Learning Multi-Valued Biological Models with Delayed Influence from Time-Series Observations

Tony Ribeiro
Morgan Magnin, Katsumi Inoue, Chiaki Sakama

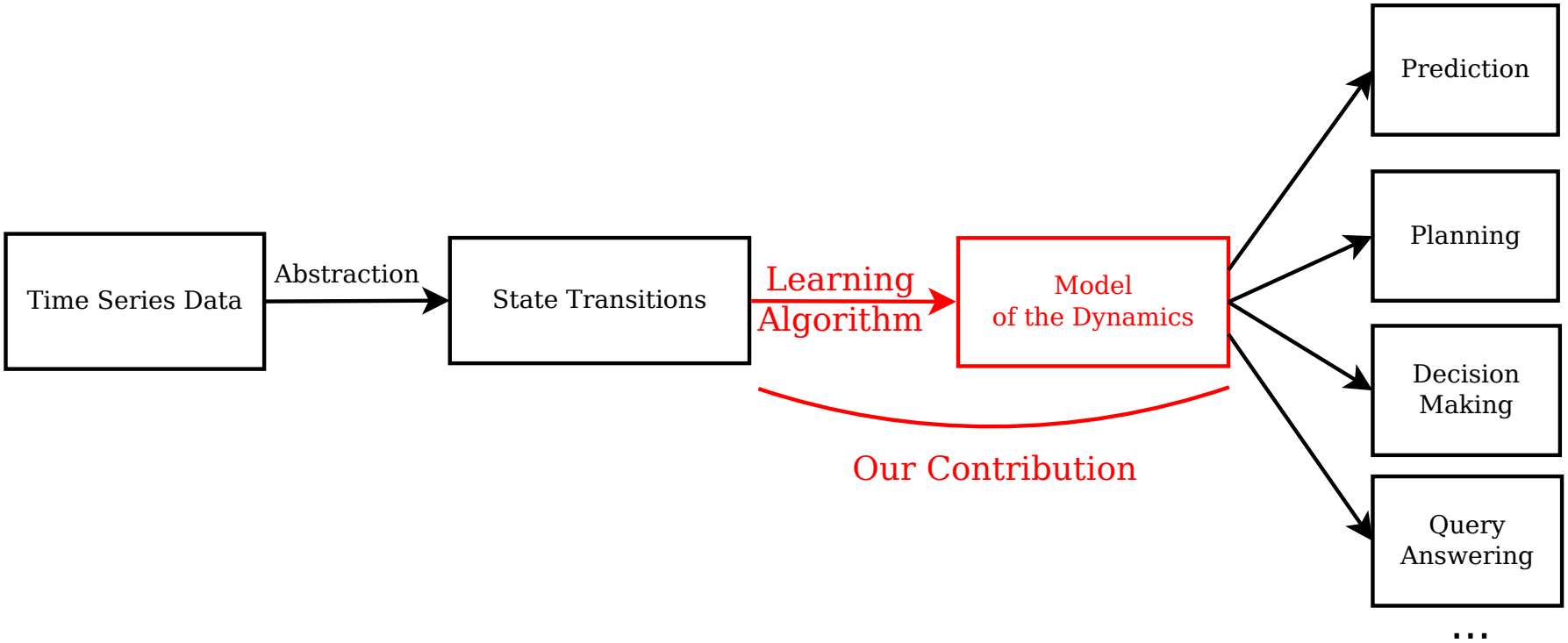
Department of Informatics
The Graduate University for Advanced Studies, Japan

National Institute of Informatics, Japan

Wakayama University, Japan

ILP, August 20th, 2015

Motivations



Problem:

- Time series data alone may be not sufficient, we need models.

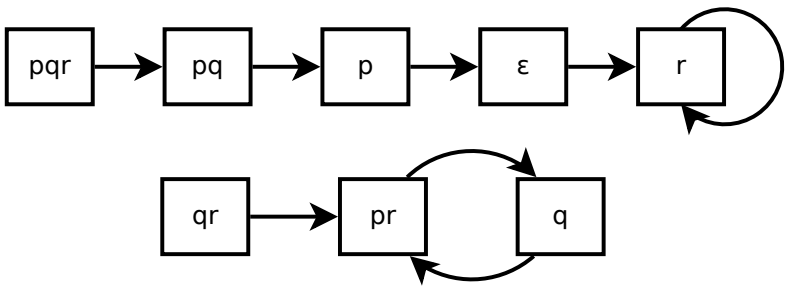
Goal:

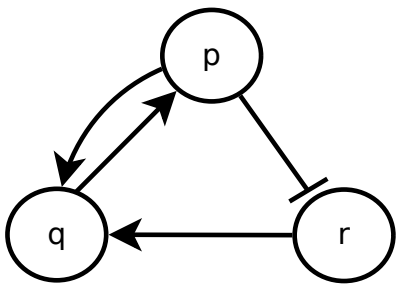
- Automated modeling of systems dynamics from these data.

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

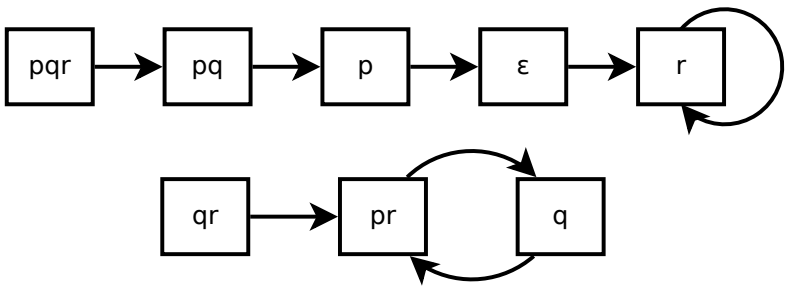
$$p(t{+}1) \leftarrow q(t).$$
$$q(t{+}1) \leftarrow p(t) \wedge r(t).$$
$$r(t{+}1) \leftarrow \neg p(t).$$

Representation: Logic Progam

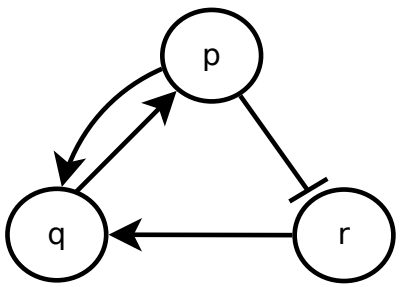
Learning From Interpretation Transitions (LFIT)

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.



What happens



Why it happens

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

Representation: Logic Progam

Learning From k-Step Transitions (LFkT)

Motivation

- Learn Systems with delayed influences: Markov(k) systems.
- Extract causality from observations, why things happen.
- Predict consequences of new observations, what will happen.

What is new?

- **Automatic** computation of delays
- **Multi-valued** variables
- **Real** benchmark evaluation

Outline

Logic Program

We consider a *logic program* as a set of *rules* of the form

$$p \leftarrow p_1 \wedge \cdots \wedge p_m \wedge \neg p_{m+1} \wedge \ldots \wedge \neg p_n \quad (1)$$

where p and p_i's are atoms ($n \geq m \geq 1$).

Definition (Herbrand Base)

The Herbrand Base of a program P, denoted by $\mathcal{B}$, is the set of all atoms in the language of P.

Example

- $R_1 = a \leftarrow b \wedge c$
- $R_2 = b \leftarrow a \wedge c$
- $P = \{R_1, R_2\}$ is a logic program
- The Herbrand Base of P is $\mathcal{B} = \{a, b, c\}$

Formalization: Markov(k) into Logic Program

Definition (Timed Herbrand base)

Let $\mathcal{B}$ be the Herbrand base of a program P and k be a natural number. The timed Herbrand base of P (with period k) denoted by $\mathcal{B}_k$, is as follows:

$$\mathcal{B}_k = \bigcup_{i=1}^{k} \{v_{t-i} | v \in \mathcal{B}\}$$

Where t is a constant term which represents the current time step.

Example

If the Herbrand base of a program P is $\mathcal{B} = \{a, b, c\}$ then

- $\mathcal{B}_1 = \{a_{t-1}, b_{t-1}, c_{t-1}\}$
- $\mathcal{B}_2 = \{a_{t-1}, b_{t-1}, c_{t-1}, a_{t-2}, b_{t-2}, c_{t-2}\}$

Markov(k) Systems

A Markov(k) system can be interpreted as a logic program.

Definition (Markov(k) system)

Let P be a logic program, $\mathcal{B}$ be the Herbrand base of P and $\mathcal{B}_k$ be the timed Herbrand base of P with period k. A Markov(k) system S with respect to P is a logic program where for all rules $R \in S$, $h(R) \in \mathcal{B}$ and all atoms appearing in b(R) belong to $\mathcal{B}_k$.

Example

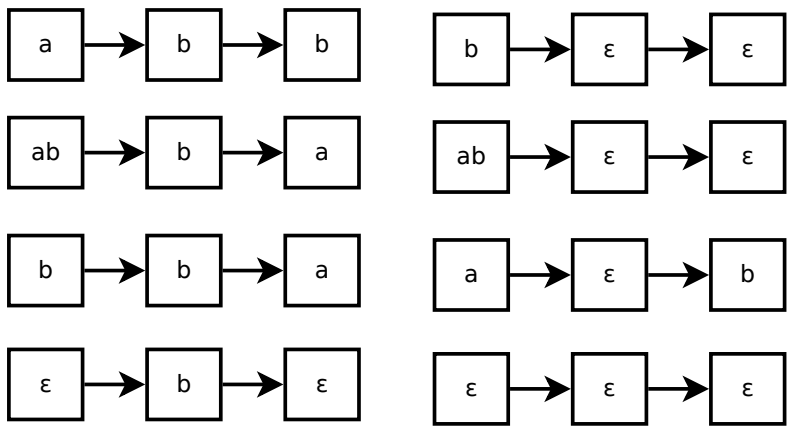
If the Herbrand base of a program P is $\mathcal{B} = \{a, b\}$ then

- $\mathcal{B}_2 = \{a_{t-1}, b_{t-1}, a_{t-2}, b_{t-2}\}$.
- Let $R_1 = a \leftarrow b_{t-1}, b_{t-2}$ and $R_2 = b \leftarrow a_{t-2}, \neg b_{t-2}$.
- $S = \{R_1, R_2\}$ is a Markov(2) system.

Example

Let S be a Markov(k) system as follows:

$$S = \{(a \leftarrow b_{t-1}, b_{t-2}), (b \leftarrow a_{t-2}, \neg b_{t-2})\}$$



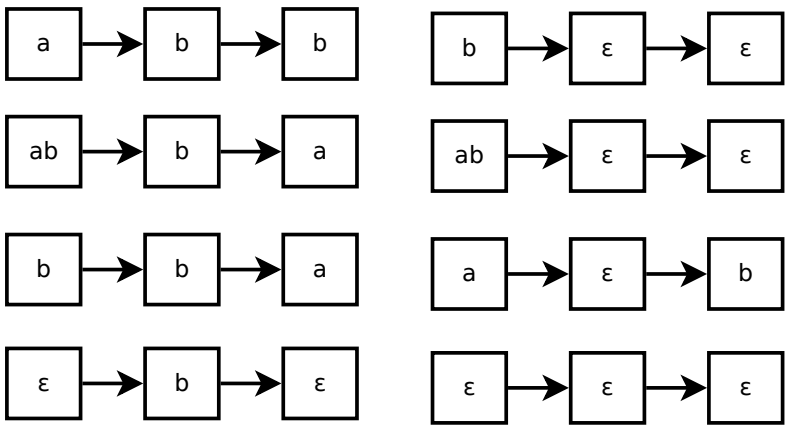
Eight traces of executions of the system S

Outline

Example: Detection of the Delay

Let S be a Markov(k) system as follows:

$$S = \{(a \leftarrow b_{t-1}, b_{t-2}), (b \leftarrow a_{t-2}, \neg b_{t-2})\}$$

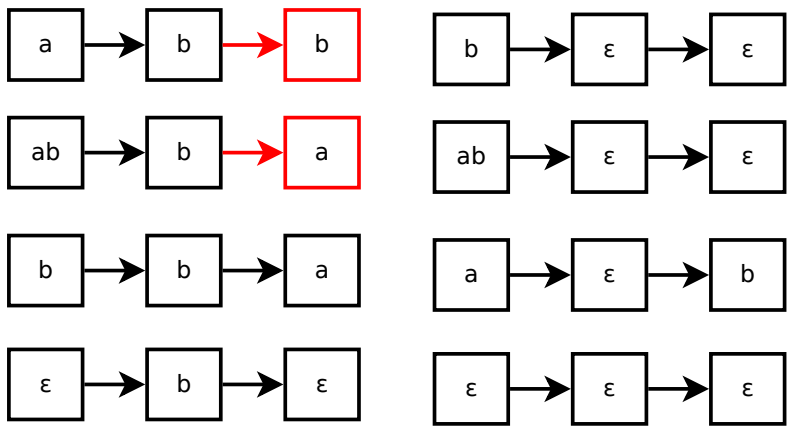


Eight traces of executions of the system S

Example: Detection of the Delay

Let S be a Markov(k) system as follows:

$$S = \{(a \leftarrow b_{t-1}, b_{t-2}), (b \leftarrow a_{t-2}, \neg b_{t-2})\}$$

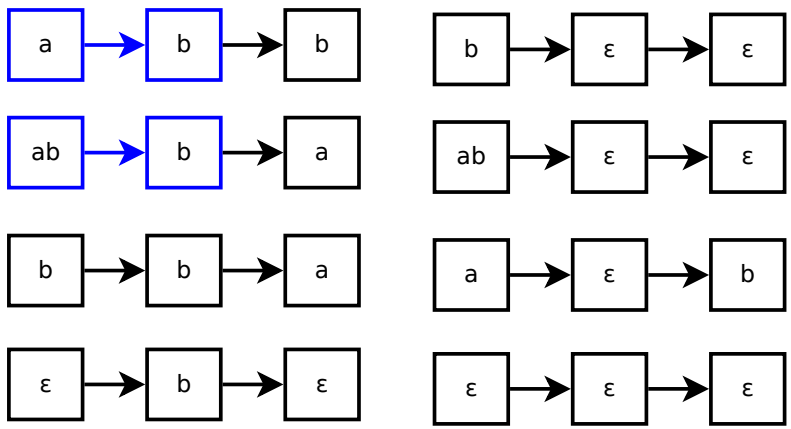


Eight traces of executions of the system S

Example: Detection of the Delay

Let S be a Markov(k) system as follows:

$$S = \{(a \leftarrow b_{t-1}, b_{t-2}), (b \leftarrow a_{t-2}, \neg b_{t-2})\}$$



Eight traces of executions of the system S

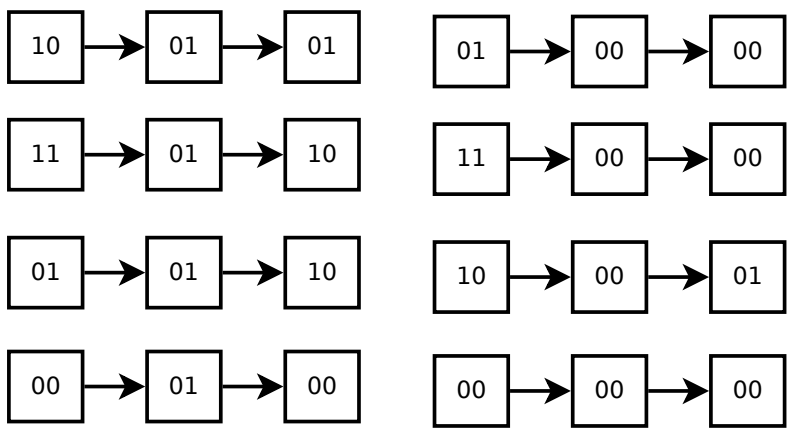
Multivalued Variables

In order to represent multi-valued variables, we now restrict all atoms of a logic program to the form var^{val}. We consider a *multi-valued logic program* as a set of *rules* of the form

$$var^{val} \leftarrow var_1^{val_1} \wedge \cdots \wedge var_n^{val_n} \tag{2}$$

where var^{val} and $var_i^{val_i}$ are atoms ($n \geq 0$).

$$S = \{(a^1 \leftarrow b^1_{t-1}, b^1_{t-2}), (b^1 \leftarrow a^1_{t-2}, b^0_{t-2})\}$$



Outline

DREAM4 input data

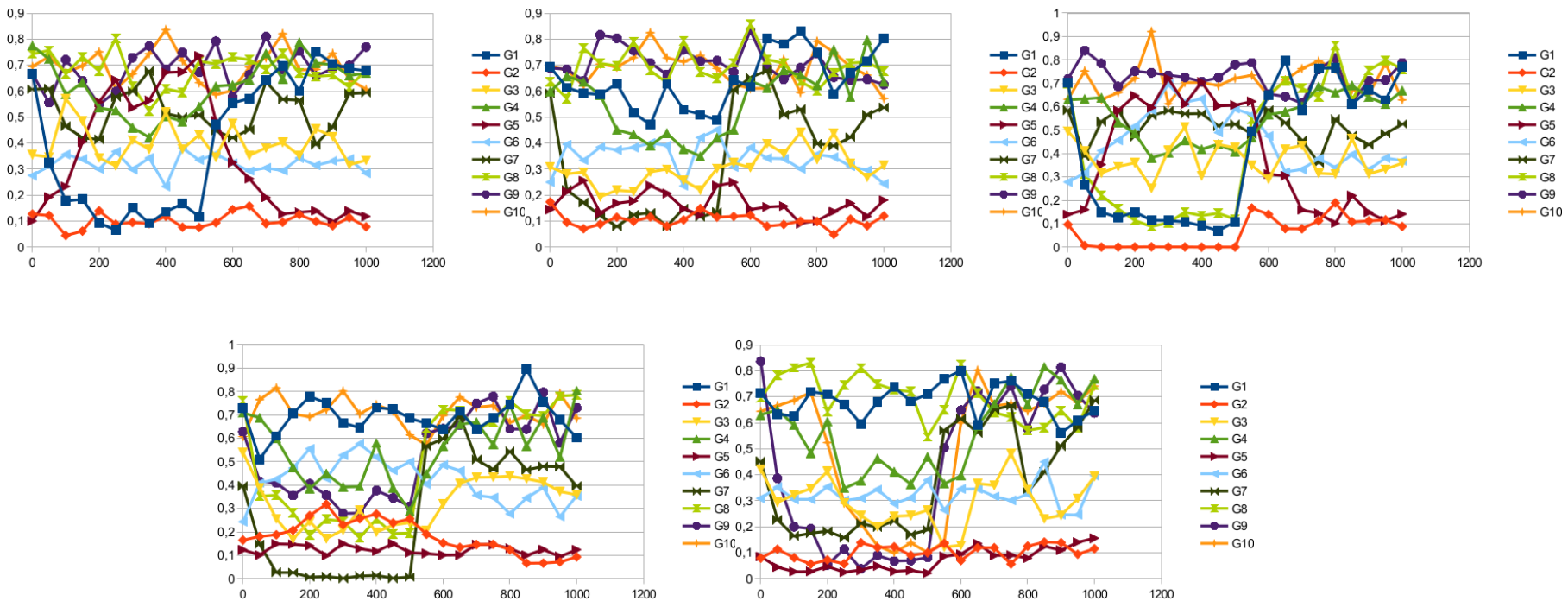
- Case studies:
  - 5 different systems each composed of 10 genes
  - 5 different systems composed of 100 genes
  - All come from E. coli and yeast networks
- Data sets available for each system of 10 genes (resp. 100):
  - 5 (resp. 10) time series data with 21 time points
  - Steady state at wild type, i.e. 1 steady state
  - Steady state after knocking out each gene, i.e. 10 steady states (resp. 100)
  - Steady state after knocking down each gene (transcription rate at 50%), i.e. 10 steady states (resp. 100)
  - Steady states after some random multifactorial perturbations, i.e. 10 steady states

DREAM4 input data

- Case studies:
  - 5 different systems each composed of 10 genes
  - 5 different systems composed of 100 genes
  - All come from E. coli and yeast networks
- Data sets available for each system of 10 genes (resp. 100):
  - 5 (resp. 10) time series data with 21 time points
  - Steady state at wild type, i.e. 1 steady state
  - Steady state after knocking out each gene, i.e. 10 steady states (resp. 100)
  - Steady state after knocking down each gene (transcription rate at 50%), i.e. 10 steady states (resp. 100)
  - Steady states after some random multifactorial perturbations, i.e. 10 steady states

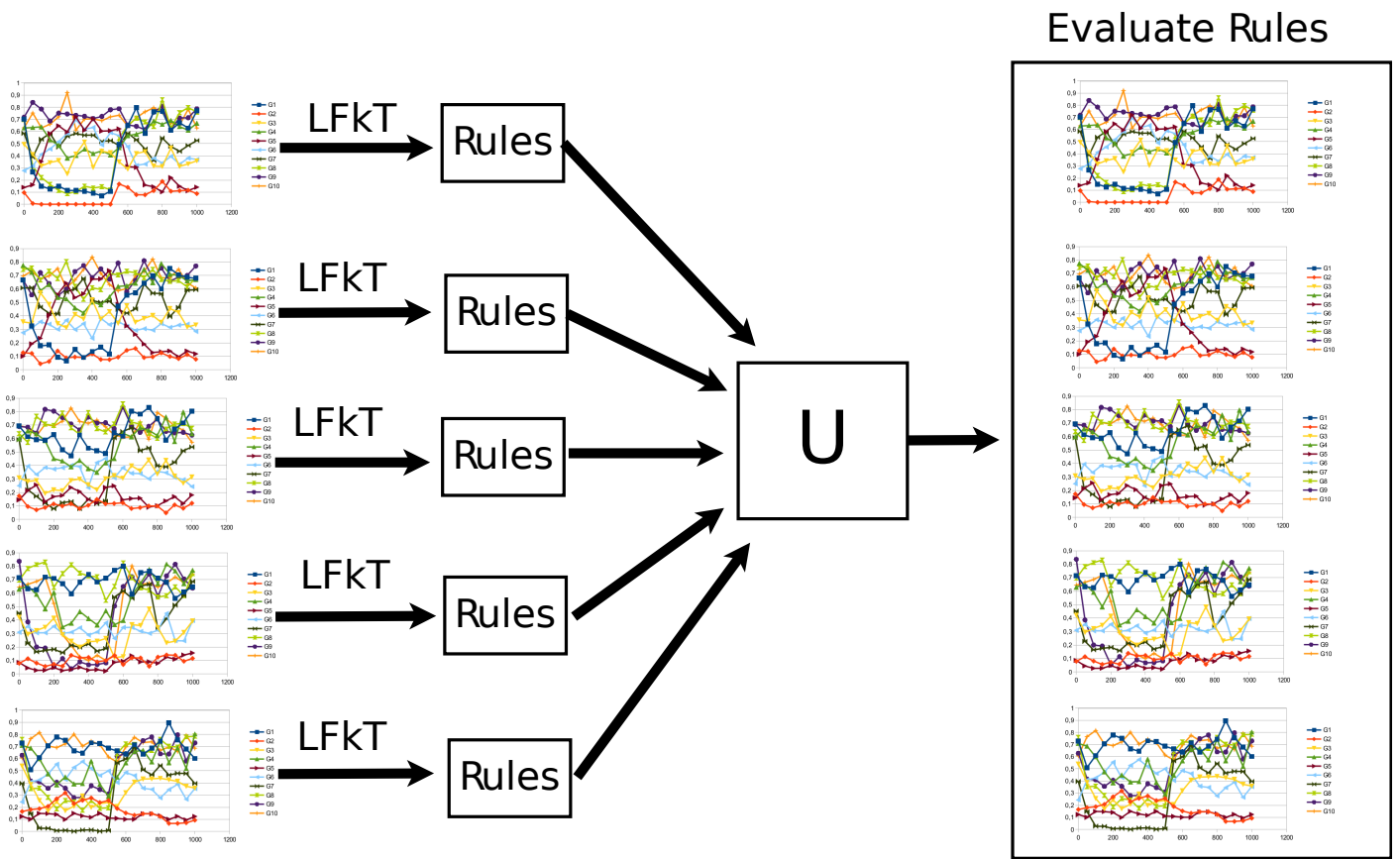
DREAM4 Challenge

- DATA: 5 different time series of genes expression.
- GOAL:
  - Predict the directed unsigned interaction graph
  - Predict steady states from dual knockouts



Learning phase

- INPUT: 5 series of 20 transitions with different perturbations.
- Approach:
  - Learn independently each series with LFkT.
  - Evaluate rules on all series (full cross-validation).



Prediction

Evaluation

- INPUT:
  - An initial state
  - 5 different conditions of dual genes to be knockout simultaneously
- GOAL:
  - Predict the point attractor

Benchmark	run time	raw output	final output	Mean squared error
insilico_size10_1	28s	118,834	359	0.073
insilico_size10_2	2m5s	401,923	462	0.064
insilico_size10_3	44s	151,021	480	0.019
insilico_size10_4	22s	90,904	387	0.031
insilico_size10_5	1m04s	297,364	326	0.091

Evaluation: precision is evaluated as the mean squared error of the difference between predicted/expected values.

Outline

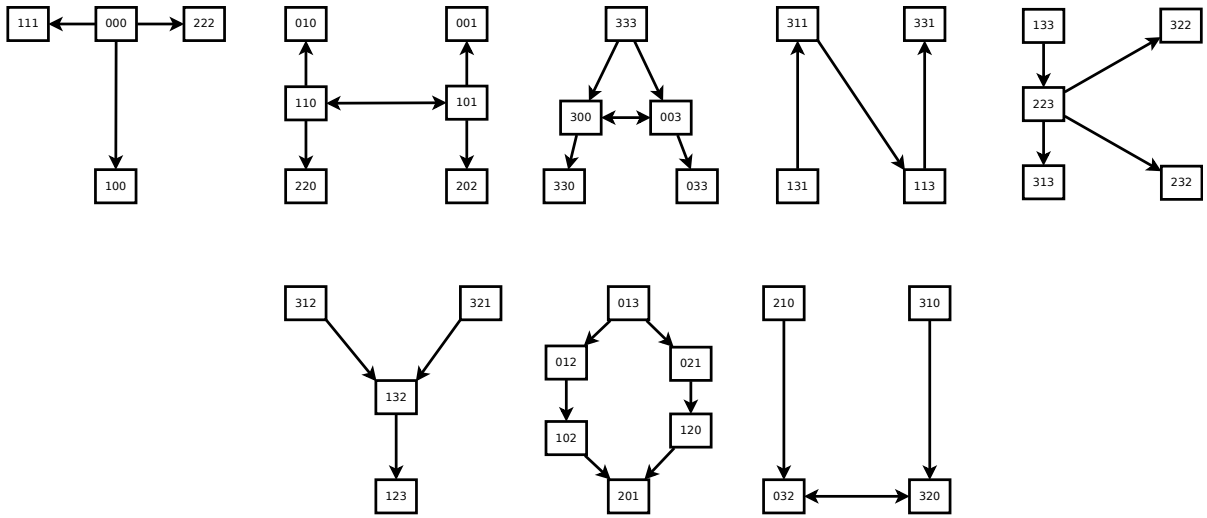
Conclusion

Contribution

- Extension of the LFkT algorithm
  - Detect delay dynamically
  - Handle multivalued variable
- Practical methods
  - Automatic discretization of time series (pre-processing)
  - Heuristic for model simplification (post-processing)

Current & Future works

- Improve performances to tackle big network (DREAM4 100)
- Design method to extract the real influences


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