Learning Biological Regulatory Networks from Time Series with LFIT: Theory and Practice

Maxime Folschette

Univ. Lille, CNRS, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, France

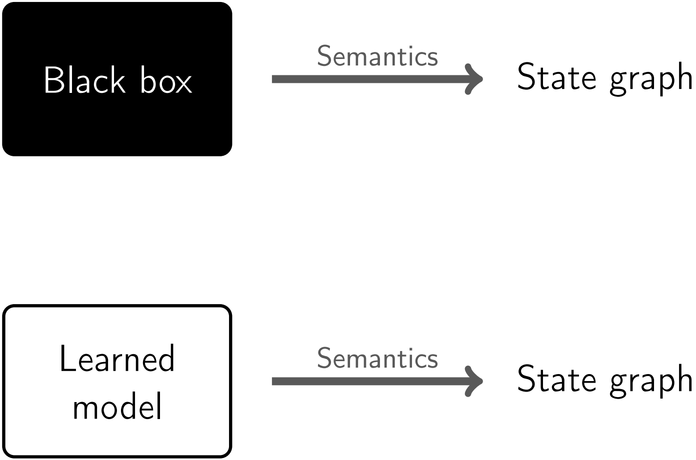
2023-07-03

Workshop on network and model inference (CIRM)

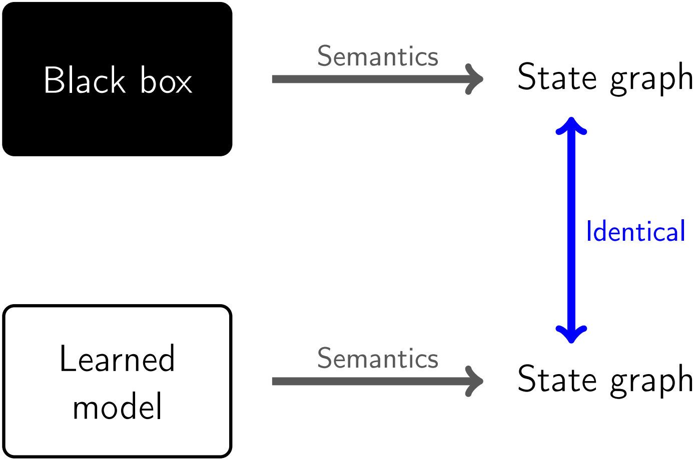
Joint work with: Tony Ribeiro (Independant researcher, France),
Omar Ikne (Univ. Lille, France),
Morgan Magnin (Centrale Nantes, France),
Katsumi Inoue (NII, Tokyo, Japan)

Outline

Introduction



Introduction



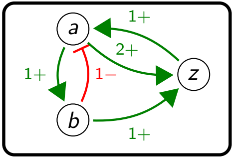
Introduction

Black box

Semantics

State graph

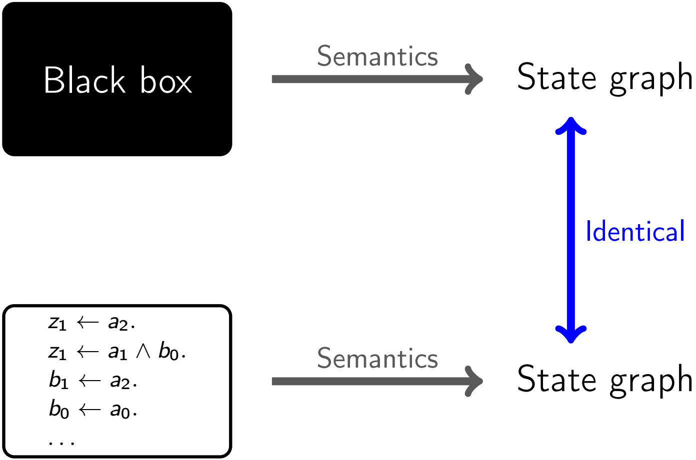
Identical



Semantics

State graph

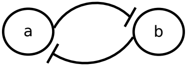
Introduction



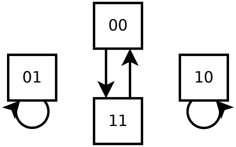
General Definitions

Dynamical Semantics

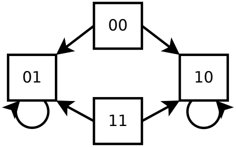
A Boolean network is a (syntactical) structure.
It must be interpreted with a semantics to run.



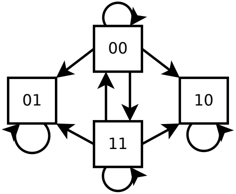
f(a) := not b.
f(b) := not a.



Synchronous



Asynchronous

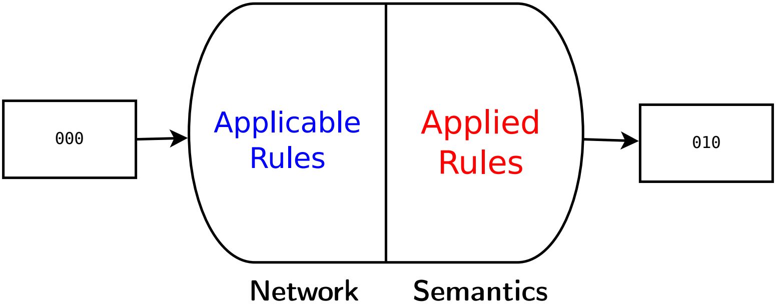


General

- **Synchronous**: all variables are updated
- **Asynchronous**: only one variable is updated
- **General**: any number of variables can be updated

Definition of Semantics

In a given state, among the possible changes permited by the network (structure), the semantics select which ones to apply and how to combine them.



Logic Rules

LFIT learns a logic program, which is a set of logic rules.
It is an alternative representation of biological networks.

$$a_1 \leftarrow a_0, b_0, c_2.$$

The network states that if a and b are at level 0 and c is at level 2, then a can change its value to 1.

$$a_1 \leftarrow c_2.$$

Whenever c is at level 2, a can change its value to 1.

$$a_1 \leftarrow .$$

a can change its value to 1 anytime.

Logic Rules

LFIT learns a logic program, which is a set of logic rules.
It is an alternative representation of biological networks.

$$a_1 \leftarrow a_0, b_0, c_2.$$

The network states that if a and b are at level 0 and c is at level 2, then a **can** change its value to 1.

$$a_1 \leftarrow c_2.$$

Whenever c is at level 2, a **can** change its value to 1.

$$a_1 \leftarrow .$$

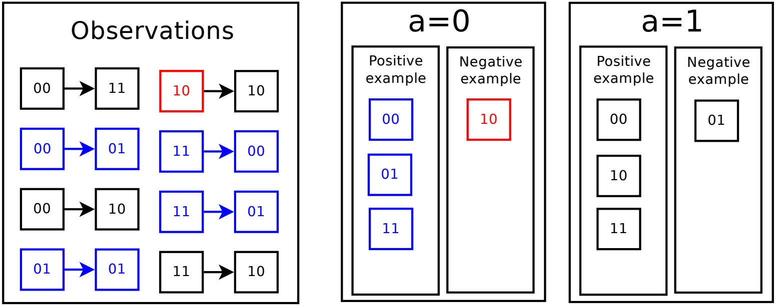
a **can** change its value to 1 anytime.

When **will** a take value 1? This depends on the semantics

Learning From Interpretation Transition (LFIT)

Learning Algorithm Intuition: Classification Problem

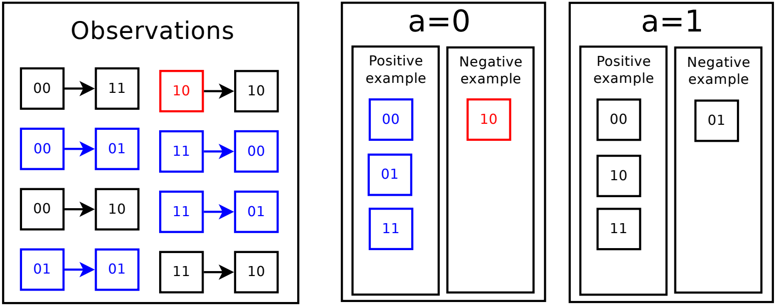
Learn applicable rules: conditions so that a variable **can** take a certain value in next state.



Equivalent to a **classification problem**: What is a typical state where a can take value 0 in the next state ? Here: when a_0 or b_1 is present.

Learning Algorithm Intuition: Classification Problem

Learn applicable rules: conditions so that a variable **can** take a certain value in next state.



Equivalent to a **classification problem**: What is a typical state where a can take value 0 in the next state ? Here: when a_0 or b_1 is present.

$$a_0 \leftarrow a_0. \qquad a_0 \leftarrow b_1.$$

Presentation of GULA

GULA = General Usage LFIT Algorithm

Input: a set of transitions $(s_1 \rightarrow s_2)$

Output: a logic program that respects:

- **Consistency**: the program allows no negative examples
- **Realization**: the program covers all positive examples
- **Completeness**: the program covers all the state space
- **Minimality** of the rules (most general conditions)

Method: start from most general rules and **specialize** iteratively.

Minimal refinements

Suppose: $\text{dom}(a) = \text{dom}(b) = \{0, 1\}$ and $\text{dom}(c) = \{0, 1, 2\}$
and the current program contains the following rules regarding a_1:

$$a_1 \leftarrow c_2. \qquad\qquad a_1 \leftarrow b_1.$$

From state $\langle a_1, b_0, c_2 \rangle$, a_1 is never observed in the next states.

Minimal refinement to make the rules inapplicable in this state:

Minimal refinements

Suppose: $\text{dom}(a) = \text{dom}(b) = \{0, 1\}$ and $\text{dom}(c) = \{0, 1, 2\}$
and the current program contains the following rules regarding a_1:

$a_1 \leftarrow c_2.$

$a_1 \leftarrow b_1.$

From state $\langle a_1, b_0, c_2 \rangle$, a_1 is never observed in the next states.

Minimal refinement to make the rules inapplicable in this state:

$a_1 \leftarrow a_0, c_2.$
$a_1 \leftarrow b_1, c_2.$
$a_1 \leftarrow c_2, c_0.$
$a_1 \leftarrow c_2, c_1.$

$a_1 \leftarrow b_1.$
(No change)

Minimal refinements

Suppose: $\text{dom}(a) = \text{dom}(b) = \{0, 1\}$ and $\text{dom}(c) = \{0, 1, 2\}$
and the current program contains the following rules regarding a_1:

$$a_1 \leftarrow c_2. \qquad a_1 \leftarrow b_1.$$

From state $\langle a_1, b_0, c_2 \rangle$, a_1 is never observed in the next states.

Minimal refinement to make the rules inapplicable in this state:

$$a_1 \leftarrow a_0, c_2.$$
$$a_1 \leftarrow b_1, c_2.$$
$$a_1 \leftarrow c_2, c_0.$$
$$a_1 \leftarrow c_2, c_1.$$

$$a_1 \leftarrow b_1.$$

Minimal refinements

Suppose: $\text{dom}(a) = \text{dom}(b) = \{0, 1\}$ and $\text{dom}(c) = \{0, 1, 2\}$
and the current program contains the following rules regarding a_1:

$$a_1 \leftarrow c_2. \qquad\qquad a_1 \leftarrow b_1.$$

From state $\langle a_1, b_0, c_2 \rangle$, a_1 is never observed in the next states.

Minimal refinement to make the rules inapplicable in this state:

$$a_1 \leftarrow a_0, c_2.$$
$$a_1 \leftarrow b_1, c_2.$$

$$a_1 \leftarrow b_1.$$
(More general)

Minimal refinements

Suppose: $\text{dom}(a) = \text{dom}(b) = \{0, 1\}$ and $\text{dom}(c) = \{0, 1, 2\}$
and the current program contains the following rules regarding a_1:

$$a_1 \leftarrow c_2. \qquad\qquad a_1 \leftarrow b_1.$$

From state $\langle a_1, b_0, c_2 \rangle$, a_1 is never observed in the next states.

Minimal refinement to make the rules inapplicable in this state:

$$a_1 \leftarrow a_0, c_2. \qquad\qquad a_1 \leftarrow b_1.$$

Results

Tony Ribeiro, Maxime Folschette, Morgan Magnin and Katsumi Inoue. **Learning any memory-less discrete semantics for dynamical systems represented by logic programs**. *Machine Learning* 111, Springer. November 2021. `https://doi.org/10.1007/s10994-021-06105-4`

- **Allows to learn the network** (structure of the model)
- **Independent of the semantics**
  (characterization of applicable memoryless semantics)

Nice in theory, but in practice?

- **Exponential complexity** $\rightarrow$ How to handle big datasets?
  (many transitions, many variables)
- **Exact learning** $\rightarrow$ How to handle noise?

Two Heuristic on LFIT

Weighted Likeliness/Unlikeliness Rules

- Use the algorithm twice to learn two logic programs:
  - likeliness rules: what is possible
  - unlikeliness rules: what is impossible
- Weight each rule by the number of observations it matches

Statistical overlay $\Rightarrow$ usable on **noisy datasets**

Likeliness rules	**Unlikeliness rules**
$(3, a_0 \leftarrow b_1)$	$(30, a_0 \leftarrow c_1)$
$(15, a_1 \leftarrow b_0)$	$(5, a_1 \leftarrow c_0)$
$\vdots$	$\vdots$

Using Weighted Likeliness/Unlikeliness Rules

Explainable predictions:

- Compare weights of applicable likeliness/unlikeliness rules
- Ratio of highest weights ⇒ **probability** P
- Rules with highest weights ⇒ **explanation** E

$$\text{predict} : (atom, state) \mapsto (P, E)$$

Likeliness rules

$(3, a_0 \leftarrow b_1)$

$(15, a_1 \leftarrow b_0)$

Unlikeliness rules

$(30, a_0 \leftarrow c_1)$

$(5, a_1 \leftarrow c_0)$

Using Weighted Likeliness/Unlikeliness Rules

Explainable predictions:

- Compare weights of applicable likeliness/unlikeliness rules
- Ratio of highest weights ⇒ **probability** P
- Rules with highest weights ⇒ **explanation** E

$$\text{predict} : (atom, state) \mapsto (P, E)$$

Likeliness rules	**Unlikeliness rules**
$(3, a_0 \leftarrow b_1)$	$(30, a_0 \leftarrow c_1)$
$(15, a_1 \leftarrow b_0)$	$(5, a_1 \leftarrow c_0)$

$\text{predict}(a_1, \langle a_1, b_1, c_0\rangle) = (0.75, ((15, a_1 \leftarrow b_0), (5, a_1 \leftarrow c_0)))$ ⇒ Likely

Using Weighted Likeliness/Unlikeliness Rules

Explainable predictions:

- Compare weights of applicable likeliness/unlikeliness rules
- Ratio of highest weights ⇒ **probability** P
- Rules with highest weights ⇒ **explanation** E

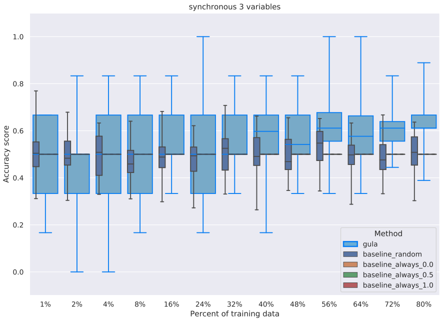
$$\text{predict} : (atom, state) \mapsto (P, E)$$

Likeliness rules	**Unlikeliness rules**
$(3, a_0 \leftarrow b_1)$	$(30, a_0 \leftarrow c_1)$
$(15, a_1 \leftarrow b_0)$	$(5, a_1 \leftarrow c_0)$

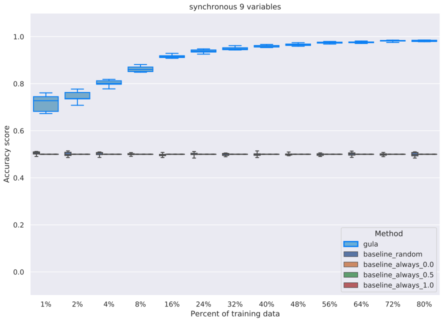
$\text{predict}(a_1, \langle a_1, b_1, c_0 \rangle) = (0.75, ((15, a_1 \leftarrow b_0), (5, a_1 \leftarrow c_0)))$ ⇒ Likely

$\text{predict}(a_0, \langle a_1, b_1, c_0 \rangle) = (0.09, ((3, a_0 \leftarrow b_1), (30, a_0 \leftarrow c_1)))$ ⇒ Unlikely

Prediction power



3 variables



9 variables

Training data = X% of transitions
Tested against unseen states (not in the training data)

PRIDE: Polynomial Alternative to GULA

GULA: **Exponential complexity** in the number of variables

PRIDE: Greedy version of **GULA** that only keeps the first compatible minimal refinement ⇒ subset of rules

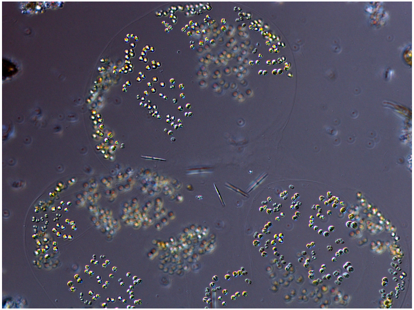
- **Consistency**: the program allows no negative examples
- **Realization**: the program covers all positive examples
- ~~**Completeness**: the program covers all the state space~~
- **Minimality** of the rules (most general conditions)

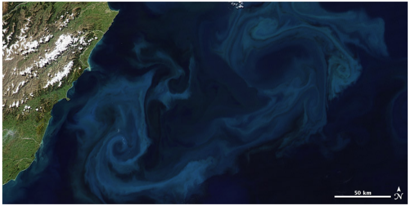
...And the results depends on the ordering of variables

Polynomial complexity ⇒ usable on **large datasets**

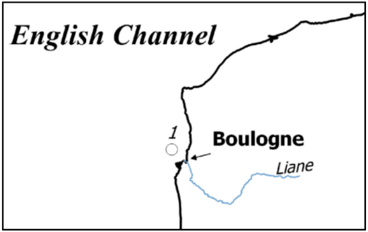
Application: Dynamics of Marine Phytoplankton

Phytoplankton Blooms





SRN Dataset



https://www.seanoe.org/data/00397/50832/

Sampling location	Sampling date	Taxon	Value	Sampling depth
001-P-015	1992-05-18	CHLOROA	6.0	Surface (0-1m)
006-P-001	2019-12-02	Chaetoceros	1000.0	Surface (0-1m)
002-P-007	1994-05-25	Pleurosigma	100.0	Surface (0-1m)
002-P-030	2005-10-19	SALI	34.83	Surface (0-1m)
006-P-007	2015-09-28	Guinardia delicatula	11400.0	Surface (0-1m)

Environmental variables (7)   Phytoplankton species (12)

Applying LFIT

Expectations

- Find known **abiotic** influences (of environment on phytoplankton)
- Find new **biotic** influences (of phytoplankton species on others)

Input

- Pre-processing: data cleaning + discretization
- 253 training transitions
- 53 testing transitions

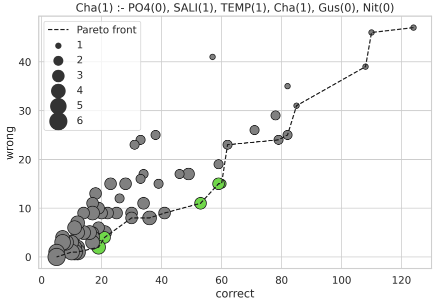
Output

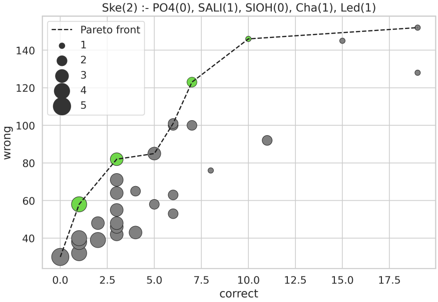
- Run time = 2.35s (**PRIDE**)
- 1683 likeliness rules
- 1981 unlikeliness rules
- Model accuracy: **0.670**

Model Improvement

Pareto frontier

- For likeliness rules : maximize correct and minimize wrong weights
- For unlikeliness rules : maximize wrong and minimize correct weights





Accuracy improvement: 0.670 → **0.716**

Likeliness rules: 1683 → **1609**

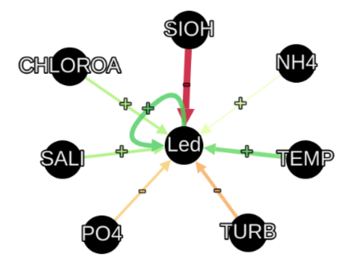
Unlikeliness rules: 1981 → **1405**

Global Influences

Process: Search and count patterns in rules that characterize an activation/inhibition
Hypotheses: Monotonous influences & same threshold for all variables
Result: Score $[-1; +1]$ between each pair of variables (no threshold)

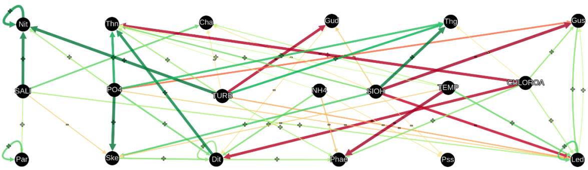
Influences on phytoplankton specie Led:

Variable	Positive	Negative	Global
PO4	$+0$	-58	-0.36
SALI	$+71$	-4	$+0.42$
CHLOROA	$+84$	-22	$+0.39$
SIOH	$+3$	-161	-0.98
NH4	$+25$	-5	$+0.12$
TEMP	$+106$	-5	$+0.63$
TURB	$+10$	-87	-0.48

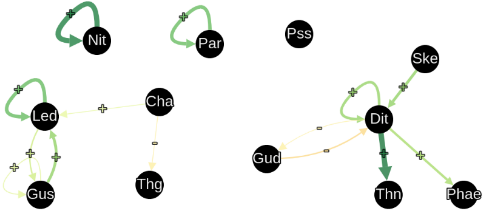


$$\text{global\_influence}(\texttt{PO4} \rightarrow \texttt{Led}) = \frac{+0 + (-58)}{161} = -0.36$$

Results



Global influence graph (biotic and abiotic interactions)



Biotic interactions (between phytoplankton only)

Very few biotic interactions...
Future work: integrate knowledge + validate results

Conclusion

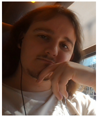
Conclusion

- **Learn** the network with LFIT (theory)
- **Heuristics** to tackle real data (practice)
- **Application** to phytoplankton

Outlooks:

- Quatify how many rules are “missed” by PRIDE
- Integrate biological knowledge to improve learning
- Improve the Biological network inference
- ...

Thanks



Tony
RIBEIRO

Omar
IKNE

Morgan
MAGNIN

Katsumi
INOUE

Cédric
LHOUSSAINE

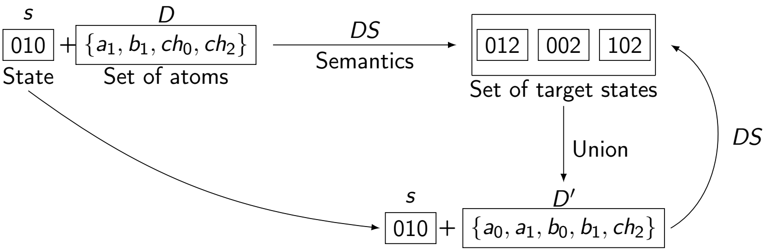
Sébastien
LEFEBVRE

Bibliography

- About GULA: Tony Ribeiro, Maxime Folschette, Morgan Magnin and Katsumi Inoue. **Learning any memory-less discrete semantics for dynamical systems represented by logic programs**. *Machine Learning* 111, Springer. November 2021. `https://doi.org/10.1007/s10994-021-06105-4`
- pyLFIT Python library: `https://github.com/Tony-sama/pylfit`
- About PRIDE: Tony Ribeiro, Maxime Folschette, Morgan Magnin and Katsumi Inoue. **Polynomial Algorithm For Learning From Interpretation Transition**. Poster at the *1st International Joint Conference on Learning & Reasoning*. October 2021, Online. `https://hal.science/hal-03347026v1`
- About the application: Omar Iken, Maxime Folschette and Tony Ribeiro. **Automatic Modeling of Dynamical Interactions Within Marine Ecosystems**. Poster in the *1st International Joint Conference on Learning & Reasoning*. October 2021, Online. `https://hal.science/hal-03347033v1`

Pseudo-idempotent semantics

GULA can model observations from any pseudo-idempotent semantics.



$$\longrightarrow DS(s, D) = DS\Big(s, \bigcup_{s' \in DS(s,D)} s'\Big)$$

where DS is the dynamical semantics, and D is set of heads of rules of a multi-valued logic program that match the sate s.