my PhD research so far :)

Athénaïs Vaginay, Taha Boukhobza, Malika Smaïl-Tabbone

VINO — August 27, 2022

My field: Systems Biology

Formal modeling and reasoning about biological systems

V: set of **components** of interest (= genes, proteins. . .)

questions in this field:

- "How do their concentration / activity evolve in time?"
- "Can we fix broken (pathological) systems by avoiding undesirable state?"
- "Can we produce more of some desirable product of interest?"

What I have vs what I want

Chemical Reaction Network (CRN) model of an enzymatic process:

$$\mathscr{R} = \overbrace{\underbrace{\mathsf{E} + \mathsf{S} \xleftrightarrow[e_{\text{off}}]{e_{\text{on}}} \mathsf{C}}_{r_{\text{off}}}}^{r_{\text{on}}} \overbrace{\xrightarrow{e_{\text{cat}}} \mathsf{E} + 2 \times \mathsf{P}}^{r_{\text{cat}}}$$

$$V = \{\mathsf{E}, \mathsf{S}, \mathsf{C}, \mathsf{P}\}$$

A **reaction** transforms **reactants** to **products**
at a given **speed** with given **net stoichiometry**

speed: function of the reactants,
and some optional **modifiers**
(activators and/or inhibitors)

$$\mathsf{S} \xrightarrow{e} 2 \times \mathsf{P},\ e = \frac{e_{cat} \times \mathsf{E}_0 \times \mathsf{S}}{K_M + \mathsf{S}}$$

$$\delta_{r_{\text{cat}}}(\mathsf{E}) = +1$$
$$\delta_{r_{\text{cat}}}(\mathsf{S}) = 0$$
$$\delta_{r_{\text{cat}}}(\mathsf{C}) = -1$$
$$\delta_{r_{\text{cat}}}(\mathsf{P}) = +2$$

What I have vs what I want

Chemical Reaction Network (CRN) model of an enzymatic process:

$$\mathscr{R} = \overbrace{\underbrace{\mathsf{E} + \mathsf{S} \underset{e_{\text{off}}}{\overset{e_{\text{on}}}{\longleftrightarrow}} \mathsf{C}}_{r_{\text{off}}}}^{r_{\text{on}}} \overbrace{\xrightarrow{e_{\text{cat}}} \mathsf{E} + 2 \times \mathsf{P}}^{r_{\text{cat}}}$$

$$V = \{\mathsf{E}, \mathsf{S}, \mathsf{C}, \mathsf{P}\}$$

A **reaction** transforms **reactants** to **products**
at a given **speed** with given **net stoichiometry**

speed: function of the reactants,
and some optional **modifiers**
(activators and/or inhibitors)

$$\mathsf{S} \xrightarrow{e} 2 \times \mathsf{P},\ e = \frac{e_{cat} \times \mathsf{E}_0 \times \mathsf{S}}{K_M + \mathsf{S}}$$

$$\delta_{r_{\text{cat}}}(\mathsf{E}) = +1$$
$$\delta_{r_{\text{cat}}}(\mathsf{S}) = 0$$
$$\delta_{r_{\text{cat}}}(\mathsf{C}) = -1$$
$$\delta_{r_{\text{cat}}}(\mathsf{P}) = +2$$

What I have vs what I want

Chemical Reaction Network (CRN) model of an enzymatic process:

$$\mathscr{R} = \overbrace{\underbrace{\mathsf{E} + \mathsf{S} \underset{e_{\text{off}}}{\overset{e_{\text{on}}}{\longleftrightarrow}} \mathsf{C}}_{r_{\text{off}}}}^{r_{\text{on}}} \overbrace{\xrightarrow{e_{\text{cat}}} \mathsf{E} + 2 \times \mathsf{P}}^{r_{\text{cat}}}$$

$$V = \{\mathsf{E}, \mathsf{S}, \mathsf{C}, \mathsf{P}\}$$

A **reaction** transforms **reactants** to **products**
at a given **speed** with given **net stoichiometry**

speed: function of the reactants,
and some optional **modifiers**
(activators and/or inhibitors)

$$\mathsf{S} \xrightarrow{e} 2 \times \mathsf{P},\ e = \frac{e_{cat} \times \mathsf{E}_0 \times \mathsf{S}}{K_M + \mathsf{S}}$$

$$\delta_{r_{\text{cat}}}(\mathsf{E}) = +1$$
$$\delta_{r_{\text{cat}}}(\mathsf{S}) = 0$$
$$\delta_{r_{\text{cat}}}(\mathsf{C}) = -1$$
$$\delta_{r_{\text{cat}}}(\mathsf{P}) = +2$$

What I have vs what I want

Chemical Reaction Network (CRN) model of an enzymatic process:

$$\mathscr{R} = \overbrace{\underbrace{\mathsf{E} + \mathsf{S} \xrightleftharpoons[e_{\text{off}}]{e_{\text{on}}} \mathsf{C}}_{r_{\text{off}}}}^{r_{\text{on}}} \overbrace{\xrightarrow{e_{\text{cat}}} \color{blue}{\mathsf{E}} + 2 \times \color{blue}{\mathsf{P}}}^{r_{\text{cat}}}$$

$$V = \{\mathsf{E}, \mathsf{S}, \mathsf{C}, \mathsf{P}\}$$

A **reaction** transforms **reactants** to **products**
at a given **speed** with given **net stoichiometry**

speed: function of the reactants,
and some optional **modifiers**
(activators and/or inhibitors)

$$\mathsf{S} \xrightarrow{e} 2 \times \mathsf{P},\ e = \frac{e_{cat} \times \mathsf{E}_0 \times \mathsf{S}}{K_M + \mathsf{S}}$$

$$\delta_{r_{\text{cat}}}(\mathsf{E}) = +1$$
$$\delta_{r_{\text{cat}}}(\mathsf{S}) = 0$$
$$\delta_{r_{\text{cat}}}(\mathsf{C}) = -1$$
$$\delta_{r_{\text{cat}}}(\mathsf{P}) = +2$$

What I have vs what I want

Chemical Reaction Network (CRN) model of an enzymatic process:

$$\mathscr{R} = \underbrace{\overbrace{\mathsf{E} + \mathsf{S} \xleftrightarrow[e_{\text{off}}]{e_{\text{on}}} \mathsf{C}}^{r_{\text{on}}}}_{r_{\text{off}}} \overbrace{\xrightarrow{\color{blue}{e_{\text{cat}}}} \mathsf{E} + 2 \times \mathsf{P}}^{r_{\text{cat}}}$$

$$V = \{\mathsf{E}, \mathsf{S}, \mathsf{C}, \mathsf{P}\}$$

A **reaction** transforms **reactants** to **products**
at a given **speed** with given **net stoichiometry**

speed: function of the reactants,
and some optional **modifiers**
(activators and/or inhibitors)

$$\mathsf{S} \xrightarrow{e} 2 \times \mathsf{P},\ e = \frac{e_{cat} \times \mathsf{E}_0 \times \mathsf{S}}{K_M + \mathsf{S}}$$

$$\delta_{r_{\text{cat}}}(\mathsf{E}) = +1$$
$$\delta_{r_{\text{cat}}}(\mathsf{S}) = 0$$
$$\delta_{r_{\text{cat}}}(\mathsf{C}) = -1$$
$$\delta_{r_{\text{cat}}}(\mathsf{P}) = +2$$

What I have vs what I want

Chemical Reaction Network (CRN) model of an enzymatic process:

$$\mathscr{R} = \overbrace{\underbrace{\mathsf{E} + \mathsf{S} \underset{e_{\text{off}}}{\overset{e_{\text{on}}}{\longleftrightarrow}} \mathsf{C}}_{r_{\text{off}}}}^{r_{\text{on}}} \overbrace{\xrightarrow{e_{\text{cat}}} \mathsf{E} + 2 \times \mathsf{P}}^{r_{\text{cat}}}$$

$$V = \{\mathsf{E}, \mathsf{S}, \mathsf{C}, \mathsf{P}\}$$

A **reaction** transforms **reactants** to **products**
at a given **speed** with given **net stoichiometry**

speed: function of the reactants,
and some optional **modifiers**
(activators and/or inhibitors)

$$\mathsf{S} \xrightarrow{e} 2 \times \mathsf{P},\ e = \frac{e_{cat} \times \mathsf{E}_0 \times \mathsf{S}}{K_M + \mathsf{S}}$$

$$\delta_{r_{\text{cat}}}(\mathsf{E}) = +1$$
$$\delta_{r_{\text{cat}}}(\mathsf{S}) = 0$$
$$\delta_{r_{\text{cat}}}(\mathsf{C}) = -1$$
$$\delta_{r_{\text{cat}}}(\mathsf{P}) = +2$$

What I have vs what I want

Chemical Reaction Network (CRN) model of an enzymatic process:

$$\mathscr{R} = \underbrace{\overbrace{\mathsf{E} + \mathsf{S} \underset{e_{\text{off}}}{\overset{e_{\text{on}}}{\longleftrightarrow}} \mathsf{C}}^{r_{\text{on}}}}_{r_{\text{off}}} \overbrace{\xrightarrow{e_{\text{cat}}} \mathsf{E} + 2 \times \mathsf{P}}^{r_{\text{cat}}}$$

$$V = \{\mathsf{E}, \mathsf{S}, \mathsf{C}, \mathsf{P}\}$$

A **reaction** transforms **reactants** to **products**
at a given **speed** with given **net stoichiometry**

speed: function of the reactants,
and some optional **modifiers**
(activators and/or inhibitors)

$$\mathsf{S} \xrightarrow{e} 2 \times \mathsf{P},\ e = \frac{e_{cat} \times \mathsf{E}_0 \times \mathsf{S}}{K_M + \mathsf{S}}$$

$$\delta_{r_{\text{cat}}}(\mathsf{E}) = +1$$
$$\delta_{r_{\text{cat}}}(\mathsf{S}) = 0$$
$$\delta_{r_{\text{cat}}}(\mathsf{C}) = -1$$
$$\delta_{r_{\text{cat}}}(\mathsf{P}) = +2$$

What I have vs **what I want**

Boolean Network (BN): set of n **Boolean expressions** encoding for Boolean functions

$$\{\forall i \in V,\ f_i : \mathbb{B}^n \to \mathbb{B}\}$$

$\mathbb{B} = \{0/\text{inactive}, 1/\text{active}\}$
state: a vector of $\mathbb{B}^n$

What I have vs **what I want**

Boolean Network (BN): set of n **Boolean expressions** encoding for Boolean functions

$$\{\forall i \in V,\ f_i : \mathbb{B}^n \to \mathbb{B}\}$$

$\mathbb{B} = \{0/\text{inactive}, 1/\text{active}\}$
state: a vector of $\mathbb{B}^n$

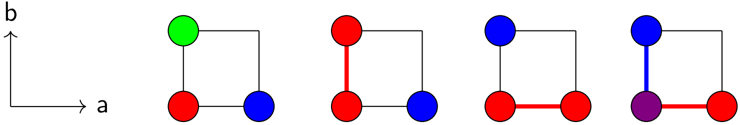
⚠ Boolean function $\neq$ Boolean expression
we consider disjonctive normal form (DNF)

$$(\neg a \wedge \neg b) \vee (a \wedge \neg b) \vee (\neg a \wedge b)$$

$$\neg a \vee (a \wedge \neg b)$$

$$\neg b \vee (\neg a \wedge b)$$

$$\neg a \vee \neg b$$



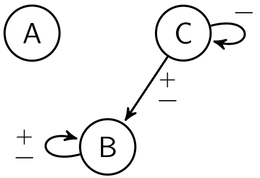
Boolean Network — an example

$$\mathscr{B} = \begin{cases} f_{\mathsf{A}} := 0 \\ f_{\mathsf{B}} := (\mathsf{B} \land \neg \mathsf{C}) \lor (\neg \mathsf{B} \land \mathsf{C}) \\ f_{\mathsf{C}} := \neg \mathsf{C} \end{cases}$$

Boolean Network — an example, its structure

$$\mathscr{B} = \begin{cases} f_{\mathsf{A}} := 0 \\ f_{\mathsf{B}} := (\mathsf{B} \wedge \neg \mathsf{C}) \vee (\neg \mathsf{B} \wedge \mathsf{C}) \\ f_{\mathsf{C}} := \neg \mathsf{C} \end{cases}$$

interaction graph:



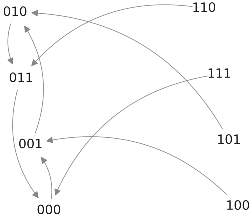
Boolean Network — an example, its dynamics

$$\mathscr{B} = \begin{cases} f_{\mathsf{A}} := 0 \\ f_{\mathsf{B}} := (\mathsf{B} \land \neg\mathsf{C}) \lor (\neg\mathsf{B} \land \mathsf{C}) \\ f_{\mathsf{C}} := \neg\mathsf{C} \end{cases}$$

state transition graph under an update scheme U:

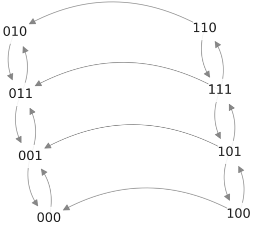
synchronous

$U = \{\{\mathsf{A}, \mathsf{B}, \mathsf{C}\}\}$



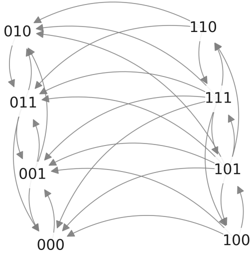
asynchronous

$U = \{\{\mathsf{A}\}, \{\mathsf{B}\}, \{\mathsf{C}\}\}$



general

$U = \mathscr{P}(\{\mathsf{A}, \mathsf{B}, \mathsf{C}\}) \setminus \emptyset$



My PhD project

Goal: synthetise (set of) BN compatible with a given CRN.
Why? Because BNs are easier to understand and to work with for certain tasks (such as control).

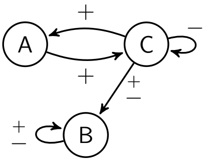
Part I:
Synthesis of BN from a given dynamics and (optional) structure

Part II:
Apply the synthesis on the structure and the dynamics of a CRN

1. Derive the dynamics of a CRN using abstract interpretation of its ODEs
2. Derive the structure and dynamics of a CRN when 1. is not applicable.

Structural Knowledge and Dynamical Data

structural knowledge: Prior Knowledge Network (PKN)
= putative interactions between the components

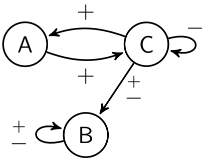


dynamical data: Time Series (TS) = concentrations of the components over time, sequence of states

t	1	2	3	4	5	6	7	8	9	10	11	12	13	...
A	0	3	7	13	20	30	49	61	100	63	36	25	2	...
B	100	86	64	57	54	53	51	49	45	37	33	28	22	...
C	0	27	36	42	60	75	54	44	38	48	60	72	88	...

Structural Knowledge and Dynamical Data

structural knowledge: Prior Knowledge Network (PKN)
= putative interactions between the components

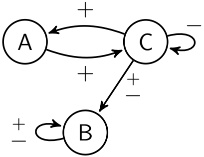


dynamical data: Time Series (TS) = concentrations of the components over time, sequence of states

t	1	2	3	4	5	6	7	8	9	10	11	12	13	...
A	0	3	7	13	20	30	49	61	100	63	36	25	2	...
B	100	86	64	57	54	53	51	49	45	37	33	28	22	...
C	0	27	36	42	60	75	54	44	38	48	60	72	88	...

Structural Knowledge and Dynamical Data

structural knowledge: Prior Knowledge Network (PKN)
= putative interactions between the components

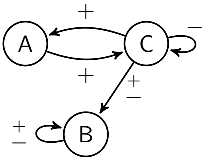


dynamical data: Time Series (TS) = concentrations of the components over time, sequence of states

t	1	2	3	4	5	6	7	8	9	10	11	12	13	...
A	0	3	7	13	20	30	49	61	100	63	36	25	2	...
B	100	86	64	57	54	53	51	49	45	37	33	28	22	...
C	0	27	36	42	60	75	54	44	38	48	60	72	88	...

Structural Knowledge and Dynamical Data

structural knowledge: Prior Knowledge Network (PKN)
= putative interactions between the components

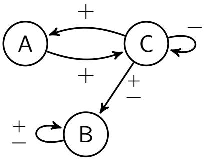


dynamical data: Time Series (TS) = concentrations of the components over time, sequence of states

t	1	2	3	4	5	6	7	8	9	10	11	12	13	...
A	0	3	7	13	20	30	49	61	100	63	36	25	2	...
B	100	86	64	57	54	53	51	49	45	37	33	28	22	...
C	0	27	36	42	60	75	54	44	38	48	60	72	88	...

Structural Knowledge and Dynamical Data

structural knowledge: Prior Knowledge Network (PKN)
= putative interactions between the components



dynamical data: Time Series (TS) = concentrations of the components over time, sequence of states

	010			→	011		→	100		→	001		
t	1	2	3	4	5	6	7	8	9	10	11	12	13	...
A	0	3	7	13	20	30	49	61	100	63	36	25	2	...
B	100	86	64	57	54	53	51	49	45	37	33	28	22	...
C	0	27	36	42	60	75	54	44	38	48	60	72	88	...

Our Wishes VS Existing Approaches

- use a signed PKN + TS
- synthesise *all* the compatible BNs (with all the equivalent minDNFs)
- no assumption on the class of functions and on the underlying update scheme of the seq. of states.

Our Wishes VS Existing Approaches

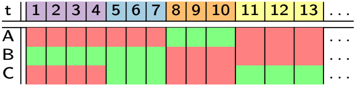
- use a signed PKN + TS
- synthesise *all* the compatible BNs (with all the equivalent minDNFs)
- no assumption on the class of functions and on the underlying update scheme of the seq. of states.

	signed PKN	all minDNF, all class		assumption on TS & config. seq.
`REVEAL`	✗	✗	✓	each timestep = sync. transition
`Best-Fit`	✗	✗	✓	each timestep = sync. transition
`caspo-TS`	✓	✓	monotonous	async. reachability

Our Wishes VS Existing Approaches

- use a signed PKN + TS
- synthesise *all* the compatible BNs (with all the equivalent minDNFs)
- no assumption on the class of functions and on the underlying update scheme of the seq. of states.

	signed PKN	all minDNF, all class		assumption on TS & config. seq.
`REVEAL`	✗	✗	✓	each timestep = sync. transition
`Best-Fit`	✗	✗	✓	each timestep = sync. transition
`caspo-TS`	✓	✓	monotonous	async. reachability



$010 \to 011 \to 100 \to 001$

$010 \to 010 \to 010 \to 010 \to 011 \to 011 \to \ldots$

$010 \to * \to 011 \to * \to 100 \to * \to 001 \to *$

Our own approach: `ASKeD-BN`

1. **local search**: generate all the possible transition functions (in minDNF) compatible with a given PKN and TS.
2. **global assembly**: produce all the possible BNs.

Our own approach: `ASKeD-BN`

1. **local search**: generate all the possible transition functions (in minDNF) compatible with a given PKN and TS.
2. **global assembly**: produce all the possible BNs.

Answer Set Programming (ASP) because. . .

- several tools are now developed with ASP in systems biology
- we can focus only on modeling the problem and not on the way to get the solutions
- we were told it is very fast and efficient, fun to learn, . . .

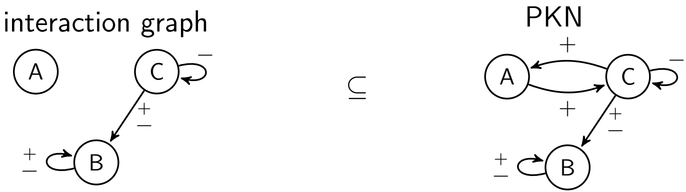
ASKeD-BN— modeling a candidate DNF

pick a subset of conjunctions among all the possible ones (given)

```
% GIVEN : conj(ID, Component, Sign}
conj(0, a,  0). conj(0, b, 0). conj(0, c,  0).
conj(1, a,  1). conj(1, b,-1). conj(1, c,  0). %  A ∧ ¬B
conj(2, a, -1). conj(1, b, 0). conj(1, c, -1). % ¬A ∧ ¬C
conj(3, a, -1). conj(3, b,-1). conj(3, c, -1). % ¬A ∧ ¬B ∧ ¬C
conj(4, a,  1). conj(4, b, 1). conj(4, c,  1). %  A ∧  B ∧  C
...
1{conjTakenID(0..maxNbPossibleConj)}.  % 3^|V| possibilities
conjTaken(I, N, V) :- conj(I, _, _); conjTakenID(I).
```

Example: taken $= \{1, 2\} \rightarrow$ candidate $= (A \wedge \neg B) \vee (\neg A \wedge \neg C)$

ASKeD-BN— structural constraints

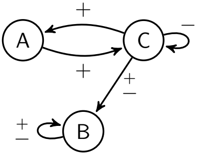


"it is false to select a conjunction that uses a literal that is not allowed by the PKN"

```
ig(ParentID, x, V):- conjTaken(ConjID, ParentID, V); V!=0.
:- ig(ParentID, x, V) ; not pkn(ParentID, x, V).
```

ASKeD-BN— dynamical constraints

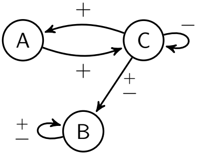
(1) Use state sequence with the parcimonious update schema possible + the PKN to build partial truth tables



$$010 \xrightarrow{\{C\}} 011 \xrightarrow{\{A,B,C\}} 100 \xrightarrow{\{A,C\}} 001$$

ASKeD-BN— dynamical constraints

(1) Use state sequence with the parcimonious update schema possible + the PKN to build partial truth tables



$$010 \xrightarrow{\{C\}} 011 \xrightarrow{\{A,B,C\}} 100 \xrightarrow{\{A,C\}} 001$$

	putative input	output
for A:	C	
0	0	0
1	1	1
for B:	B, C	
0	00	
1	01	
2	10	
3	11	0
for C:	A, C	
0	00	
1	01	0
2	10	1
3	11	

ASKeD-BN— dynamical constraints

(2) discard candidates that doesn't match the truth table

examples of eliminated candidates

for A:

- 0
- $\neg$C

for B:

- 1
- B $\vee$ C
- B $\wedge$ C
- (A $\wedge$ B) $\vee$ ($\neg$A $\wedge$ $\neg$B)

for C:

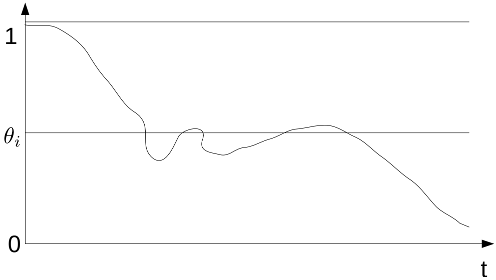
- 0
- 1
- C

	putative input	output
for A:	C	
0	0	0
1	1	1
for B:	B, C	
0	00	
1	01	
2	10	
3	11	0
for C:	A, C	
0	00	
1	01	0
2	10	1
3	11	

ASKeD-BN— dynamical constraints

(3) Optional: minimize the error (to avoid UNSAT)

```
#minimize{MAE@2 : mae(MAE)}. % highest priority
```



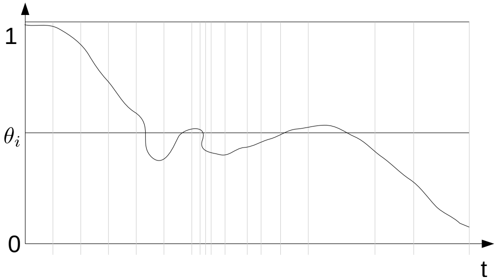
i_t: continuous value of i at time t

θ_i: binarisation threshold for i

ASKeD-BN— dynamical constraints

(3) Optional: minimize the error (to avoid UNSAT)

```
#minimize{MAE@2 : mae(MAE)}. % highest priority
```



i_t: continuous value of i at time t

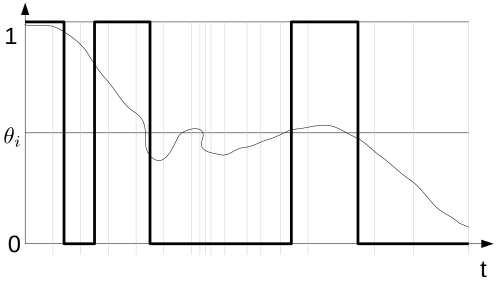
θ_i: binarisation threshold for i

T: # time steps

ASKeD-BN— dynamical constraints

(3) Optional: minimize the error (to avoid UNSAT)

```
#minimize{MAE@2 : mae(MAE)}. % highest priority
```



i_t: continuous value of i at time t

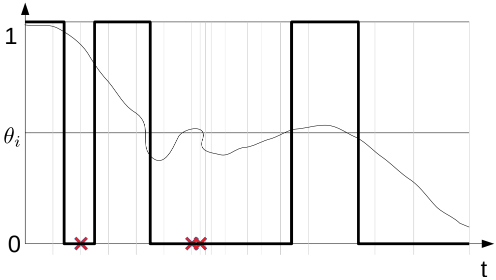
θ_i: binarisation threshold for i

T: # time steps

ASKeD-BN— dynamical constraints

(3) Optional: minimize the error (to avoid UNSAT)

```
#minimize{MAE@2 : mae(MAE)}. % highest priority
```



i_t: continuous value of i at time t

θ_i: binarisation threshold for i

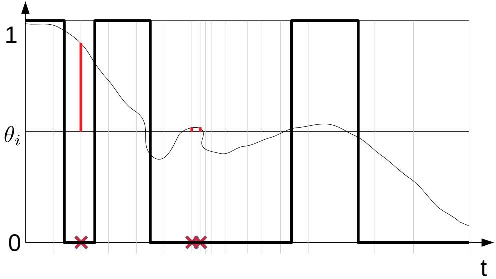
T: # time steps

$\mathscr{U}$: set of unexplained time steps

ASKeD-BN— dynamical constraints

(3) Optional: minimize the error (to avoid UNSAT)

```
#minimize{MAE@2 : mae(MAE)}. % highest priority
```



i_t: continuous value of i at time t
θ_i: binarisation threshold for i
T: # time steps
$\mathscr{U}$: set of unexplained time steps

minimise the Mean Absolute Error (ideally 0)

$$\mathrm{MAE}_{f_i} = \frac{\sum_{t \in \mathscr{U}_{f_i}} |\theta_i - i_t|}{T}$$

ASKeD-BN— minimality constraint

Find the smallest minDNF(s) among the minDNFs compatible with the (partial) truth table

	putative input	output	possible guess			
0	00		0	1	0	1
1	01	0	0	0	0	0
2	10	1	1	1	1	1
3	11		0	0	1	1
		minDNF	$\neg A \wedge B$	$\neg A$	B	$\neg A \vee B$
		size	2	1	1	2

```
sizeconj(C, S):-conjTakenID(C);S=#sum{|V|,N:conj(C, N, V)} .
sizeDNF(S):- S=#sum{N,C: sizeconj(C, N), conjTakenID(C)} .
% N elements in conjunction C
#minimize{S@1 : sizeDNF(S)}. % lower priority
```

Part II:
Apply `ASKeD-BN` on the structure ($\sim$ PKN)
and the dynamics ($\sim$ state sequence) of a CRN

1. use abstract interpretation of differential equations
   $\rightarrow$ applying a joint work done with Joachim Niehren and Cristian Versari
2. use dedicated simulation tools

Differential semantics of a CRN

$$\mathscr{R} = \overbrace{\underbrace{\mathsf{E} + \mathsf{S} \xleftrightarrow[e_{\text{off}}]{e_{\text{on}}} \mathsf{C}}_{r_{\text{off}}}}^{r_{\text{on}}} \overbrace{\xrightarrow{e_{\text{cat}}} \mathsf{E} + 2 \times \mathsf{P}}^{r_{\text{cat}}}$$

one ODE per component

$$\left\{\forall i \in V,\ \frac{\mathrm{d}i}{\mathrm{d}t} = \textstyle\sum_{r \in \mathscr{R}} e_r \times \delta_r(i)\right\}$$

$$\frac{\mathrm{d}[\mathsf{C}]}{\mathrm{d}t} = \underbrace{e_{\text{on}} \times 1}_{r_{\text{on}}} + \underbrace{e_{\text{off}} \times -1}_{r_{\text{off}}} + \underbrace{e_{\text{cat}} \times -1}_{r_{\text{cat}}}$$

Abstract Euler simulation of an ODE system — intuition

$$\dot{a} = 0, \dot{v} = a, \dot{x} = v$$
$$a(0) = 1, v(0) = 0, x(0) = 0$$

Abstract Euler simulation of an ODE system — intuition

$$\dot{a} = 0, \dot{v} = a, \dot{x} = v$$
$$a(0) = 1, v(0) = 0, x(0) = 0$$

with the Euler algorithm, $\Delta_t = 1$:
$i(t) = i(t - \Delta_t) + \dot{i}(t - \Delta_t) \times \Delta_t$

t	0	1	2	3
a	1	1	1	1
v	0	1	1	1
x	0	0	1	2

Abstract Euler simulation of an ODE system — intuition

$$\dot{a} = 0, \dot{v} = a, \dot{x} = v$$
$$a(0) = 1, v(0) = 0, x(0) = 0$$

with the Euler algorithm, $\Delta_t = 1$:
$i(t) = i(t - \Delta_t) + \dot{i}(t - \Delta_t) \times \Delta_t$

t	0	1	2	3
a	1	1	1	1
v	0	1	1	1
x	0	0	1	2

with the analytical solution:

t	0	1	2	3
a	1	1	1	1
v	0	1	1	1
x	0	1	2	3

Abstract Euler simulation of an ODE system — intuition

$$\dot{a} = 0, \dot{v} = a, \dot{x} = v$$
$$a(0) = 1, v(0) = 0, x(0) = 0$$

with the Euler algorithm, $\Delta_t = 1$:
$i(t) = i(t - \Delta_t) + \dot{i}(t - \Delta_t) \times \Delta_t$

t	0	1	2	3
a	1	1	1	1
v	0	1	1	1
x	0	0	1	2

with the analytical solution:

t	0	1	2	3
a	1	1	1	1
v	0	1	1	1
x	0	1	2	3

abstraction from $\mathbb{R}^+ \rightarrow \mathbb{B}$:

$$\begin{cases} 0 \text{ if the value is } 0, \\ 1 \text{ otherwise} \end{cases}$$

abstract state sequence $[a, v, x]$:
100 $\rightarrow$110 $\rightarrow$111 $\circlearrowright$ vs 100 $\rightarrow$111 $\circlearrowright$

Abstract Euler simulation of an ODE system — intuition

$$\dot{a} = 0, \dot{v} = a, \dot{x} = v$$
$$a(0) = 1, v(0) = 0, x(0) = 0$$

with the Euler algorithm, $\Delta_t = 1$:
$i(t) = i(t - \Delta_t) + \dot{i}(t - \Delta_t) \times \Delta_t$

t	0	1	2	3
a	1	1	1	1
v	0	1	1	1
x	0	0	1	2

with the analytical solution:

t	0	1	2	3
a	1	1	1	1
v	0	1	1	1
x	0	1	2	3

abstraction from $\mathbb{R}^+ \rightarrow \mathbb{B}$:

$$\begin{cases} 0 \text{ if the value is } 0, \\ 1 \text{ otherwise} \end{cases}$$

abstract state sequence $[a, v, x]$:
100 $\rightarrow$110 $\rightarrow$111 $\circlearrowright$ vs 100 $\rightarrow$111 $\circlearrowright$

Abstract Euler simulation of the ODEs,
which keeps **causality** between the events,
to compute the abstract state transition graph

ODE system in FOL — syntax

Signature: $\Sigma = \mathscr{V} \cup \mathbb{R} \cup \{+, *\}$

Terms: $e, e' \in \mathscr{E}_\Sigma(\mathscr{V}) \quad ::= x \mid \rho \mid e + e' \mid e * e'$
where $x \in \mathscr{V}$, $\rho \in \mathbb{R}$,

Formula: $\phi, \phi' \in \mathscr{F}_\Sigma(\mathscr{V}) \quad ::= \quad e = e' \mid \exists x \phi \mid \phi \wedge \phi' \mid \neg \phi$
where $x \in \mathscr{V}$, $x \in \mathscr{V}$, and $e \in \mathscr{E}_\Sigma(\mathscr{V})$

ODE system in FOL — syntax

Signature: $\Sigma = \mathscr{V} \cup \mathbb{R} \cup \{+, *\}$

Terms: $e, e' \in \mathscr{E}_\Sigma(\mathscr{V}) \quad ::= x \mid \rho \mid e + e' \mid e * e'$
where $x \in \mathscr{V}$, $\rho \in \mathbb{R}$,

Formula: $\phi, \phi' \in \mathscr{F}_\Sigma(\mathscr{V}) \quad ::= \quad e = e' \mid \exists x \phi \mid \phi \wedge \phi' \mid \neg\phi$
where $x \in \mathscr{V}$, $x \in \mathscr{V}$, and $e \in \mathscr{E}_\Sigma(\mathscr{V})$

variable $\mathscr{V} = \{i \, \forall i \in V\} \cup \{\mathring{i} \, \forall i \in V\}$, ODE system = a FOL formula:

$$odes(R) \ =_{def} \ \bigwedge_{A \in V} \mathring{A} = \sum_{r \in R} \delta_r(A) * e_r \wedge A \geq 0$$

Creation of a new formula from the ODE formula

$$\mathscr{V} = \{i\, \forall i \in V\} \cup \{\mathring{i}\, \forall i \in V\} \cup \{\overrightarrow{i}\, \forall i \in V\} \cup \{\overrightarrow{\mathring{i}}\, \forall i \in V\}$$

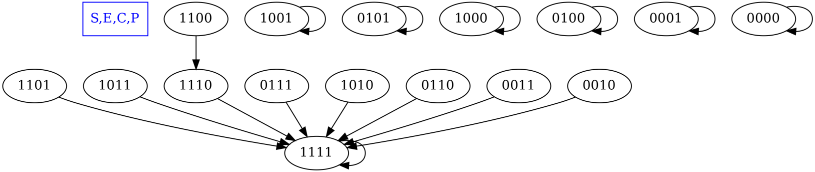
$$\begin{array}{lllllll}
\phi = \exists \mathring{v} \exists \mathring{x}\, \overrightarrow{\mathring{v}}\, \exists\, \overrightarrow{\mathring{x}}\, . & & & & & & \\
 & \mathring{a} = 0 & \wedge & \mathring{v} = a & \wedge & \mathring{x} = v \\
\wedge & \overrightarrow{\mathring{a}} = 0 & \wedge & \overrightarrow{\mathring{v}} = \overrightarrow{a} & \wedge & \overrightarrow{\mathring{x}} = \overrightarrow{v} \\
\wedge & \overrightarrow{a} = a + \mathring{a} & \wedge & \overrightarrow{v} = v + \mathring{v} & \wedge & \overrightarrow{x} = x + \mathring{x} \\
\wedge & a \leq \overrightarrow{a} & \wedge & v \leq \overrightarrow{v} & \wedge & x \leq \overrightarrow{x} \\
\wedge & a \geq 0 & \wedge & v \geq 0 & \wedge & x \geq 0
\end{array}$$

$$fv(\phi) = \{a, v, x, \overrightarrow{a}, \overrightarrow{v}, \overrightarrow{x}\}$$

$\rightarrow$ An abstract state transition graph is made from all the assignment of the free variables that make ϕ true.
Domain = $\mathbb{S} = \{-1, 0, 1\}$ (but on $\mathbb{B}$ for the free variables)

Resulting abstract STG for the enzymatic process

$$\mathscr{R} = \underbrace{\overbrace{\mathsf{E}+\mathsf{S} \xleftrightarrow[e_{\text{off}}]{e_{\text{on}}} \mathsf{C}}^{r_{\text{on}}}}_{r_{\text{off}}} \overbrace{\xrightarrow{e_{\text{cat}}} \mathsf{E} + 2 \times \mathsf{P}}^{r_{\text{cat}}}$$



Future work: run `ASKeD-BN` on this dynamics

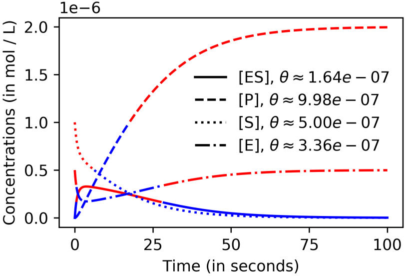
Dynamics of a real-world CRN model, from simulation

$$\mathscr{R} = \overbrace{\underbrace{\mathsf{E} + \mathsf{S} \xleftrightarrow[e_{\text{off}}]{e_{\text{on}}} \mathsf{C}}_{r_{\text{off}}}}^{r_{\text{on}}} \overbrace{\xrightarrow{e_{\text{cat}}} \mathsf{E} + 2 \times \mathsf{P}}^{r_{\text{cat}}}$$

$+$ event, rules, constraints...

$\rightarrow$ Systems Biology Markup Language (SBML)

(1) numerical simulation of the SBML model, (2) binarisation

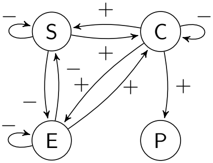


What is the Structure of CRN?

$$\mathscr{R} = \underbrace{\overbrace{\mathsf{E} + \mathsf{S} \xleftrightarrow[e_{\text{off}}]{e_{\text{on}}} \mathsf{C}}^{r_{\text{on}}}}_{r_{\text{off}}} \overbrace{\xrightarrow{e_{\text{cat}}} \mathsf{E} + 2 \times \mathsf{P}}^{r_{\text{cat}}}$$

There is a reaction r in which...

1. X is a reactant and Y disappears **then** X $\xrightarrow{-}$ Y
2. X is an inhibitor and Y appears **then** X $\xrightarrow{-}$ Y
3. X is a reactant or an activator and Y appears **then** X $\xrightarrow{+}$ Y
4. X is an inhibitor and Y disappears **then** X $\xrightarrow{+}$ Y

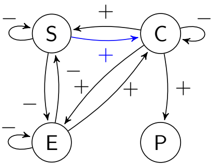


What is the Structure of CRN?

$$\mathscr{R} = \underbrace{\overbrace{\mathsf{E} + \mathsf{S} \xleftrightarrow[e_{\text{off}}]{e_{\text{on}}} \mathsf{C}}^{r_{\text{on}}}}_{r_{\text{off}}} \overbrace{\xrightarrow{e_{\text{cat}}} \mathsf{E} + 2 \times \mathsf{P}}^{r_{\text{cat}}}$$

There is a reaction r in which...

1. X is a reactant and Y disappears **then** X $\xrightarrow{-}$ Y
2. X is an inhibitor and Y appears **then** X $\xrightarrow{-}$ Y
3. X is a reactant or an activator and Y appears **then** X $\xrightarrow{+}$ Y
4. X is an inhibitor and Y disappears **then** X $\xrightarrow{+}$ Y



Evaluation of the simulation based conversion

Nice results on our criteria, on this (**very specific**) application case

- IG $\subseteq$ PKN (by construction)
- the general STG of the synthesised BNs recover a good proportion of the transitions of the sequence.
- small number of BNs syntesised (thanks to mincard minDNF)

Remaining Things to Investigate

When using simulation:

- overfitting to *the* given seq. of state? (drawback of mincard minDNF)
- choice binarisation procedure and error measure?

Math and computer science for a biologist: ✓

Thanks for your attention.

athenais.vaginay@loria.fr

(looking for "write a PhD thesis" and "find a post-doc" advice :)))

$$
\mathscr{R} = \begin{cases} r_{\mathrm{on}} = e_{\mathrm{on}} & : \mathsf{S} + \mathsf{E} \quad \rightarrow \mathsf{C} \\ r_{\mathrm{off}} = e_{\mathrm{off}} & : \mathsf{C} \qquad\quad \rightarrow \mathsf{S} + \mathsf{E} \\ r_{\mathrm{cat}} = e_{\mathrm{cat}} & : \mathsf{C} \qquad\quad \rightarrow \mathsf{E} + 2 \times \mathsf{P} \end{cases}
$$

$$\mathscr{R} = \begin{cases} r_{\text{on}} = e_{\text{on}} & : \mathsf{S} + \mathsf{E} \quad \rightarrow \mathsf{C} \\ r_{\text{off}} = e_{\text{off}} & : \mathsf{C} \qquad\quad \rightarrow \mathsf{S} + \mathsf{E} \\ r_{\text{cat}} = e_{\text{cat}} & : \mathsf{C} \qquad\quad \rightarrow \mathsf{E} + 2 \times \mathsf{P} \end{cases}$$

$$\mathscr{R} = \begin{cases} r_{\text{on}} = e_{\text{on}} & : \mathsf{S} + \mathsf{E} \quad \to \mathsf{C} \\ r_{\text{off}} = e_{\text{off}} & : \mathsf{C} \qquad\quad\; \to \mathsf{S} + \mathsf{E} \\ r_{\text{cat}} = e_{\text{cat}} & : \mathsf{C} \qquad\quad\; \to \mathsf{E} + 2 \times \mathsf{P} \end{cases}$$

$$\mathscr{R} = \begin{cases} r_{\text{on}} = e_{\text{on}} & : \mathsf{S} + \mathsf{E} \rightarrow \color{blue}{\mathsf{C}} \\ r_{\text{off}} = e_{\text{off}} & : \mathsf{C} \rightarrow \color{blue}{\mathsf{S}} + \color{blue}{\mathsf{E}} \\ r_{\text{cat}} = e_{\text{cat}} & : \mathsf{C} \rightarrow \color{blue}{\mathsf{E}} + 2 \times \color{blue}{\mathsf{P}} \end{cases}$$

$$\mathscr{R} = \begin{cases} r_{\text{on}} = e_{\text{on}} & : \mathsf{S} + \mathsf{E} \rightarrow \mathsf{C} \\ r_{\text{off}} = e_{\text{off}} & : \mathsf{C} \rightarrow \mathsf{S} + \mathsf{E} \\ r_{\text{cat}} = e_{\text{cat}} & : \mathsf{C} \rightarrow \mathsf{E} + 2 \times \mathsf{P} \end{cases}$$

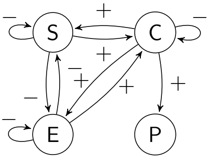
Reaction Network — Structure and Dynamics

$$\mathscr{R} = \overbrace{\underbrace{\mathsf{E} + \mathsf{S} \xleftrightarrow[e_{\text{off}}]{e_{\text{on}}} \mathsf{C}}_{r_{\text{off}}}}^{r_{\text{on}}} \overbrace{\xrightarrow{e_{\text{cat}}} \mathsf{E} + 2 \times \mathsf{P}}^{r_{\text{cat}}}$$

Structure:

1. **If** Y is a reactant and X disappears
   then $\mathsf{Y} \xrightarrow{-} \mathsf{X}$
2. **If** Y is a reactant and X appears
   then $\mathsf{Y} \xrightarrow{+} \mathsf{X}$

[Fages et al. 2008]



Dynamics:
numerical simulation of the ODEs
+ binarisation

	putative input	output	candidate functions							
for B:	B, C									
0	00		0	1	0	1	0	1	0	1
1	01		0	0	1	1	0	0	1	1
2	10		0	0	0	0	1	1	1	1
3	11	0	0	0	0	0	0	0	0	0
			rota	rota	rota	rota	rota	rota	rota	rota

	putative input	output	with guess			
for C:	A, C					
0	00		0	1	0	1
1	01	0	0	0	0	0
2	10	1	1	1	1	1
3	11		0	0	1	1
			$\neg A \land B$			

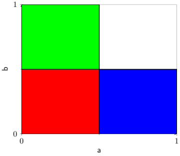
ASKeD-BN— minimality constraints

$\rightarrow$ For finding the minDNF(s) given a truth table

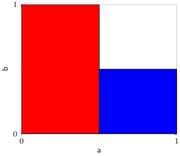
	putative input	output
0	00	1
1	01	1
2	10	1
3	11	0

Several candidate DNFs, but only one minimal

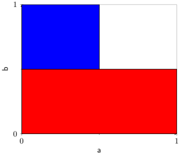
$\neg A \wedge \neg B$
$\vee (A \wedge \neg B)$
$\vee (\neg A \wedge B)$



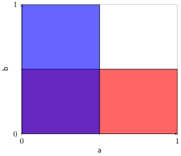
$\neg A$
$\vee (A \wedge \neg B)$



$\neg B$
$\vee (\neg A \wedge B)$



$\neg A \vee \neg B$



What I have vs what I want

Existing mathematical models (**ODEs**, Petri Nets, ...) encoded as CRNs

CRN

$$\mathscr{R} = \overbrace{\mathsf{E} + \mathsf{S} \underset{e_{\mathrm{off}}}{\overset{e_{\mathrm{on}}}{\longleftrightarrow}} \mathsf{C}}^{r_{\mathrm{on}}} \overbrace{\xrightarrow{e_{\mathrm{cat}}} \mathsf{E} + 2 \times \mathsf{P}}^{r_{\mathrm{cat}}}$$

$$\underbrace{\mathsf{E} + \mathsf{S} \longleftrightarrow \mathsf{C}}_{r_{\mathrm{off}}}$$

ODEs system

$$\left\{\forall i \in V,\ \frac{\mathrm{d}i}{\mathrm{d}t} = f(t, x)\right\}$$

differential semantics of a CRN :

$$\left\{\forall i \in V,\ \frac{\mathrm{d}i}{\mathrm{d}t} = \sum_{r \in \mathscr{R}} e_r \times \delta_r(i)\right\}$$

$$\frac{\mathrm{d}[\mathsf{C}]}{\mathrm{d}t} = \underbrace{e_{\mathrm{on}} \times 1}_{r_{\mathrm{on}}} + \underbrace{e_{\mathrm{off}} \times -1}_{r_{\mathrm{off}}} + \underbrace{e_{\mathrm{cat}} \times -1}_{r_{\mathrm{cat}}}$$

ODE system in FOL — semantics

Semantics:
the truth value $||\phi||^{\alpha,S} \in \mathbb{B}$ computed given an interpretation
(= a Σ-structure S + a variable assignment $\alpha : \mathscr{V} \to dom(S)$)

Solutions of ϕ on a given structure = the assignements that make ϕ true: $sol^S(\phi)=\{\alpha_{|fv\phi} \mid \alpha : \mathscr{V} \to dom(S),\ ||\phi||^{\alpha,S} = true\}$.

Athénaïs Vaginay, Taha Boukhobza, Malika Smaïl-Tabbone.
Automatic Synthesis of Boolean Networks from Biological Knowledge and Data. *OLA21: International Conference on Optimization and Learning, Jun 2021, online.*

Joachim Niehren, Athénaïs Vaginay, Cristian Versari.
Abstract Simulation of Reaction Networks via Boolean Networks*.* *CMSB2022: 20th International Conference on Computational Methods in Systems Biology, Sep 2022, Bucarest, Romania.*

Athénaïs Vaginay, Taha Boukhobza, Malika Smaïl-Tabbone.
From quantitative SBML models to boolean networks*. CNA2021: 10th International Conference on Complex Networks and their Applications, Nov 2021, Madrid, Spain.*