Automata learning: a categorical perspective

A tribute to Prakash Panangaden

Alexandra Silva

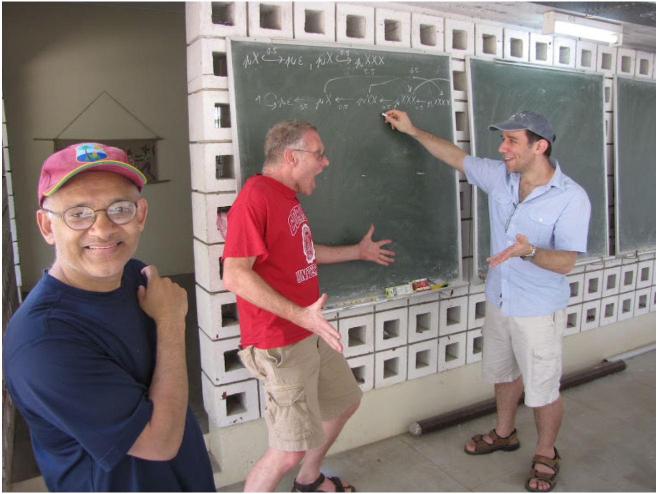
Radboud University Nijmegen
and
Centrum Wiskunde & Informatica

Prakash fest, 24 May 2014

A tribute to Prakash Panangaden

terrified student

A tribute to Prakash Panangaden



not so terrified student

A tribute to Prakash Panangaden

co-author

A tribute to Prakash Panangaden

sous-chef

A tribute to Prakash Panangaden

co-explorer (of port wine!)

A tribute to Prakash Panangaden

co-organizer

A tribute to Prakash Panangaden

SIGLOG

A tribute to Prakash Panangaden

SIGLOG

SIGLOG Executive Committee
Prakash Panangaden: Chair
Luke Ong: Vice-chair
Natarajan Shankar: Treasurer
Alexandra Silva: Secretary

A tribute to Prakash Panangaden

Thanks for the friendship and inspiration!

Automata learning: encounters

- 2011 : Frits Vaandrager.

  Printed Angluin's paper *Learning Regular Sets from Queries and Counterexamples*.

Automata learning: encounters

- 2011 : Frits Vaandrager.

  Printed Angluin's paper *Learning Regular Sets from Queries and Counterexamples*.

- 2012: One night in Dagstuhl

  I have this feeling that category theory has something to say about automata learning. (Prakash)

Automata learning: encounters

- 2011 : Frits Vaandrager.

  Printed Angluin's paper *Learning Regular Sets from Queries and Counterexamples*.

- 2012: One night in Dagstuhl

  I have this feeling that category theory has something to say about automata learning. (Prakash)

  Printed Angluin's paper *Learning Regular Sets from Queries and Counterexamples*.

Automata learning: encounters

- 2013: Prakash's volume

Automata learning: encounters

- 2013: Prakash's volume

  Printed Angluin's paper *Learning Regular Sets from Queries and Counterexamples*...

- ...and read it with categorical glasses. Joint work with Bart Jacobs.

The L^* algorithm: ingredients

- Master language $\mathcal{L}\colon A^* \to 2$ (regular language).

The L^* algorithm: ingredients

- Master language $\mathcal{L}\colon A^* \to 2$ (regular language).
- The teacher, omniscient, answers 2 types of queries
  - $w \in \mathcal{L}$?
  - *Guess correct?* If no, counter-example.

The L^* algorithm: ingredients

- Master language $\mathcal{L} \colon A^* \to 2$ (regular language).
- The teacher, omniscient, answers 2 types of queries
  - $w \in \mathcal{L}$?
  - *Guess correct?* If no, counter-example.
- L^*-algorithm (Angluin 87)
  - Incrementally builds an *observation table*.
  - Table closed & consistent = finite automaton accepting $\mathcal{L}$

The L^* algorithm: ingredients

- Master language $\mathcal{L}\colon A^* \to 2$ (regular language).
- The teacher, omniscient, answers 2 types of queries
  - $w \in \mathcal{L}$?
  - *Guess correct?* If no, counter-example.
- L^*-algorithm (Angluin 87)
  - Incrementally builds an *observation table*.
  - Table closed & consistent = finite automaton accepting $\mathcal{L}$ and minimal!

The L^* algorithm: the observation table

An observation table is a triple (S, E, row), where

$$row\colon (S \cup S \cdot A) \to 2^E$$

with $S, E \subseteq 2^{A^*}$.

The L^* algorithm: the observation table

An observation table is a triple (S, E, row), where

$$row\colon (S \cup S \cdot A) \to 2^E$$

with $S, E \subseteq 2^{A^*}$.

Closed and Consistent Table

(S, E, row) is *closed* if for all $t \in S \cdot A$ there exists an $s \in S$ such that $row(t) = row(s)$.

The L^* algorithm: the observation table

An observation table is a triple (S, E, row), where

$$row\colon (S \cup S \cdot A) \to 2^E$$

with $S, E \subseteq 2^{A^*}$.

Closed and Consistent Table

(S, E, row) is *closed* if for all $t \in S \cdot A$ there exists an $s \in S$ such that $row(t) = row(s)$.

(S, E, row) is *consistent* if whenever $s_1, s_2 \in S$ are such that $row(s_1) = row(s_2)$, for all $a \in A$, $row(s_1 a) = row(s_2 a)$.

The L^* algorithm: the observation table

(S, E, row) is *closed* if for all $t \in S \cdot A$ there exists an $s \in S$ such that $row(t) = row(s)$.

The L^* algorithm: the observation table

$$
\begin{tikzcd}
S \arrow[r, two heads, "e"] \arrow[rrd, bend left, "row"] & \bullet \arrow[rd, tail, "m"] & \\
 & & 2^E \\
S \cdot A \arrow[r, two heads, "e_A"'] \arrow[rru, bend right, "row"'] & \bullet \arrow[ru, tail, "m_A"'] \arrow[uu, dotted, "\exists i"] &
\end{tikzcd}
$$

closed

(S, E, row) is *closed* if for all $t \in S \cdot A$ there exists an $s \in S$ such that $row(t) = row(s)$.

The L^* algorithm: the observation table

$$
\begin{tikzcd}
S \arrow[r, two heads, "e"] \arrow[rrd, bend left, "row"] & \bullet \arrow[rd, tail, "m"] & \\
 & & 2^E \\
S \cdot A \arrow[r, two heads, "e_A"'] \arrow[rru, bend right, "row"'] & \bullet \arrow[ru, tail, "m_A"'] \arrow[uu, dotted, "\exists i"] &
\end{tikzcd}
$$

closed

(S, E, row) is *consistent* if whenever $s_1, s_2 \in S$ are such that $row(s_1) = row(s_2)$, for all $a \in A$, $row(s_1 a) = row(s_2 a)$.

The L^* algorithm: the observation table

$$
\begin{tikzcd}
S \arrow[r, two heads, "e"] \arrow[drr, bend left, "row"] & \bullet \arrow[dr, tail, "m"] & \\
 & & 2^E \\
S \cdot A \arrow[r, two heads, "e_A"'] \arrow[urr, bend right, "row"'] & \bullet \arrow[ur, tail, "m_A"'] \arrow[uu, dotted, "\exists i"] &
\end{tikzcd}
\qquad
\begin{tikzcd}
 & \bullet \arrow[r, tail, "m"] \arrow[dd, dotted, "\exists j"] & 2^E \\
S \arrow[ur, two heads, "e"] \arrow[dr, two heads, "e_\Lambda"'] \arrow[urr, bend left, "row"] \arrow[drr, bend right, "\Lambda(row)"'] & & \\
 & \bullet \arrow[r, tail, "m_\Lambda"'] & (2^E)^A
\end{tikzcd}
$$

closed *consistent*

(S, E, row) is *consistent* if whenever $s_1, s_2 \in S$ are such that $row(s_1) = row(s_2)$, for all $a \in A$, $row(s_1 a) = row(s_2 a)$.

The L^* algorithm: the observation table

$$
\begin{tikzcd}
S \arrow[r, two heads, "e"] \arrow[drr, bend left, "row"] & \bullet \arrow[dr, tail, "m"] & \\
& & 2^E \\
S \cdot A \arrow[r, two heads, "e_A"'] \arrow[urr, bend right, "row"'] & \bullet \arrow[uu, dotted, "\exists i"] \arrow[ur, tail, "m_A"'] &
\end{tikzcd}
$$

closed

$$
\begin{tikzcd}
& \bullet \arrow[r, tail, "m"] \arrow[dd, dotted, "\exists j"] & 2^E \\
S \arrow[ur, two heads, "e"] \arrow[dr, two heads, "e_\Lambda"'] \arrow[urr, bend left, "row"] \arrow[drr, bend right, "\Lambda(row)"'] & & \\
& \bullet \arrow[r, tail, "m_\Lambda"'] & (2^E)^A
\end{tikzcd}
$$

consistent

The L^* algorithm: from table to automaton

Closed and consistent table (S, E) to DFA (Q, q_0, δ, F):

- Q is a finite set of states: $Q = \{row(s) \mid s \in S\}$.

The L^* algorithm: from table to automaton

Closed and consistent table (S, E) to DFA (Q, q_0, δ, F):

- Q is a finite set of states: $Q = \{row(s) \mid s \in S\}$.
- $F \subseteq Q$ is a set of final states:
  $F = \{row(s) \mid s \in S, row(s)(\lambda) = 1\}$.

The L^* algorithm: from table to automaton

Closed and consistent table (S, E) to DFA (Q, q_0, δ, F):

- Q is a finite set of states: $Q = \{row(s) \mid s \in S\}$.
- $F \subseteq Q$ is a set of final states:
  $F = \{row(s) \mid s \in S, row(s)(\lambda) = 1\}$.
- $q_0 \in Q$ is the initial state: $q_0 = row(\lambda)$.

The L^* algorithm: from table to automaton

Closed and consistent table (S, E) to DFA (Q, q_0, δ, F):

- Q is a finite set of states: $Q = \{row(s) \mid s \in S\}$.
- $F \subseteq Q$ is a set of final states:
  $F = \{row(s) \mid s \in S, row(s)(\lambda) = 1\}$.
- $q_0 \in Q$ is the initial state: $q_0 = row(\lambda)$.
- $\delta \colon Q \times A \to Q$ is the transition function:
  $\delta(row(s), a) = row(sa)$.

The L^* algorithm: from table to automaton

Closed and consistent table (S, E) to DFA (Q, q_0, δ, F):

- Q is a finite set of states: $Q = \{row(s) \mid s \in S\}$.
- $F \subseteq Q$ is a set of final states:
  $F = \{row(s) \mid s \in S, row(s)(\lambda) = 1\}$.
- $q_0 \in Q$ is the initial state: $q_0 = row(\lambda)$.
- $\delta\colon Q \times A \to Q$ is the transition function:
  $\delta(row(s), a) = row(sa)$.

Note: well-definedness of automaton uses closed & consistent.

The L^* algorithm: from table to automaton

$$\begin{tikzcd} S \arrow[r, two heads, "e"] \arrow[d, "\varphi"'] & Q \arrow[d, "\psi"] \arrow[dl, dashed, "\delta"'] \\ Q^A \arrow[r, tail, "m^A"] & (2^E)^A \end{tikzcd}$$

where $\begin{cases} \varphi = \Lambda(i \circ e_A) \\ \psi = m_\Lambda \circ j. \end{cases}$

Another butterfly!

$$\begin{tikzcd}
1 \arrow[d, "\lambda"'] \arrow[dr, "init"] & & 2 \\
S \arrow[r, two heads, "e"] \arrow[rr, bend right, "row"'] & Q \arrow[r, tail, "m"] \arrow[ur, "final"] \arrow[d, "\delta"'] & 2^E \arrow[u, "ev_\lambda"'] \\
& Q^A &
\end{tikzcd}$$

Another butterfly!

$$
\begin{tikzcd}
1 \arrow[d, "\lambda"'] \arrow[dr, "\mathit{init}"] & & 2 \\
S \arrow[r, two heads, "e"] \arrow[rr, bend right, "\mathit{row}"'] & Q \arrow[r, tail, "m"] \arrow[ur, "\mathit{final}"] \arrow[d, "\delta"'] & 2^E \arrow[u, "ev_\lambda"'] \\
& Q^A &
\end{tikzcd}
$$

Theorem

The automaton associated with a closed and consistent observation table is minimal.

Proof of minimality: the usual butterfly!

$$
\begin{tikzcd}
1 \arrow[d, "\lambda"'] \arrow[dr, "init"] & & 2 \\
A^* \arrow[r, dashed, "r"'] \arrow[d, "c"'] & Q \arrow[ur, "final"] \arrow[r, dashed, "o"'] \arrow[d, "\delta"] & 2^{A*} \arrow[u, "ev_\lambda"'] \arrow[d, "\partial"] \\
(A^*)^A \arrow[r, dashed, "r^A"'] & Q^A \arrow[r, dashed, "o^A"'] & (2^{A*})^A
\end{tikzcd}
$$

The L^* algorithm: learning the table

function LEARNER
  $S \leftarrow \{\lambda\}$; $E \leftarrow \{\lambda\}$.
  repeat
    while (S, E) is not closed or not consistent **do**
      if (S, E) is not consistent **then**
        find $s_1, s_2 \in S$, $a \in A$, and $e \in E$ such that
          $row(s_1) = row(s_2)$ and $\mathcal{L}(s_1 ae) \neq \mathcal{L}(s_2 ae)$
        $E \leftarrow E \cup \{ae\}$.
      end if
      if (S, E) is not closed **then**
        find $s_1 \in S$, $a \in A$ such that
          $row(s_1 a) \neq row(s)$, for all $s \in S$
        $S \leftarrow S \cup \{s_1 a\}$.
      end if
    end while
    Make the conjecture $M(S, E)$.
    if the Teacher replies **no** to the conjecture, with a counter-example t **then**
      $S \leftarrow S \cup \downarrow t$.
    end if
  until the Teacher replies **yes** to the conjecture $M(S, E)$.
  return $M(S, E)$.
end function

The L^* algorithm: example

$$\mathcal{L} = \{u \in \{a, b\}^* \mid \text{the number of } a\text{'s in } u \text{ is divisible by } 3\}.$$

The L^* algorithm: example

$\mathcal{L} = \{u \in \{a,b\}^* \mid$ the number of a's in u is divisible by $3\}$.

$$
\begin{array}{rc|c}
 & & \lambda \\
\hline
S\,\{ & \lambda & 1 \\
\hline
\multirow{2}{*}{$S \cdot A$} & a & 0 \\
 & b & 1
\end{array}
$$

(S, E) consistent? ✓

The L^* algorithm: example

$$\mathcal{L} = \{u \in \{a,b\}^* \mid \text{the number of } a\text{'s in } u \text{ is divisible by } 3\}.$$

$$\begin{array}{r|c|c} & & \lambda \\ \hline S\ \{ & \lambda & 1 \\ \hline S \cdot A \left\{ \begin{array}{l} \\ \\ \end{array} \right. & \begin{array}{c} a \\ b \end{array} & \begin{array}{c} 0 \\ 1 \end{array} \end{array}$$

(S, E) consistent? ✓

(S, E) closed? No.

The L^* algorithm: example

$$\mathcal{L} = \{u \in \{a, b\}^* \mid \text{the number of } a\text{'s in } u \text{ is divisible by } 3\}.$$

$$\begin{array}{r|c|c}
 & & \lambda \\ \hline
S & \lambda & 1 \\ \hline
S \cdot A & a & 0 \\
 & b & 1
\end{array}$$

(S, E) consistent? ✓
(S, E) closed? No.

if (S, E) is not closed **then**
    find $s_1 \in S$, $a \in A$ such that
        $row(s_1 a) \neq row(s)$, for all $s \in S$
    $S \leftarrow S \cup \{s_1 a\}$.
end if

The L^* algorithm: example

$$\mathcal{L} = \{u \in \{a, b\}^* \mid \text{the number of } a\text{'s in } u \text{ is divisible by } 3\}.$$

$$\begin{array}{r|c|c}
 & & \lambda \\ \hline
S\ \{ & \lambda & 1 \\ \hline
S \cdot A\ \Bigg\{ & \begin{array}{c} a \\ b \end{array} & \begin{array}{c} 0 \\ 1 \end{array}
\end{array}$$

(S, E) consistent? ✓
(S, E) closed? No.
Then, $S \leftarrow S \cup \{a\}$.

```
if (S, E) is not closed then
    find s_1 ∈ S, a ∈ A such that
        row(s_1 a) ≠ row(s), for all s ∈ S
    S ← S ∪ {s_1 a}.
end if
```

The L^* algorithm: example

	λ
λ	1
a	0
b	1
aa	0
ab	0

The L^* algorithm: example

(S, E) consistent? ✓
(S, E) closed? ✓

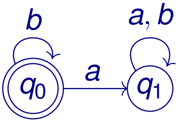
	λ
λ	1
a	0
b	1
aa	0
ab	0

The L^* algorithm: example

(S, E) consistent? ✓
(S, E) closed? ✓

	λ
λ	1
a	0
b	1
aa	0
ab	0

Guess



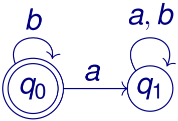
$$q_0 = row(\lambda)$$
$$q_1 = row(a)$$

The L^* algorithm: example

(S, E) consistent? ✓
(S, E) closed? ✓

	λ
λ	1
a	0
b	1
aa	0
ab	0

Guess



$$q_0 = row(\lambda)$$
$$q_1 = row(a)$$

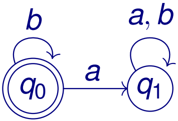
Teacher replies with counter-example *aaa*.

The L^* algorithm: example

(S, E) consistent? ✓
(S, E) closed? ✓

	λ
λ	1
a	0
b	1
aa	0
ab	0

Guess



$$q_0 = row(\lambda)$$
$$q_1 = row(a)$$

Teacher replies with counter-example aaa.

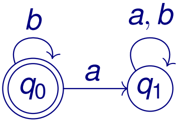
1: Make the conjecture $M(S, E)$.
2: **if** the Teacher replies **no** to the conjecture, with a counter-example t **then**
3: $S \leftarrow S \cup \downarrow t$.
4: **end if**

The L^* algorithm: example

(S, E) consistent? ✓
(S, E) closed? ✓

	λ
λ	1
a	0
b	1
aa	0
ab	0

Guess



$$q_0 = row(\lambda)$$
$$q_1 = row(a)$$

Teacher replies with counter-example aaa.
$S \leftarrow S \cup \{a, aa, aaa\}$.

Make the conjecture $M(S, E)$.
if the Teacher replies **no** to the conjecture, with a counter-example t **then**
    $S \leftarrow S \cup \downarrow t$.
end if

The L^* algorithm: example

	λ
λ	1
a	0
aa	0
aaa	1
b	1
ab	0
aab	0
$aaaa$	0
$aaab$	1

(S, E) consistent?

The L^* algorithm: example

	λ
λ	1
a	0
aa	0
aaa	1
b	1
ab	0
aab	0
$aaaa$	0
$aaab$	1

(S, E) consistent?
No, $row(a) = row(aa)$ but $row(aa) \neq row(aaa)$.

The L^* algorithm: example

	λ
λ	1
a	0
aa	0
aaa	1
b	1
ab	0
aab	0
$aaaa$	0
$aaab$	1

(S, E) consistent?
No, $row(a) = row(aa)$ but $row(aa) \neq row(aaa)$.

1: **if** (S, E) is not consistent **then**
2: find $s_1, s_2 \in S$, $a \in A$, and $e \in E$ such that
3: $row(s_1) = row(s_2)$ and $\mathcal{L}(s_1 ae) \neq \mathcal{L}(s_2 ae)$
4: $E \leftarrow E \cup \{ae\}$.
5: **end if**

The L^* algorithm: example

	λ
λ	1
a	0
aa	0
aaa	1
b	1
ab	0
aab	0
$aaaa$	0
$aaab$	1

(S, E) consistent?
No, $row(a) = row(aa)$ but $row(aa) \neq row(aaa)$.
$E \leftarrow E \cup \{a\}$.

if (S, E) is not consistent **then**
  find $s_1, s_2 \in S$, $a \in A$, and $e \in E$ such that
    $row(s_1) = row(s_2)$ and $\mathcal{L}(s_1 ae) \neq \mathcal{L}(s_2 ae)$
  $E \leftarrow E \cup \{ae\}$.
end if

The L^* algorithm: example

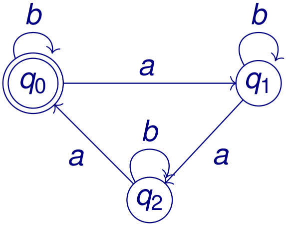
	λ	a
λ	1	0
a	0	0
aa	0	1
aaa	1	0
b	1	0
ab	0	0
aab	0	1
$aaaa$	0	0
$aaab$	1	0

(S, E) consistent? ✓
(S, E) closed? ✓

The L^* algorithm: example

	λ	a
λ	1	0
a	0	0
aa	0	1
aaa	1	0
b	1	0
ab	0	0
aab	0	1
$aaaa$	0	0
$aaab$	1	0

(S, E) consistent? ✓
(S, E) closed? ✓
Second guess:



The teacher replies **yes.**

The generalizations

- Table, automaton, proof of minimality: independent of output set.

$$\mathcal{L}\colon A^* \to 2 \qquad\qquad \mathcal{L}\colon A^* \to B$$

- Change in functor: Moore and Mealy machines.

The generalizations

- Table, automaton, proof of minimality: independent of output set.

$$\mathcal{L}\colon A^* \to 2 \qquad\qquad \mathcal{L}\colon A^* \to B$$

- Change in functor: Moore and Mealy machines.
- Category with factorization structure.
- Change in category: linear weighted automata.

Examples in the paper.

Conclusions

- Trivial but yet insightful (at least for Bart and me ;-)) categorical understanding of Angluin's algorithm.
- Mealy example: several papers justifying it.
- Applications of learning are vast, rich playground and source of examples.
- Future work: learning from incomplete information, heuristics, ...

Conclusions

- Trivial but yet insightful (at least for Bart and me ;-)) categorical understanding of Angluin's algorithm.
- Mealy example: several papers justifying it.
- Applications of learning are vast, rich playground and source of examples.
- Future work: learning from incomplete information, heuristics, . . .
- *Category Theory does have something to say about learning!*

Conclusions

- Trivial but yet insightful (at least for Bart and me ;-)) categorical understanding of Angluin's algorithm.
- Mealy example: several papers justifying it.
- Applications of learning are vast, rich playground and source of examples.
- Future work: learning from incomplete information, heuristics, ...
- *Category Theory does have something to say about learning!*

Happy birthday!