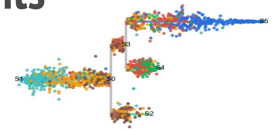
Stéphanie Chevalier,
PhD student
Université Paris-Saclay, LISN (ex LRI)
& Institut Curie, U900

supervisors:
Loïc Paulevé, LaBRI
Andrei Zinovyev, Institut Curie
Christine Froidevaux, LISN

Automatically design Boolean networks from static and dynamical knowledge on a system

example of application:

synthesis of Boolean networks from single-cell trajectory-based constraints



Automatically design models from knowledge on a system

(BOOLEAN NETWORKS) (STRUCTURE & BEHAVIORS)

Boolean network *(BN)*

BOOLEAN NETWORK: **discrete dynamical system**

A Boolean network of dimension *n*

is a function $f : \{0, 1\}^n \rightarrow \{0, 1\}^n$

$\forall i \in [n],\ f_i : \{0, 1\}^n \rightarrow \{0, 1\}$

A **configuration** is a vector $x \in \{0, 1\}^n$

example for a BN with 3 nodes:

➔ *the configuration 011 means:*

- *gene 1 is silenced*
- *genes 2 & 3 are expressed*

Automatically design models from knowledge on a system

(BOOLEAN NETWORKS) (STRUCTURE & BEHAVIORS)

Boolean network *(BN)*

BOOLEAN NETWORK: **discrete dynamical system**

A Boolean network of dimension *n*

is a function $f : \{0, 1\}^n \rightarrow \{0, 1\}^n$

$\forall i \in [n],\ f_i : \{0, 1\}^n \rightarrow \{0, 1\}$

A **configuration** is a vector $x \in \{0, 1\}^n$

example of a BN with 3 nodes:

$$\begin{aligned} f_1(x) &:= \neg x_2 \\ f_2(x) &:= \neg x_1 \\ f_3(x) &:= \neg x_1 \wedge x_2 \end{aligned}$$

Automatically design **models** from **knowledge** on a system

(**BOOLEAN NETWORKS**) (**STRUCTURE & BEHAVIORS**)

Boolean network *(BN)*

BOOLEAN NETWORK: **discrete dynamical system**

A Boolean network of dimension *n*

is a function $f : \{0, 1\}^n \to \{0, 1\}^n$

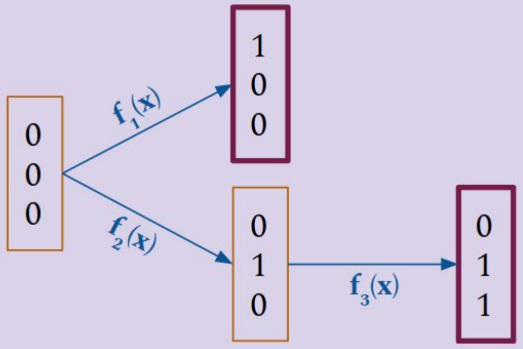
$\forall i \in [n],\ f_i : \{0, 1\}^n \to \{0, 1\}$

A **configuration** is a vector $x \in \{0, 1\}^n$

example of a BN with 3 nodes:

$$f_1(x) := \neg x_2$$
$$f_2(x) := \neg x_1$$
$$f_3(x) := \neg x_1 \wedge x_2$$

Dynamics of a BN:



fully asynchronous dynamics of *f*

→ transition ▢ stable state

Automatically design models from knowledge on a system

(BOOLEAN NETWORKS) (STRUCTURE & BEHAVIORS)

➔ *The aim :*

Direct enumeration of the BNs compatible with the input data (static and dynamical knowledge)

➔ *The methodology :*

Logical inference of a Boolean network from constraints on:

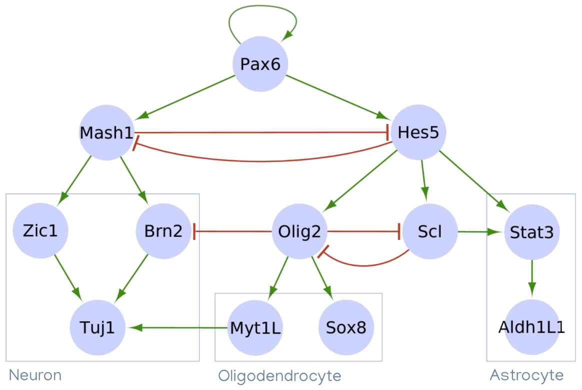
◆ **the domain of its Boolean functions** ◆ **its dynamics**	**⇔ to respect ⇔**	◆ **the knowledge about the structure** ◆ **the observations**

Automatically design **models** from **knowledge** on a system

(**BOOLEAN NETWORKS**) (**STRUCTURE** & **BEHAVIORS**)

The data

STRUCTURE: **known and putative interactions between components**

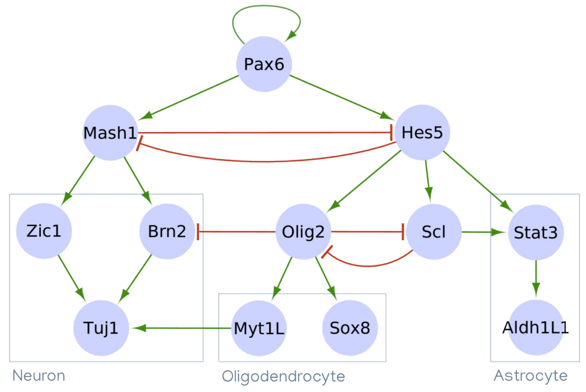


Automatically design **models** from **knowledge** on a system

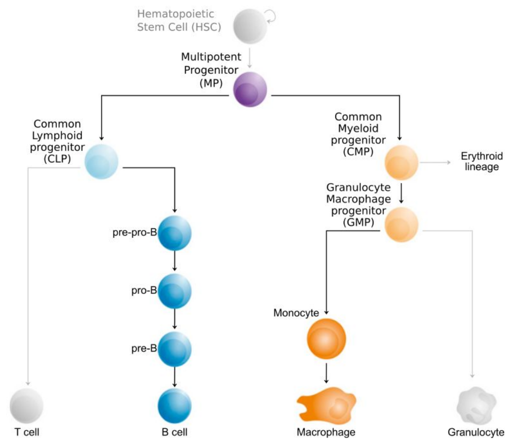
(BOOLEAN NETWORKS) **(STRUCTURE & BEHAVIORS)**

The data

STRUCTURE: **known and putative interactions between components**



BEHAVIORS: **dynamics of biological observations along processes** which are (most of the time) ***partial observations*** of the system

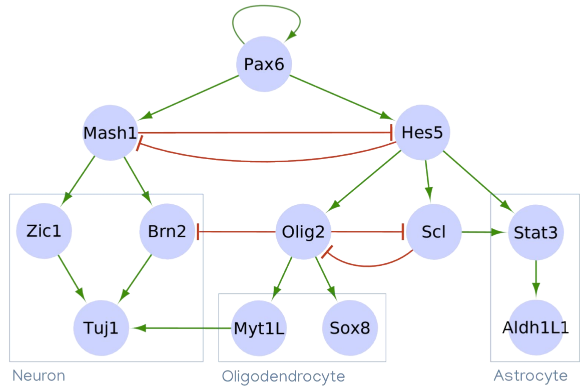

Hematopoietic Stem Cell (HSC)
Multipotent Progenitor (MP)
Common Lymphoid progenitor (CLP)
Common Myeloid progenitor (CMP)
Erythroid lineage
Granulocyte Macrophage progenitor (GMP)
pre-pro-B
pro-B
pre-B
Monocyte
T cell
B cell
Macrophage
Granulocyte

Automatically design **models** from **knowledge** on a system

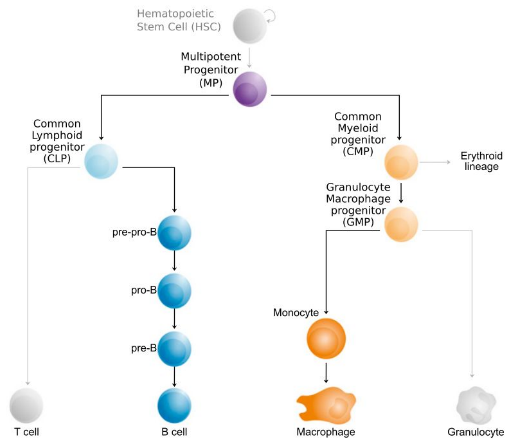
(**BOOLEAN NETWORKS**) (**STRUCTURE** & **BEHAVIORS**)

The data

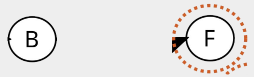
STRUCTURE: **known and putative interactions between components**



BEHAVIORS: **dynamics of biological observations along processes** which are (most of the time) ***partial observations*** of the system


Hematopoietic Stem Cell (HSC)
Multipotent Progenitor (MP)
Common Lymphoid progenitor (CLP)
Common Myeloid progenitor (CMP)
Erythroid lineage
Granulocyte Macrophage progenitor (GMP)
pre-pro-B
pro-B
pre-B
Monocyte
T cell
B cell
Macrophage
Granulocyte

example:



gene expr. in CMP:	*gene expr. in macrophage:*	

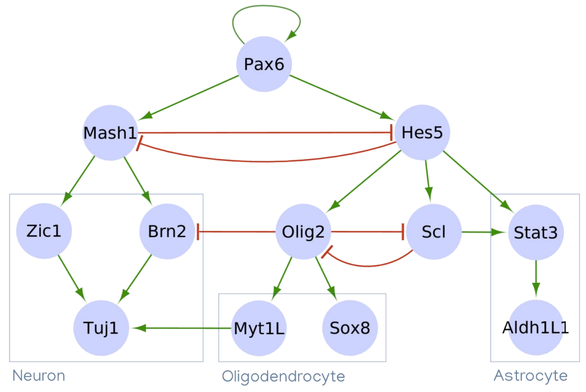
Flt3 = 1
Gfi1 = 0
...
	Flt3 = 1
	Gfi1 = 1
	...
		...

Automatically design **models** from **knowledge** on a system

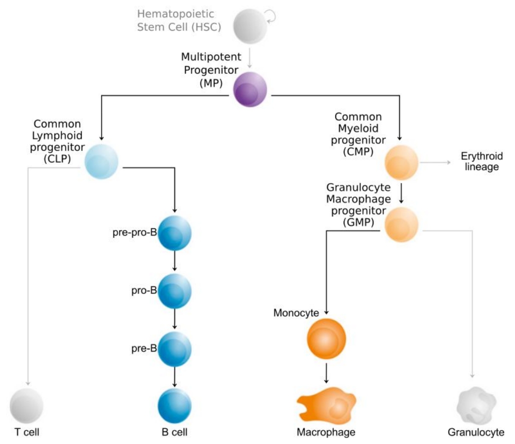
(BOOLEAN NETWORKS) **(STRUCTURE & BEHAVIORS)**

The data

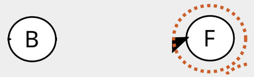
STRUCTURE: **known and putative interactions between components**



BEHAVIORS: **dynamics of biological observations along processes** which are (most of the time) ***partial observations*** of the system

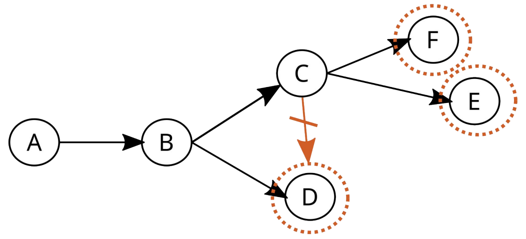

Hematopoietic Stem Cell (HSC)
Multipotent Progenitor (MP)
Common Lymphoid progenitor (CLP)
Common Myeloid progenitor (CMP)
Erythroid lineage
Granulocyte Macrophage progenitor (GMP)
pre-pro-B
pro-B
pre-B
Monocyte
T cell
B cell
Macrophage
Granulocyte

example:



gene expr. in CMP:	*gene expr. in macrophage:*	

Flt3 = 1
Gfi1 = 0
...
	Flt3 = 1
	Gfi1 = 1
	...
		...

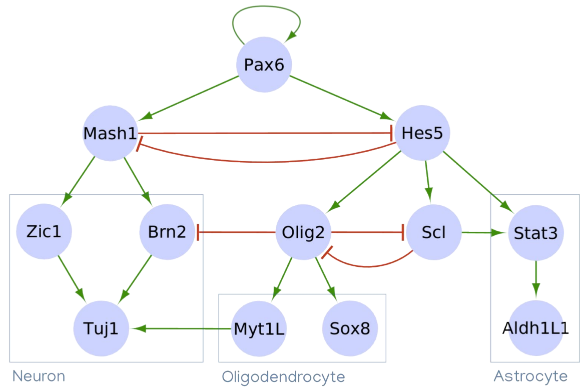


Automatically design **models** from **knowledge** on a system

(**BOOLEAN NETWORKS**) (**STRUCTURE** & **BEHAVIORS**)

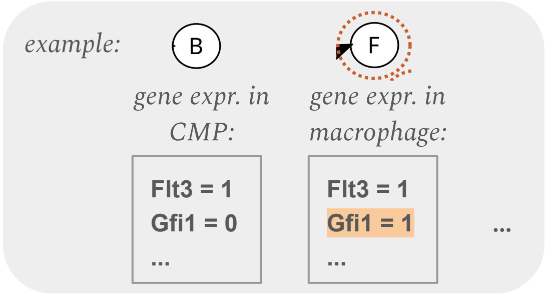
main point: **in input, the data are**

1) static knowledge (PKN)



constrains the domain of the Boolean functions of the models

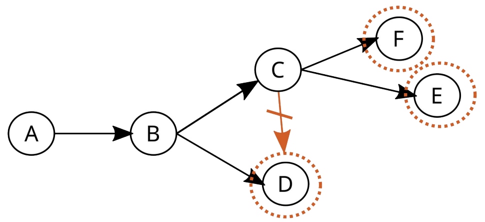
2) dynamical knowledge (observations)



example:

gene expr. in CMP:	*gene expr. in macrophage:*	

Flt3 = 1
Gfi1 = 0
...
	Flt3 = 1
	Gfi1 = 1
	...
		...



constrains the dynamics of the models

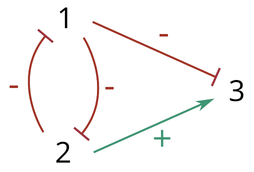
Automatically design **models** from **knowledge** on a system

(**BOOLEAN NETWORKS**) (**STRUCTURE & BEHAVIORS**)

Boolean network inference: a complex problem

STRUCTURE: **known and putative interactions between components**

specifies the domain of the compatible BNs



Possible rules for node 3:

$f_3(x) = 0$; $f_3(x) = x_2$
$f_3(x) = 1$; $f_3(x) = \neg x_1 \wedge x_2$
$f_3(x) = \neg x_1$; $f_3(x) = \neg x_1 \vee x_2$

BEHAVIORS: **dynamics of observations along processes** which are (most of the time) ***partial observations*** of the system

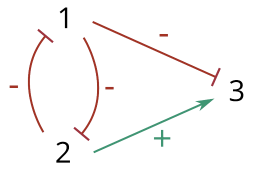
Automatically design **models** from **knowledge** on a system

(**BOOLEAN NETWORKS**) (**STRUCTURE & BEHAVIORS**)

Boolean network inference: a complex problem

STRUCTURE: **known and putative interactions between components**

specifies the domain of the compatible BNs



Possible rules for node 3:

$f_3(x) = 0$;	$f_3(x) = x_2$
$f_3(x) = 1$;	$f_3(x) = \neg x_1 \wedge x_2$
$f_3(x) = \neg x_1$;	$f_3(x) = \neg x_1 \vee x_2$

Combinatorial problem:

indegree	*# monotonic Boolean functions*
0	2
2	6
4	168
6	7,828,354
8	56,130,437,228,687,557,907,788

BEHAVIORS: **dynamics of observations along processes** which are (most of the time) ***partial observations*** of the system

Automatically design **models** from **knowledge** on a system

(**BOOLEAN NETWORKS**) (**STRUCTURE & BEHAVIORS**)

Boolean network inference: a complex problem

STRUCTURE: **known and putative interactions between components**

indegree	*# monotonic Boolean functions*
0	2
2	6
4	168
6	7,828,354
8	56,130,437,228,687,557,907,788

combinatorial explosion

BEHAVIORS: **dynamics of observations along processes** which are (most of the time) ***partial observations*** of the system

Automatically design **models** from **knowledge** on a system

(**BOOLEAN NETWORKS**) (**STRUCTURE & BEHAVIORS**)

Boolean network inference: a complex problem

STRUCTURE: **known and putative interactions between components**

indegree	*# monotonic Boolean functions*
0	2
2	6
4	168
6	7,828,354
8	56,130,437,228,687,557,907,788

combinatorial explosion

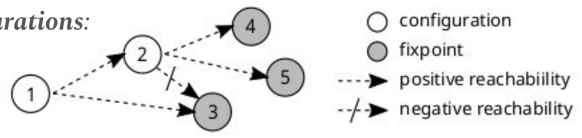
BEHAVIORS: **dynamics of observations along processes** which are (most of the time) ***partial observations*** of the system

a BN is compatible if, in its dynamics, *configurations compatible* with the *partial observations* respect the behaviors (reachability, stable properties)

*Example of **observation** ↔ **configuration compatibility**:*

obs.		*compatible conf.*	
$x_1 = 1$, $x_2 = 0$	→	$x_1 = 1$, $x_2 = 0$, $x_3 = 0$	$x_1 = 1$, $x_2 = 0$, $x_3 = 1$

*Example of **dynamics of compatible configurations**:*



Automatically design **models** from **knowledge** on a system

(**BOOLEAN NETWORKS**) (**STRUCTURE & BEHAVIORS**)

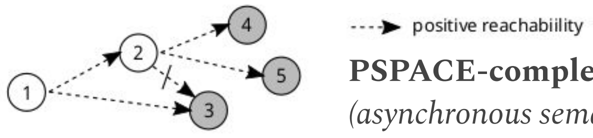
Boolean network inference: a complex problem

STRUCTURE: **known and putative interactions between components**

indegree	*# monotonic Boolean functions*
0	2
2	6
4	168
6	7,828,354
8	56,130,437,228,687,557,907,788

combinatorial explosion

BEHAVIORS: **dynamics of observations along processes** which are (most of the time) ***partial observations*** of the system



PSPACE-complete
(asynchronous semantics)

hard complexity

Automatically design models from knowledge on a system

(BOOLEAN NETWORKS) (STRUCTURE & BEHAVIORS)

Boolean network inference: a complex problem

combinatorial explosion & high complexity

➪ strategy:

Formulate the inference as a Boolean satisfiability problem

Answer-Set Programming: designed for solving combinatorial satisfaction problem

Domain & observations **taken into account during the enumeration**: ~~model checking~~

Principle of the synthesis method

Satisfiability problem

We use **logic programming** with **Answer-Set Programming** to encode the synthesis problem:

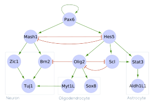
➪ we obtain a big equation, where variables relate to the logical functions in the Boolean network

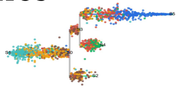
Each solution = **BN showing the complete bifurcation process matching with scRNA-seq data**

Solver: clingo
Can scale to **BNs with thousands of components** (genes) **depending on the properties** ➤ *see ICTAI 2019 paper*

Main lines of the logic program:

- the description of a BN
- the domain of its functions
  = PKN
- the way to compute its dynamic
  = semantics
- the properties of its dynamics
  = observations





The solver enumerates the solutions
(solutions = BNs compatible with data = models)

Automatically design models from knowledge on a system

(BOOLEAN NETWORKS) (STRUCTURE & BEHAVIORS)

Encoding

Brief overview of ASP syntax :

> A Logic Program in ASP is a set of logical rules of the form:
> $a_0 \leftarrow a_1, \ldots, a_n, \text{not } a_{n+1}, \ldots, \text{not } a_{n+k}.$
> with integrity constraints as:
> $\leftarrow a_1, \ldots, a_n, \text{not } a_{n+1}, \ldots, \text{not } a_{n+k}.$

Suitable for solving combinatorial satisfaction problem

Computes **stable models** *[Gelfond and Lifschitz, 1988]*
(minimal sets of a_i satisfying the rules)

Implementation of BN:

Boolean function:
expressed in propositional logic under Disjunctive Normal Form

example: $f_a(x) = \boxed{x_c} \vee \boxed{(\neg x_a \wedge x_b)}$

↓

is encoded:
```
clause(a,1,c,1).
clause(a,2,a,-1).
clause(a,2,b,1).
```

encoded by clause(N,C,L,S) predicates such that:

- atom L
- with sign S (-1, 1)
- is included in the Cth clause
- of f_N

Encoding of the canonicity for exhaustive enumeration:

2 solutions = 2 non-equivalent BNs
➪ enforced by a total ordering between the clauses

Automatically design **models** from **knowledge** on a system

(**BOOLEAN NETWORKS**) (**STRUCTURE & BEHAVIORS**)

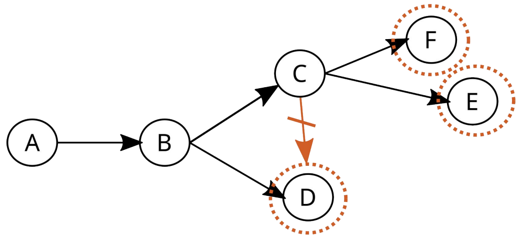
encoding: **2 families of dynamical constraints:**
existence of a property vs **universality of a property**

Existential dynamical constraints: ∃ ...

- checks that, in the BN dynamics, it exists a configuration that respects the property.

Universal dynamical constraints: ∀ ... ∃ ...

- checks the respect of a property over the whole BN dynamics.



Automatically design **models** from **knowledge** on a system

(**BOOLEAN NETWORKS**) (**STRUCTURE & BEHAVIORS**)

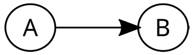
encoding: **2 families of dynamical constraints:**

existence of a property vs **universality of a property**

Existential dynamical constraints:

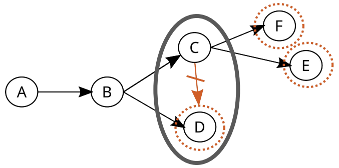
time series: **positive reachability**

∃ path between configurations compatible with successive observations.



bifurcating trajectories: **negative reachability**

∄ path between configurations compatible with bifurcating observations.



stable behaviors:

- fixpoint

A config. compatible with a stable observation is a fixpoint.

- trapspace:

Given an obs. with stability hypotheses on some nodes, these nodes are fixed from a compatible configuration.

Automatically design **models** from **knowledge** on a system

(**BOOLEAN NETWORKS**) (**STRUCTURE & BEHAVIORS**)

encoding: **2 families of dynamical constraints:**
existence of a property vs **universality of a property**

Universal dynamical constraints:

stable behaviors:
- universality in the properties of the reachable fixed points:
we can ensure that, from a time point, no other fixed points than those given are reachable
we can account for observations in different mutants

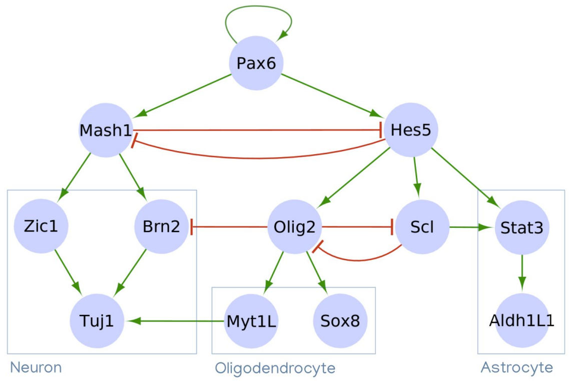
2QBF ($\forall x \exists y.\varphi$ or $\exists y \forall x.\varphi$, with φ a propositional formula without quantifier)
➪ ASP: *saturation technique [Eiter & Gottlob - 1995])*
(disjunctive rule + saturation on the term subject to the disjunction)

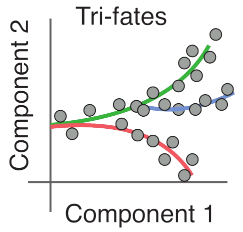
Automatically design models from knowledge on a system

(BOOLEAN NETWORKS) (STRUCTURE & BEHAVIORS)

Test of <u>constraint impact</u>: on a biological application

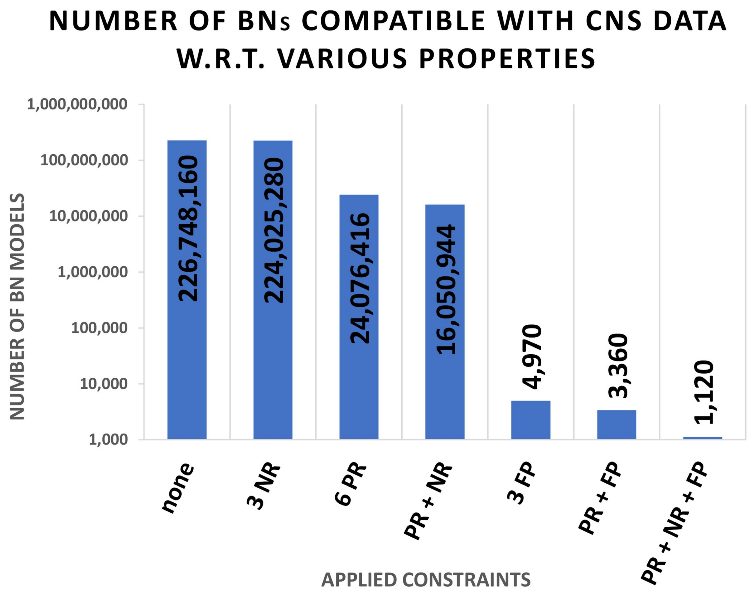
central nervous system development





6 pos. reach (PR)
3 neg. reach (NR)
3 fixpoint (FP)

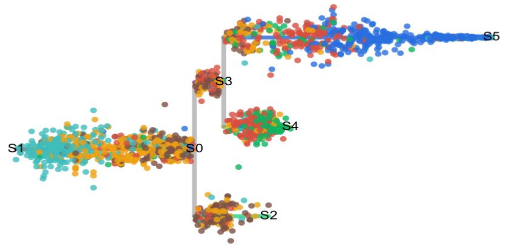
Impact of the constraints:



Methodology to model from scRNA-seq

scRNA-seq differentiation data: gene measurements across cells at different stage of differentiation

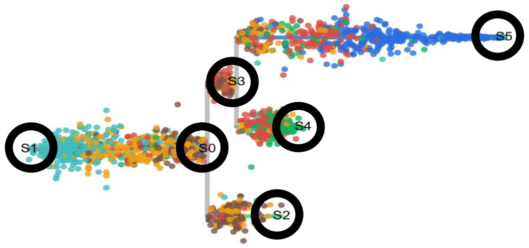
1) From data, we use **trajectory reconstruction** (e.g. STREAM) **to obtain differentiation branches and bifurcation points**



Methodology to model from scRNA-seq

From scRNA-seq data to dynamical constraints

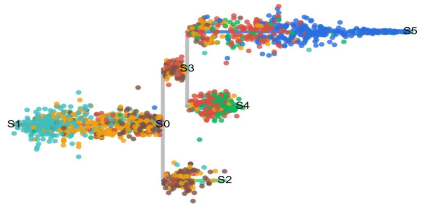
1) From data, we use **trajectory reconstruction** (e.g. STREAM) **to obtain differentiation branches and bifurcation points**



2) Nearby the ends of branches, **a group of cells** is selected. Per gene, the expression data is binarized and the majority value among cells of the time point is retained.

Methodology to model from scRNA-seq

From scRNA-seq data to dynamical constraints



3) **We translate the branches into Boolean dynamical properties**:

   a) positive reachability:
   there is **a path from the beginning to the end of each branch**

   b) negative reachability:
   there is **no path between the diverging branches**

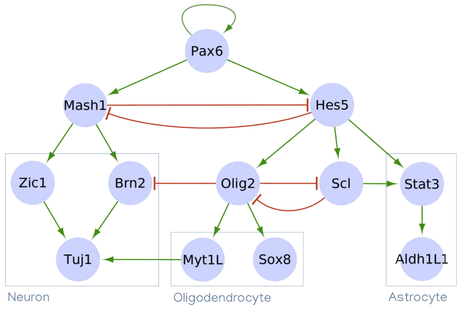
   c) stable properties:
   leafs of the graph are interpreted as **trap spaces** or **attractors** (for now fixed points)

   d) universality in the properties of the reachable fixed points:
   - we can ensure that, from a time point, **no other fixed points than those given are reachable**
   - we can account for observations in **different mutants**

Methodology to model from scRNA-seq

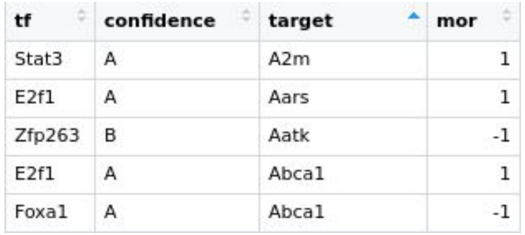
Domain of interactions

4) The possible Boolean functions are generated from a **prior knowledge network** (**PKN**)



Can be extract from interaction databases

e.g. could be a full export of DoRothEA



tf	confidence	target	mor
Stat3	A	A2m	1
E2f1	A	Aars	1
Zfp263	B	Aatk	-1
E2f1	A	Abca1	1
Foxa1	A	Abca1	-1

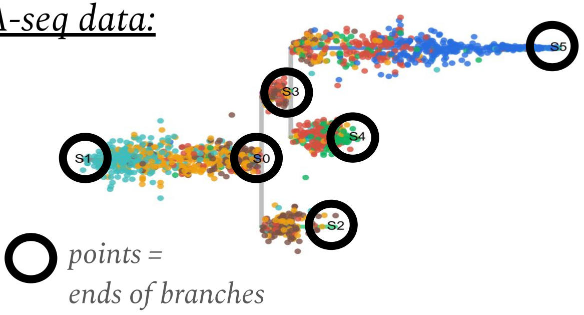
Pruning of the domain (keep only the necessary nodes to explain the dynamical data) thanks to the logic program with optimization.

Automatically design **models** from **knowledge** on a system

(BOOLEAN NETWORKS) **(STRUCTURE & BEHAVIORS)**

Application with scRNA-seq data to study blood cell differentiation

scRNA-seq data:



at each point: around 2400 genes with a binarized value

5 positive reachability (trajectory between successive points)

1 negative reachability (no trajectory between branches)

3 fixpoints (branches ended in a stable state with final measurements)

Prior knowledge network: **DoRothEA** (confidence A & B) ➪ 3112 nodes & 6314 edges

TF → TF & TF → measured genes ➪ 599 nodes & 1396 edges

1) optimisation for PKN reduction (with pos. & neg. reachability, existence of fixpoints for the end of branches)
   ➪ 234 nodes & 554 edges: connected graph with max SCC of 37 nodes
2) model enumeration on the reduced graph

Conclusion: **Automatic design of Boolean networks modelling biological process**

Models = solutions of a logic program
Dynamics described by constraints able to model biological data:

- **with bifurcations** (cell differentiation) :
  -> negative reachability constraint
- **with phenotypic divergence depending on conditions/mutations**
  -> universal fixed point

by considering as domain of knowledge:

- **whole interaction database**
  (DoRothEA, SIGNOR, ...)

Thank you for your attention !

Do you have questions?

stephanie.chevalier@universite-paris-saclay.fr
loic.pauleve@labri.fr
andrei.zinovyev@curie.fr

Our tool "BoNesis": github.com/bioasp/bonesis

Synthesis of Boolean Networks from Biological Dynamical Constraints using Answer-Set Programming
Stéphanie Chevalier, Christine Froidevaux, Andrei Zinovyev, Loïc Paulevé

Synthesis and Simulation of Ensembles of Boolean Networks for Cell Fate Decision
Stéphanie Chevalier, Vincent Noël, Laurence Calzone, Andrei Zinovyev, Loïc Paulevé

Reconciling qualitative, abstract, and scalable modeling of biological networks
Loïc Paulevé, Juraj Kolcak, Thomas Chatain, Stefan Haar