```
git clone https://github.com/pauleve/caspots
```



10 commits 1 branch 0 releases 1 contributor

Branch: master New pull request Find file Clone or download

File	Commit	Time ago
pauleve	add --networks option to identify to force domain	Latest commit 842c17e 15 hours ago
caspots	add --networks option to identify to force domain	15 hours ago
datasets	renaming benchmarks	5 days ago
.gitignore	initial import	6 days ago
Dockerfile	initial import	6 days ago
MANIFEST.in	initial import	6 days ago
README.md	update doc	22 hours ago
cli.py	initial import	6 days ago
results2csv	initial import	6 days ago
setup.py	setup.py: dependencies	23 hours ago

README.md

**Caspots - Boolean network inference from time series data with perturbations**

All inclusive distribution: docker container (prototype)

```
docker pull pauleve/caspots
```

```
docker run --volume "$PWD":/wd --workdir /wd pauleve/caspots
```



# Toy example: cardinal minimality

**2 networks with sub-optimal score (10%) and minimal size**

**Both networks include a loop**

**The over-approximation was exact**

```
(caspo-env)alycastr@MacNamara asiegel$ caspots mse --family mincard --weight-tolerance 10
--check-exact, pkn.sif dataset.csv
MSE_discrete = 6.79443890251
MSE_sample >= MSE_discrete
MSE_sample is exact
```

**The minimal score is smaller than the scores reported by caspo (two-point) networks : 7.0897**

# Toy example: subset minimality

The image shows a file explorer window with 24 thumbnails of network diagrams, labeled network-1.png through network-24.png. Below the thumbnails is a terminal window with the following output:

```
(caspo-env)alycastre:MacNamara asiegel$ docker run --rm --volume "$PWD":/wd --workdir /wd pauleve/caspots identify pkn.sif dataset.csv network_caspoTS_subset.csv --mincard-tolerance 0 --weight-tolerance 10 --family subset --true-positives  
# start initial solving  
# initial solve took 1.02552819252  
# optimizations = [0]  
# begin enumeration  
# enumeration took 46.8882110119es  
24 solution(s) for the over-approximation  
24/24 true positives [rate: 100.00%]
```

The network diagram shows interactions between proteins: **sos**, **raf1**, **erk**, **egf**, **pi3k**, **gsk3**, **map3k1**, **ap1**, **p38**, **tnfa**, **ikb**, and **nfk**. Red and green lines represent different types of interactions. Two loops are highlighted in red and green, with the text "Both loops are recovered" in red. A red dashed arrow points from the terminal output to the network diagram.

**Both loops are recovered**

**24 networks subset-minimal and sub-optimal (10%) Exact overapproximation**

A small heatmap visualization in the bottom right corner shows a grid of black and white squares, with a color scale on the left ranging from 0 to 10.

# Performance and accuracy

## Tests on synthetic time-series data.

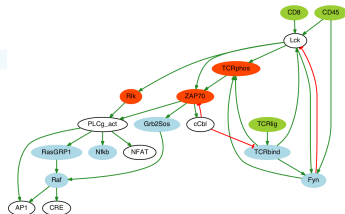
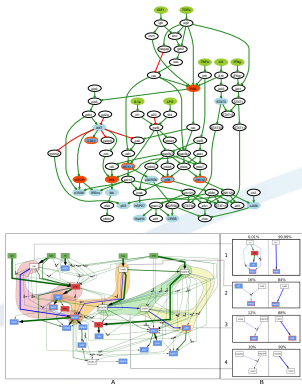
Model	Space	cardinal-minimal			subset-minimal		
		First	Total	TP	First	Total	TP
Case-Study A	$2^{21}$	<1s	8 (1s)	100%	< 1s	54 (2s)	100%
TNF $\alpha$ -EGF [5]		1s	12 (5s)	100%	1s	64 (3s)	100%
13 nodes, 16 edges		<1s	4 (1s)	100%	<1s	36 (3s)	100%
Case-Study B.1	$2^{37}$	1s	18 (5s)	100%	1s	5,544 (3min)	100%
TCR signaling [20]		1s	2 (5s)	100%	1s	2,901 (90s)	100%
14 nodes, 22 edges		1s	8 (5s)	100%	1s	6,510 (4min)	100%
Case-Study B.2	$2^{49}$	2s	4 (12s)	100%	1s	73,962 (1h40)	100%
TCR signaling [20]		3s	4 (25s)	0%	1s	68,338 (1h30)	78%
16 nodes, 25 edges		3s	20 (23s)	90%	1s	74,757 (1h40)	96%
Case-Study B.3	$2^{106}$	4s	8 (90s)	-	5s	>100,000	-
TCR signaling* [20]		6s	8 (90s)	-	5s	>100,000	-
40 nodes, 58 edges		4s	8 (60s)	-	5s	>100,000	-
Case-Study C	$2^{174}$	7s	19 (7min)	42%	6s	>100,000	-
ERBB [21]		3s	2 (2min)	100%	5s	>100,000	-
19 nodes, 50 edges		5s	69 (6min)	19%	5s	>100,000	-

[Paulevé et al, Biosystems (in revision)]

- **Performance** Cardinal minimality is **very efficient**  
[faster by several orders of magnitude than MILP implementation]
- **Enumeration mode** **Subset minimality explodes** in terms of solutions
- **Accuracy** Model-checking reported that **most over-approximated networks** are correct.

**Nearly all inferred BN verifying the over-approximated constraint also satisfied the "real" time-series constraint**

## Partial summary



**Time-series:** 2,901 model with minimal mse and subset minimality property

**Early response:** 3,506 models with minimal size when adding 10% noise to the optimal mse.

→ **Still too many models !!**

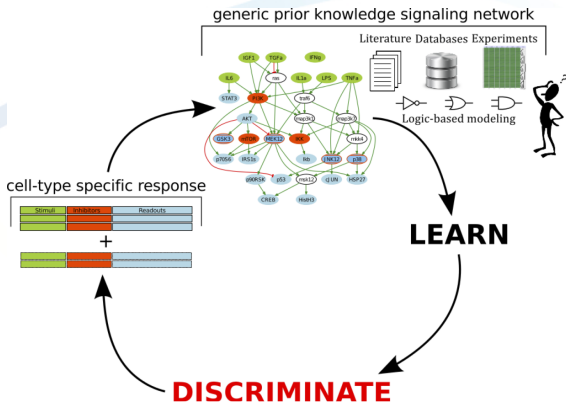
- Not enough observations.
- Variability within single-cells ?

→ **Too many uncertainties to choose a single model within the family**

# Towards discriminations of data?

(intermediate) take-home message

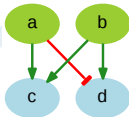
- Numerous sub-optimal models
- Many explanation to such a variability



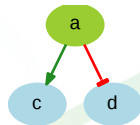
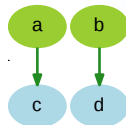
Can we reduce the size of the sub-optimal family by adding experimentations?

# Illustration of the discrimination process

#	Stimuli		Readouts	
	a	b	c	d
1	0	1	0	1
2	1	0	1	0
3	0	0	0	0
4	1	1	1	0



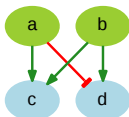
Experiments  
#1, #2  
→  
No tolerance



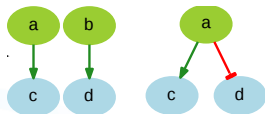
**The models can be discriminated either by exp. 3 or by exp. 4**

# Illustration of the discrimination process

#	Stimuli		Readouts	
	a	b	c	d
1	0	1	0	1
2	1	0	1	0
3	0	0	0	0
4	1	1	1	0

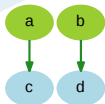


Experiments  
#1, #2  
→  
No tolerance

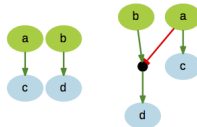


The models can be discriminated either by exp. 3 or by exp. 4

Experiments  
#1, #2, #3  
→  
No tolerance

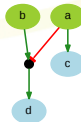


Experiments  
#1, #2, #3  
→  
Size tolerance



Introducing size tolerance reports new models that can be discriminated by exp. 4

Experiments  
#1, #2, #3, #4  
→  
No tolerance



A loop for experimental design requires to play with tolerances

# Experimental design as combinatorial optimization

State-of-the-art [Sharan'13, see also ECCB'14]

Find an input maximizing the difference of the outputs of the rival models

- Optimize *Shannon entropy* wrt possible experiments
- find one experiment to be performed in the same time.
- ILP-based sketched algorithm



# Experimental design as combinatorial optimization

State-of-the-art [Sharan'13, see also ECCB'14]

Find an input maximizing the difference of the outputs of the rival models

- Optimize *Shannon entropy* wrt possible experiments
- find one experiment to be performed in the same time.
- ILP-based sketched algorithm

Cell-type specific experimental data

Stimuli	Inhibitors	Readouts
0 0 0 0 1 0 0	0 0 0 0 0 0 0	0.23 0.84 0.15 0.45 0.98
0 0 0 0 0 0 0	0 0 0 0 0 0 1	0.12 0.78 0.01 0.32 0.02
0 1 0 0 0 0 0	0 0 0 0 0 1 0	0.98 0.34 0.29 0.13 0.75

Combinatorial  
perturbations

Cellular  
response

**Main issue: technologies perform many experimentations  
at the same time!**

# Extended combinatorial problem

Find a set of experimentations to be performed at the same time  
to reduce the variability

- Reduce the family of studied networks to **early-response truth-tables**
- Find the **minimum number of perturbations** which can discriminate all pairs of truth-tables
- Maximize the **sum of pairwise differences** over all pairs
- Minimize the number of **active stimuli and inhibitions**

$$(\forall \beta, \beta' \in \mathbf{B} :: (\exists p \in \mathbf{D} :: \beta(p) \neq \beta'(p))) .$$

Let us denote with  $\mathcal{D}_{(k,s,i)}$  the set of all  $\mathbf{D} \subseteq \mathbf{P}$  with  $|\mathbf{D}| = k$

$$\Theta_{diff}(\mathbf{B}, \mathbf{D}) = \sum_{\beta, \beta' \in \mathbf{B}} \sum_{p \in \mathbf{D}} \mathcal{H}(\beta(p), \beta'(p)) \quad (2)$$

where  $\mathcal{H}$  denotes the Hamming distance over Boolean vectors,

$$\mathbf{D}_{(k,s,i)}^* = \arg \max_{\mathbf{D} \in \mathcal{D}_{(k,s,i)}} \Theta_{diff}(\mathbf{B}, \mathbf{D}) .$$

$$\forall \mathbf{D}^* \in \mathcal{D}_{(k,s,i)}^*, \quad \Theta_U(\mathbf{D}^*) = \sum_{p \in \mathbf{D}^*} \sum_{u_j \in U} p_j$$

$$\mathcal{D}_{opt} = \arg \min_{\mathbf{D}^* \in \mathcal{D}_{(k,s,i)}^*} (\Theta_{V_s}(\mathbf{D}^*), \Theta_{V_t}(\mathbf{D}^*))$$

**Novel problem formulation for experimental design** [Videla et al, Frontiers, 2015]

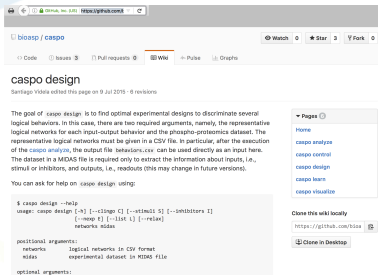
# Implementation: caspo design

## Modeling & solving with ASP

- incremental solving (on the number of experiments)
- lexicographic multi-objective optimization

## Python package

<https://github.com/bioasp/caspo/wiki/caspo-design>



The screenshot shows the GitHub repository page for 'caspo design'. The page title is 'caspo design' and it was edited by Santiago Videla on 9 Jul 2015. The main content is a README file with the following text:

The goal of `caspo design` is to find optimal experimental designs to discriminate several logical behaviors. In this case, there are two required arguments, namely, the representative logical networks for each input-output behavior and the phospho-proteomics dataset. The representative logical networks must be given in a CSV file. In particular, after the execution of the `caspo analyze`, the output file `behaviors.csv` can be used directly as an input here. The dataset in a MIDAS file is required only to extract the information about inputs, i.e., stimuli or inhibitors, and outputs, i.e., readouts (this may change in future versions).

You can ask for help on `caspo design` using:

```
$ caspo design --help
usage: caspo design [-h] [--clingo C] [--stimuli S] [--inhibitors I]
                  [--mmap I] [--list l] [--relax]
                  networks midas
```

positional arguments:

networks	logical networks in CSV format
midas	experimental dataset in MIDAS file

optional arguments:

Pages

- Home
- caspo analyze
- caspo control
- caspo design
- caspo learn
- caspo visualize

Clone this wiki locally

<https://github.com/bioasp/caspo>

Clone in Desktop

## All inclusive distribution: docker container

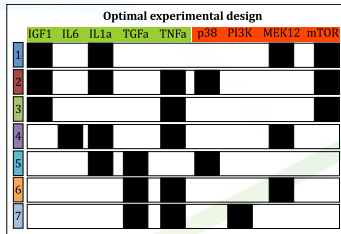
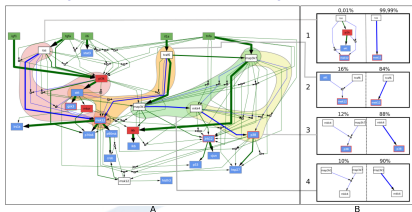
`docker pull svidela/caspo`

`docker run -v -ti /absolute-path-to/output:/opt/out svidela/caspo`



# Scalability?

Search perturbations with up to 3 stimuli and 2 inhibitors (572 exps)



tolerance	behaviors	experiments	$t_{exp}$	$t_{opt}$
2%	4	2	0.061s	0.061s
4%	31	5	5.297s	146.5s
6%	38	5	9.329s	152.5s
8%	66	7	70.52s	~ 5h
10%	91	7	160.1s	~ 18h

## Highlights

- 7 experiments needed to discriminate all behaviors pairwise
- $\binom{572}{7} = 3.8 \times 10^{15}$  possible experimental designs

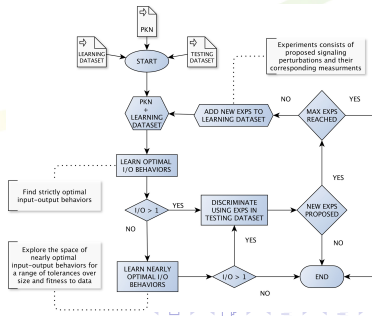
Highly computationally demanding but handled by ASP

# How to test the soundness of the algorithm?

Prerequisite: *Database*  $\mathbb{DB}$  of experimentations:  
family of *perturbations* coupled with their *impact on readouts*

- **Init** Select a set of experimentations  $\mathbb{E}_{learn}$  to train Boolean Networks.
- **Learn** a family  $\mathbf{BN}(\mathbb{E}_{learn})$  optimizing the MSE according to  $\mathbb{E}_{learn}$ .
- **Discriminate**  $\mathbf{BN}(\mathbb{E}_{learn})$  with the best perturbations  $\mathbf{P}$  in  $\mathbb{DB}$ .
- **Increment the family of experimentations**  
$$\mathbb{E}_{learn} \leftarrow \mathbb{E}_{learn} \cup \{\text{result of the discriminative perturbations in } \mathbf{P}\}$$
- **Iterate** Learn a new family  $\mathbf{BN}(\mathbb{E}_{learn})$  (...)
- When there is a single BN, **extend the search space** of BNs to suboptimals.

**Iterative workflow to test the impact of the discrimination procedure**

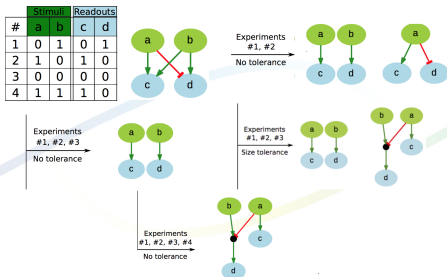


# Soundness?

Do we recover the best BN?

- The learning procedure may be too restrictive enough to select the good BNs.

Behavior of the minimal score of  $\mathbf{BN}(E_{learn})$   
with respect to the complete database of perturbations  $\mathbb{DB}$  ?



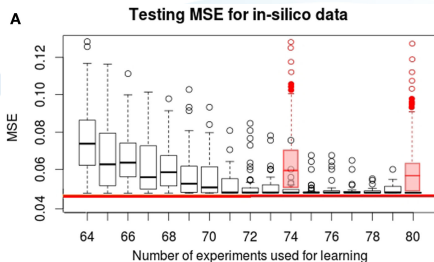
## Artificial case-study

- **Exhaustive  $\mathbb{D}\mathbb{B}$** : all possible  $2^{14}$  experiments simulated from a golden network.
- **Init**: 64 exp. with 0 or 1 stimuli and inhibitors & more complex exp. (from 10 to 16).
- **Discrimination criteria**: at most 5 experiments at each run
- **Ending criteria** 80 perturbations in  $\mathbb{E}_{learn}$ .



## Artificial case-study

- **Exhaustive DB**: all possible  $2^{14}$  experiments simulated from a golden network.
- **Init**: 64 exp. with 0 or 1 stimuli and inhibitors & more complex exp. (from 10 to 16).
- **Discrimination criteria**: at most 5 experiments at each run
- **Ending criteria** 80 perturbations in  $\mathbb{E}_{learn}$ .



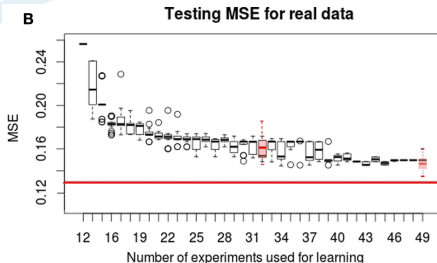
*Average score wrt to the exhaustive perturbation database, procedure applied 100 times*

- The best MSE wrt to the full database is **non monotonous**.
- **Optimal BNs are nearly always identified**
- Much better results than random procedure.

**Good convergence to the best MSE wrt to the full database after 10 experimentations.**

## Real case-study

- **Network**: PKN from [Melas et al, 2012] (12 stimuli, 3 inhibitors, 16 readouts)
- **Partial IDB**: 120 combinatorial experimentations (real data)
- **Init**: 12 screening perturbations (only 1 stimuli/inhibitor in each experiment).
- **Discrimination criteria**: at most 5 experiments at each run
- **Ending criteria** 50 perturbations in  $\mathbb{E}_{learn}$ .



*Average score wrt to the 120 perturbation database, procedure applied 100 times*

**The Best MSE slowly decreases but does not reach the optimal one**

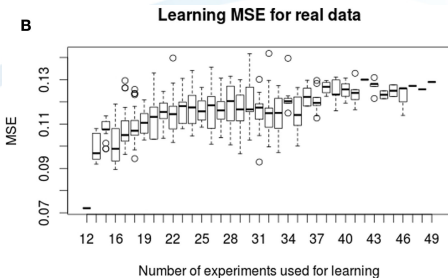
*[Videla et al, Frontiers, 2015]*

## What did happen?

- **Init** The initial set  $\mathbb{E}_{learn}$  used to learn **BN** consisted of 12 different perturbations of a single node.
  - not enough combinatorial process to constrain the search
- **Database of perturbations** There were only 120 different perturbations to select.
  - not enough variability to discriminate

## What did happen?

- **Init** The initial set  $\mathbb{E}_{learn}$  used to learn **BN** consisted of 12 different perturbations of a single node.  
→ not enough combinatorial process to constrain the search
- **Database of perturbations** There were only 120 different perturbations to select.  
→ not enough variability to discriminate



Average score wrt to the ongoing learnt perturbations  $\mathbb{E}_{learn}$

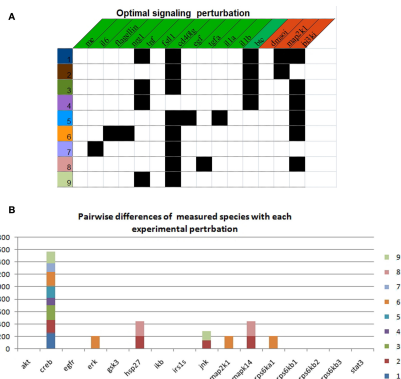
- At first step (screening data), very good MSE.
- When adding the readouts of new perturbations, the best score becomes ugly.

**The discrimination procedure highly depends on the initial perturbation datasets and experimental possibilities**

[Videla et al, Frontiers, 2015]

# Application to the complete 120 perturbation datasets

The best follow-up set of perturbations to 120 existing perturbations can be computed



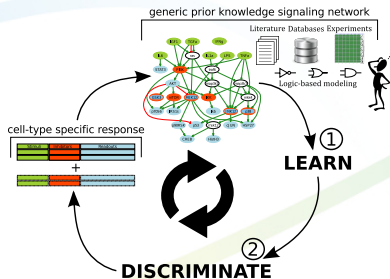
... Although the process is far from being ended

# Conclusion

## Revisiting the loop relying on Answer Set Programming allows gaining robustness

- uniformity
- completeness
- performance

... Although there are still many issues to solve...



Experimental design: Impact of loop and asynchronous dynamics?

Biology: **Test experimental design on real experimentations?**

- Core-reason of variability: single-cell studies?
- Impact of the parsimony assumption

## A more generic question...

The main trick that we used: early steady state, causal abstraction, three value abstraction... allow us to highly simplify the dynamics by reasoning on a single attractor.

**Morality:** We reason over input/output behaviors rather than on the dynamics. ‘

One logical network  $\rightarrow$  one truth table at (pseudo)-steady state

## A more generic question...

The main trick that we used: early steady state, causal abstraction, three value abstraction... allow us to highly simplify the dynamics by reasoning on a single attractor.

**Morality:** We reason over input/output behaviors rather than on the dynamics. ‘

One logical network  $\rightarrow$  one truth table at (pseudo)-steady state

**Question 1:** can we define an extended truth table by mapping an initial state to several attractors ?

**Question 2:** can we reason over such an extended truth table?



# Coming back to dynamical systems

## Historical motivation

Modeling the evolution of a set of components  $\mathbb{A}$  of a system over time over a domain  $\mathbb{T}$ .

## Mathematical framework

$$F : \begin{array}{ccc} \mathbb{T} & \times & \mathbb{S} \\ (t & , & \mathbf{z}) \\ \text{(time} & , & \text{state)} \end{array} \begin{array}{c} \rightarrow \\ \mapsto \end{array} \begin{array}{c} \mathbb{S} \\ F(t, \mathbf{z}) \\ \text{new state at time } t \end{array}$$

## Physics-inspired hypotheses

- Physical laws are precisely set up.
- Sensors enable the measurements of high-level number of components.
- Components are independent.

# Coming back to dynamical systems

## Historical motivation

Modeling the evolution of a set of components  $\mathbb{A}$  of a system over time over a domain  $\mathbb{T}$ .

## Mathematical framework

$$F : \begin{array}{l} \mathbb{T} \times \mathbb{S} \\ (t, \mathbf{z}) \\ \text{(time, state)} \end{array} \begin{array}{l} \rightarrow \\ \mapsto \end{array} \mathbb{S} \begin{array}{l} F(t, \mathbf{z}) \\ \text{new state at time } t \end{array}$$

## Physics-inspired hypotheses

- Physical laws are precisely set up.
- Sensors enable the measurements of high-level number of components.
- Components are independent.

## Biological hypotheses?

- **Biological laws are empirical.**
- Sensors are rather limited.
- **Components are not independent** : we often recover the same compound under several shapes (gene, complex, protein...) within the same network.

**Meta-question: how the hidden dependencies impact the analyses that we are currently performing?**

**Which novel paradigms are required to handle dependencies?**

# Credits

## Dyliss - IRISA, Rennes, France

- Anne Siegel
- **Santiago Videla**
- Jacques Nicolas
- Sven Thiele (now at Max Planck)



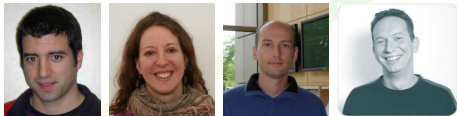
## IRCCYN - Ecole Centrale Nantes, France & LRI (Orsay)

- **Carito Guziolowski**
- Loic Paulevé



## EBI, UK & Greece

- Julio Saez-Rodriguez
- Federica Eduarti
- Thomas Coekaler
- Leonidas Alexopoulos



## Potsdam university, Germany

- Torsten Schaub
- Martin Gebser
- Roland Kaminski
- Max Ostrowski

