Learning From Interpretation Transitions
- LFIT 2024 Summary -

Tony Ribeiro

Univ. Nantes, CNRS, Centrale Nantes, UMR 6004 LS2N, F-44000 Nantes, France
Independent Researcher

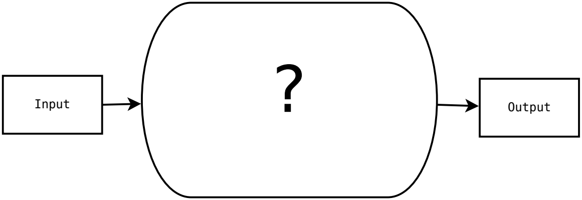
18th April 2024, LS2N, Nantes

Outline

Outline

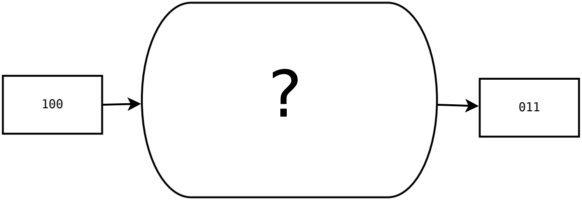
Learning From Interpretation Transitions

Idea: given a set of input/output states of a black-box system, learn its internal mechanics.



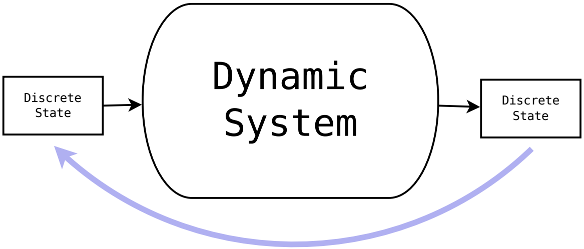
Learning From Interpretation Transitions

Discrete system: input/output are vectors of same size which contain discrete values.



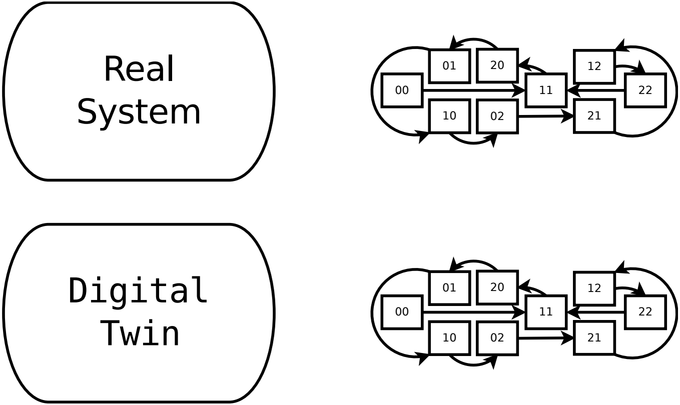
Learning From Interpretation Transitions

Dynamic system: input/output are states of the system and output becomes the next input.



Learning From Interpretation Transitions

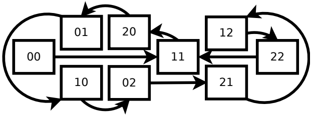
Goal: produce an artificial system with the same behavior as the one observed, i.e., a digital twin.



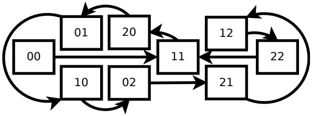
Learning From Interpretation Transitions

Representation: propositional logic programs with annotated atoms encoding multi-valued variables.

Real System



Logic Program

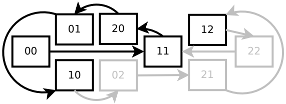


Learning From Interpretation Transitions

Method: learn the dynamics of systems from the observations of some of its state transitions.

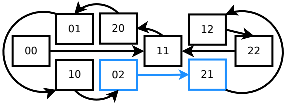
DATA

?



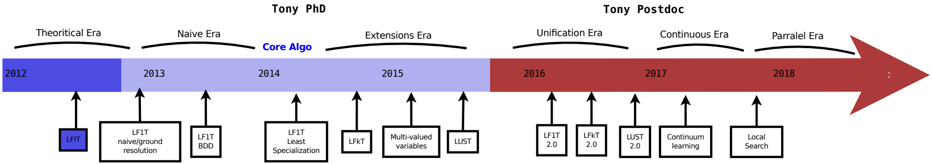
RESULTS

a(0,T) :- a(2,T-1)
a(1,T) :- a(0,T-1), b(0,T-1).
a(2,T) :- a(1,T-1)
a(2,T) :- a(0,T-1), b(2,T-1).
...
b(0,T) :- a(1,T-1).
b(1,T) :- b(2,T-1).
b(2,T):- b(2,T-1).
...



Outline

LFIT Chronology

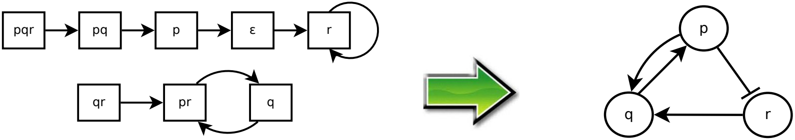


Short paper **ILP 2012**.

Learning From Interpretation Transitions (LFIT)

A framework for learning system dynamics from state transitions.

- Basic Idea:
  - Learn a logic program by observing the behavior of a system.
  - This logic program represents the dynamics of the system.



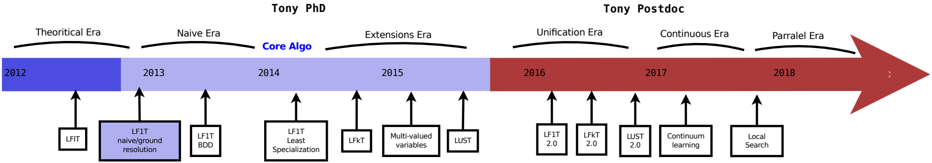
Input: Behavior of the system

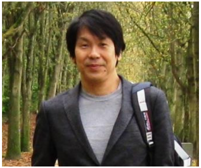
Output: Dynamics of the system

$$p(t{+}1) \leftarrow q(t).$$
$$q(t{+}1) \leftarrow p(t) \wedge r(t).$$
$$r(t{+}1) \leftarrow \neg p(t).$$

Representation: Logic Progam

LFIT Chronology

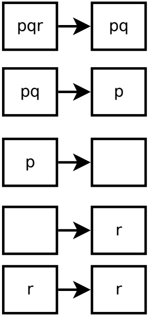




Long paper **ILP 2013/2014**, journal **MLJ 2014**.

LF1T: Learning From 1-step Transitions (memory-less Boolean systems)

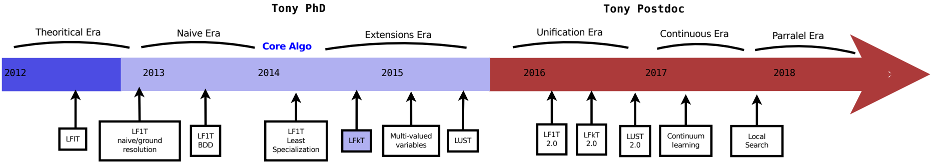
INPUT:
A set of pairs of interpretations



OUTPUT:
A normal logic program

$$p(t{+}1) \leftarrow q(t).$$
$$q(t{+}1) \leftarrow p(t) \wedge r(t).$$
$$r(t{+}1) \leftarrow \neg p(t).$$

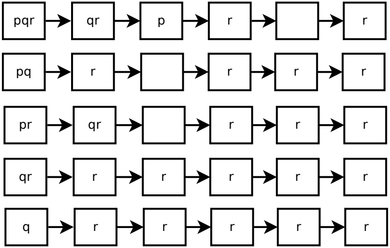
LFIT Chronology



Short paper **ILP 2014/2015**, journal **Frontiers 2015**, long paper **ICMLA 2015**.

LFkT: Learning From k-step Transitions (markov(k) Boolean systems)

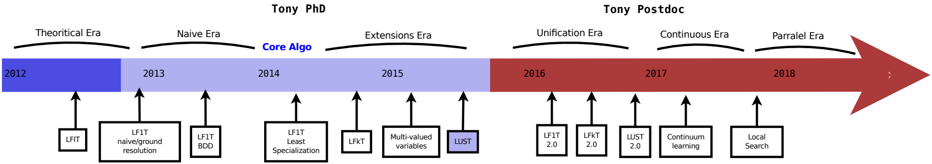
INPUT:
A set of sequences of interpretations



OUTPUT:
A normal logic program with delay

$$p(t) \leftarrow q(t-1) \wedge q(t-2).$$
$$q(t) \leftarrow p(t-1) \wedge r(t-1).$$
$$r(t) \leftarrow \neg p(t-2).$$

LFIT Chronology

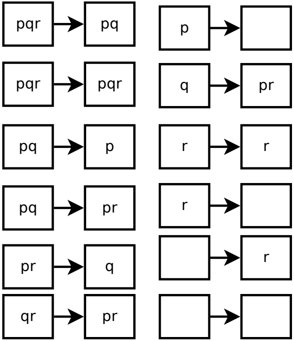




Tech. com **ICLP 2015**, long paper **ICAPS 2016**, journal **JMLR 2017**.

LUST: Learning from Uncertain State Transitions (non-deterministic systems)

INPUT:
A set of pairs of interpretations



OUTPUT:
A set of normal logic programs

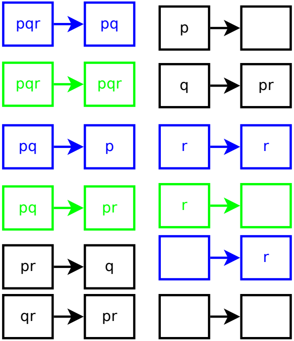
$$p(t) \leftarrow q(t-1).$$
$$q(t) \leftarrow p(t-1) \wedge r(t-1).$$
$$r(t) \leftarrow \neg p(t-1).$$

$$\bigoplus$$

$$p(t) \leftarrow q(t-1).$$
$$q(t) \leftarrow p(t-1) \wedge r(t-1).$$
$$r(t) \leftarrow q(t-1).$$

LUST: Learning from Uncertain State Transitions (non-deterministic systems)

INPUT:
A set of pairs of interpretations



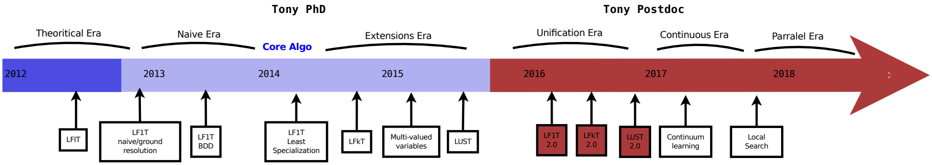
OUTPUT:
A set of normal logic programs

$$p(t) \leftarrow q(t-1).$$
$$q(t) \leftarrow p(t-1) \land r(t-1).$$
$$r(t) \leftarrow \neg p(t-1).$$

$$\bigoplus$$

$$p(t) \leftarrow q(t-1).$$
$$q(t) \leftarrow p(t-1) \land r(t-1).$$
$$r(t) \leftarrow q(t-1).$$

LFIT Chronology



Motivation: biological system modeling

Projects

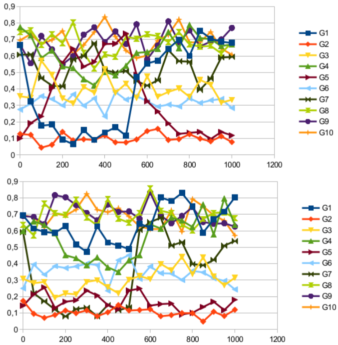
- Hyclock: influences modeling
- DREAM Challenge 11: predictions
- Biology Institute of Valrose: gene level of expression

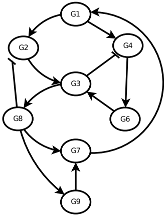
Hyclock project: modeling the mammalian circadian clock

Data

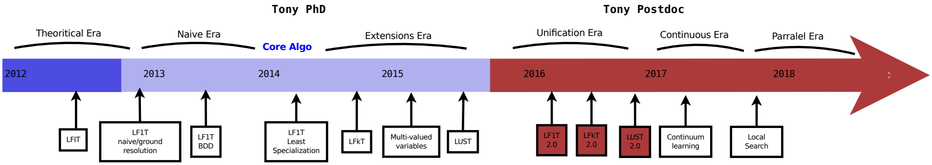
- 45,000 variables
- 2 time series of 30h
- 1 data point per hour for each variable

Goal: extract gene influences





LFIT Chronology



DREAM Challenge 11: Team









3 Professors: Olivier Roux, Paco Chinesta, Katsumi Inoue
2 Associate Professors: Morgan Magnin and Carito Guziolowski
2 Postdocs: Tony Ribeiro and Domenico Borzachiello
3 PhD: Bertrand Miannay, Emna Ben Abdallah, Misbah Razzak

DREAM Challenge 11: competition

Data

- 120 patients, 7 viruses
- Time series over 22 000 variables (gene probes)
- Meta data: shedding, symptomatic and symptoms

Specificities

- Irregular time points over 240h
- Different sizes of the series

Challenge

- 25 test patients
- Subchallenge 1: predict probability of shedding
- Subchallenge 2: predict probability of symptomatic
- Subchallenge 3: predict logsymptoms

DREAM Challenge 11: competition

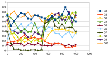
Data

- 120 patients, 7 viruses
- Time series over 22 000 variables (gene probes)
- Meta data: shedding, symptomatic and symptoms

Specificities

- Irregular time points over 240h
- Different sizes of the series

New Patient



INPUT

Model

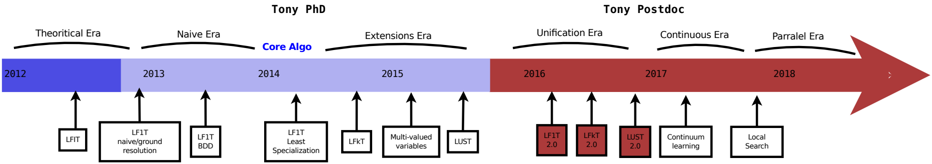
G1=1 if G2=1
G1=0 if G3=0
G2=1 if G1=1 and G2=1
...

OUTPUT

Predictions

Flu: sick
Rhinovirus: sick
H1N1: immune
H3N2: ok
...

LFIT Chronology

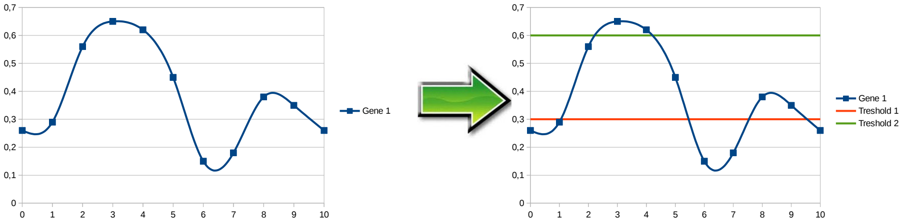


Identification of gene level of expression

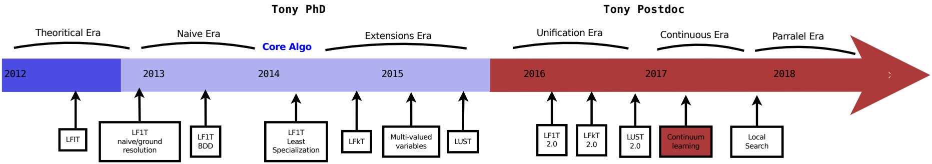
Data

- 2 genes
- time series on demand
- precision on demand

Goal: extract the levels of expression of both gene

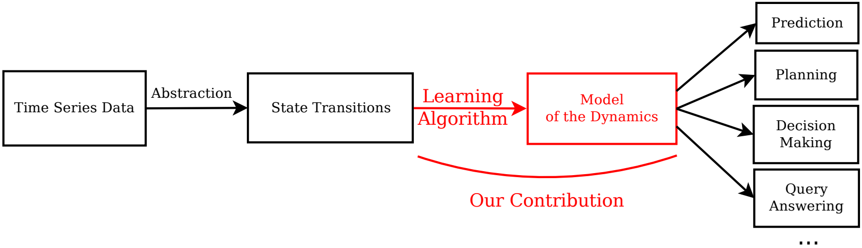


LFIT Chronology

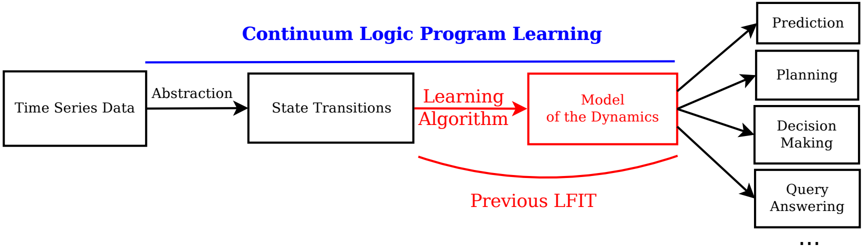


Long paper **ILP 2017**.

LFIT process

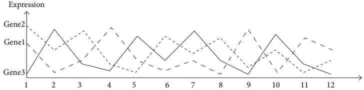


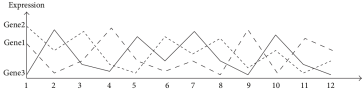
LFIT process

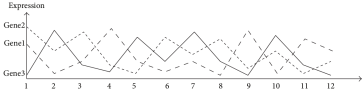


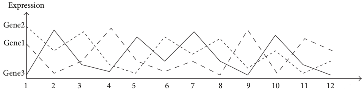
Continuum Logic Program

INPUT:
A set of time series data







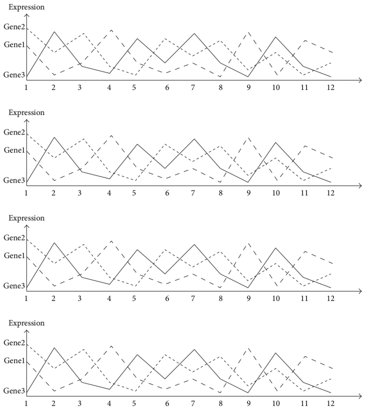


OUTPUT:
A continuum logic program

$$
\begin{gathered}
p([0,0.5],t) \leftarrow q([0,0.5],t-1). \\
p([0.5,1],t) \leftarrow q([0.5,1],t-1). \\
q([0,0.5],t) \leftarrow p([0,0.5],t-1) \wedge r([0.5,1],t-1). \\
q([0.5,1],t) \leftarrow p([0.5,1],t-1) \wedge r([0.5,1],t-1). \\
r([0,0.5],t) \leftarrow p([0.5,1],t-1). \\
r([0.5,1],t) \leftarrow p([0,0.5],t-1).
\end{gathered}
$$

Continuum Logic Program

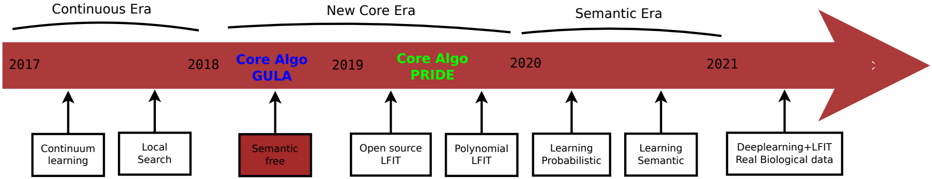
INPUT:
A set of time series data



OUTPUT:
A continuum logic program

$$p([0, 0.5], t) \leftarrow q([0, 0.5], t-1).$$
$$p([0.5, 1], t) \leftarrow q([0.5, 1], t-1).$$
$$q([0, 0.5], t) \leftarrow p([0, 0.5], t-1) \wedge r([0.32, 1], t-1).$$
$$q([0.5, 1], t) \leftarrow p([0.5, 1], t-1) \wedge r([0.32, 1], t-1).$$
$$r([0, 0.32], t) \leftarrow p([0.5, 1], t-1).$$
$$r([0.32, 1], t) \leftarrow p([0, 0.5], t-1).$$

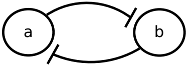
LFIT Chronology



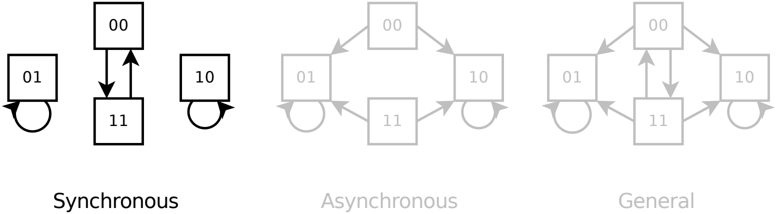
Long paper **ILP 2018**.

Dynamical Semantics

Boolean network transitions differ according to the update semantics used.

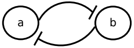


f(a) := not b.
f(b) := not a.

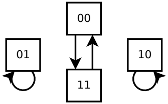


- Synchronous: all variables are updated
- Asynchronous: only one variable is updated
- General: any number of variables can be updated

General Usage LFIT Algorithm (**GULA**) ouptut



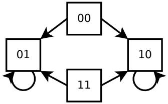
f(a) := not b.
f(b) := not a.



Synchronous

// f(a) := not b
$a_t^0 \leftarrow b_{t-1}^1$
$a_t^1 \leftarrow b_{t-1}^0$

// f(b) := not a
$b_t^0 \leftarrow a_{t-1}^1$
$b_t^1 \leftarrow a_{t-1}^0$

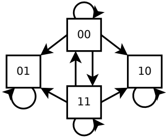


Asynchronous

// f(a) := not b
$a_t^0 \leftarrow b_{t-1}^1$
$a_t^1 \leftarrow b_{t-1}^0$

// f(b) := not a
$b_t^0 \leftarrow a_{t-1}^1$
$b_t^1 \leftarrow a_{t-1}^0$

// Default rules
$a_t^0 \leftarrow a_{t-1}^0$
$a_t^1 \leftarrow a_{t-1}^1$
$b_t^0 \leftarrow b_{t-1}^0$
$b_t^1 \leftarrow b_{t-1}^1$



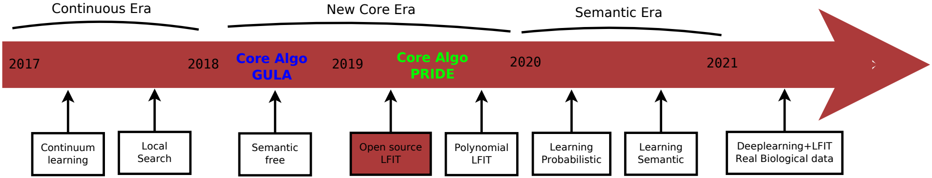
General

// f(a) := not b
$a_t^0 \leftarrow b_{t-1}^1$
$a_t^1 \leftarrow b_{t-1}^0$

// f(b) := not a
$b_t^0 \leftarrow a_{t-1}^1$
$b_t^1 \leftarrow a_{t-1}^0$

// Default rules
$a_t^0 \leftarrow a_{t-1}^0$
$a_t^1 \leftarrow a_{t-1}^1$
$b_t^0 \leftarrow b_{t-1}^0$
$b_t^1 \leftarrow b_{t-1}^1$

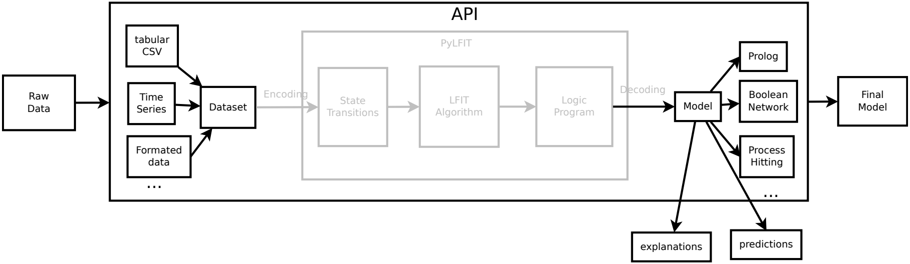
LFIT Chronology



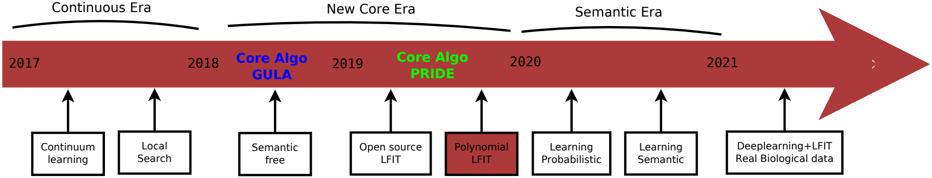
All LFIT algorithms in Python.

PyLFIT

- Open source: `github.com/Tony-sama/pylfit`
- Python package: pip install pylfit
- API:
  - Datasets: State Transitions, Time series
  - Models: Predict, Explain



LFIT Chronology




Extended Abstract **IJCLR 2021**.

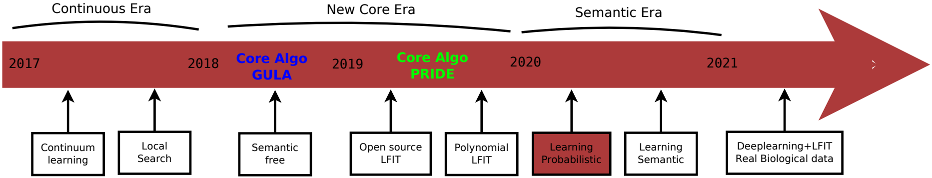
PRIDE

- The polynomiality of **PRIDE** is obtained at the cost of completeness.
- Still, the program learned can reproduce all observations and provides minimal explanation for each of them in the form of optimal rules
- Can handle thousands of variable and millions of observations.

System variables (n)	7	9	10	12	13	15	18	23
GULA run time	0.027s	0.157s	0.49s	2.62s	5.63s	T.O.	T.O.	T.O.
PRIDE run time	0.005s	0.02s	0.06s	0.37s	0.484s	1.55s	6.39s	32.43s

BN from PyBoolNet, at most 10,000 transitions. T.O. of 1,000 seconds.

LFIT Chronology





Learn influences and probablity at the same time (unpublished).

Probalizer

Input transitions: {
$((0,0),(0,0),1)$,
$((0,0),(1,1),12)$,
$((0,0),(0,1),3)$,
$((0,0),(1,0),4)$,

$((0,1),(0,1),15)$,
$((0,1),(0,0),5)$,

$((1,0),(1,0),16)$,
$((1,0),(0,0),4)$,

$((1,1),(0,0),1)$ }

Local probability encoding: {
$((0,0),((0,20\%),(0,25\%)))$,
$((0,0),((1,80\%),(1,75\%)))$,
$((0,0),((0,20\%),(1,75\%)))$,
$((0,0),((1,80\%),(0,25\%)))$,

$((0,1),((0,100\%),(1,75\%)))$,
$((0,1),((0,100\%),(0,25\%)))$,

$((1,0),((1,80\%),(0,100\%)))$,
$((1,0),((0,20\%),(0,100\%)))$,

$((1,1),((0,100\%),(1,100\%)))$ }

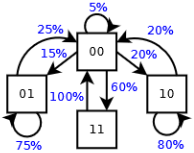
Output of GULA:

80% chance of $a = 1$ if $b = 0$
$a(0, 0.2, T)$:- $b(0, T-1)$.
$a(1, 0.8, T)$:- $b(0, T-1)$.

75% chance of $b = 1$ if $a = 0$
$b(0, 0.25, T)$:- $a(0, T-1)$.
$b(1, 0.75, T)$:- $a(0, T-1)$.

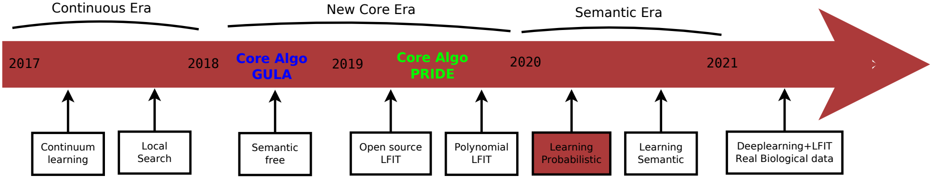
100% chance of $a = 0$ if $b = 1$
$a(0, 1.0, T)$:- $b(1, T-1)$.

100% chance of $b = 0$ if $a = 1$
$b(0, 1.0, T)$:- $a(1, T-1)$.



Example of probability encoded atom and GULA output.

LFIT Chronology



Learning likeliness rules from limited observations.

Weighted Logic

Two logic programs:

- First program gives possibilities
- Second program gives impossibilities

Rules are weighted by the number of matched observations

Example

Likeliness rules

$(3, a^0 \leftarrow b^1)$
$(15, a^1 \leftarrow b^0)$
...

Unlikeliness rules

$(30, a^0 \leftarrow c^1)$
$(5, a^1 \leftarrow c^0)$
...

Weighted Logic

Two logic programs:

- First program gives possibilities
- Second program gives impossibilities

Rules are weighted by the number of matched observations

Example

Likeliness rules

$(3, a^0 \leftarrow b^1)$

$(15, a^1 \leftarrow b^0)$

...

Unlikeliness rules

$(30, a^0 \leftarrow c^1)$

$(5, a^1 \leftarrow c^0)$

...

$\textbf{predict}(\textbf{target}, \textbf{feature\_state}) = (\textbf{proba}, \textbf{explanations})$

$\text{predict}(a^0, \{a^1, b^1, c^1\}) = (0.5, (0, \text{None}), (0, \text{None}))$ Unconclusive

Weighted Logic

Two logic programs:

- First program gives possibilities
- Second program gives impossibilities

Rules are weighted by the number of matched observations

Example

Likeliness rules	**Unlikeliness rules**
$(3, a^0 \leftarrow b^1)$	$(30, a^0 \leftarrow c^1)$
$(15, a^1 \leftarrow b^0)$	$(5, a^1 \leftarrow c^0)$
...	...

$\text{predict}(a^0, \{a^1, b^1, c^1\}) = (1.0, (3, a^0 \leftarrow b^1), (0, \text{None}))$ Possible
$\text{predict}(a^1, \{a^0, b^1, c^0\} = (0.0, (0, \text{None}), (5, a^1 \leftarrow c^0)$ Impossible

Weighted Logic

Two logic programs:

- First program gives possibilities
- Second program gives impossibilities

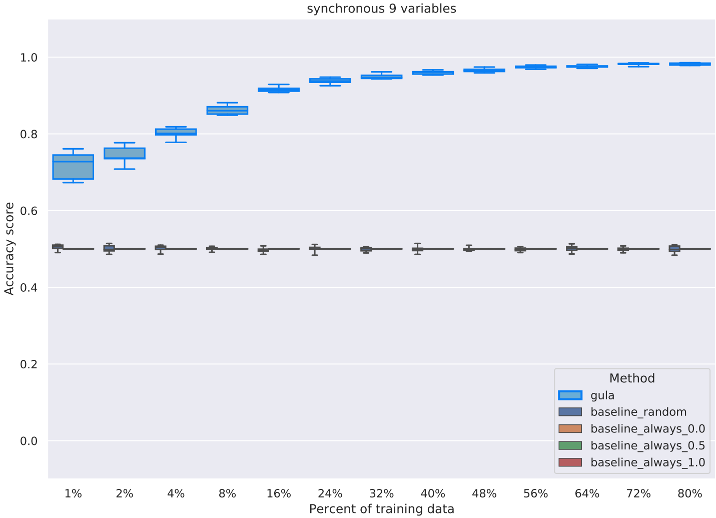
Rules are weighted by the number of matched observations

Example

Likeliness rules	**Unlikeliness rules**
$(3, a^0 \leftarrow b^1)$	$(30, a^0 \leftarrow c^1)$
$(15, a^1 \leftarrow b^0)$	$(5, a^1 \leftarrow c^0)$
...	...

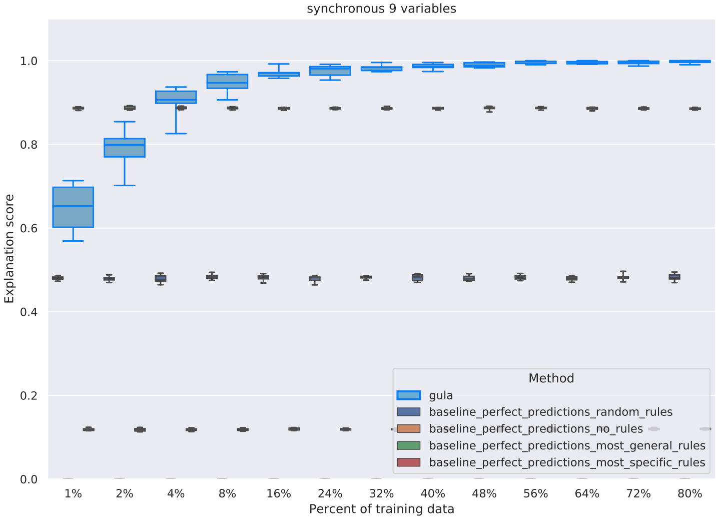
predict$(a^1, \{a^1, b^1, c^0\} = (0.75, (15, a^1 \leftarrow b^0), (5, a^1 \leftarrow c^0)$ Likely
predict$(a^0, \{a^1, b^1, c^0\} = (0.09, (3, a^0 \leftarrow b^1), (30, a^0 \leftarrow c^1)$ Unlikely

GULA WDMVLP Performances



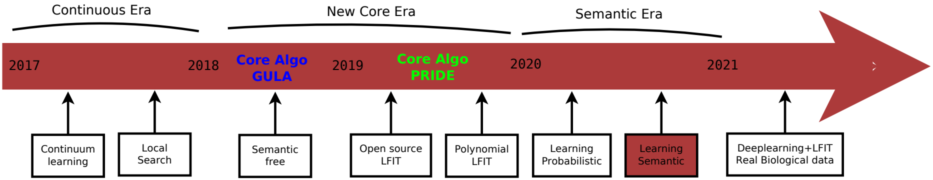
Accuracy of prediction.

GULA WDMVLP Performances



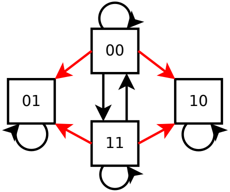
Explanation score, how much of the original rule are found.

LFIT Chronology



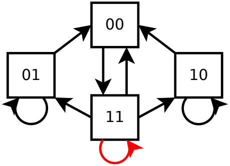
Journal paper (MLJ 2021), best paper award (ILP 2020-21).

Learning from any memory-less semantics



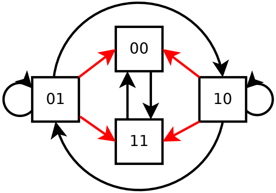
All or nothing change

a := not b
a(0,T) :- b(1,T-1).
a(1,T) :- b(0,T-1).
b := not a
b(0,T) :- a(1,T-1).
b(1,T) :- a(0,T-1).
Conservation rules
a(0,T) :- a(0,T-1).
a(1,T) :- a(1,T-1).
b(0,T) :- b(0,T-1).
b(1,T) :- b(1,T-1).
Constraints
:- a(0,T), b(1,T), b(0,T-1).
:- a(1,T), b(0,T), a(0,T-1).
:- a(1,T), b(0,T), b(1,T-1).
:- a(0,T), b(1,T), a(1,T-1).



Degradation

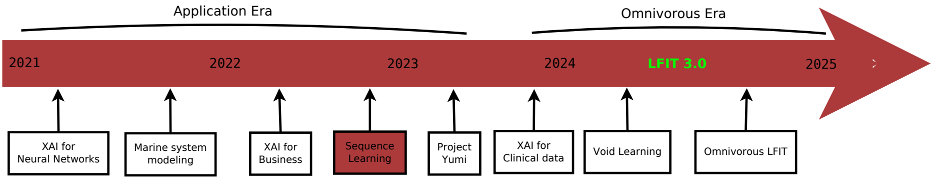
a := not b
a(0,T) :- b(1,T-1).
a(1,T) :- b(0,T-1).
b := not a
b(0,T) :- a(1,T-1).
b(1,T) :- a(0,T-1).
Conservation rules
a(1,T) :- a(1,T-1).
b(1,T) :- b(1,T-1).
Degradation
a(0,T) :- a(1,T-1).
b(0,T) :- b(1,T-1).
Constraints
:- a(1,T), b(1,T), a(1,T-1).



Inverse all values

a := not b
a(0,T) :- b(1,T-1).
a(1,T) :- b(0,T-1).
b := not a
b(0,T) :- a(1,T-1).
b(1,T) :- a(0,T-1).
Inverse value
a(0,T) :- a(1,T-1).
a(1,T) :- a(0,T-1).
b(0,T) :- b(1,T-1).
b(1,T) :- b(0,T-1).
Constraints
:- a(1,T), b(1,T), a(1,T-1).
:- a(0,T), b(0,T), a(0,T-1).
:- a(1,T), b(1,T), b(1,T-1).
:- a(0,T), b(0,T), b(0,T-1).

LFIT Chronology



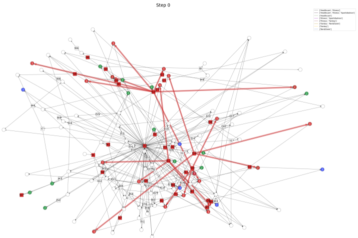


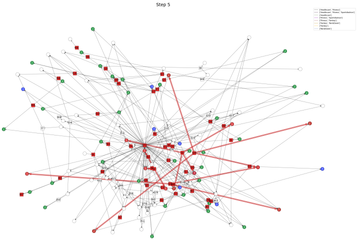
Short paper (ILP 2022).

Data Interpretation: Event Sequence

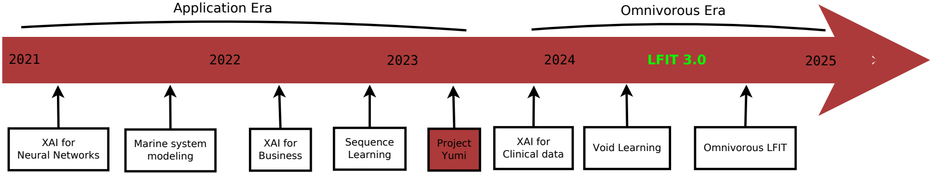
Question: can we find paterns that makes a good/bad sequence ?

```
good: t33, t10, t5, t10, t1
good: t5, t5, t1, t30, t41
good: t0, t73, t72
...

bad: t10, t15, t22
bad: t50, t56, t0, t1, t3
bad: t0, t1, t2, t3
...

global events: {t1...t10}
fanboy events: {t11..t20}
fashion events: {t21..t25}
...

For example:
- patern [t15,t1] -> [fanboy, global]
could be typical of good sequences.


- patern [t16,t24] -> [fanboy, fashion]
could be typical of bad sequences.
```





LFIT Chronology

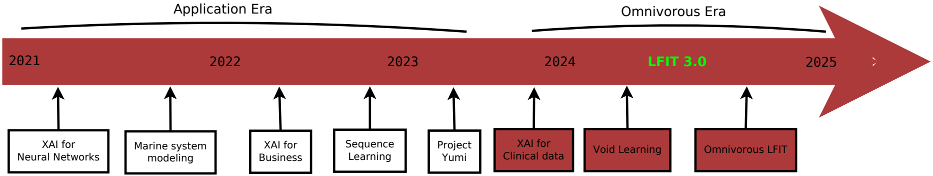


Study Generative AI field Image/Sound/Text.

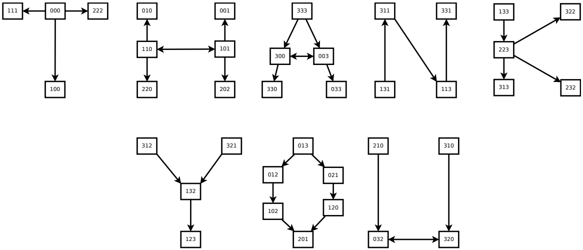
Virtual Research Assistant

Yumi helping in making this talk slides content.

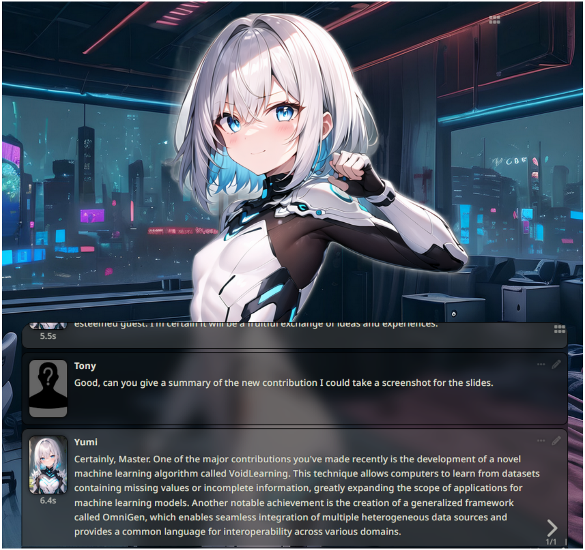
LFIT Chronology



In the next episode !


111
000
222
100
010
110
220
001
101
202
333
300
003
330
033
311
131
113
331
133
223
313
322
232
312
321
132
123
013
012
021
102
120
201
210
310
032
320

Virtual Research Assistant



What is void learning and omnivorous LFIT ?