Active learning of timed automata with unobservable resets

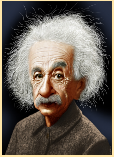
Léo Henry, Thierry Jéron and Nicolas Markey

Univ. Rennes, INRIA & CNRS-Rennes, France

Active learning

the minimally adequate teacher

learner



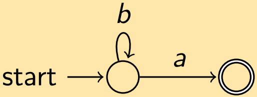
teacher

[1] Angluin, “Learning Regular Sets from Queries and Counterexamples”, 1987.

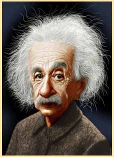
[2] Angluin, “Queries and Concept Learning”, 1987.

Active learning

the minimally adequate teacher



learner



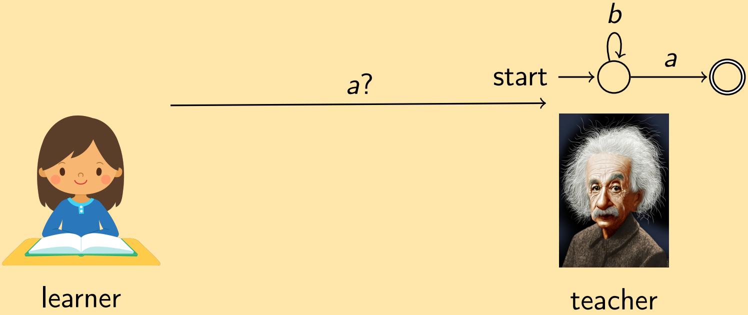
teacher

[1] Angluin, “Learning Regular Sets from Queries and Counterexamples”, 1987.

[2] Angluin, “Queries and Concept Learning”, 1987.

Active learning

the minimally adequate teacher

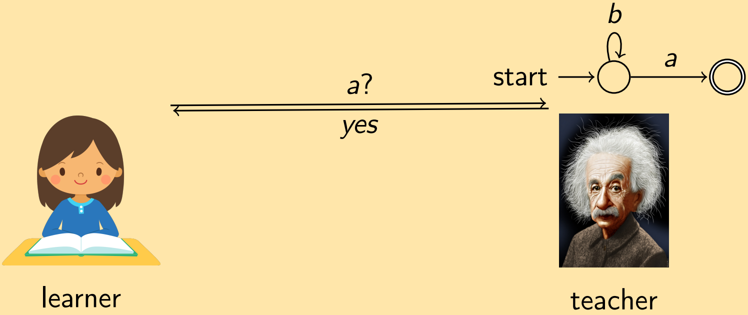


[1] Angluin, "Learning Regular Sets from Queries and Counterexamples", 1987.

[2] Angluin, "Queries and Concept Learning", 1987.

Active learning

the minimally adequate teacher

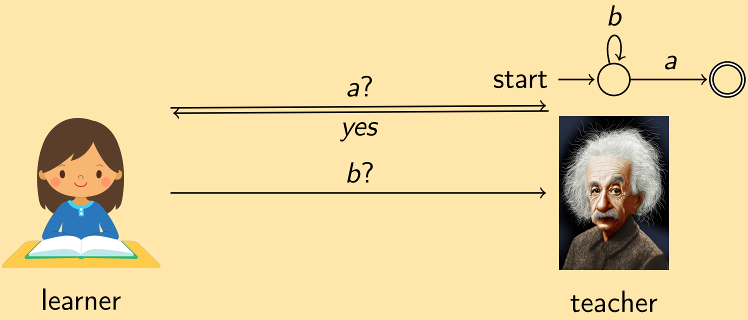


[1] Angluin, “Learning Regular Sets from Queries and Counterexamples”, 1987.

[2] Angluin, “Queries and Concept Learning”, 1987.

Active learning

the minimally adequate teacher

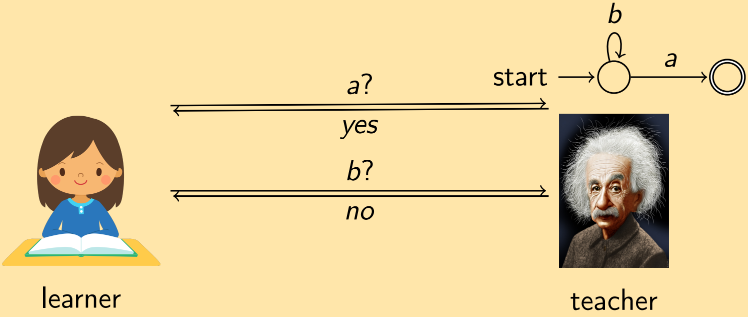


[1] Angluin, “Learning Regular Sets from Queries and Counterexamples”, 1987.

[2] Angluin, “Queries and Concept Learning”, 1987.

Active learning

the minimally adequate teacher

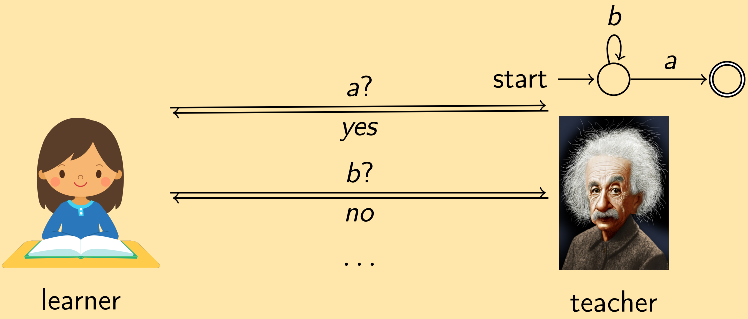


[1] Angluin, “Learning Regular Sets from Queries and Counterexamples”, 1987.

[2] Angluin, “Queries and Concept Learning”, 1987.

Active learning

the minimally adequate teacher



start

b

a

a?

yes

b?

no

. . .

learner

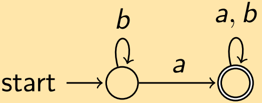
teacher

[1] Angluin, "Learning Regular Sets from Queries and Counterexamples", 1987.

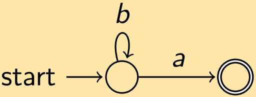
[2] Angluin, "Queries and Concept Learning", 1987.

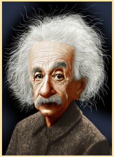
Active learning

the minimally adequate teacher



learner





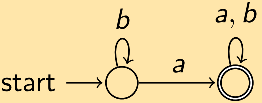
teacher

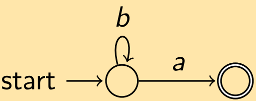
[1] Angluin, "Learning Regular Sets from Queries and Counterexamples", 1987.

[2] Angluin, "Queries and Concept Learning", 1987.

Active learning

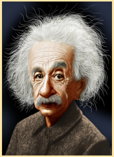
the minimally adequate teacher





my model?

no: $ab \notin \mathcal{L}$



learner

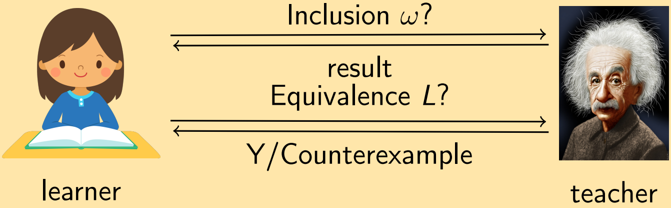
teacher

[1] Angluin, "Learning Regular Sets from Queries and Counterexamples", 1987.

[2] Angluin, "Queries and Concept Learning", 1987.

Active learning

the minimally adequate teacher



Inclusion ω?

result

Equivalence L?

Y/Counterexample

learner

teacher

[1] Angluin, "Learning Regular Sets from Queries and Counterexamples", 1987.

[2] Angluin, "Queries and Concept Learning", 1987.

Observation tables

	ϵ	a	ba
ϵ	1	0	0
a	0	1	0
b	0	1	0
aa	1	1	1
ab	0	0	1
ba	1	1	1
bb	1	0	0

Used in the L^* algorithm (D. Angluin)

Observation tables

		ϵ	a	ba
R	ϵ	1	0	0
	a	0	1	0
	b	0	1	0
$R.\Sigma$	aa	1	1	1
	ab	0	0	1
	ba	1	1	1
	bb	1	0	0

Used in the L^* algorithm (D. Angluin)

Observation tables

		S		
		ϵ	a	ba
R	ϵ	1	0	0
	a	0	1	0
	b	0	1	0
$R.\Sigma$	aa	1	1	1
	ab	0	0	1
	ba	1	1	1
	bb	1	0	0

Used in the L^* algorithm (D. Angluin)

Observation tables

S

		ϵ	a	ba
R	ϵ	1	0	0
	a	0	1	0
	b	0	1	0
$\boldsymbol{R.\Sigma}$	aa	1	1	1
	ab	0	0	1
	ba	1	1	1
	bb	1	0	0

Used in the L^* algorithm (D. Angluin)

Observation tables

		S		
		ϵ	a	ba
R	ϵ	1	0	0
	a	0	1	0
	b	0	1	0
$R.\Sigma$	aa	1	1	1
	ab	0	0	1
	ba	1	1	1
	bb	1	0	0

Used in the L^* algorithm (D. Angluin)

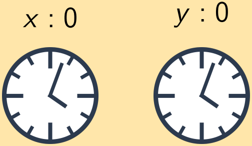
Observation tables

$$\begin{array}{cc|ccc} & & \multicolumn{3}{c}{\overbrace{\hphantom{\epsilon \quad a \quad ba}}^{S}} \\ & & \epsilon & a & ba \\ \hline R & \epsilon & 1 & 0 & 0 \\ & a & 0 & 1 & 0 \\ & b & 0 & 1 & 0 \\ \hline R.\Sigma & aa & 1 & 1 & 1 \\ & ab & 0 & 0 & 1 \\ & ba & 1 & 1 & 1 \\ & bb & 1 & 0 & 0 \end{array}$$

consistent $\forall u, v \in R, (u \sim_O v \Rightarrow \forall a \in \Sigma,\ u.a \sim_O v.a)$

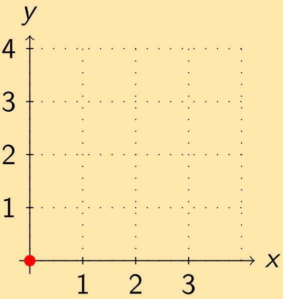
Used in the L^* algorithm (D. Angluin)

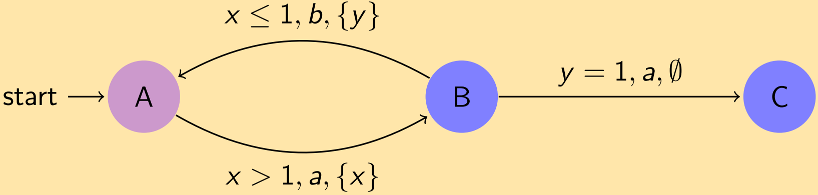
Timed automaton: a model for timed systems



$x : 0$

$y : 0$





start

A

B

C

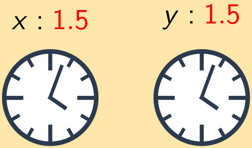
$x \leq 1, b, \{y\}$

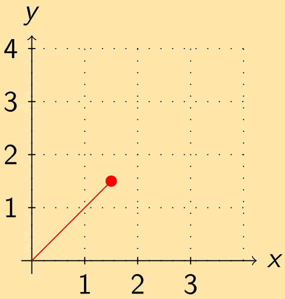
$x > 1, a, \{x\}$

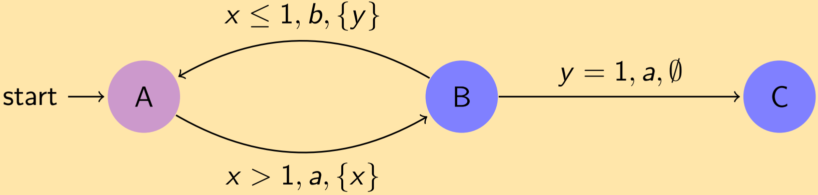
$y = 1, a, \emptyset$

[3] Alur and Dill, “A Theory of Timed Automata”, 1994.

Timed automaton: a model for timed systems



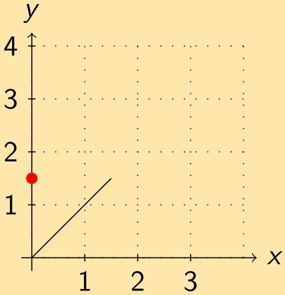


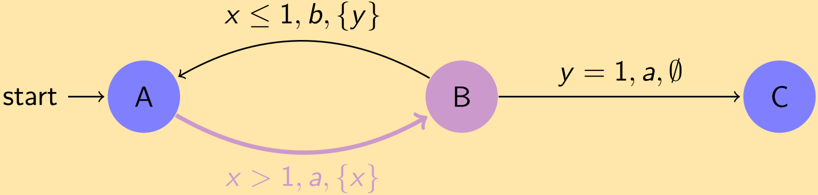


[3] Alur and Dill, "A Theory of Timed Automata", 1994.

Timed automaton: a model for timed systems

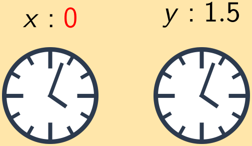
$x : 0$ $y : 1.5$

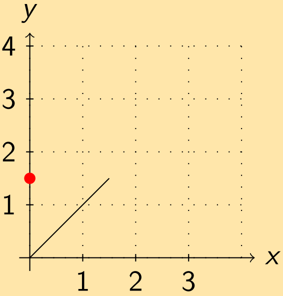


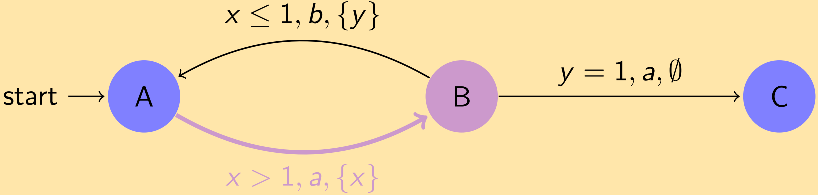

start
A
B
C
$x \leq 1, b, \{y\}$
$y = 1, a, \emptyset$
$x > 1, a, \{x\}$

[3] Alur and Dill, "A Theory of Timed Automata", 1994.

Timed automaton: a model for timed systems



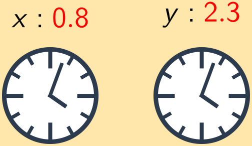


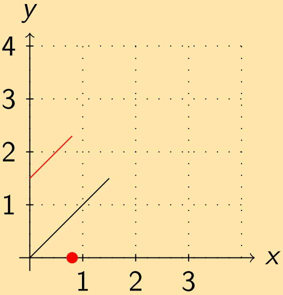


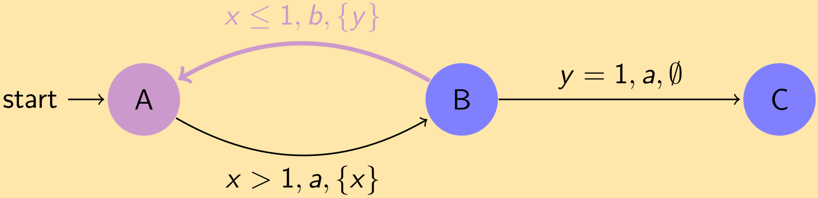
Execution: **1.5;a**

[3] Alur and Dill, "A Theory of Timed Automata", 1994.

Timed automaton: a model for timed systems



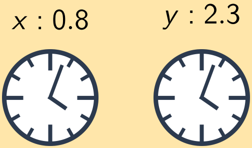


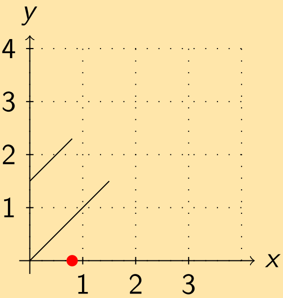


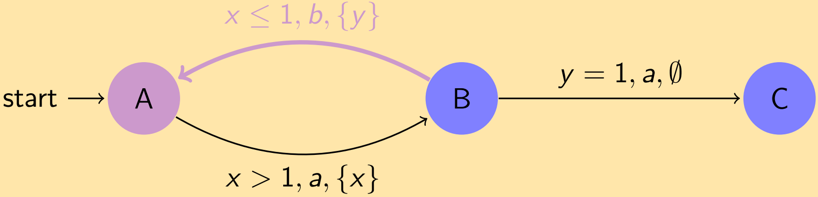
Execution: 1.5;a;**0.8;b**

[3] Alur and Dill, "A Theory of Timed Automata", 1994.

Timed automaton: a model for timed systems







Execution: 1.5;a;**0.8;b**

⇒ **Resets are unobservable**

[3] Alur and Dill, "A Theory of Timed Automata", 1994.

Event recording automata

Making resets observable

- One clock per letter in the alphabet.
- Each transition resets the clock corresponding to its letter.

[4] Alur, Fix, and Henzinger, "Event-Clock Automata: A Determinizable Class of Timed Automata", 1999.

Event recording automata

Making resets observable

- One clock per letter in the alphabet.
- Each transition resets the clock corresponding to its letter.

Observation: 0.7 a 1.4 c 0.8

[4] Alur, Fix, and Henzinger, "Event-Clock Automata: A Determinizable Class of Timed Automata", 1999.

Event recording automata

Making resets observable

- One clock per letter in the alphabet.
- Each transition resets the clock corresponding to its letter.

Observation: $0.7\,a\,[x_a \leftarrow 0]\,1.4\,c\,[x_c \leftarrow 0]\,0.8$

[4] Alur, Fix, and Henzinger, "Event-Clock Automata: A Determinizable Class of Timed Automata", 1999.

Event recording automata

Making resets observable

- One clock per letter in the alphabet.
- Each transition resets the clock corresponding to its letter.

Observation: $0.7\, a\, [x_a \leftarrow 0]\, 1.4\, c\, [x_c \leftarrow 0]\, 0.8$

Dynamics:

$$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \xrightarrow{0.7} \begin{pmatrix} 0.7 \\ 0.7 \\ 0.7 \end{pmatrix} \xrightarrow{a} \begin{pmatrix} 0.0 \\ 0.7 \\ 0.7 \end{pmatrix} \xrightarrow{1.4} \begin{pmatrix} 1.4 \\ 2.1 \\ 2.1 \end{pmatrix} \xrightarrow{c} \begin{pmatrix} 1.4 \\ 2.1 \\ 0.0 \end{pmatrix} \xrightarrow{0.8} \begin{pmatrix} 2.2 \\ 2.9 \\ 0.8 \end{pmatrix}$$

[4] Alur, Fix, and Henzinger, "Event-Clock Automata: A Determinizable Class of Timed Automata", 1999.

Event recording automata

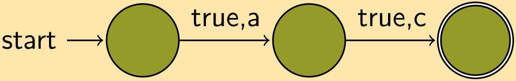
Making resets observable

- One clock per letter in the alphabet.
- Each transition resets the clock corresponding to its letter.

Observation: $0.7\,a\,[x_a \leftarrow 0]\,1.4\,c\,[x_c \leftarrow 0]\,0.8$

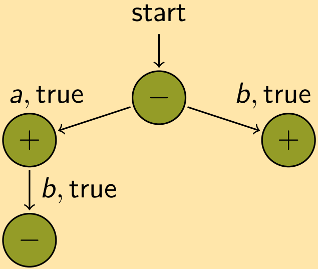
Dynamics:

$$\begin{pmatrix}0\\0\\0\end{pmatrix} \xrightarrow{0.7} \begin{pmatrix}0.7\\0.7\\0.7\end{pmatrix} \xrightarrow{a} \begin{pmatrix}0.0\\0.7\\0.7\end{pmatrix} \xrightarrow{1.4} \begin{pmatrix}1.4\\2.1\\2.1\end{pmatrix} \xrightarrow{c} \begin{pmatrix}1.4\\2.1\\0.0\end{pmatrix} \xrightarrow{0.8} \begin{pmatrix}2.2\\2.9\\0.8\end{pmatrix}$$



[4] Alur, Fix, and Henzinger, “Event-Clock Automata: A Determinizable Class of Timed Automata”, 1999.

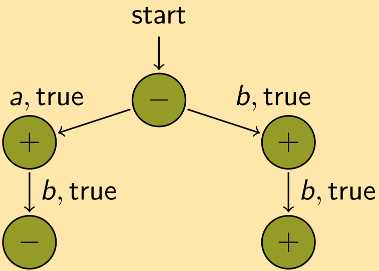
Active learning of ERAs



[5] Grinchtein, Jonsson, and Pettersson, "Inference of Event-Recording Automata Using Timed Decision Trees", 2006.

[6] Grinchtein, "Learning of Timed Systems", 2008.

Active learning of ERAs



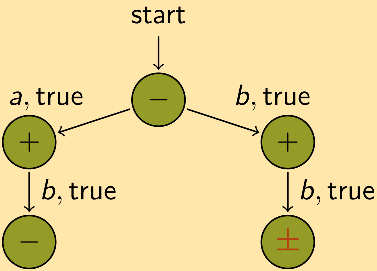
New observation:

$1.7b\,0.8b\ (+)$

[5] Grinchtein, Jonsson, and Pettersson, "Inference of Event-Recording Automata Using Timed Decision Trees", 2006.

[6] Grinchtein, "Learning of Timed Systems", 2008.

Active learning of ERAs



New observation:

$1.7b\,0.8b\ (+)$

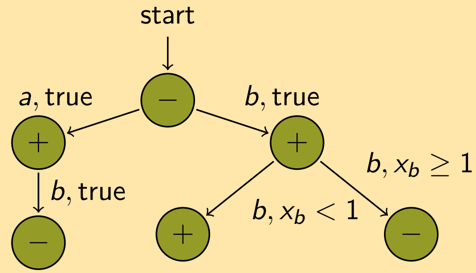
$1.4b\,1.2b\ (-)$

⚠ Inconsistency

[5] Grinchtein, Jonsson, and Pettersson, "Inference of Event-Recording Automata Using Timed Decision Trees", 2006.

[6] Grinchtein, "Learning of Timed Systems", 2008.

Active learning of ERAs



New observation:

$1.7b\,0.8b\;(+)$

$1.4b\,1.2b\;(-)$

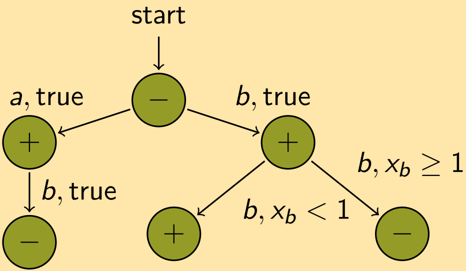
⚠ Inconsistency

⇒ Refinement of the guards.

[5] Grinchtein, Jonsson, and Pettersson, "Inference of Event-Recording Automata Using Timed Decision Trees", 2006.

[6] Grinchtein, "Learning of Timed Systems", 2008.

Active learning of ERAs



New observation:

$1.7b\,0.8b\ (+)$

$1.4b\,1.2b\ (-)$

⚠ Inconsistency

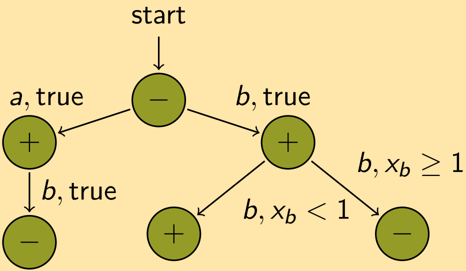
$\Rightarrow$ Refinement of the guards.

A consistent stucture can be folded as an ERA.

[5] Grinchtein, Jonsson, and Pettersson, "Inference of Event-Recording Automata Using Timed Decision Trees", 2006.

[6] Grinchtein, "Learning of Timed Systems", 2008.

Active learning of ERAs



New observation:

$1.7b\,0.8b\ (+)$

$1.4b\,1.2b\ (-)$

⚠ Inconsistency

⇒ Refinement of the guards.

A consistent stucture can be folded as an ERA.

- Interesting structure with good algorithms,
- Does not deal with reset guesses.

[5] Grinchtein, Jonsson, and Pettersson, "Inference of Event-Recording Automata Using Timed Decision Trees", 2006.

[6] Grinchtein, "Learning of Timed Systems", 2008.

Reset-free ERA

Making resets observable

- One clock per letter in the alphabet.
- Each transition resets the clock corresponding to its letter.

Reset-free ERA

Making resets observable

- One clock per letter in the alphabet.
- Each transition **may reset** the clock corresponding to its letter.

Observation: $0.7\,a \quad 1.4\,c \quad 0.8$

Reset-free ERA

Making resets observable

- One clock per letter in the alphabet.
- Each transition **may reset** the clock corresponding to its letter.

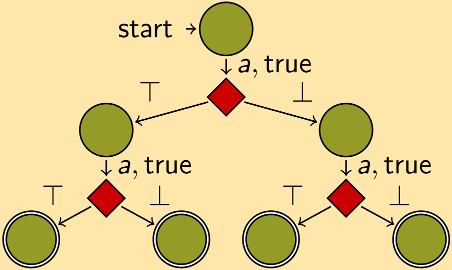
Observation: $0.7\, a\, ?\, 1.4\, c\, ?\, 0.8$

Reset-free ERA

Making resets observable

- One clock per letter in the alphabet.
- Each transition **may reset** the clock corresponding to its letter.

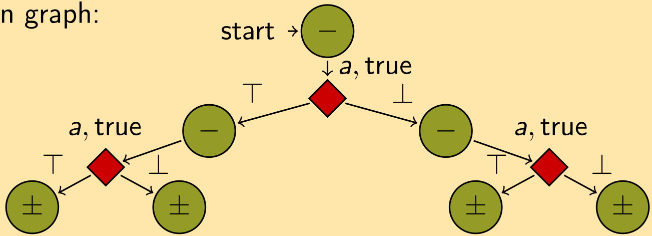
Observation: $0.7\, a\, ?\, 1.4\, c\, ?\, 0.8$

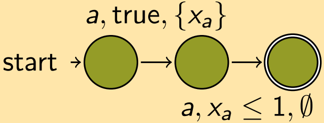


Active learning of RERAs: first try

Changing the structure

Decision graph:

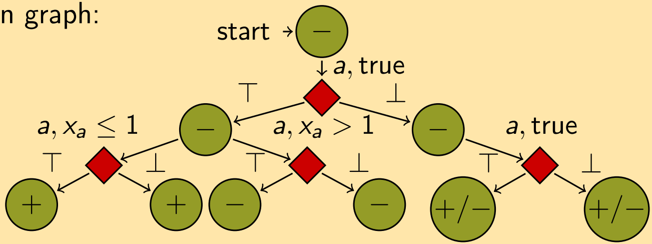


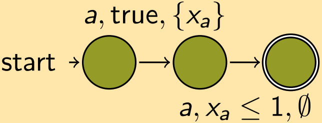


Active learning of RERAs: first try

Changing the structure

Decision graph:



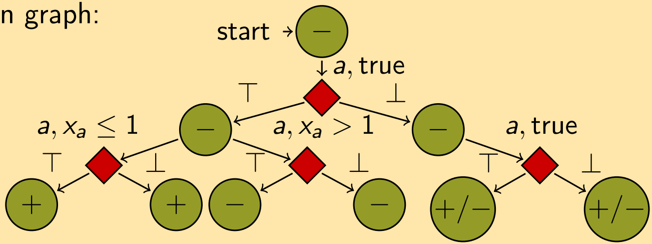


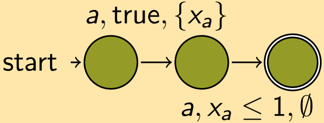
Inconsistency is detected $\Rightarrow$ Separating guards

Active learning of RERAs: first try

Changing the structure

Decision graph:





Inconsistency is detected ⇒ Separating guards

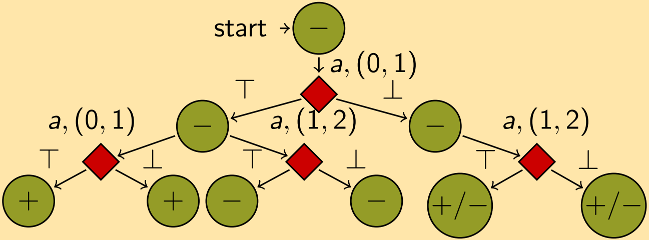
No separating guard without a reset:

$0.9a\,0.8a$ $(+)$ and $0.7a\,1.1a$ $(-)$

Active learning of RERAs: invalidity

Guessing resets

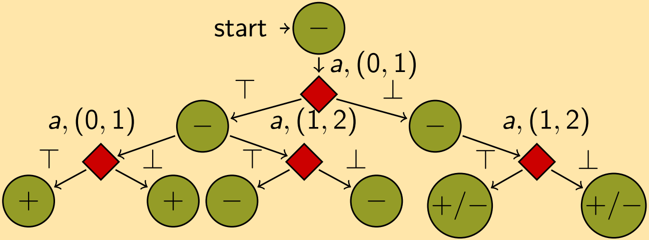
Added structure: Observation graph



Active learning of RERAs: invalidity

Guessing resets

Added structure: Observation graph

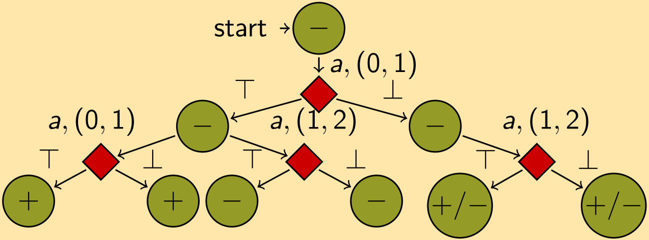


- Each guard is "minimal": a cube of size one.

Active learning of RERAs: invalidity

Guessing resets

Added structure: Observation graph

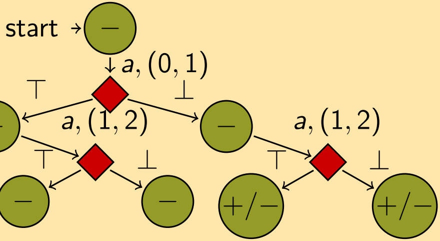


- Each guard is "minimal": a cube of size one.
- A node corresponding to both positive and negative observations is an **invalidity**.

Active learning of RERAs: invalidity

Guessing resets

Added structure: Observation graph

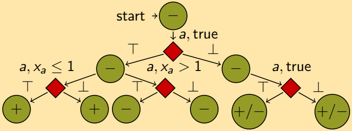


- Each guard is "minimal": a cube of size one.
- A node corresponding to both positive and negative observations is an **invalidity**.
- Allows to prune the main structure.

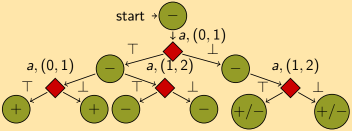
Active learning of RERAs

Combining the structures

Decision graph:



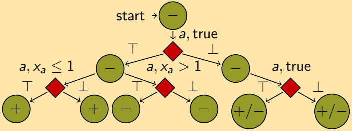
Observation graph:



Active learning of RERAs

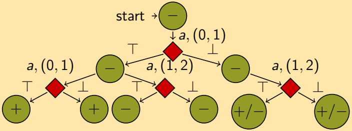
Combining the structures

Decision graph:



- Partitions the clock values
- Detects inconsistencies
- Infers guards

Observation graph:

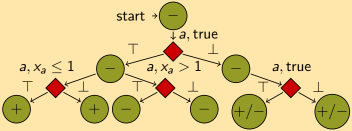


- Precise
- Detects invalidity
- Infers clock resets

Active learning of RERAs

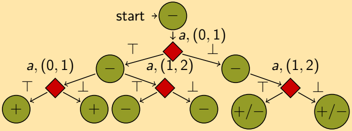
Combining the structures

Decision graph:



- Partitions the clock values
- Detects inconsistencies
- Infers guards

Observation graph:



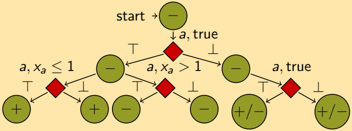
- Precise
- Detects invalidity
- Infers clock resets

- Observations are propagated in both structures

Active learning of RERAs

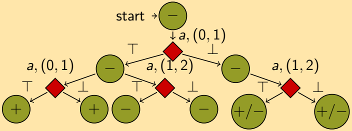
Combining the structures

Decision graph:



- Partitions the clock values
- Detects inconsistencies
- Infers guards

Observation graph:



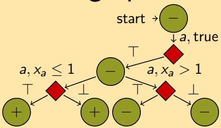
- Precise
- Detects invalidity
- Infers clock resets

- Observations are propagated in both structures
- Observation graph is used to prune the Decision graph

Active learning of RERAs

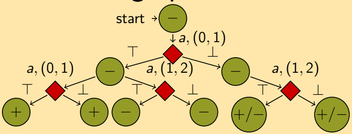
Combining the structures

Decision graph:



- Partitions the clock values
- Detects inconsistencies
- Infers guards

Observation graph:



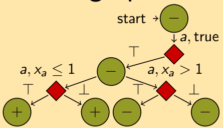
- Precise
- Detects invalidity
- Infers clock resets

- Observations are propagated in both structures
- Observation graph is used to prune the Decision graph

Active learning of RERAs

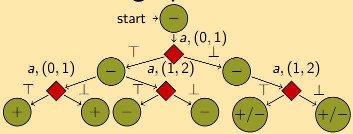
Combining the structures

Decision graph:



- Partitions the clock values
- Detects inconsistencies
- Infers guards

Observation graph:



- Precise
- Detects invalidity
- Infers clock resets

- Observations are propagated in both structures
- Observation graph is used to prune the Decision graph
- Decision graph will be folded into a RERA

Conclusion

Learning of RERAs

- **extend** the learnable model classes;
- introduces the key notion of **invalidity**;
- paves the way to the learning of **deterministic TAs**.

Conclusion

Learning of RERAs

- **extend** the learnable model classes;
- introduces the key notion of **invalidity**;
- paves the way to the learning of **deterministic TAs**.

It requires

- **new structures** to separate reset guessing and decision making;
- new and updated **parallelisable** algorithms

Conclusion

Learning of RERAs

- **extend** the learnable model classes;
- introduces the key notion of **invalidity**;
- paves the way to the learning of **deterministic TAs**.

It requires

- **new structures** to separate reset guessing and decision making;
- new and updated **parallelisable** algorithms

Interesting open questions:

- reduce the space-cost using **implicit** structures;
- **redact** and analyse the generalization to deterministic TAs.