Learning Generalised Tree Automata

Gerco van Heerdt Tobias Kappé Jurriaan Rot
Matteo Sammartino Alexandra Silva

June 25, 2020

$\mathtt{L}^\star$ setup for DFAs

Finite alphabet A

System behaviour captured by a **regular language** $\mathcal{L} \subseteq A^*$

$\mathtt{L}^\star$ learns *minimal* DFA for $\mathcal{L}$

$\mathtt{L}^\star$ setup for DFAs

Finite alphabet A

System behaviour captured by a **regular language** $\mathcal{L} \subseteq A^*$

$\mathtt{L}^\star$ learns *minimal* DFA for $\mathcal{L}$ assuming an *oracle* that answers

- **Membership queries**

$$\boxed{w \in \mathcal{L}?}$$

$\mathrm{L}^\star$ setup for DFAs

Finite alphabet A

System behaviour captured by a **regular language** $\mathcal{L} \subseteq A^*$

$\mathrm{L}^\star$ learns *minimal* DFA for $\mathcal{L}$ assuming an *oracle* that answers

- **Membership queries**

$$w \in \mathcal{L}?$$

- **Equivalence queries**

$$\mathcal{L}(H) = \mathcal{L}?$$

Negative result $\implies$ *counterexample*

L* observation table

L* maintains $S, E \subseteq A^*$ inducing a table

		E: ε	E: a
S	ε	1	0
S	a	0	1
S	aa	1	0
$S \cdot A$	aaa	0	1

$\mathtt{L}^\star$ observation table

$\mathtt{L}^\star$ maintains $S, E \subseteq A^*$ inducing a table

E = columns; S = rows ε, a, aa; $S \cdot A$ = row aaa

	ε	a
ε	1	0
a	0	1
aa	1	0
aaa	0	1

Prepend *row label* to *column label* and pose **membership query**

$$(s, e) \mapsto \begin{cases} 1 & \text{if } se \in \mathcal{L} \\ 0 & \text{if } se \notin \mathcal{L} \end{cases}$$

L* observation table

L* maintains $S, E \subseteq A^*$ inducing a table

		E: ε	E: a
S	ε	1	0
S	a	0	1
S	aa	1	0
$S \cdot A$	aaa	0	1

$\mathcal{L} = \{a^n \mid n \text{ is even}\}$

$aa \cdot a \notin \mathcal{L}$

Prepend *row label* to *column label* and pose **membership query**

$$(s, e) \mapsto \begin{cases} 1 & \text{if } se \in \mathcal{L} \\ 0 & \text{if } se \notin \mathcal{L} \end{cases}$$

L* hypothesis DFA

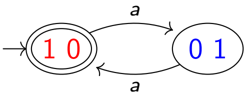
Hypothesis states are upper rows of the table

	ε	a
ε	1	0
a	0	1
aa	1	0
aaa	0	1

L* hypothesis DFA

Hypothesis states are upper rows of the table;
transitions append symbols to row labels

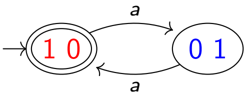
	ε	a
ε	1	0
a	0	1
aa	1	0
aaa	0	1



L⋆ hypothesis DFA

Hypothesis states are upper rows of the table;
transitions append symbols to row labels

	ε	a
ε	1	0
a	0	1
aa	1	0
aaa	0	1



Requires properties **closedness** and **consistency** to be well-defined

$L^\star$ algorithm overview

1. Initialise $S = E = \{\varepsilon\}$

L* algorithm overview

table updated using
membership queries

1. Initialise $S = E = \{\varepsilon\}$
2. Satisfy closedness and consistency (by augmenting S and E)

L* algorithm overview

table updated using
membership queries

1. Initialise $S = E = \{\varepsilon\}$
2. Satisfy closedness and consistency (by augmenting S and E)
3. Construct hypothesis

L* algorithm overview

table updated using
membership queries

1. Initialise $S = E = \{\varepsilon\}$
2. Satisfy closedness and consistency (by augmenting S and E)
3. Construct hypothesis
4. Pose equivalence query

L* algorithm overview

table updated using
membership queries

1. Initialise $S = E = \{\varepsilon\}$
2. Satisfy closedness and consistency (by augmenting S and E)
3. Construct hypothesis
4. Pose equivalence query
5. On a counterexample, add its prefixes to S and repeat from 2

DFAs vs (bottom-up) generalised tree automata

DFA

$$\begin{array}{c} 1 + Q \times A \\ \downarrow \\ Q \end{array}$$

accept/reject map $Q \to 2$

DFAs vs (bottom-up) generalised tree automata

DFA

$$\begin{array}{c} 1 + Q \times A \\ \downarrow \\ Q \end{array}$$

accept/reject map $Q \to 2$

tree automaton

$$\begin{array}{c} I + FQ \\ \downarrow \\ Q \end{array}$$

accept/reject map $Q \to 2$

DFAs vs (bottom-up) generalised tree automata

DFA	tree automaton
$\begin{array}{c} 1 + Q \times A \\ \downarrow \\ Q \end{array}$	$\begin{array}{c} I + FQ \\ \downarrow \\ Q \end{array}$
accept/reject map $Q \to 2$	accept/reject map $Q \to 2$

Semantics: language of trees generated by F and I

DFAs vs (bottom-up) generalised tree automata

DFA	tree automaton
$1 + Q \times A \rightarrow Q$	$I + FQ \rightarrow Q$
accept/reject map $Q \to 2$	accept/reject map $Q \to 2$

Semantics: language of trees generated by F and I

$F\colon$ **Set** $\to$ **Set** *strongly finitary*

Trees generated by functor

Trees generated by F with leaves in I:

$$F^* I = \mathrm{lfp}(I + F(-))$$

F^* is the *free monad* over F

Trees generated by functor

Trees generated by F with leaves in I:

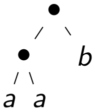
$$F^* I = \mathrm{lfp}(I + F(-))$$

F^* is the *free monad* over F

Example:

$$FX = X \times X \qquad\qquad I = \{a, b\}$$

Then e.g.



$\in F^* I$

Strongly finitary functor

Finitary: $u \in FX$ "contains" finitely many elements of X

- $FX = \mathcal{P}X = \{U \mid U \subseteq X\}$ is not but
- $FX = \mathcal{P}_{\text{fin}}X = \{U \mid U \subseteq X, U \text{ finite}\}$ is

Strongly finitary functor

Finitary: $u \in FX$ "contains" finitely many elements of X

- $FX = \mathcal{P}X = \{U \mid U \subseteq X\}$ is not but
- $FX = \mathcal{P}_{\mathrm{fin}}X = \{U \mid U \subseteq X, U \text{ finite}\}$ is

Strongly finitary: F also preserves finite sets

- $FX = X^*$ is not but
- $FX = \mathcal{P}_{\mathrm{fin}}X$ is

Tree automaton example

$I = \{a, b\}$, $FX = X \times X$

Tree automaton example

$I = \{a, b\}$, $FX = X \times X$

$Q = \{q_{\text{even}}, q_{\text{odd}}\}$

Tree automaton example

$I = \{a, b\}$, $FX = X \times X$

$Q = \{q_{\text{even}}, q_{\text{odd}}\}$

$$a \mapsto q_{\text{odd}} \qquad b \mapsto q_{\text{even}}$$

Tree automaton example

$I = \{a, b\}$, $FX = X \times X$

$Q = \{q_{\text{even}}, q_{\text{odd}}\}$

$$a \mapsto q_{\text{odd}} \qquad b \mapsto q_{\text{even}}$$

$$(q_{\text{even}}, q_{\text{even}}) \mapsto q_{\text{even}} \qquad (q_{\text{even}}, q_{\text{odd}}) \mapsto q_{\text{odd}}$$

$$(q_{\text{odd}}, q_{\text{even}}) \mapsto q_{\text{odd}} \qquad (q_{\text{odd}}, q_{\text{odd}}) \mapsto q_{\text{even}}$$

Tree automaton example

$I = \{a, b\}$, $FX = X \times X$

$Q = \{q_{\text{even}}, q_{\text{odd}}\}$

$$a \mapsto q_{\text{odd}} \qquad b \mapsto q_{\text{even}}$$

$$(q_{\text{even}}, q_{\text{even}}) \mapsto q_{\text{even}} \qquad (q_{\text{even}}, q_{\text{odd}}) \mapsto q_{\text{odd}}$$

$$(q_{\text{odd}}, q_{\text{even}}) \mapsto q_{\text{odd}} \qquad (q_{\text{odd}}, q_{\text{odd}}) \mapsto q_{\text{even}}$$

$$o(q_{\text{even}}) = 1 \qquad o(q_{\text{odd}}) = 0$$

Tree automaton example

$I = \{a, b\}$, $FX = X \times X$

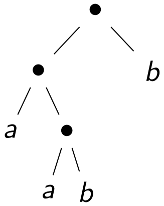
$Q = \{q_{\text{even}}, q_{\text{odd}}\}$

$$a \mapsto q_{\text{odd}} \qquad b \mapsto q_{\text{even}}$$

$$(q_{\text{even}}, q_{\text{even}}) \mapsto q_{\text{even}} \qquad (q_{\text{even}}, q_{\text{odd}}) \mapsto q_{\text{odd}}$$

$$(q_{\text{odd}}, q_{\text{even}}) \mapsto q_{\text{odd}} \qquad (q_{\text{odd}}, q_{\text{odd}}) \mapsto q_{\text{even}}$$

$$o(q_{\text{even}}) = 1 \qquad o(q_{\text{odd}}) = 0$$



Tree automaton example

$I = \{a, b\}$, $FX = X \times X$

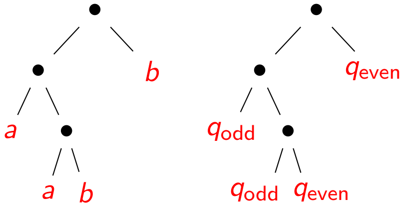
$Q = \{q_{\text{even}}, q_{\text{odd}}\}$

$$a \mapsto q_{\text{odd}} \qquad b \mapsto q_{\text{even}}$$

$$(q_{\text{even}}, q_{\text{even}}) \mapsto q_{\text{even}} \qquad (q_{\text{even}}, q_{\text{odd}}) \mapsto q_{\text{odd}}$$

$$(q_{\text{odd}}, q_{\text{even}}) \mapsto q_{\text{odd}} \qquad (q_{\text{odd}}, q_{\text{odd}}) \mapsto q_{\text{even}}$$

$$o(q_{\text{even}}) = 1 \qquad o(q_{\text{odd}}) = 0$$



Tree automaton example

$I = \{a, b\}$, $FX = X \times X$

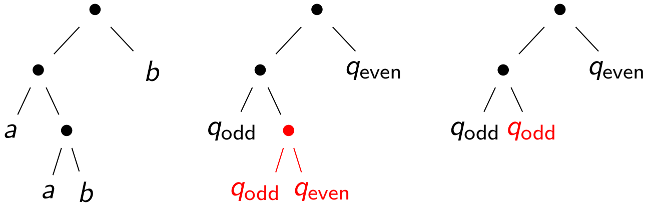
$Q = \{q_{\text{even}}, q_{\text{odd}}\}$

$$a \mapsto q_{\text{odd}} \qquad b \mapsto q_{\text{even}}$$

$$(q_{\text{even}}, q_{\text{even}}) \mapsto q_{\text{even}} \qquad (q_{\text{even}}, q_{\text{odd}}) \mapsto q_{\text{odd}}$$

$$(q_{\text{odd}}, q_{\text{even}}) \mapsto q_{\text{odd}} \qquad (q_{\text{odd}}, q_{\text{odd}}) \mapsto q_{\text{even}}$$

$$o(q_{\text{even}}) = 1 \qquad o(q_{\text{odd}}) = 0$$



Tree automaton example

$$I = \{a, b\},\ FX = X \times X$$

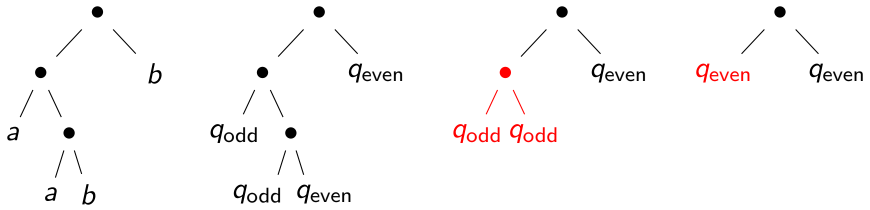
$$Q = \{q_{\text{even}}, q_{\text{odd}}\}$$

$$a \mapsto q_{\text{odd}} \qquad b \mapsto q_{\text{even}}$$

$$(q_{\text{even}}, q_{\text{even}}) \mapsto q_{\text{even}} \qquad (q_{\text{even}}, q_{\text{odd}}) \mapsto q_{\text{odd}}$$

$$(q_{\text{odd}}, q_{\text{even}}) \mapsto q_{\text{odd}} \qquad (q_{\text{odd}}, q_{\text{odd}}) \mapsto q_{\text{even}}$$

$$o(q_{\text{even}}) = 1 \qquad o(q_{\text{odd}}) = 0$$



Tree automaton example

$I = \{a, b\},\ FX = X \times X$

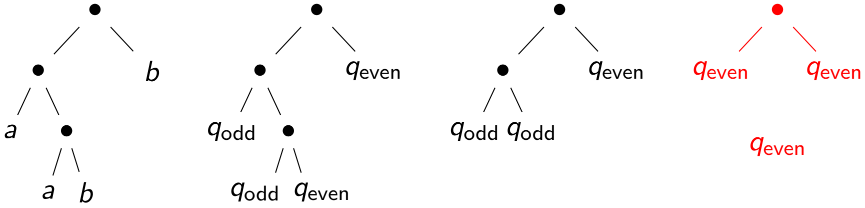
$Q = \{q_{\text{even}}, q_{\text{odd}}\}$

$$a \mapsto q_{\text{odd}} \qquad b \mapsto q_{\text{even}}$$

$$(q_{\text{even}}, q_{\text{even}}) \mapsto q_{\text{even}} \qquad (q_{\text{even}}, q_{\text{odd}}) \mapsto q_{\text{odd}}$$

$$(q_{\text{odd}}, q_{\text{even}}) \mapsto q_{\text{odd}} \qquad (q_{\text{odd}}, q_{\text{odd}}) \mapsto q_{\text{even}}$$

$$o(q_{\text{even}}) = 1 \qquad o(q_{\text{odd}}) = 0$$



Tree automaton example

$I = \{a, b\}$, $FX = X \times X$

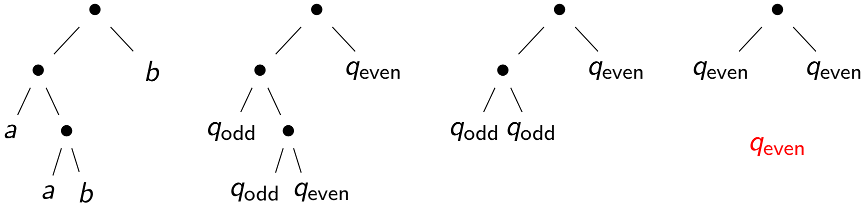
$Q = \{q_{\text{even}}, q_{\text{odd}}\}$

$$a \mapsto q_{\text{odd}} \qquad b \mapsto q_{\text{even}}$$

$$(q_{\text{even}}, q_{\text{even}}) \mapsto q_{\text{even}} \qquad (q_{\text{even}}, q_{\text{odd}}) \mapsto q_{\text{odd}}$$

$$(q_{\text{odd}}, q_{\text{even}}) \mapsto q_{\text{odd}} \qquad (q_{\text{odd}}, q_{\text{odd}}) \mapsto q_{\text{even}}$$

$$o(q_{\text{even}}) = 1 \qquad o(q_{\text{odd}}) = 0$$



Contexts

Contexts over F: $F^*(I + \{\Box\})$

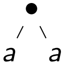
$\Box$ placeholder to plug in trees

Contexts

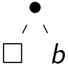
Contexts over F: $F^*(I + \{\Box\})$

$\Box$ placeholder to plug in trees

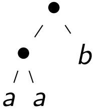
Example: plugging



into



gives



Observation table

		E	
		□	•(□, *a*)
S	a	0	1
	b	1	0
FS	•(*a*, *a*)	1	0
	•(*a*, *b*)	0	1
	•(*b*, *a*)	0	1
	•(*b*, *b*)	1	0

Observation table

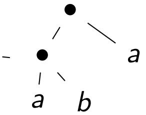
		E: $\square$	E: $\bullet(\square, a)$
S	a	0	1
S	b	1	0
FS	$\bullet(a, a)$	1	0
FS	$\bullet(a, b)$	0	1
FS	$\bullet(b, a)$	0	1
FS	$\bullet(b, b)$	1	0

$\mathcal{L} = \{t \in F^* I \mid \text{even } a\text{'s in } t\}$

Observation table

		E: $\square$	E: $\bullet(\square, a)$
S	a	0	1
S	b	1	0
FS	$\bullet(a, a)$	1	0
FS	$\bullet(a, b)$	0	1
FS	$\bullet(b, a)$	0	1
FS	$\bullet(b, b)$	1	0

$\mathcal{L} = \{t \in F^* I \mid \text{even } a\text{'s in } t\}$



$\in \mathcal{L}$

L* observation table, abstractly

$S, E \subseteq A^*$

reachability — language

$$S \xrightarrow{\alpha} Q \xrightarrow{\beta} 2^E$$

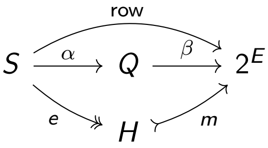
L* observation table, abstractly

$S, E \subseteq A^*$

$$S \xrightarrow{\alpha} Q \xrightarrow{\beta} 2^E, \qquad \mathsf{row} : S \to 2^E$$

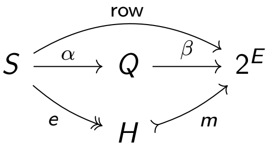
L* observation table, abstractly

$S, E \subseteq A^*$



L* observation table, abstractly

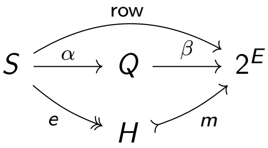
$S, E \subseteq A^*$

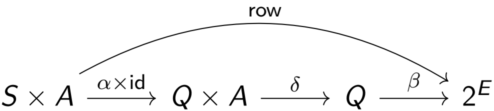


$$S \times A \xrightarrow{\alpha \times \mathsf{id}} Q \times A \xrightarrow{\delta} Q \xrightarrow{\beta} 2^E$$

L* observation table, abstractly

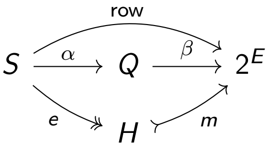
$S, E \subseteq A^*$

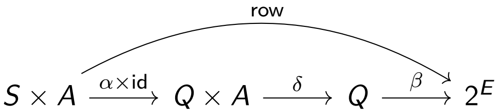


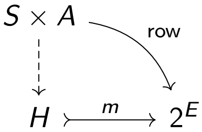


L* observation table, abstractly

$S, E \subseteq A^*$



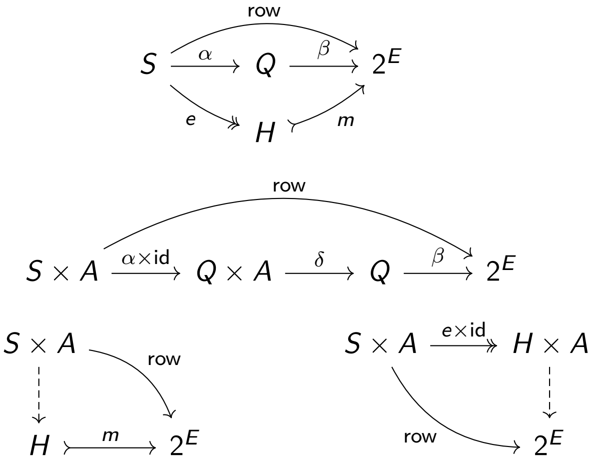




closedness

L* observation table, abstractly

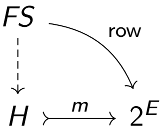
$S, E \subseteq A^*$



closedness

consistency

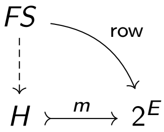
Closedness and consistency for tree automata



closedness:

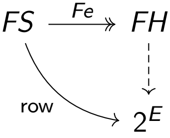
$\forall x \in FS\, \exists s \in S.$
$\mathrm{row}(s) = \mathrm{row}(x)$

Closedness and consistency for tree automata



closedness:

$$\forall x \in FS\, \exists s \in S.$$
$$\mathrm{row}(s) = \mathrm{row}(x)$$



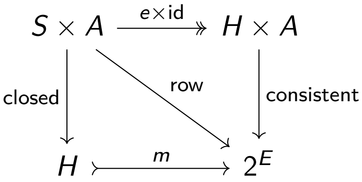
consistency:

$$\forall x \in F(S \times S).$$
$$F(\mathrm{row} \circ \pi_1)(x) = F(\mathrm{row} \circ \pi_2)(x)$$
$$\Longrightarrow$$
$$(\mathrm{row} \circ F\pi_1)(x) = (\mathrm{row} \circ F\pi_2)(x)$$

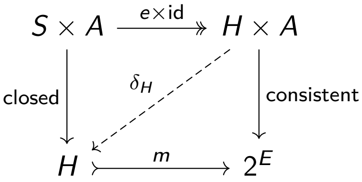
Hypothesis, abstractly

$$
\begin{array}{ccc}
S \times A & \xrightarrow{e \times \mathrm{id}} & H \times A \\
\text{closed} \downarrow & & \downarrow \text{consistent} \\
H & \xrightarrow{m} & 2^{E}
\end{array}
$$

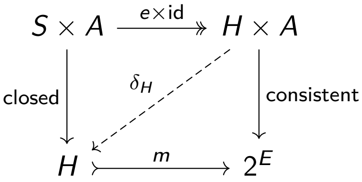
Hypothesis, abstractly



Hypothesis, abstractly

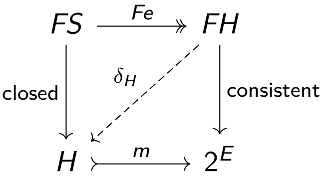


Hypothesis, abstractly



$$\delta_H(\mathrm{row}(s), a) = \mathrm{row}(sa)$$

Hypothesis for tree automata



$$\delta_H(Fe(x)) = \mathsf{row}(x)$$

Hypothesis example

		E: $\square$	E: $\bullet(\square, a)$
S	a	0	1
S	b	1	0
FS	$\bullet(a, a)$	1	0
FS	$\bullet(a, b)$	0	1
FS	$\bullet(b, a)$	0	1
FS	$\bullet(b, b)$	1	0

Hypothesis example

		E: $\square$	E: $\bullet(\square, a)$
S	a	0	1
S	b	1	0
FS	$\bullet(a, a)$	1	0
FS	$\bullet(a, b)$	0	1
FS	$\bullet(b, a)$	0	1
FS	$\bullet(b, b)$	1	0

$a \mapsto q_0$

$b \mapsto q_1$

Hypothesis example

		E: $\square$	E: $\bullet(\square, a)$
S	a	0	1
S	b	1	0
FS	$\bullet(a, a)$	1	0
FS	$\bullet(a, b)$	0	1
FS	$\bullet(b, a)$	0	1
FS	$\bullet(b, b)$	1	0

$a \mapsto q_0$

$b \mapsto q_1$

$o(q_0) = 0, o(q_1) = 1$

Hypothesis example

E = { $\square$, $\bullet(\square, a)$ }

		$\square$	$\bullet(\square, a)$
S	a	0	1
	b	1	0
FS	$\bullet(a, a)$	1	0
	$\bullet(a, b)$	0	1
	$\bullet(b, a)$	0	1
	$\bullet(b, b)$	1	0

$a \mapsto q_0$

$b \mapsto q_1$

$o(q_0) = 0, o(q_1) = 1$

$(q_0, q_0) \mapsto q_1$

Hypothesis example

		E: $\square$	E: $\bullet(\square, a)$
S	a	0	1
S	b	1	0
FS	$\bullet(a, a)$	1	0
FS	$\bullet(a, b)$	0	1
FS	$\bullet(b, a)$	0	1
FS	$\bullet(b, b)$	1	0

$$a \mapsto q_0$$
$$b \mapsto q_1$$

$$o(q_0) = 0, o(q_1) = 1$$

$$(q_0, q_0) \mapsto q_1$$

$$(q_0, q_1) \mapsto q_0$$

Hypothesis example

E = { □, •(□, a) }

		□	•(□, a)
S	a	0	1
	b	1	0
FS	•(a, a)	1	0
	•(a, b)	0	1
	•(b, a)	0	1
	•(b, b)	1	0

$$a \mapsto q_0$$

$$b \mapsto q_1$$

$$o(q_0) = 0, o(q_1) = 1$$

$$(q_0, q_0) \mapsto q_1$$

$$(q_0, q_1) \mapsto q_0$$

$$(q_1, q_0) \mapsto q_0$$

Hypothesis example

		E: $\square$	E: $\bullet(\square, a)$
S	a	0	1
	b	1	0
FS	$\bullet(a, a)$	1	0
	$\bullet(a, b)$	0	1
	$\bullet(b, a)$	0	1
	$\bullet(b, b)$	1	0

$$a \mapsto q_0$$
$$b \mapsto q_1$$

$$o(q_0) = 0, o(q_1) = 1$$

$$(q_0, q_0) \mapsto q_1$$
$$(q_0, q_1) \mapsto q_0$$
$$(q_1, q_0) \mapsto q_0$$
$$(q_1, q_1) \mapsto q_1$$

Algorithm overview

table updated using
membership queries

1. Initialise $S = I$, $E = \{\Box\}$
2. Satisfy closedness and consistency (by augmenting S and E)
3. Construct hypothesis
4. Pose equivalence query
5. On a counterexample, add its subtrees to S and repeat from 2

Contributions

Abstract version of $\mathtt{L}^\star$

- On any category satisfying some (mild) conditions
- Abstract iterations, counterexamples
- Termination proof

Contributions

Abstract version of $\mathtt{L}^\star$

- On any category satisfying some (mild) conditions
- Abstract iterations, counterexamples
- Termination proof

Instantiation to learn **generalised** tree automata in **Set**

Future directions

Generalise monad F^* to arbitrary monad

- Learn pomset automata

Future directions

Generalise monad F^* to arbitrary monad

- Learn pomset automata

Tree automata in other categories

Future directions

Generalise monad F^* to arbitrary monad

- Learn pomset automata

Tree automata in other categories

Regular ω-tree languages