QEST + FORMATS Artifacts Evaluation Available

QEST + FORMATS Artifacts Evaluation Functional

京都大学 KYOTO UNIVERSITY

Learning Alternating Real-Time Automata

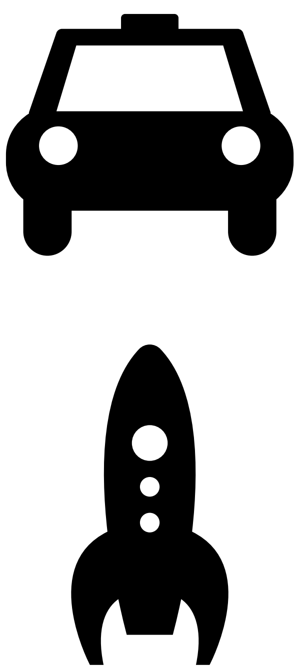
Kazuki Kinoshita[1], **Masaki Waga**[1,2]

Kyoto University[1], National Institute of Informatics[2]

2nd September 2026, QEST+FORMATS 2026

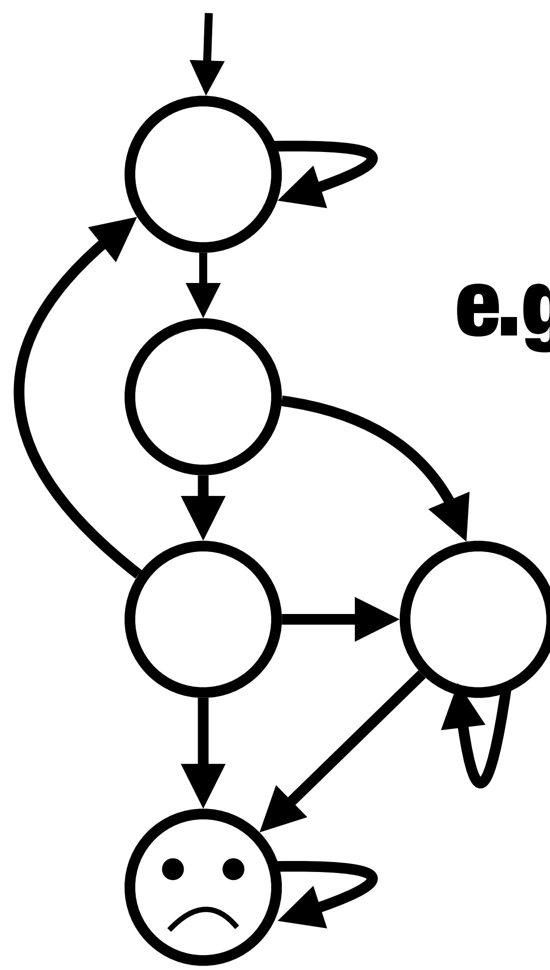
This work is partially supported by the JST BOOST Grant No. JPMJBY24H8, JST PRESTO Grant No. JPMJPR22CA, and JSPS KAKENHI Grant No. 22K17873

Formal modeling is essential



⋮

Formal Modeling



Verification
e.g. Model Checking

✔

✘

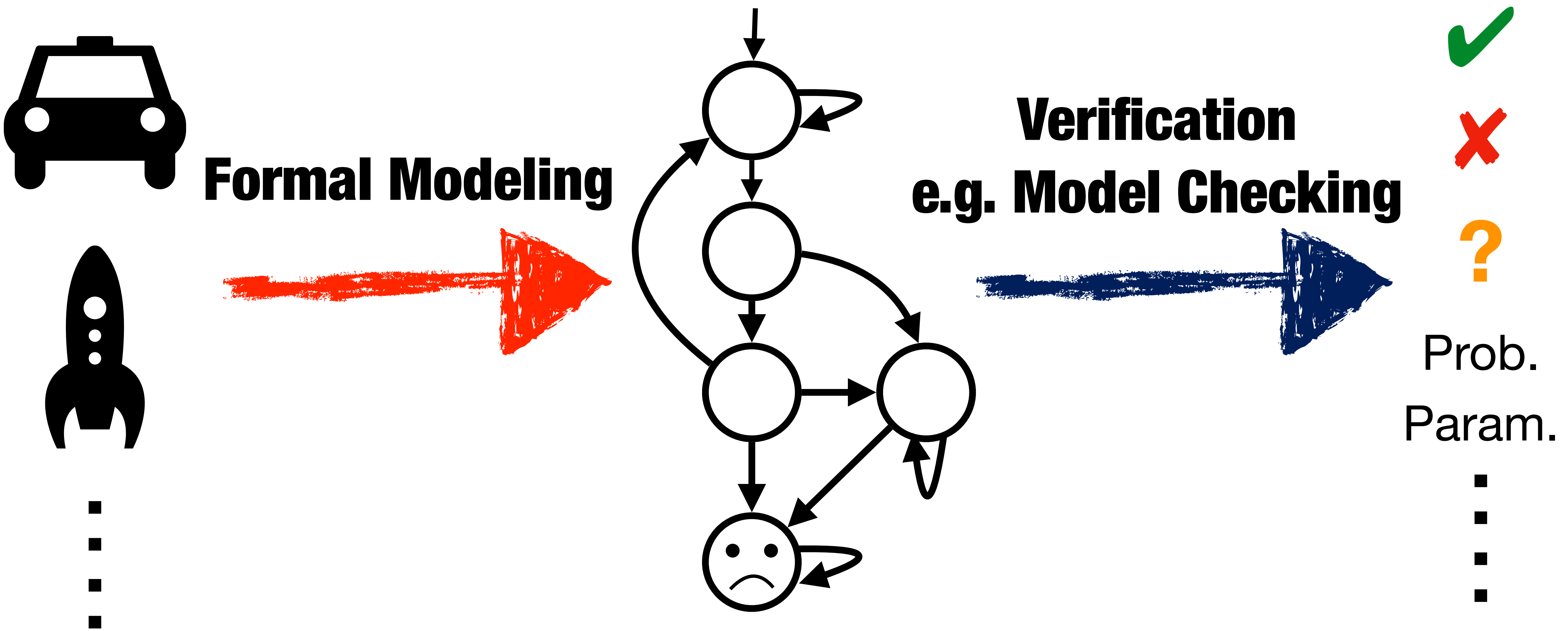
?

Prob.

Param.

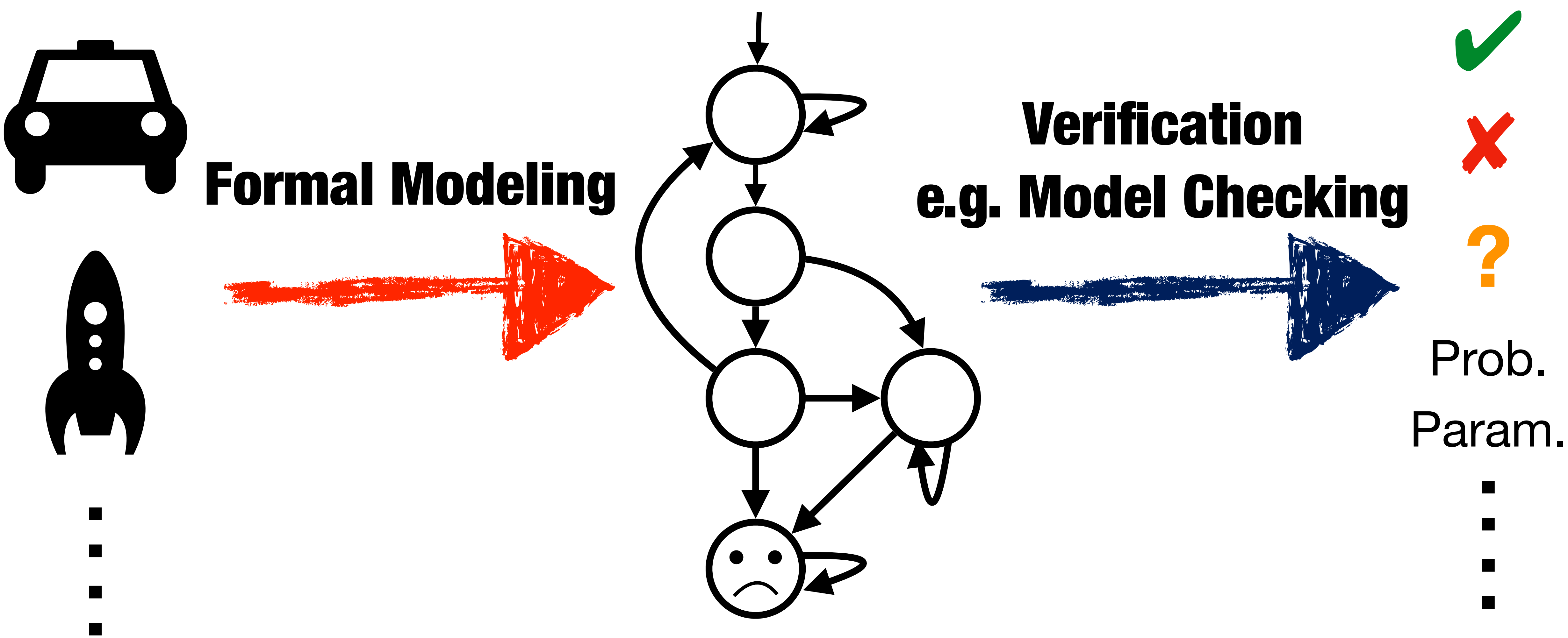
⋮

Formal modeling is essential **but usually difficult**



Formal modeling is essential but usually difficult

Q. How about automatically learn a formal model?



Active DFA Learning

[Angluin, Inform Comput'87]

Membership

Equivalence

Questions: $w \overset{?}{\in} L_{\mathrm{tgt}},\ \mathscr{L}(\mathscr{A}_{\mathrm{hyp}}) \overset{?}{=} L_{\mathrm{tgt}}$

Answers: Yes, No, Evidence $w \in \mathscr{L}(\mathscr{A}_{\mathrm{hyp}}) \triangle L_{\mathrm{tgt}}$

Active DFA Learning

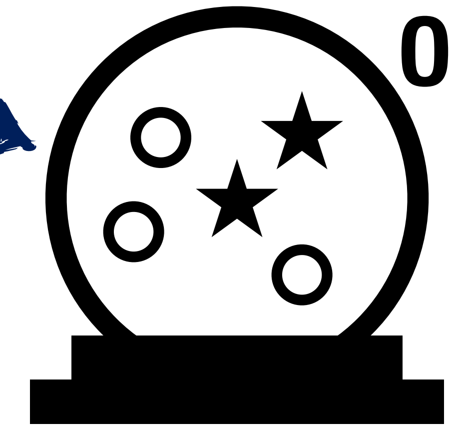
[Angluin, Inform Comput'87]

Membership **Equivalence**

Questions: $w \overset{?}{\in} L_{\mathrm{tgt}},\ \mathscr{L}(\mathscr{A}_{\mathrm{hyp}}) \overset{?}{=} L_{\mathrm{tgt}}$

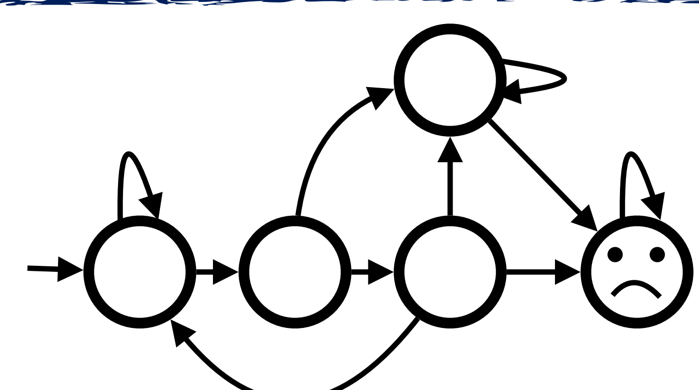
Learner

Oracle



Answers: Yes, No, Evidence $w \in \mathscr{L}(\mathscr{A}_{\mathrm{hyp}}) \bigtriangleup L_{\mathrm{tgt}}$

is modeled by



Active DFA Learning

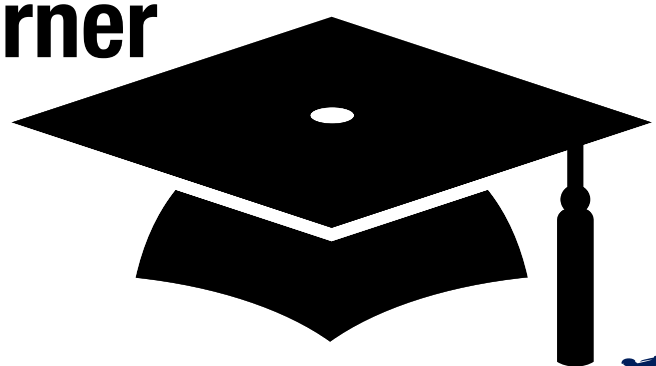
[Angluin, Inform Comput'87]

Membership

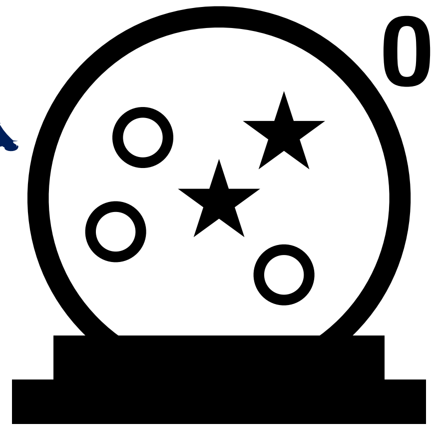
Equivalence

Questions: $w \overset{?}{\in} L_{\mathrm{tgt}}$, $\mathscr{L}(\mathscr{A}_{\mathrm{hyp}}) \overset{?}{\doteq} L_{\mathrm{tgt}}$

Learner

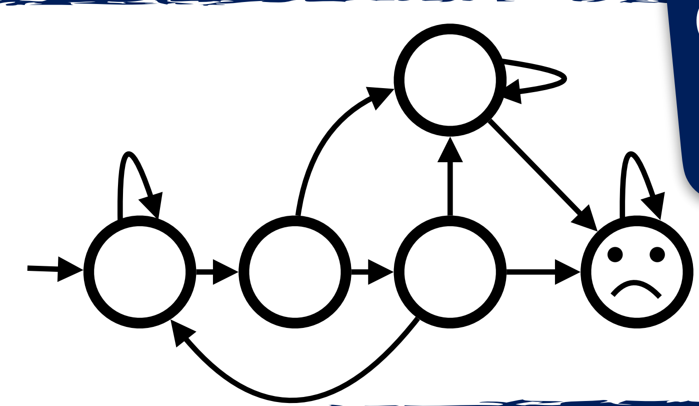


Oracle



Answers: Yes, No, Evidence $w \in \mathscr{L}(\mathscr{A}_{\mathrm{hyp}}) \bigtriangleup L_{\mathrm{tgt}}$

is modeled by



Correctness with finite queries

Active DFA Learning

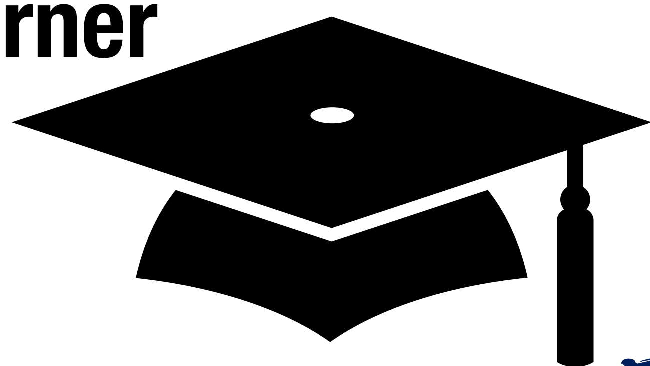
[Angluin, Inform Comput'87]

Membership

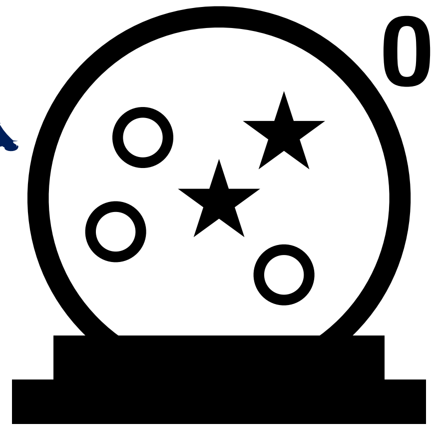
Equivalence

Questions: $w \overset{?}{\in} L_{\mathrm{tgt}},\ \mathscr{L}(\mathscr{A}_{\mathrm{hyp}}) \overset{?}{=} L_{\mathrm{tgt}}$

Learner

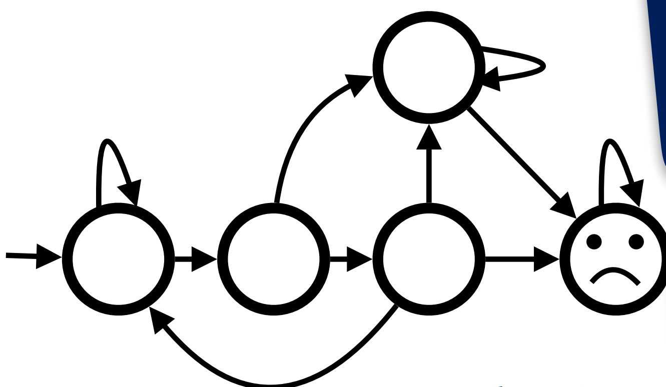


Oracle



Answers: Yes, No, Evidence $w \in \mathscr{L}(\mathscr{A}_{\mathrm{hyp}}) \triangle L_{\mathrm{tgt}}$

is modeled by



Correctness with finite queries

Maybe huge
No timing info.

Q. Can we learn succinct automata with time?

Learning nondeterministic or alternating automata
e.g. NL* [Bollig et al., IJCAI'09], AL* [Angluin et al., IJCAI'15]

Q. Can we learn succinct automata with time?

Learning nondeterministic or alternating automata
e.g. NL* [Bollig et al., IJCAI'09], AL* [Angluin et al., IJCAI'15]

Q. Can we learn succinct automata with time?

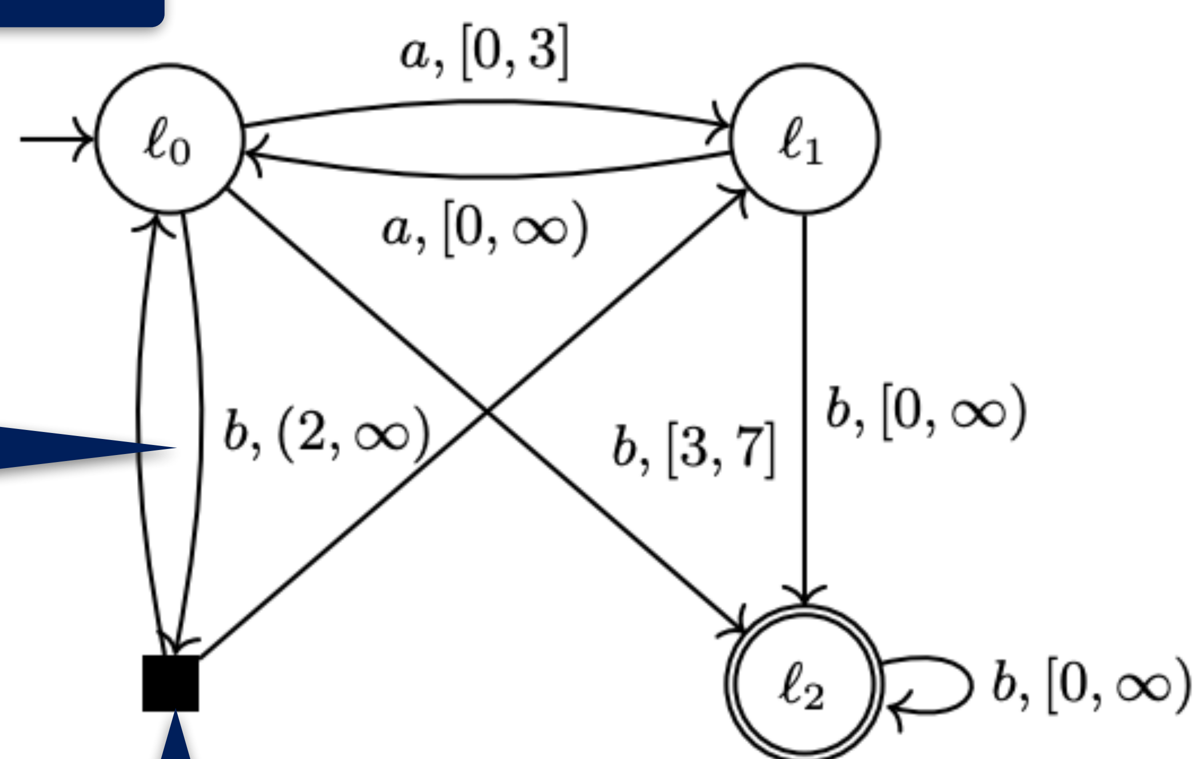
Learning a subclass of timed automata

Alternating Real-Time Automata

[Contribution]

ARTA: AFA + constraint on delays at each transition

“Boolean” branching rather than “or” branching in NFAs



Available if
Event: b
Delay: $d \in (2,\infty)$

“and” branching in alternating automata

Active ARTA Learning

[Contribution]

Timed word: Sequence of events + delay
e.g. (a, 2.7), (a, 3.1), (b, 2.3), (b, 3.2)

ARTA

Questions: $w \overset{?}{\in} L_{\mathrm{tgt}}, \ \mathscr{L}(\mathscr{A}_{\mathrm{hyp}}) \overset{?}{=} L_{\mathrm{tgt}}$

Learner

Oracle

Answers: Yes, No, Evidence $w \in \mathscr{L}(\mathscr{A}_{\mathrm{hyp}}) \triangle L_{\mathrm{tgt}}$

Active ARTA Learning

[Contribution]

Timed word: Sequence of events + delay
e.g. (a, 2.7), (a, 3.1), (b, 2.3), (b, 3.2)

ARTA

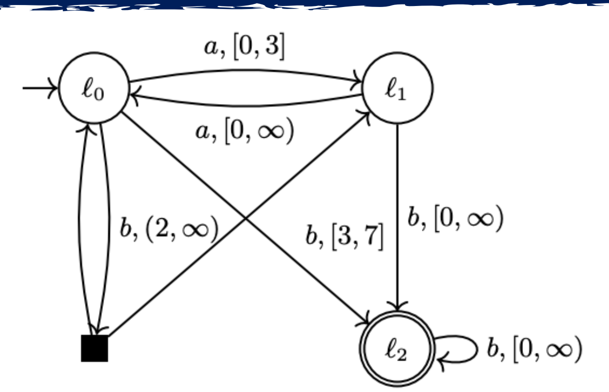
Questions: $w \stackrel{?}{\in} L_{\mathrm{tgt}},\ \mathscr{L}(\mathscr{A}_{\mathrm{hyp}}) \stackrel{?}{=} L_{\mathrm{tgt}}$

Learner

Oracle

Answers: Yes, No, Evidence $w \in \mathscr{L}(\mathscr{A}_{\mathrm{hyp}}) \triangle L_{\mathrm{tgt}}$

is modeled by



Active ARTA Learning

[Contribution]

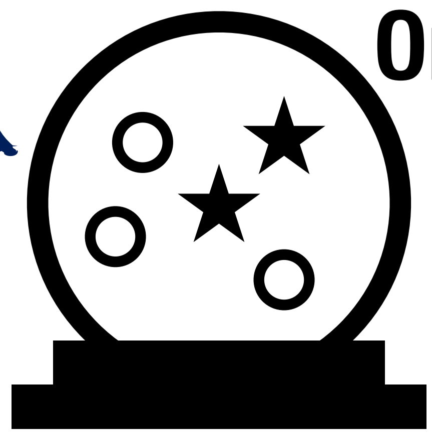
Timed word: Sequence of events + delay
e.g. (a, 2.7), (a, 3.1), (b, 2.3), (b, 3.2)

ARTA

Questions: $w \overset{?}{\in} L_{\mathrm{tgt}},\ \mathscr{L}(\mathscr{A}_{\mathrm{hyp}}) \overset{?}{=} L_{\mathrm{tgt}}$

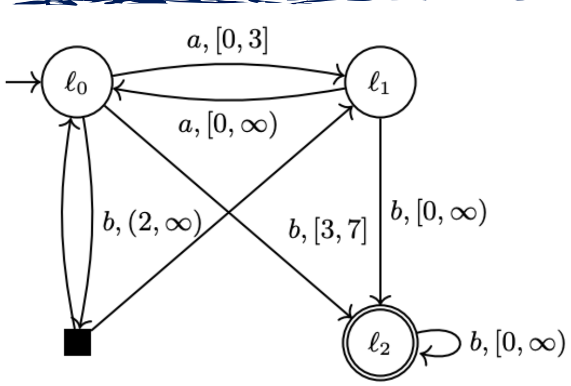
Learner

Oracle



Answers: Yes, No, Evidence $w \in \mathscr{L}(\mathscr{A}_{\mathrm{hyp}}) \bigtriangleup L_{\mathrm{tgt}}$

 **is modeled by**



Correctness with finite queries

Active ARTA Learning

[Contribution]

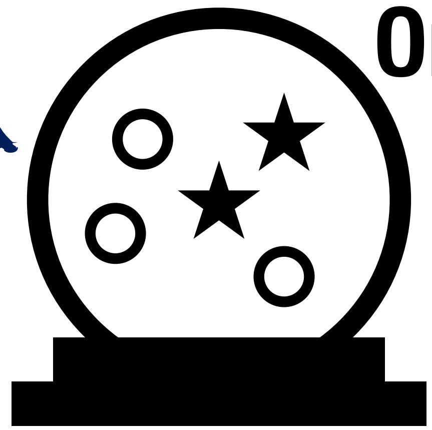
Timed word: Sequence of events + delay
e.g. (a, 2.7), (a, 3.1), (b, 2.3), (b, 3.2)

ARTA

Questions: $w \overset{?}{\in} L_{\mathrm{tgt}},\ \mathscr{L}(\mathscr{A}_{\mathrm{hyp}}) \overset{?}{=} L_{\mathrm{tgt}}$

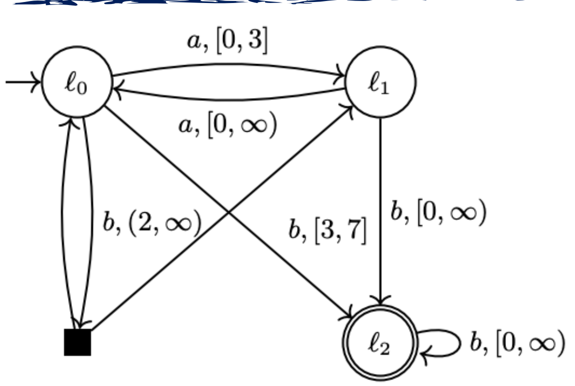
Learner

Oracle



Answers: Yes, No, Evidence $w \in \mathscr{L}(\mathscr{A}_{\mathrm{hyp}}) \triangle L_{\mathrm{tgt}}$

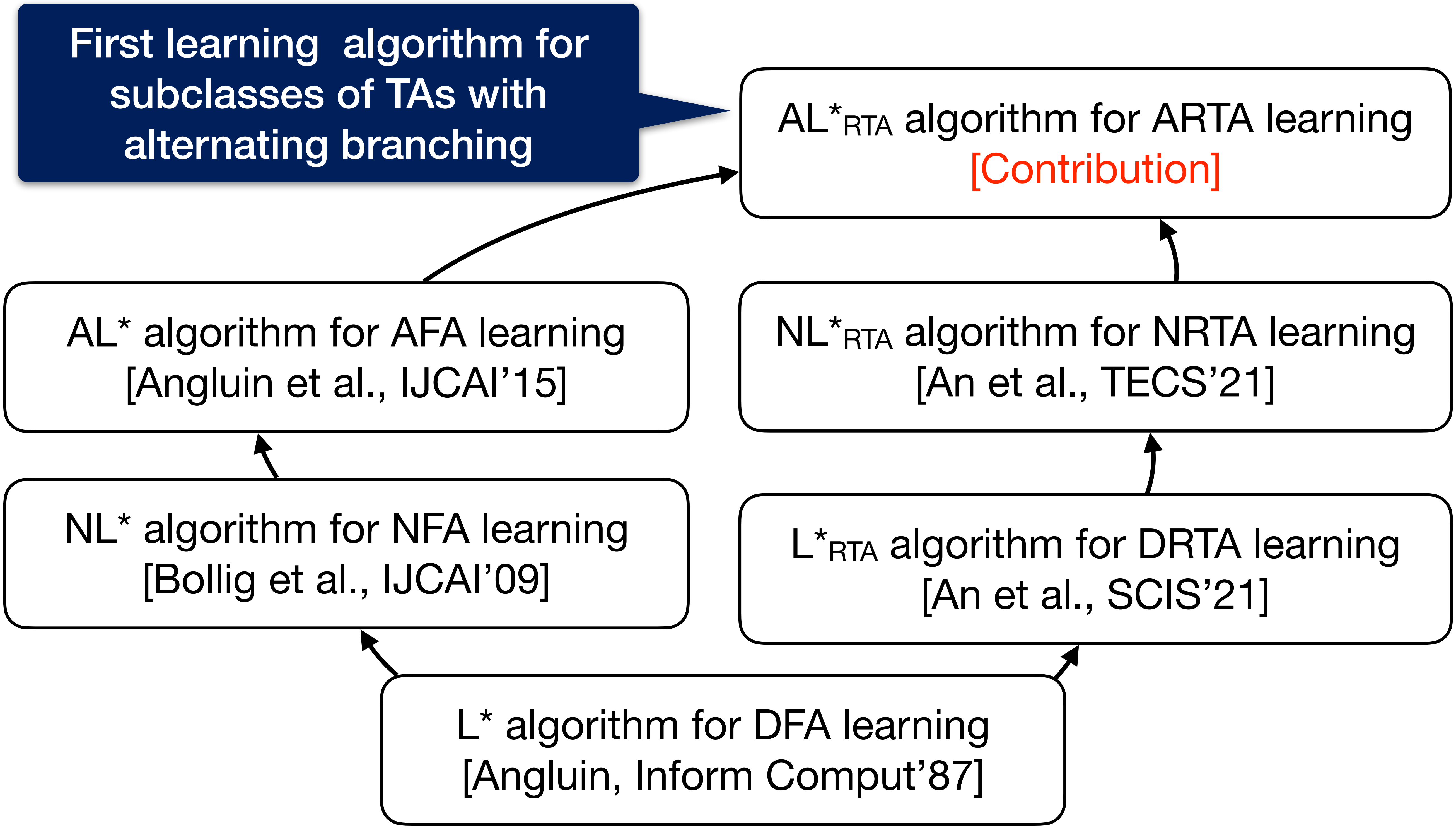
is modeled by



Correctness with finite queries

Succinct autom. with timing info.

Related Learning Algorithms



Contributions

- Show the succinctness theorem of ARTAs wrt DRTAs
  - Similar to the property of AFAs
- L*-style active learning algorithm for ARTAs
  - We call it AL^*_{RTA}
- Implementation + experiments
  → Tends to learn smaller automata than NL^*_{RTA}

Outline

- Formalism: Alternating real-time automata
  - Alternating branching + constraints on delays
- Existing learning algorithms
  - Learning alternating automata (AL*)
  - Learning real-time automata (L^*_{RTA})
- AL^*_{RTA} for active learning of ARTAs
- Experiments

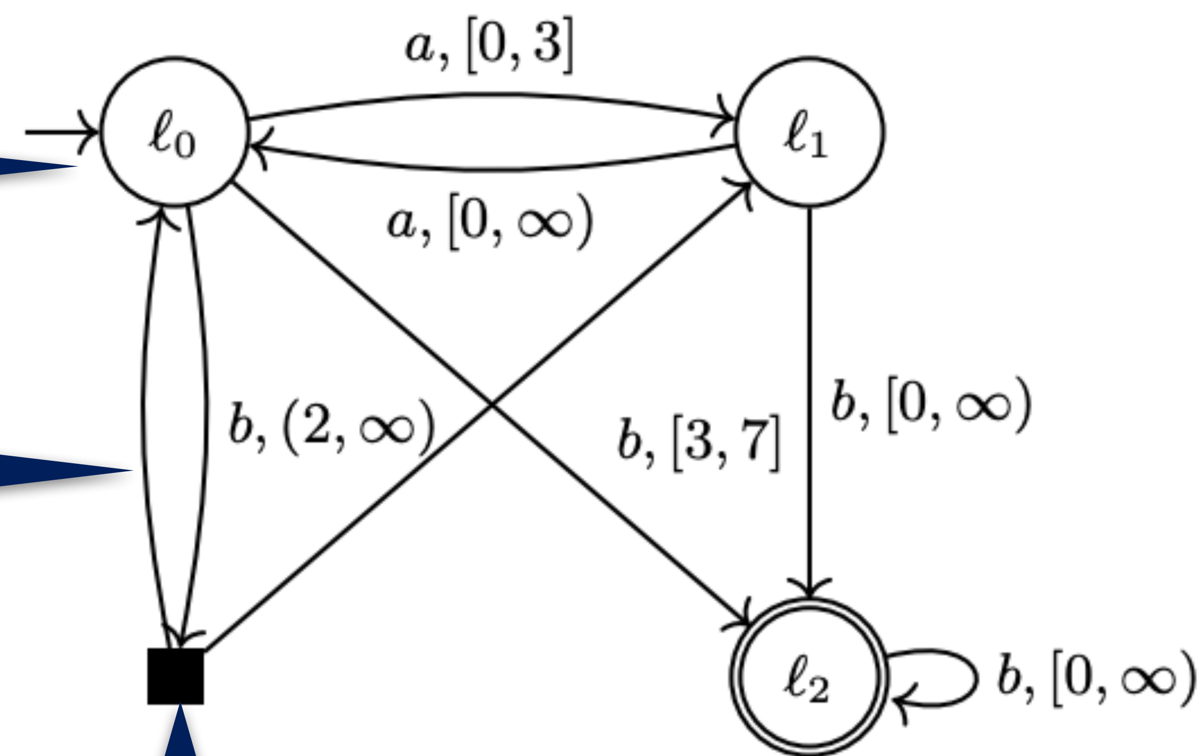
Alternating Real-Time Automata

[Contribution]

ARTA: AFA + constraint on delays at each transition

Successor: positive Boolean formula over states (rather than a state)

Initial state is also generalized to a positive Boolean formula



Available if
Event: b
Delay: $d \in (2,\infty)$

"and" branching in alternating automata

Alternating Real-Time Automata

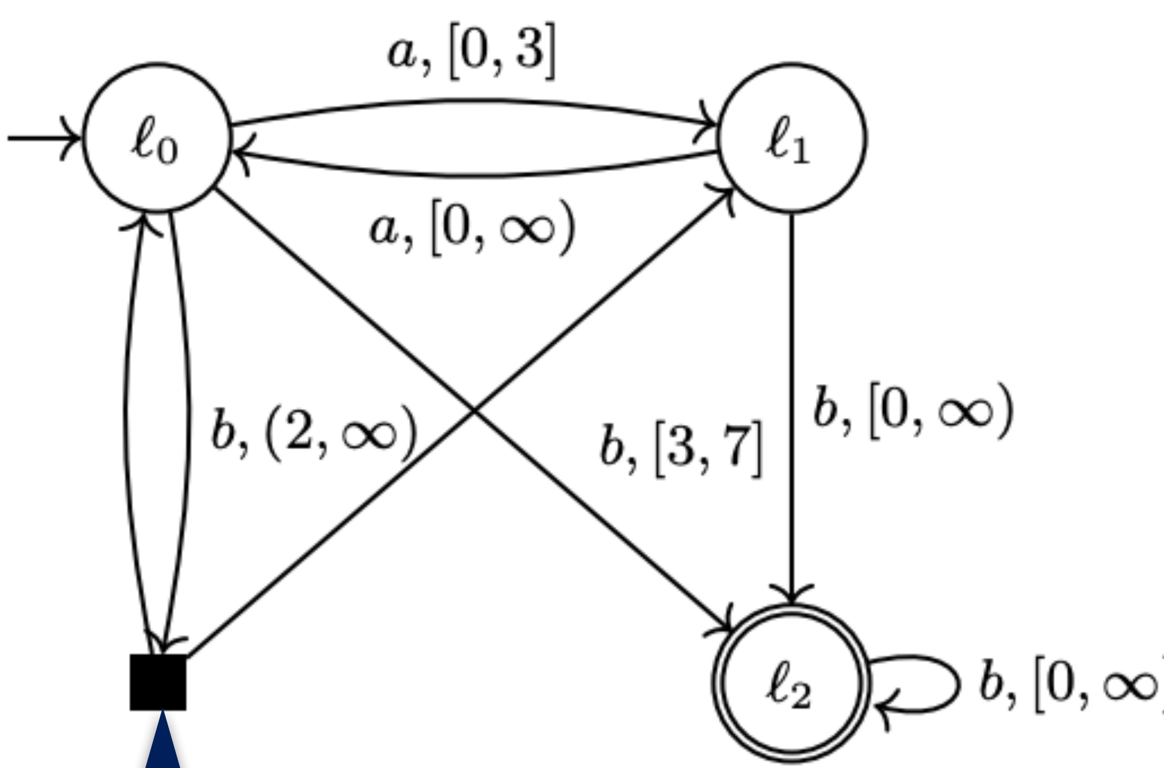
[Contribution]

ARTA: AFA + constraint on delays at each transition

Successor: positive Boolean formula over states (rather than a state)

- $(l_0, b, (2,3), l_0 \wedge l_1)$
- $(l_0, b, [3,7], l_2 \vee (l_0 \wedge l_1))$
- $(l_0, b, (7,\infty), l_0 \wedge l_1)$
- …

Interval with integer endpoints



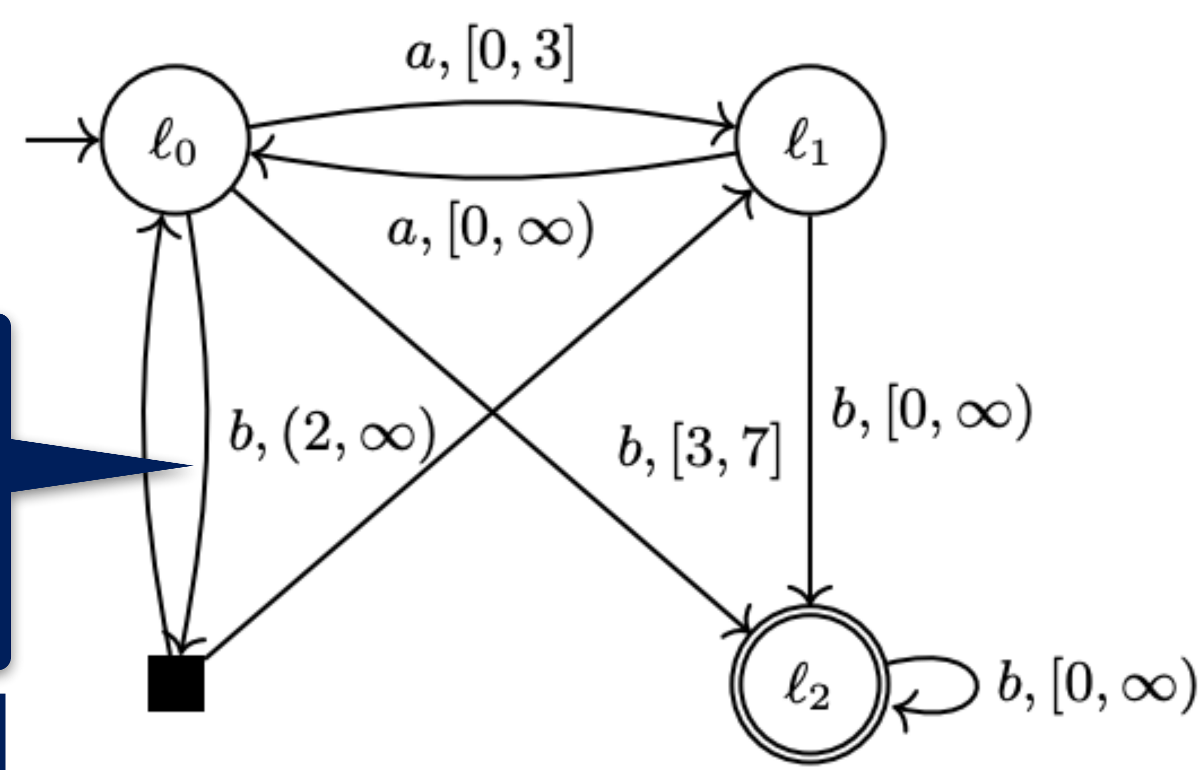
“and” branching in alternating automata

Region-based Abstraction for ARTAs

Idea: Normalize intervals with $(d, d+1)$, $[d, d]$, and (K, ∞)

- K: maximum constant in the guards of the ARTA
- ARTA → AFA by $\{(d, d+1) \mid d < K\} \cup \{[d, d] \mid d \leq K\} \cup \{(K, \infty)\}$
  - Using the alphabet $\Sigma \times Reg_K$
  - Encoding event + interval by a finite subset of $\Sigma \times Reg_K$
- This abstraction "preserves" the language

Encoded by
$(b, (2,3)), (b, [3,3]), \ldots, (b, (7,\infty))$
where $K = 7$



Properties on ARTAs

Expressiveness: ARTAs and deterministic RTAs have the same expressive power.

Sketch: Generalize the powerset construction for AFAs using a region-based abstraction.

Succinctness: There is a family of ARTAs that is doubly exponentially more succinct than the minimum deterministic RTAs.

Sketch: Generalize the succinctness of AFAs by mapping $a \in \Sigma$ to $(a, [0,\infty))$

ARTA learning can be useful for learning succinct automata

Outline

- Formalism: Alternating real-time automata
  - Alternating branching + constraints on delays
- Existing learning algorithms
  - Learning DFAs (L^*) and AFAs (AL^*)
  - Learning real-time automata (L^*_{RTA})
- AL^*_{RTA} for active learning of ARTAs
- Experiments

Active Automata Learning

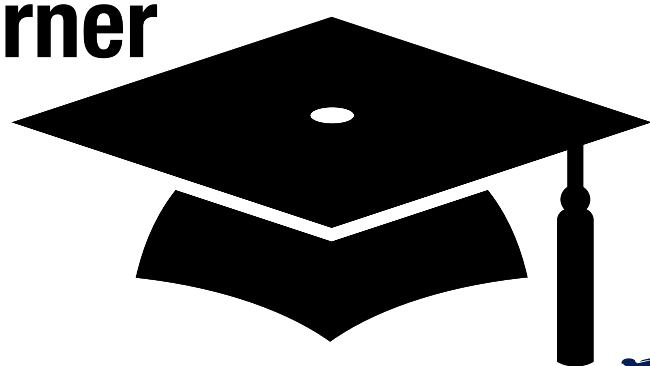
[Angluin, Inform Comput'87]

Membership

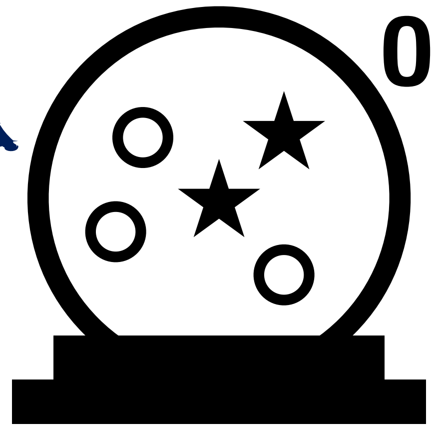
Equivalence

Questions: $w \overset{?}{\in} L_{\mathrm{tgt}}$, $\mathscr{L}(\mathscr{A}_{\mathrm{hyp}}) \overset{?}{=} L_{\mathrm{tgt}}$

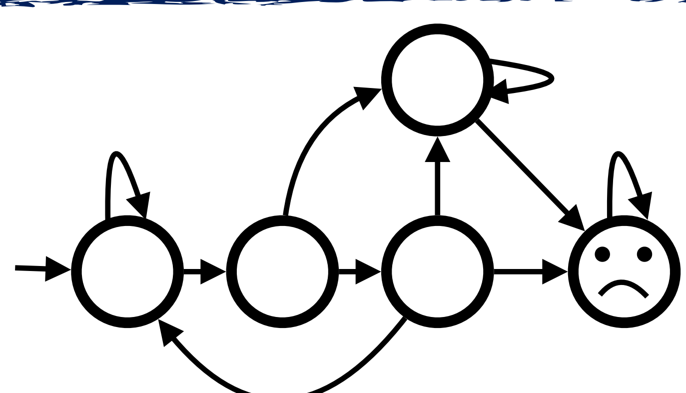
Learner



Oracle



Answers: Yes, No, Evidence $w \in \mathscr{L}(\mathscr{A}_{\mathrm{hyp}}) \triangle L_{\mathrm{tgt}}$

 **is modeled by** 

Active Automata Learning

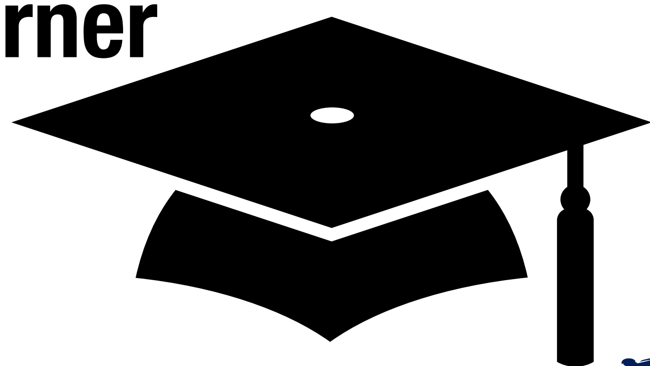
[Angluin, Inform Comput'87]

Membership

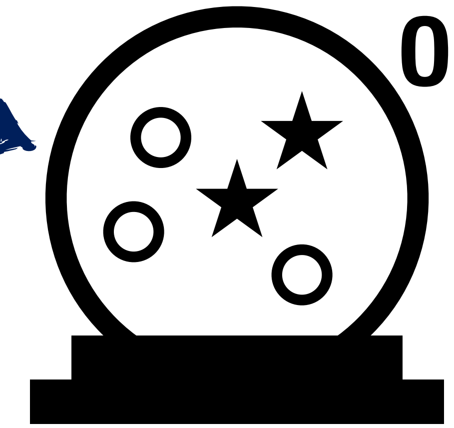
Equivalence

Questions: $w \overset{?}{\in} L_{\mathrm{tgt}},\ \mathscr{L}(\mathscr{A}_{\mathrm{hyp}}) \overset{?}{=} L_{\mathrm{tgt}}$

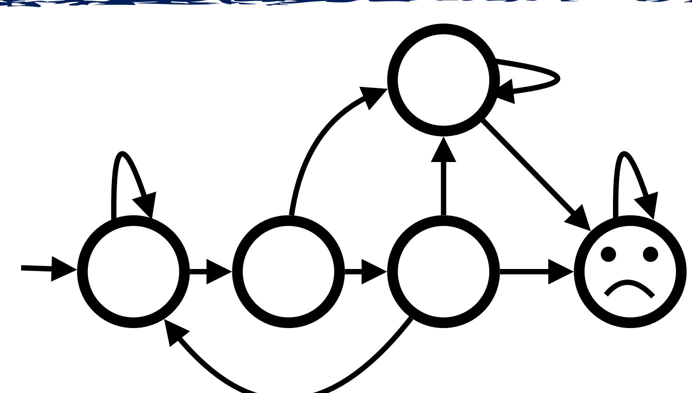
Learner



Oracle



Answers: Yes, No, Evidence $w \in \mathscr{L}(\mathscr{A}_{\mathrm{hyp}}) \triangle L_{\mathrm{tgt}}$

 **is modeled by** 

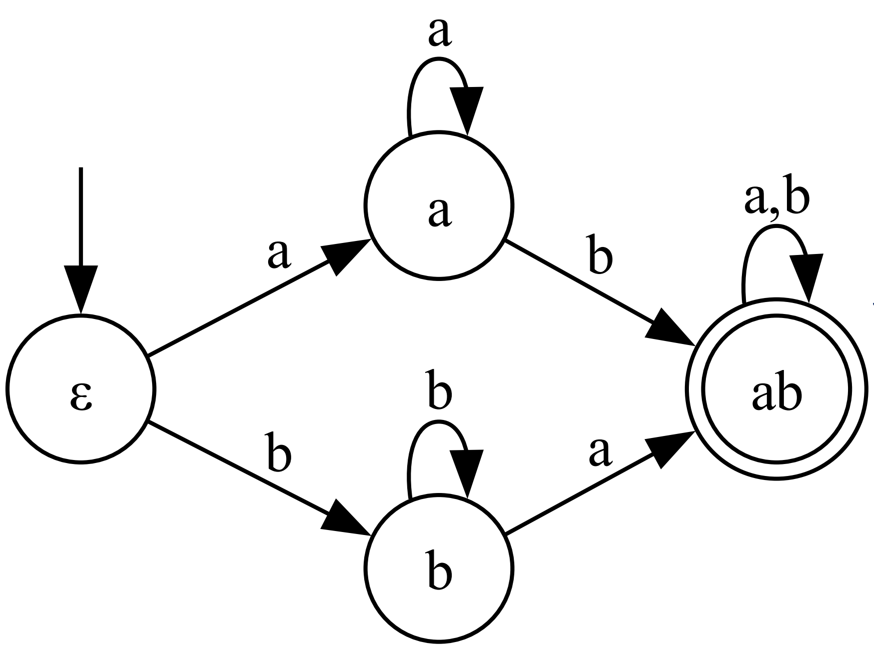
Language Identification

Prefixes to cover states

Suffixes to distinguish prefixes

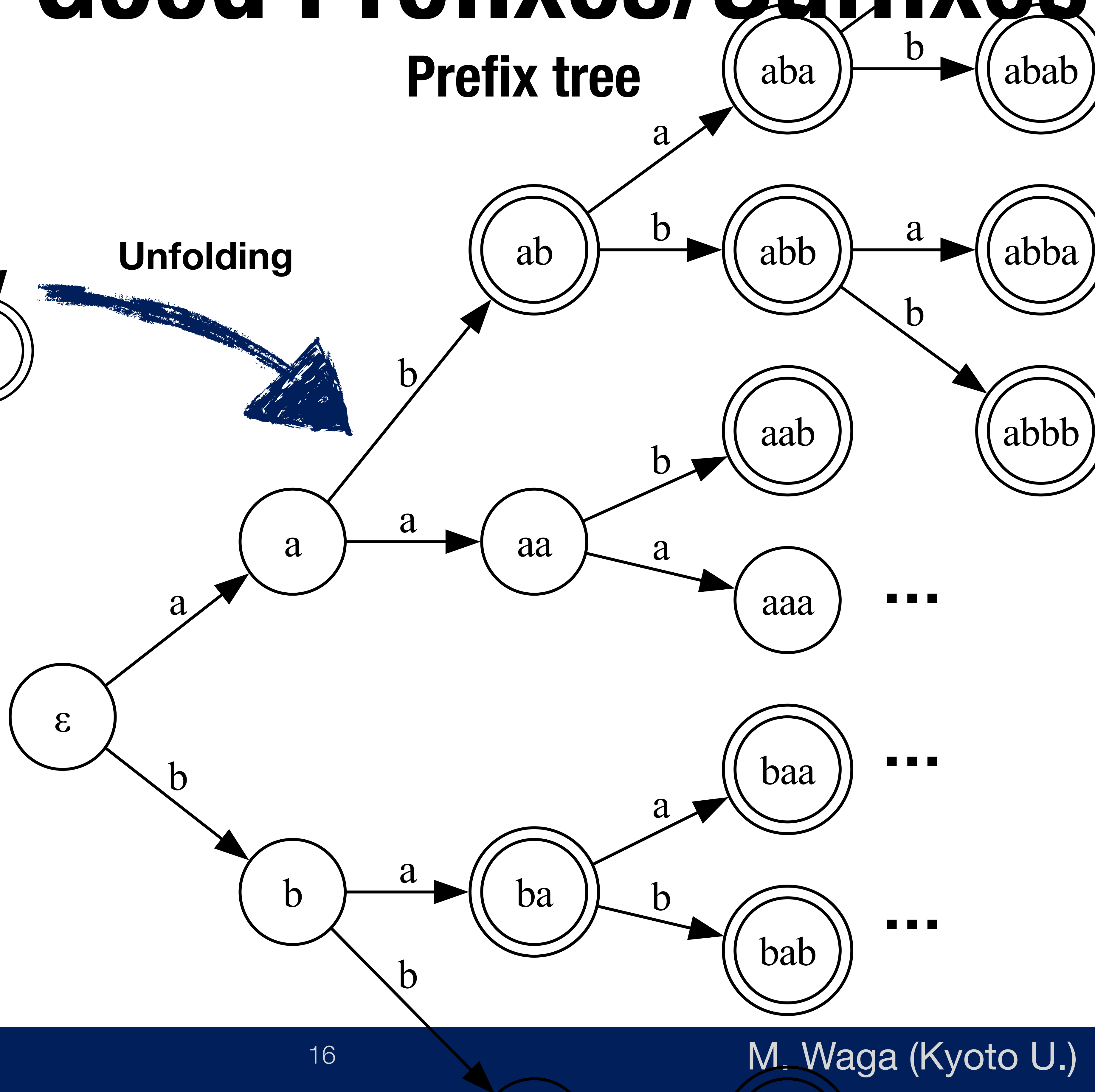
Idea: Find Good Prefixes/Suffixes

Target DFA

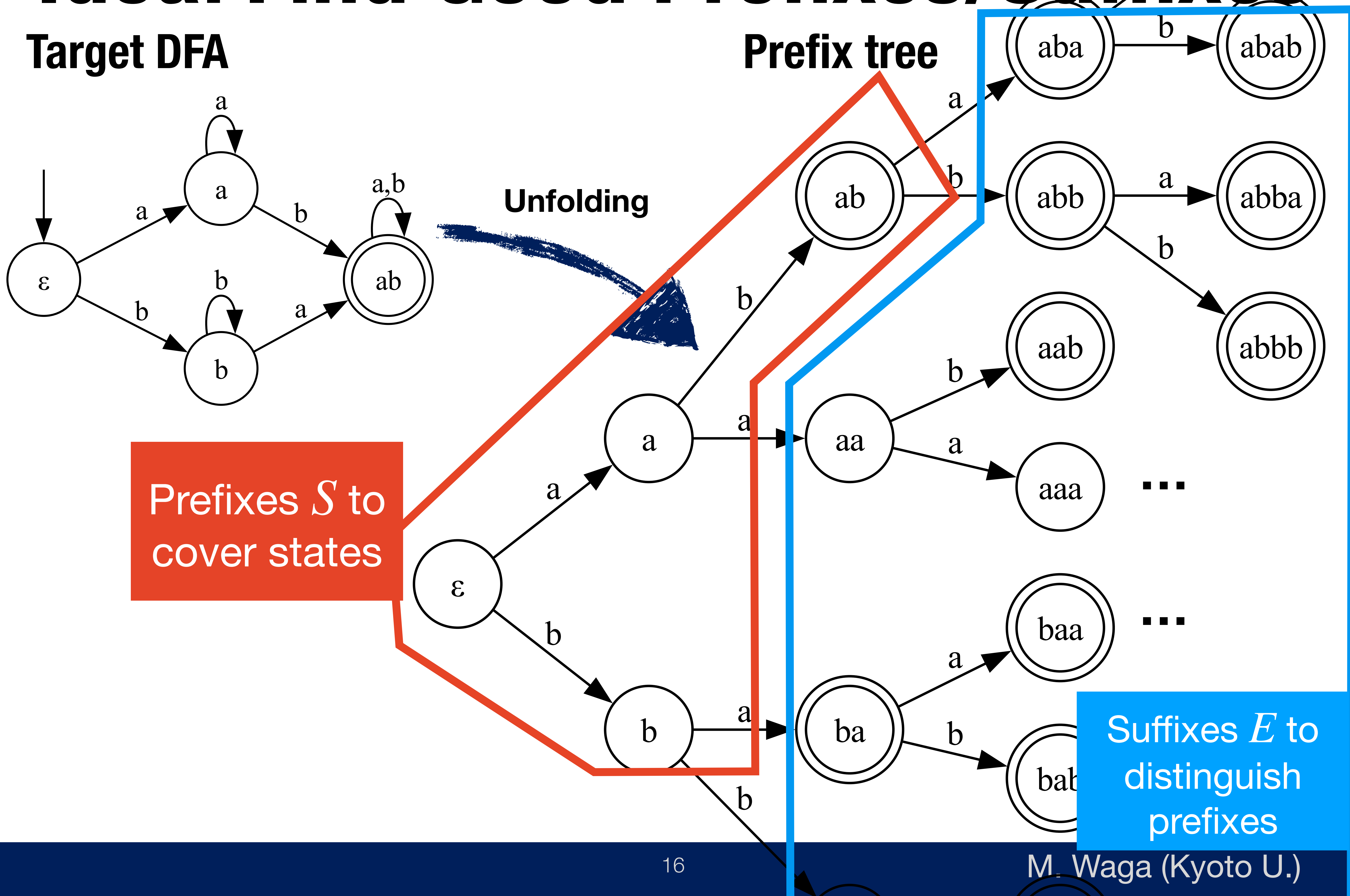


Unfolding

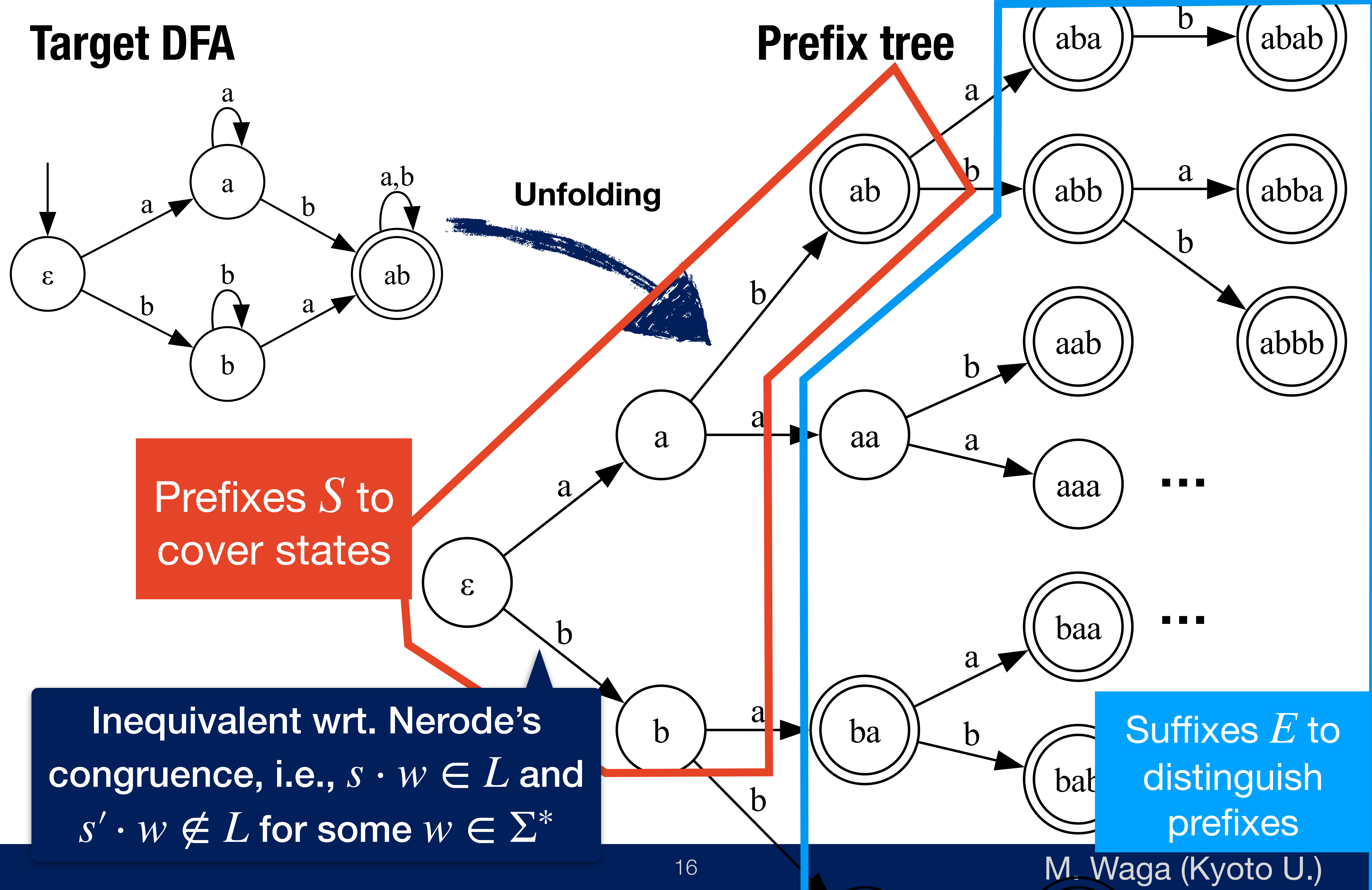
Prefix tree



Idea: Find Good Prefixes/Suffixes



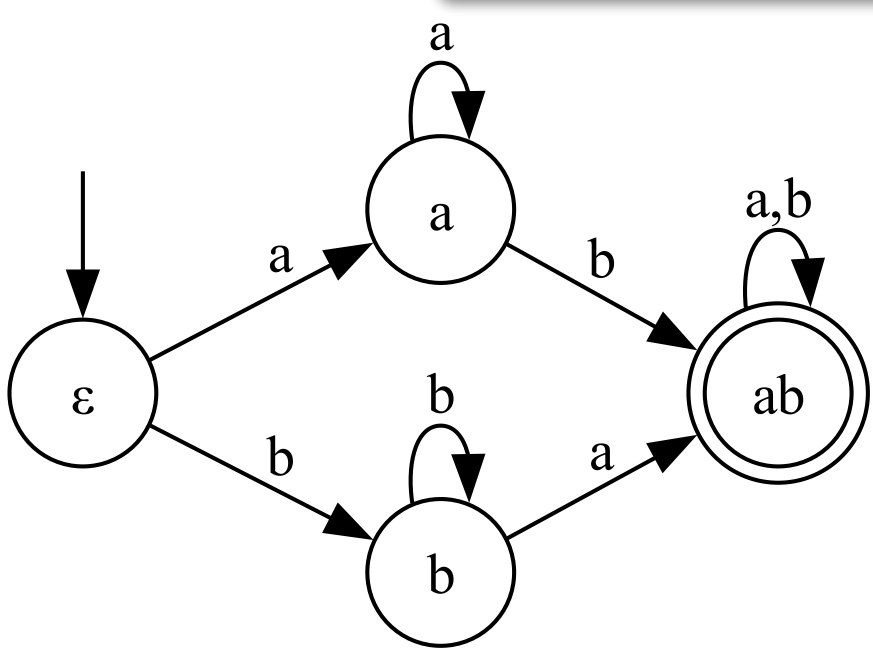
Idea: Find Good Prefixes/Suffixes



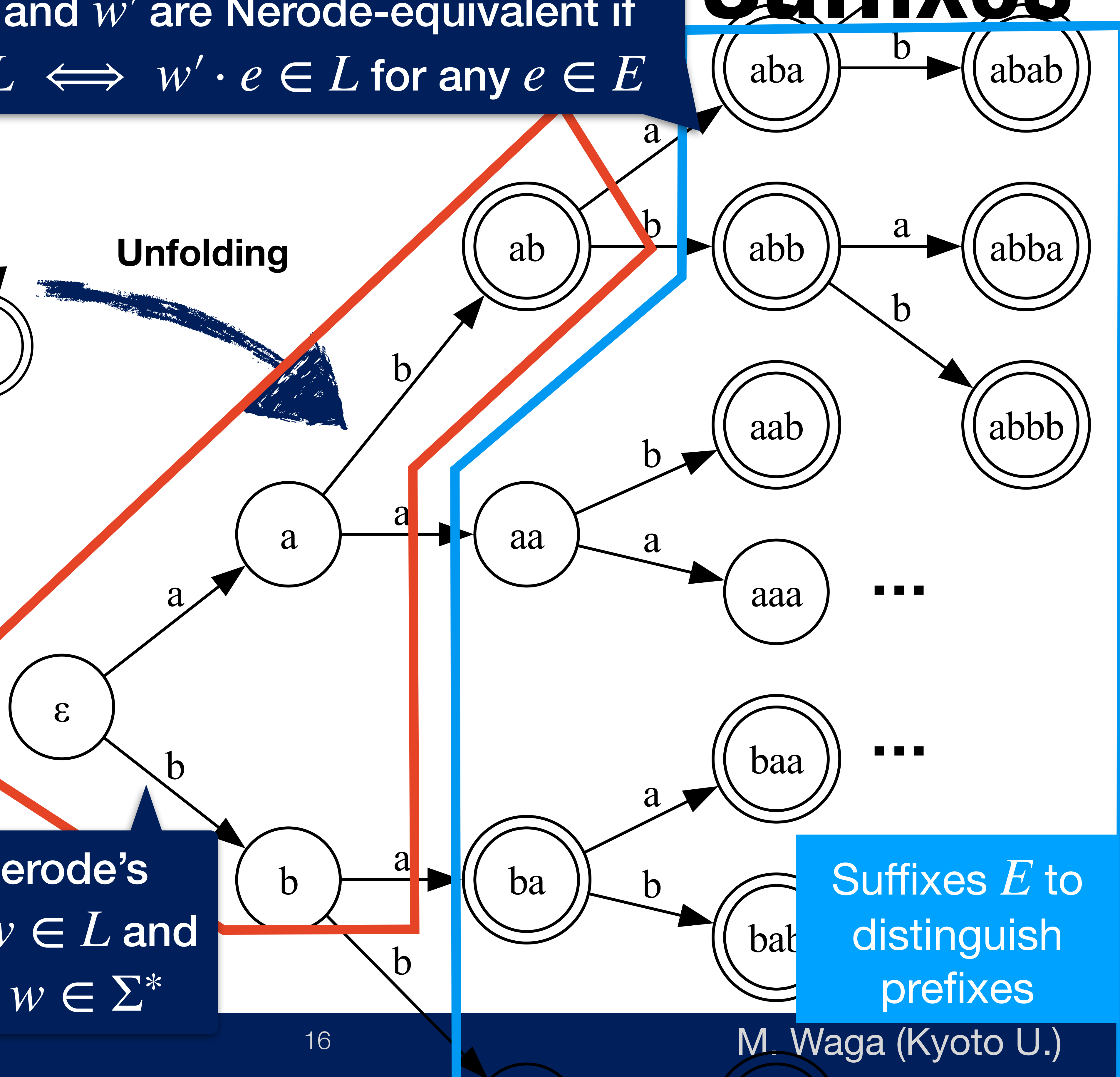
Idea: /Suffixes

Approximate Nerode-equivalence, i.e., deem w and w' are Nerode-equivalent if $w \cdot e \in L \iff w' \cdot e \in L$ for any $e \in E$

Target DFA



Unfolding



Prefixes S to cover states

Suffixes E to distinguish prefixes

Inequivalent wrt. Nerode's congruence, i.e., $s \cdot w \in L$ and $s' \cdot w \notin L$ for some $w \in \Sigma^*$

Observation Table in L* for DFAs

[Angluin, Inf. Comput.'87]

Table showing the membership of concatenated words

Shows if $s \cdot e \in L$
e.g. if $\mathrm{a} \cdot \varepsilon = \mathrm{a} \in L$

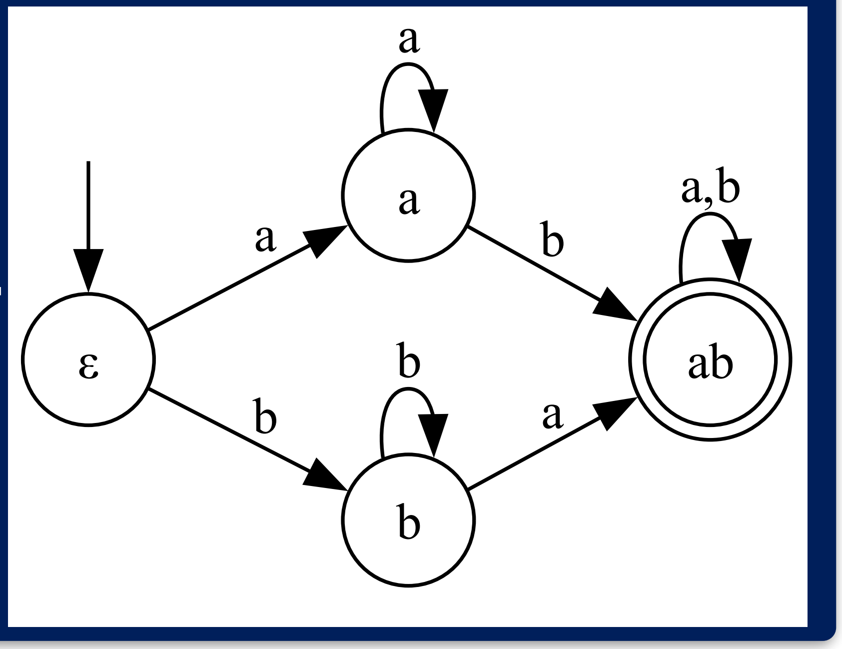
Suffixes E to distinguish prefixes

Prefixes S to cover states

$S \cdot \Sigma \backslash S$

	ε	a	b
ε	$\bot$	$\bot$	$\bot$
a	$\bot$	$\bot$	$\top$
b	$\bot$	$\top$	$\bot$
ab	$\top$	$\top$	$\top$
aa	$\bot$	$\bot$	$\top$
ba	$\top$	$\top$	$\top$
bb	$\bot$	$\top$	$\bot$
aba	$\top$	$\top$	$\top$
abb	$\top$	$\top$	$\top$

Table for



Observation Table in L* for DFAs

[Angluin, Inf. Comput.'87]

Table showing the membership of concatenated words

Shows if $s \cdot e \in L$
e.g. if $\mathrm{a} \cdot \varepsilon = \mathrm{a} \in L$

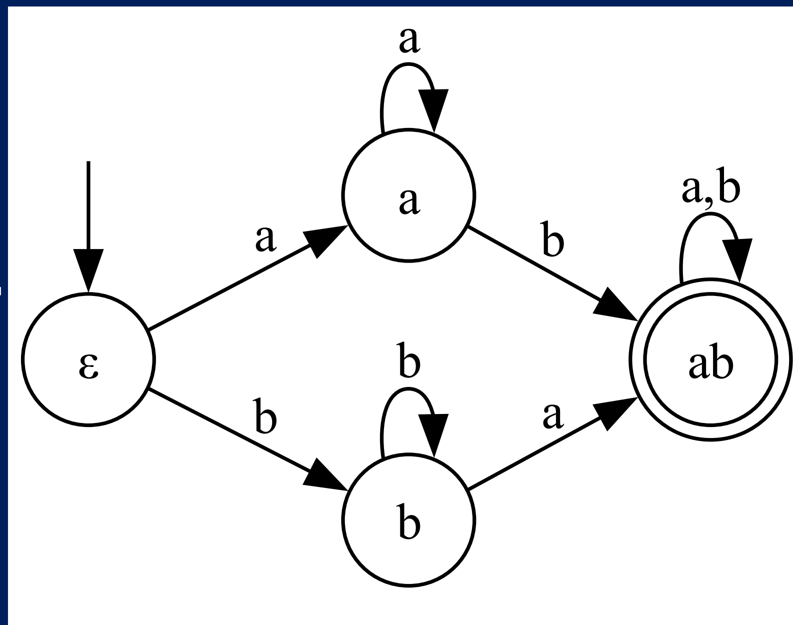
Suffixes E to distinguish prefixes

Prefixes S to cover states

$S \cdot \Sigma \backslash S$

	ε	a	b
ε	$\perp$	$\perp$	$\perp$
a	$\perp$	$\perp$	$\top$
b	$\perp$	$\top$	$\perp$
ab	$\top$	$\top$	$\top$
aa	$\perp$	$\perp$	$\top$
ba	$\top$	$\top$	$\top$
bb	$\perp$	$\top$	$\perp$
aba	$\top$	$\top$	$\top$
abb	$\top$	$\top$	$\top$

Table for



Each $s \in S$ corresponds to one state

Observation Table in L* for DFAs

[Angluin, Inf. Comput.'87]

Table showing the membership of concatenated words

Shows if $s \cdot e \in L$
e.g. if $\mathrm{a} \cdot \varepsilon = \mathrm{a} \in L$

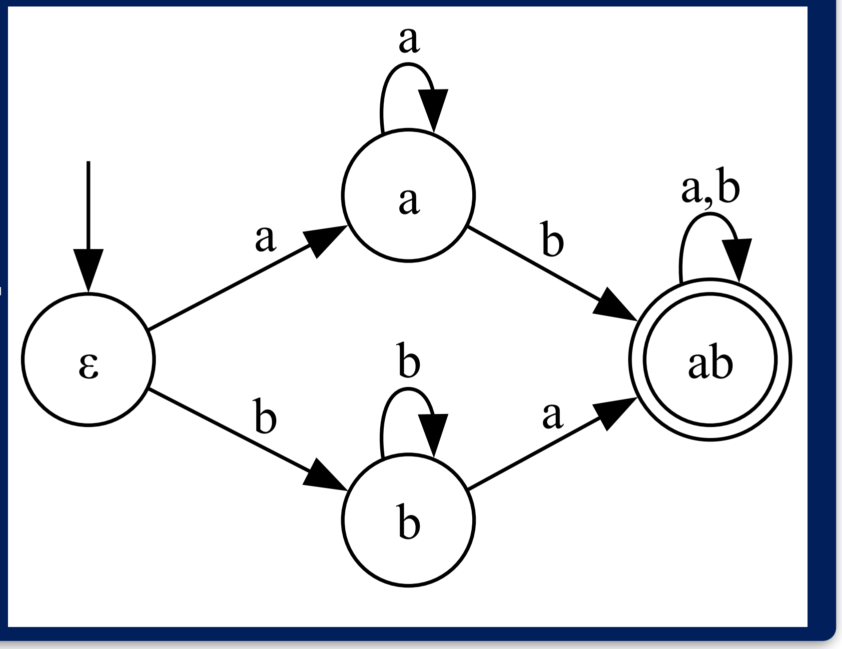
Suffixes E to distinguish prefixes

Prefixes S to cover states

$S \cdot \Sigma \backslash S$

	ε	a	b
ε	$\bot$	$\bot$	$\bot$
a	$\bot$	$\bot$	$\top$
b	$\bot$	$\top$	$\bot$
ab	$\top$	$\top$	$\top$
aa	$\bot$	$\bot$	$\top$
ba	$\top$	$\top$	$\top$
bb	$\bot$	$\top$	$\bot$
aba	$\top$	$\top$	$\top$
abb	$\top$	$\top$	$\top$

Table for



Each $s \in S$ corresponds to one state

Used to define successors

Observation Table in L* for DFAs

[Angluin, Inf. Comput.'87]

Table showing the membership of concatenated words

Shows if $s \cdot e \in L$
e.g. if $\mathrm{a} \cdot \varepsilon = \mathrm{a} \in L$

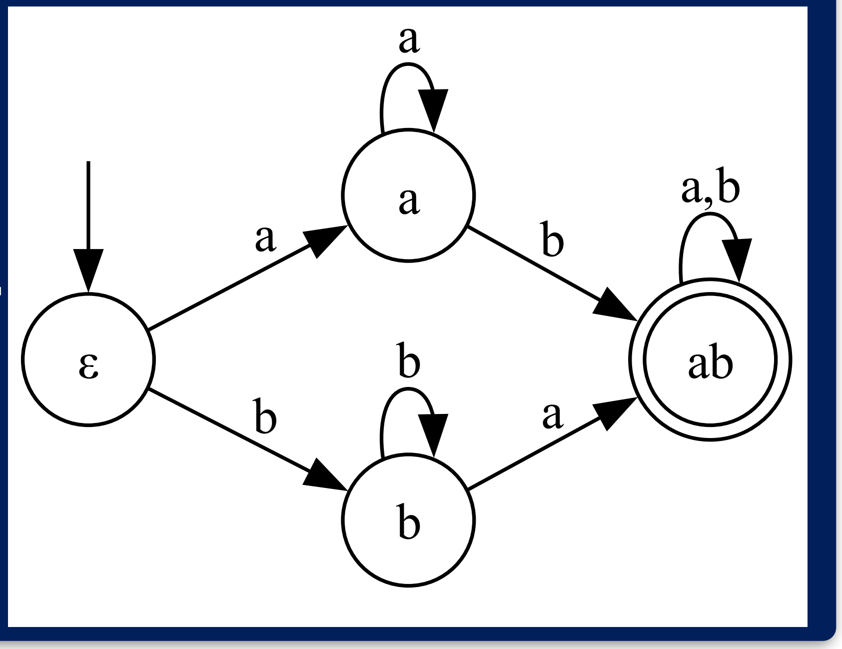
Suffixes E to distinguish prefixes

Prefixes S to cover states

$S \cdot \Sigma \backslash S$

	ε	a	b
ε	$\bot$	$\bot$	$\bot$
a	$\bot$	$\bot$	$\top$
b	$\bot$	$\top$	$\bot$
ab	$\top$	$\top$	$\top$
aa	$\bot$	$\bot$	$\top$
ba	$\top$	$\top$	$\top$
bb	$\bot$	$\top$	$\bot$
aba	$\top$	$\top$	$\top$
abb	$\top$	$\top$	$\top$

Table for



Each $s \in S$ corresponds to one state

Used to define successors

Observation Table in L* for DFAs

[Angluin, Inf. Comput.'87]

Table showing the membership of concatenated words

Shows if $s \cdot e \in L$
e.g. if $\mathrm{a} \cdot \varepsilon = \mathrm{a} \in L$

Suffixes E to distinguish prefixes

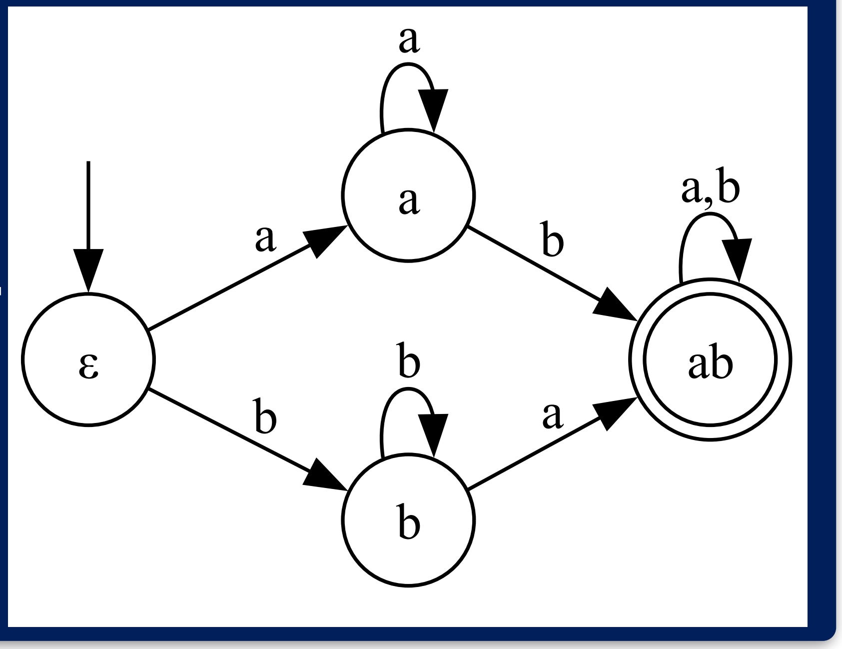
Prefixes S to cover states

Successor with a

$S \cdot \Sigma \backslash S$

	ε	a	b
ε	$\bot$	$\bot$	$\bot$
a	$\bot$	$\bot$	$\top$
b	$\bot$	$\top$	$\bot$
ab	$\top$	$\top$	$\top$
aa	$\bot$	$\bot$	$\top$
ba	$\top$	$\top$	$\top$
bb	$\bot$	$\top$	$\bot$
aba	$\top$	$\top$	$\top$
abb	$\top$	$\top$	$\top$

Table for



Each $s \in S$ corresponds to one state

Used to define successors

Observation Table in L* for DFAs

[Angluin, Inf. Comput.'87]

Table showing the membership of concatenated words

Shows if $s \cdot e \in L$
e.g. if $\mathrm{a} \cdot \varepsilon = \mathrm{a} \in L$

Suffixes E to distinguish prefixes

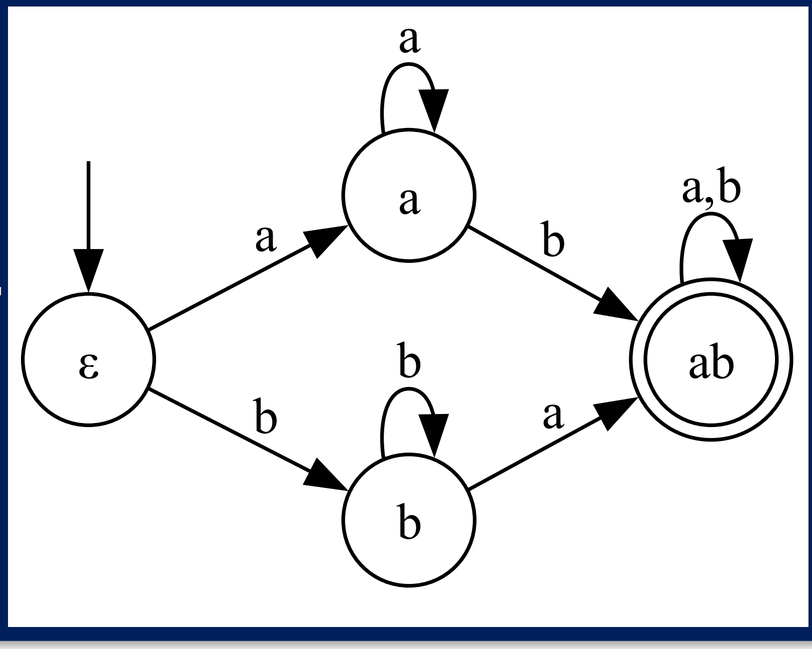
	ε	a	b
ε	$\bot$	$\bot$	$\bot$
a	$\bot$	$\bot$	$\top$
b	$\bot$	$\top$	$\bot$
ab	$\top$	$\top$	$\top$
aa	$\bot$	$\bot$	$\top$
ba	$\top$	$\top$	$\top$
bb	$\bot$	$\top$	$\bot$
aba	$\top$	$\top$	$\top$
abb	$\top$	$\top$	$\top$

Prefixes S to cover states

Successor with a

$S \cdot \Sigma \setminus S$

Table for



Equivalent!

Each $s \in S$ corresponds to one state

Used to define successors

Observation Table in L* for DFAs

[Angluin, Inf. Comput.'87]

Table showing the membership of concatenated words

Shows if $s \cdot e \in L$
e.g. if $\mathrm{a} \cdot \varepsilon = \mathrm{a} \in L$

Suffixes E to distinguish prefixes

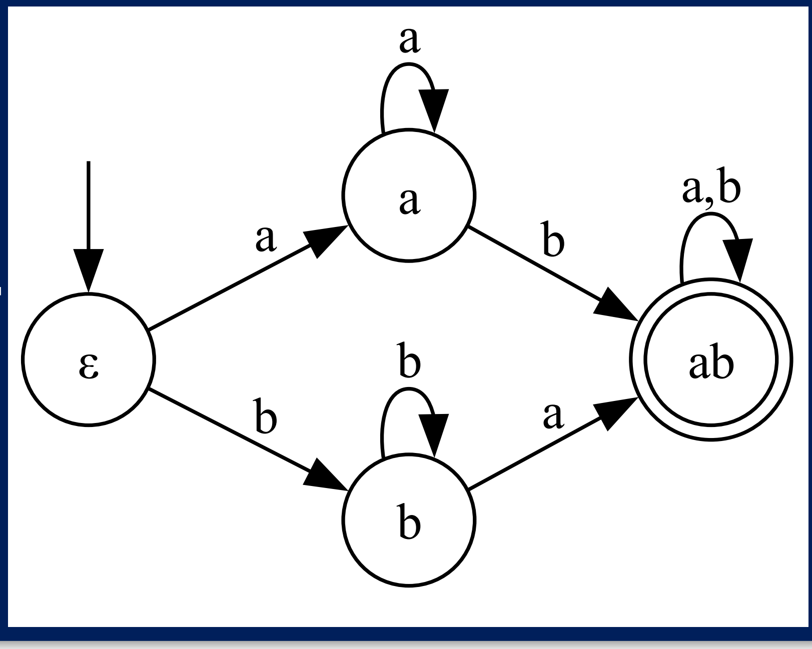
Prefixes S to cover states

Successor with b

$S \cdot \Sigma \backslash S$

	ε	a	b
ε	$\bot$	$\bot$	$\bot$
a	$\bot$	$\bot$	$\top$
b	$\bot$	$\top$	$\bot$
ab	$\top$	$\top$	$\top$
aa	$\bot$	$\bot$	$\top$
ba	$\top$	$\top$	$\top$
bb	$\bot$	$\top$	$\bot$
aba	$\top$	$\top$	$\top$
abb	$\top$	$\top$	$\top$

Table for



Each $s \in S$ corresponds to one state

Used to define successors

Observation Table in AL* for AFAs

[Angluin et al., IJCAI'15]

Idea: Generalize equivalence to Boolean representability

Shows if $s \cdot e \in L$
e.g. if $\mathrm{a} \cdot \varepsilon = \mathrm{a} \in L$

Suffixes E to distinguish prefixes

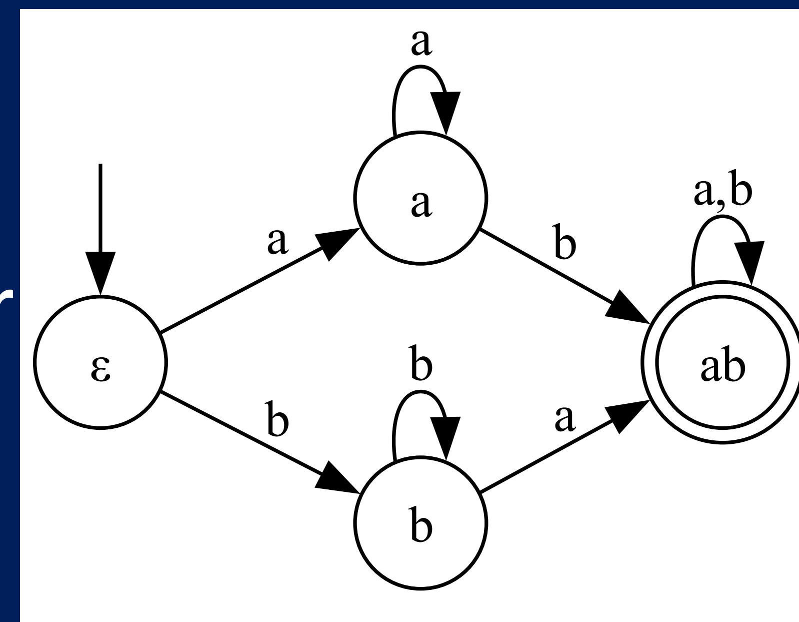
Prefixes S to cover states

$S \cdot \Sigma \backslash S$

	ε	a	b
ε	$\bot$	$\bot$	$\bot$
a	$\bot$	$\bot$	$\top$
b	$\bot$	$\top$	$\bot$
ab	$\top$	$\top$	$\top$
aa	$\bot$	$\bot$	$\top$
ba	$\top$	$\top$	$\top$
bb	$\bot$	$\top$	$\bot$
aba	$\top$	$\top$	$\top$
abb	$\top$	$\top$	$\top$

Equivalent!

Table for



Observation Table in AL* for AFAs

[Angluin et al., IJCAI'15]

Idea: Generalize equivalence to Boolean representability

Shows if $s \cdot e \in L$
e.g. if $\mathrm{a} \cdot \varepsilon = \mathrm{a} \in L$

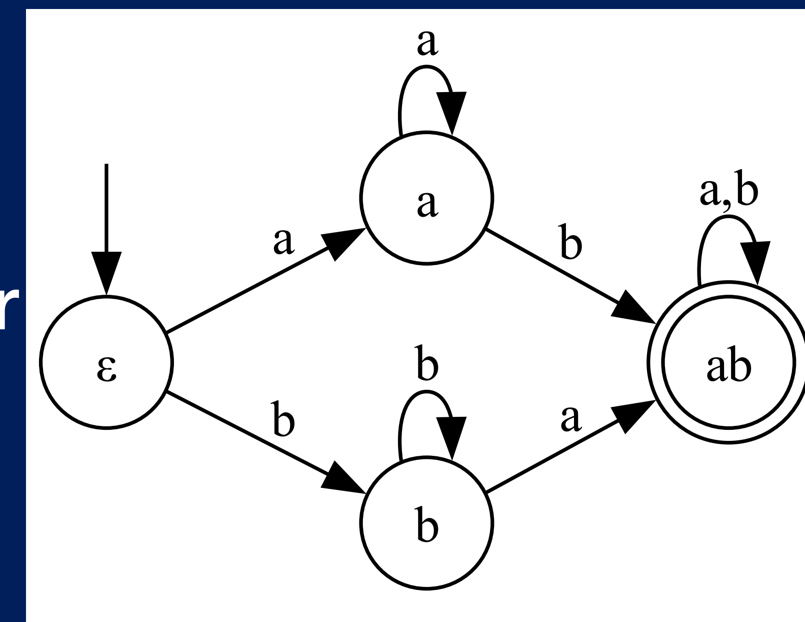
Suffixes E to distinguish prefixes

Prefixes S to cover states

$S \cdot \Sigma \backslash S$

	ε	a	b
ε	$\bot$	$\bot$	$\bot$
a	$\bot$	$\bot$	$\top$
b	$\bot$	$\top$	$\bot$
ab	$\top$	$\top$	$\top$
aa	$\bot$	$\bot$	$\top$
ba	$\top$	$\top$	$\top$
bb	$\bot$	$\top$	$\bot$
aba	$\top$	$\top$	$\top$
abb	$\top$	$\top$	$\top$

Table for



Equivalent to a ∧ b

→ ε is represented by a ∧ b

Observation Table in AL* for AFAs

[Angluin et al., IJCAI'15]

Idea: Generalize equivalence to Boolean representability

Shows if $s \cdot e \in L$
e.g. if $\mathrm{a} \cdot \varepsilon = \mathrm{a} \in L$

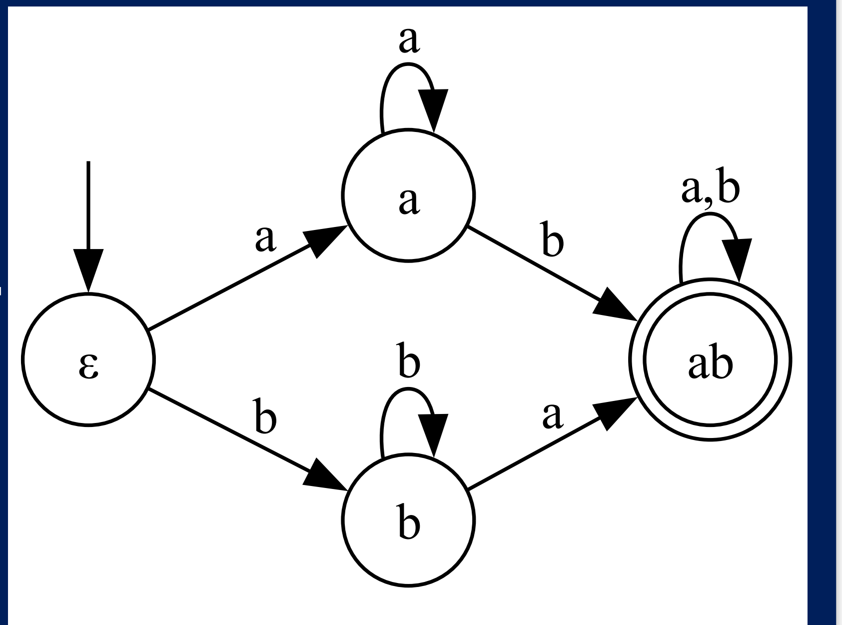
Suffixes E to distinguish prefixes

Prefixes S to cover states

$S \cdot \Sigma \backslash S$

	ε	a	b
ε	$\perp$	$\perp$	$\perp$
a	$\perp$	$\perp$	$\top$
b	$\perp$	$\top$	$\perp$
ab	$\top$	$\top$	$\top$
aa	$\perp$	$\perp$	$\top$
ba	$\top$	$\top$	$\top$
bb	$\perp$	$\top$	$\perp$
aba	$\top$	$\top$	$\top$
abb	$\top$	$\top$	$\top$

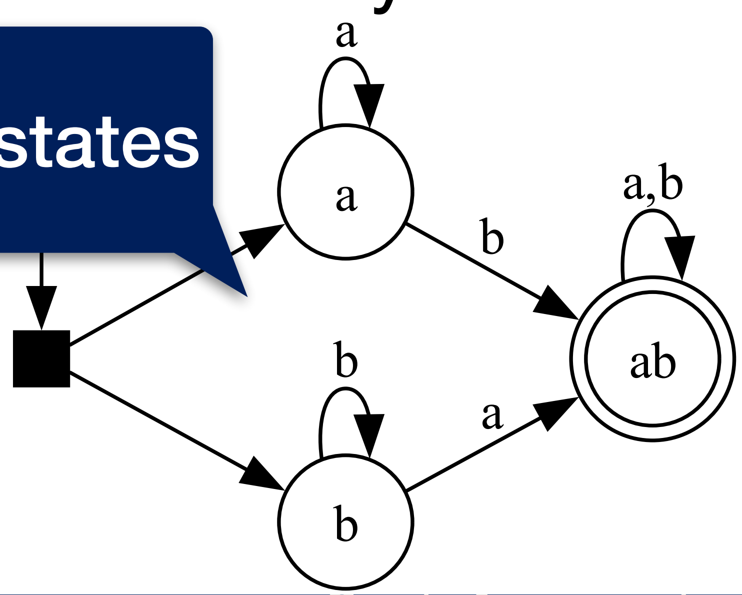
Table for



Equivalent to a ∧ b

→ ε is represented by a ∧ b

AFA w/ 3 states



Observation Table in AL* for AFAs

[Angluin et al., IJCAI'15]

Idea: Generalize equivalence to Boolean representability

Shows if $s \cdot e \in L$
e.g. if $\mathrm{a} \cdot \varepsilon = \mathrm{a} \in L$

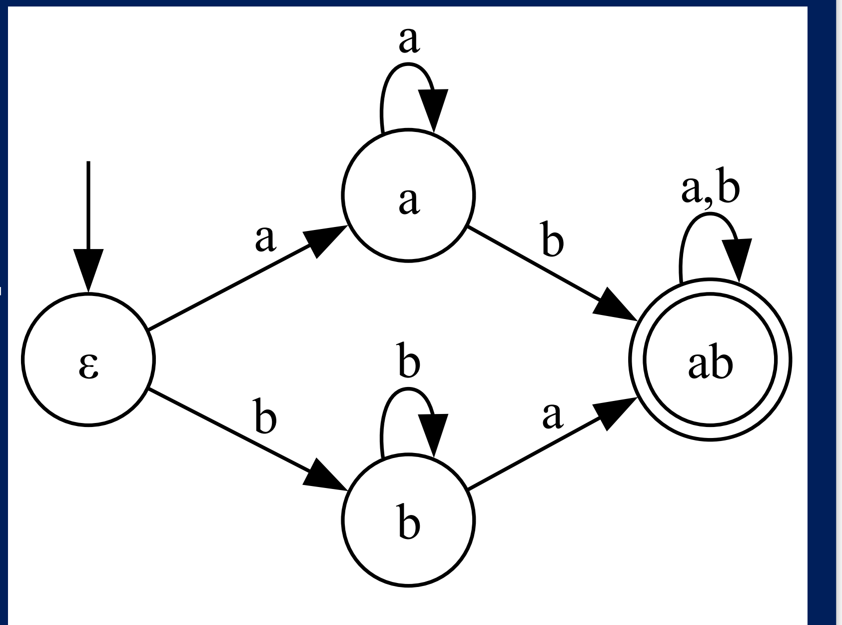
Suffixes E to distinguish prefixes

Prefixes S to cover states

$S \cdot \Sigma \backslash S$

	ε	a	b
ε	$\bot$	$\bot$	$\bot$
a	$\bot$	$\bot$	$\top$
b	$\bot$	$\top$	$\bot$
ab	$\top$	$\top$	$\top$
aa	$\bot$	$\bot$	$\top$
ba	$\top$	$\top$	$\top$
bb	$\bot$	$\top$	$\bot$
aba	$\top$		
abb	$\top$		

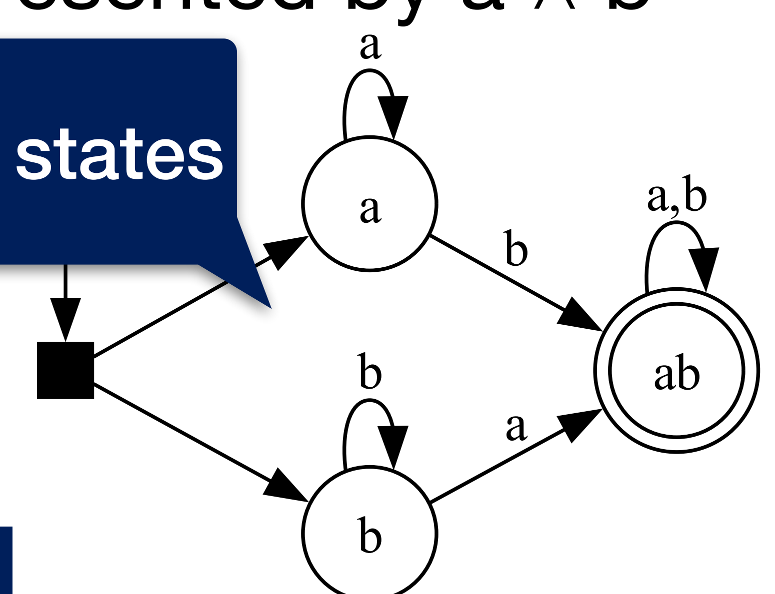
Table for



Equivalent to a ∧ b

→ ε is represented by a ∧ b

AFA w/ 3 states



Identify $P \subseteq S$ sufficient for representing S

DRTA Construction in L^*_{RTA}

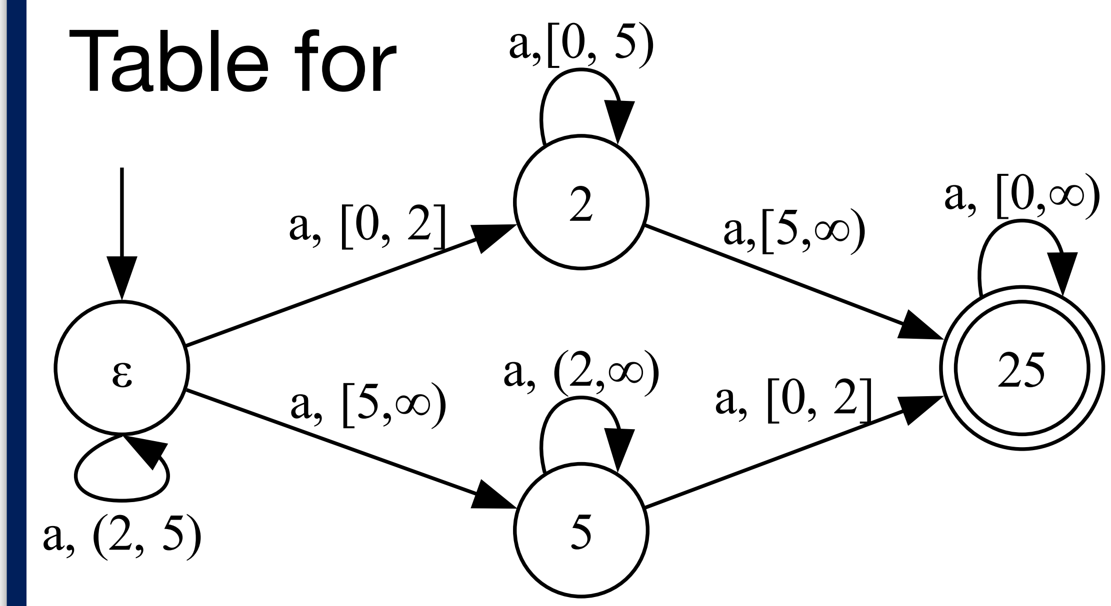
[An et al., SCIS'21]

Learn DFA for finite $\subsetneq \Sigma \times \mathbb{R}_{\geq 0}$ → Make intervals from $\mathbb{R}_{\geq 0}$

Shows if $s \cdot e \in L$
e.g. if $(a,0) \cdot \varepsilon = (a,0) \in L$

	ε	(a,0)	(a,5)
ε	$\bot$	$\bot$	$\bot$
(a,0)	$\bot$	$\bot$	$\top$
(a,5)	$\bot$	$\top$	$\bot$
(a,0)(a,5)	$\top$	$\top$	$\top$
(a,2.5)	$\bot$	$\bot$	$\bot$
(a,0)(a,0)	$\bot$	$\bot$	$\top$
(a,0)(a,2.5)	$\bot$	$\bot$	$\top$
(a,5)(a,0)	$\top$	$\top$	$\top$
(a,5)(a,2.5)	$\bot$	$\top$	$\bot$
(a,5)(a,5)	$\bot$	$\top$	$\bot$
(a,0)(a,5)(a,0)	$\top$	$\top$	$\top$
(a,0)(a,5)(a,2.5)	$\top$	$\top$	$\top$

Table for



DRTA Construction in L^*_{RTA}

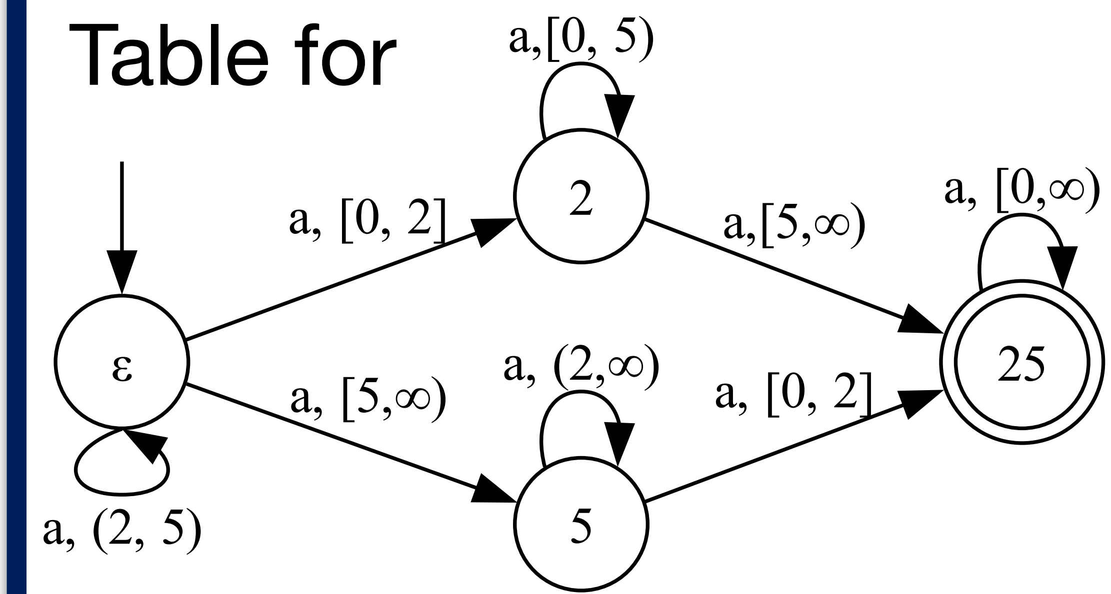
[An et al., SCIS'21]

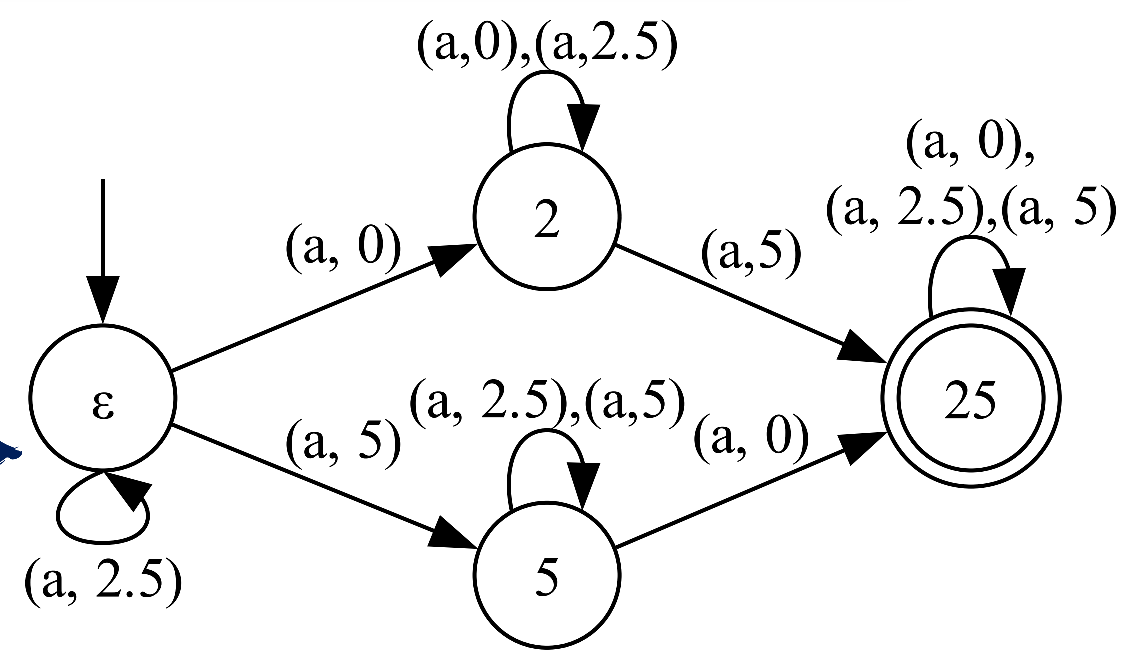
Learn DFA for finite $\subsetneq \Sigma \times \mathbb{R}_{\geq 0}$ → Make intervals from $\mathbb{R}_{\geq 0}$

Shows if $s \cdot e \in L$
e.g. if $(a,0) \cdot \varepsilon = (a,0) \in L$

	ε	(a,0)	(a,5)
ε	$\bot$	$\bot$	$\bot$
(a,0)	$\bot$	$\bot$	$\top$
(a,5)	$\bot$	$\top$	$\bot$
(a,0)(a,5)	$\top$	$\top$	$\top$
(a,2.5)	$\bot$	$\bot$	$\bot$
(a,0)(a,0)	$\bot$	$\bot$	$\top$
(a,0)(a,2.5)	$\bot$	$\bot$	$\top$
(a,5)(a,0)	$\top$	$\top$	$\top$
(a,5)(a,2.5)	$\bot$	$\top$	$\bot$
(a,5)(a,5)	$\bot$	$\top$	$\bot$
(a,0)(a,5)(a,0)	$\top$	$\top$	$\top$
(a,0)(a,5)(a,2.5)	$\top$	$\top$	$\top$

Table for





DRTA Construction in L^*_{RTA}

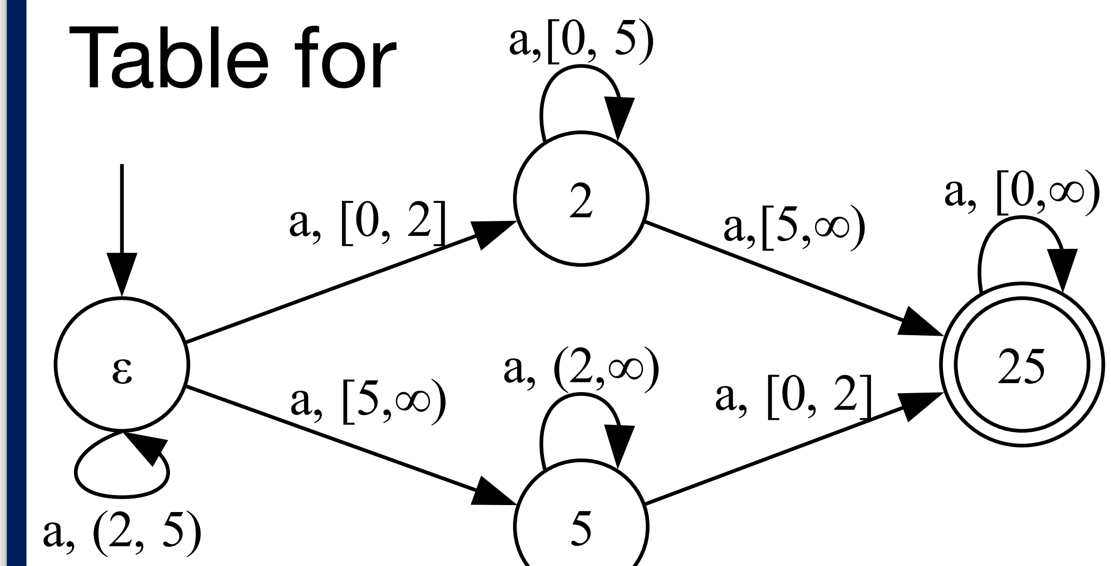
[An et al., SCIS'21]

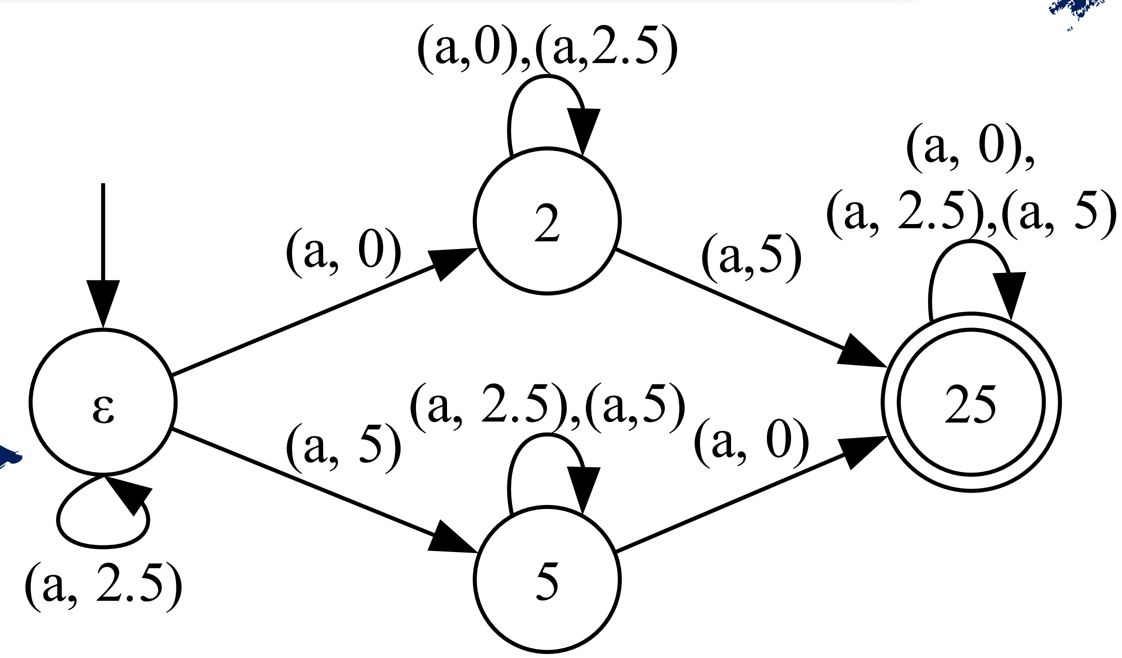
Learn DFA for finite $\subsetneq \Sigma \times \mathbb{R}_{\geq 0}$ → Make intervals from $\mathbb{R}_{\geq 0}$

Shows if $s \cdot e \in L$
e.g. if $(a,0) \cdot \varepsilon = (a,0) \in L$

	ε	(a,0)	(a,5)
ε	$\perp$	$\perp$	$\perp$
(a,0)	$\perp$	$\perp$	$\top$
(a,5)	$\perp$	$\top$	$\perp$
(a,0)(a,5)	$\top$	$\top$	$\top$
(a,2.5)	$\perp$	$\perp$	$\perp$
(a,0)(a,0)	$\perp$	$\perp$	$\top$
(a,0)(a,2.5)	$\perp$	$\perp$	$\top$
(a,5)(a,0)	$\top$	$\top$	$\top$
(a,5)(a,2.5)	$\perp$	$\top$	$\perp$
(a,5)(a,5)	$\perp$	$\top$	$\perp$
(a,0)(a,5)(a,0)	$\top$	$\top$	$\top$
(a,0)(a,5)(a,2.5)	$\top$	$\top$	$\top$

Table for





Summary so far…

- Learn automata by identifying “essential” prefixes/suffixes
  - Data structure: observation table
- For alternating automata:
  - Row equivalence → Boolean representability

Target: state

Target: Boolean formula over states

  - Identify set of prefixes to represent all the rows
- For real-time automata, relax requirements on totality to
  1. learn a partial transition function wrt delays
  2. generalize concrete delays to intervals

Outline

- Formalism: Alternating real-time automata
  - Alternating branching + constraints on delays
- Existing learning algorithms
  - Learning alternating automata (AL*)
  - Learning real-time automata (L^*_{RTA})
- AL^*_{RTA} for active learning of ARTAs
- Experiments

AL^*_{RTA} Algorithm

[Contribution]

Idea: Learn a partial AFA with AL^* → synthesize constraints

AL*$_{RTA}$ Algorithm

[Contribution]

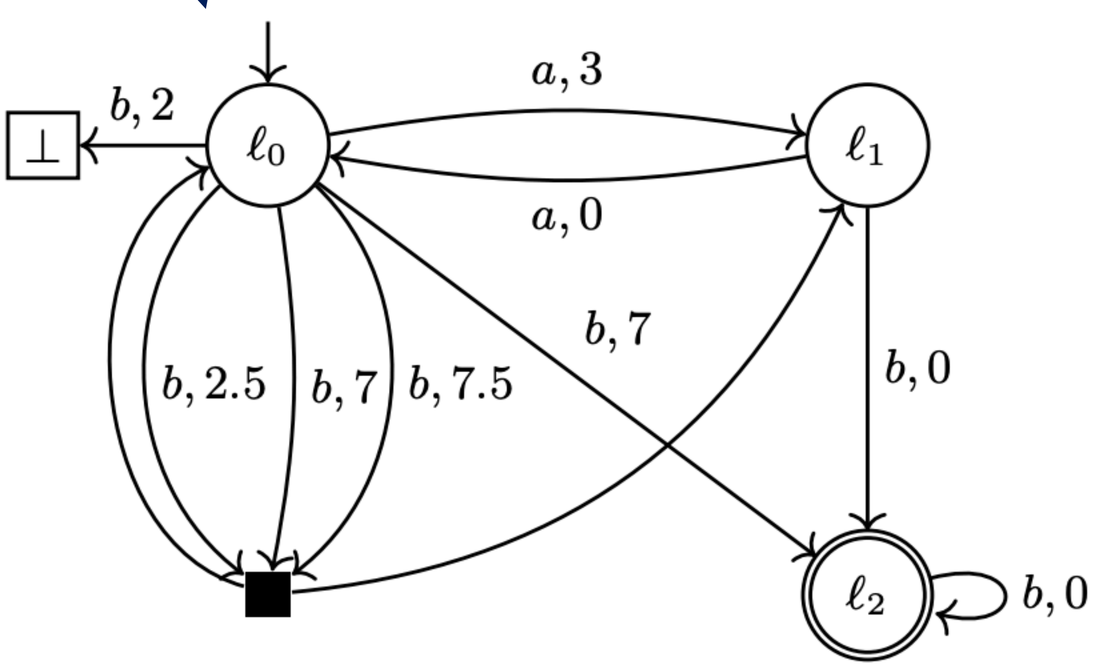
Idea: Learn a partial AFA with AL* → synthesize constraints

AL*$_{\text{RTA}}$ Algorithm

[Contribution]

Idea: Learn a partial AFA with AL* → synthesize constraints

Learn an AFA over a part of $\Sigma \times \mathbb{R}_{\geq 0}$



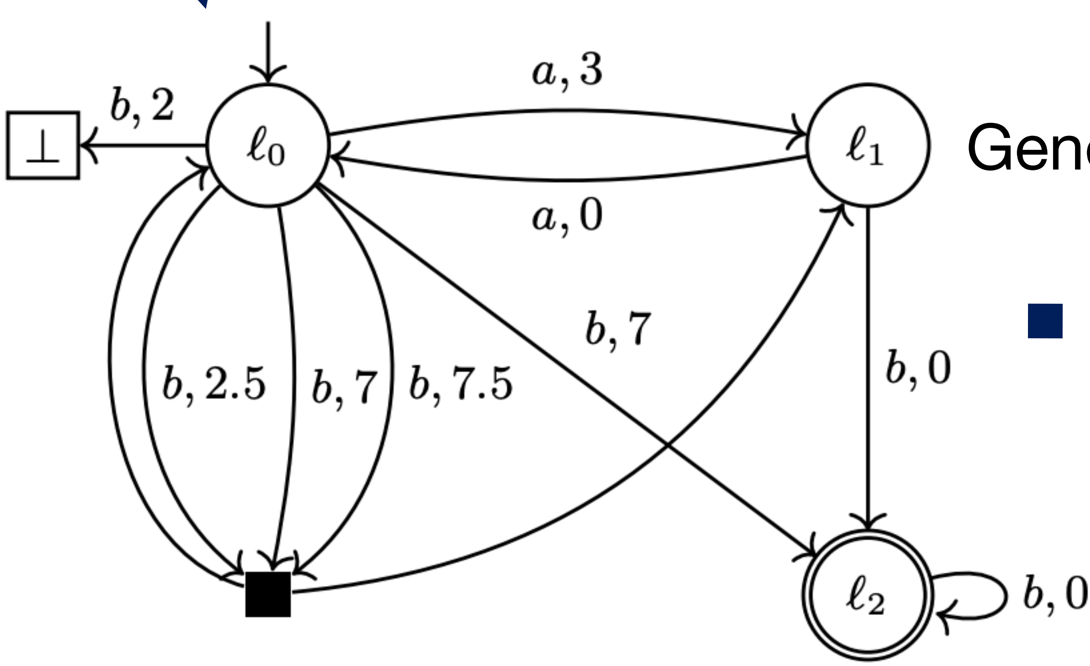
AL*$_{\text{RTA}}$ Algorithm

[Contribution]

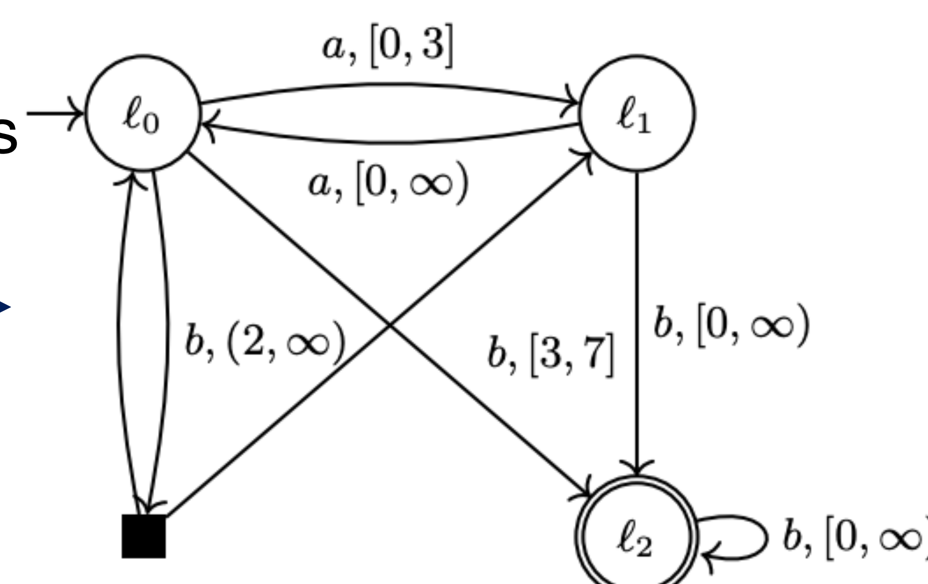
Idea: Learn a partial AFA with AL* → synthesize constraints

Learn an AFA over a part of $\Sigma \times \mathbb{R}_{\geq 0}$

Idea is similar to L*$_{\text{RTA}}$



Generalize concrete delays into intervals



Technical Ingredients of AL*$_{\text{RTA}}$

From AL*

- Compute minimum subset P of prefixes S to represent S
- Boolean representability by P rather than row eq.

From L*$_{\text{RTA}}$

- Observation table over a subset of $\Sigma \times \mathbb{R}_{\geq 0}$
- Generalize delays into intervals for AFA → ARTA
- Partial extension for successors

Technical Ingredients of AL*$_{\text{RTA}}$

From AL*

- Compute minimum subset P of prefixes S to represent S
- Boolean representability by P rather than row eq.

From L*$_{\text{RTA}}$

- Observation table over a subset of $\Sigma \times \mathbb{R}_{\geq 0}$
- Generalize delays into intervals for AFA → ARTA
- Partial extension for successors

Adjusted the definition for learning alternation

Outline of AL*$_{\text{RTA}}$

1. Initialize $S \leftarrow \{\varepsilon\}, E \leftarrow \{\varepsilon\}, P \leftarrow \varnothing$
2. If there is $s \in S$ not representable by a positive Boolean formula over P, add such s to P
3. If some "successors" are missing, extend S and go back to 2
4. Minimize P ; go back to 3 if we found a smaller P
5. Construct an ARTA $\mathscr{A}_{\mathrm{hyp}}$ and ask eq. Query
   → if $\mathscr{L}(\mathscr{A}_{\mathrm{hyp}}) = L_{\mathrm{tgt}}$, the learning finishes
6. Refine E using $w \in \mathscr{L}(\mathscr{A}_{\mathrm{hyp}}) \bigtriangleup L_{\mathrm{tgt}}$
   → go back to 2.

Correctness and Complexity

Theorem

For any real-time language L_{tgt}, AL*$_{\mathrm{RTA}}$ returns an ARTA $\mathscr{A}_{\mathrm{hyp}}$ satisfying $L_{\mathrm{tgt}} = \mathscr{L}(\mathscr{A}_{\mathrm{hyp}})$ with finite queries.

Theorem

The number of membership queries is exponential in the size of the minimum DRTA recognizing L_{tgt} and polynomial in $|\Sigma|$, K, and the max. length of cex.

Correctness and Complexity

Theorem

For any real-time language L_{tgt}, AL*$_{\mathrm{RTA}}$ returns an ARTA $\mathscr{A}_{\mathrm{hyp}}$ satisfying $L_{\mathrm{tgt}} = \mathscr{L}(\mathscr{A}_{\mathrm{hyp}})$ with finite queries.

same in AL*

Theorem

The number of membership queries is exponential in the size of the minimum DRTA recognizing L_{tgt} and polynomial in $|\Sigma|$, K, and the max. length of cex.

Outline

- Formalism: Alternating real-time automata
  - Alternating branching + constraints on delays
- Existing learning algorithms
  - Learning alternating automata (AL*)
  - Learning real-time automata (L^*_{RTA})
- AL^*_{RTA} for active learning of ARTAs
- Experiments

Setting of Experiments

- Implemented AL^*_{RTA} in Rust
- Baseline: NRTALearning [An et al., TECS'21]
    - NL^*_{RTA} for learning nondeterministic RTAs
    - Python implementation
- Used 17 groups of randomly generated NRTAs consisting of 190 NRTAs
    - Taken from NRTALearning
- Report the average numbers
- Intel Xeon w5-3435X 125 GiB RAM with Ubuntu 24.04 LTS

Summary of Experiment

$(\lvert L\rvert, \lvert\Sigma\rvert)$	$\mathrm{AL}^*_{\mathrm{RTA}}$				$\mathrm{NL}^*_{\mathrm{RTA}}$			
	# EqQ	# MemQ	$\lvert L_{\mathcal{A}_{\mathrm{hyp}}}\rvert$	Total Time	# EqQ	# MemQ	$\lvert L_{\mathcal{A}_{\mathrm{hyp}}}\rvert$	Total Time
(3,2)	13.1	975.05	**3**	**0:00.01**	**10.9**	**303.75**	4	0:00.07
(4,2)	17.15	2449.9	**4.05**	**0:00.02**	**15.65**	**810.4**	5.05	0:00.12
(5,2)	22.6	6280.5	**5**	**0:00.06**	**20.7**	**1279.1**	6	0:00.17
(6,2)	**22.3**	10440.1	**6.1**	**0:00.18**	22.4	**1737.5**	7.2	0:00.31
(8,2)	31.3	28058.3	**8**	0:00.77	**27.6**	**3093.6**	9.1	**0:00.45**
(8,4)	48.4	55362.2	**8**	**0:00.66**	**46.4**	**8449.5**	9.1	0:02.05
(10,2)	38.2	50558.9	**10**	0:01.61	**36.4**	**5665.3**	11.1	**0:01.02**
(10,4)	52.4	106987.2	**10**	**0:02.28**	**49.2**	**9664.4**	11.1	0:02.42
(10,6)	71.7	171537.9	**10**	**0:02.92**	**62.8**	**17346.3**	11	0:06.16
(10,8)	97	443801.3	**10**	0:20.75	**83.1**	**20298.7**	11	**0:07.55**
(10,10)	114.2	723024.4	**10**	0:33.37	**92**	**26558.5**	11	**0:13.48**
(12,2)	56	144836.1	**12.3**	0:08.06	**48.4**	**9245.6**	13.3	**0:02.13**
(12,4)	70	295512.5	**12**	0:06.59	**66**	**14968.3**	13	**0:04.39**
(14,4)	81.7	590253.4	**14.1**	0:40.19	**74**	**20697.2**	15.1	**0:07.69**
(16,4)	91.4	812425.1	**16**	1:39.27	**85.5**	**41004.4**	17.1	**0:28.44**
(18,4)	100.7	1136254	**18.3**	2:08.25	**97.5**	**35404**	19.4	**0:18.07**
(20,4)	105.7	1122431.9	**19.8**	1:01.14	**103.9**	**56806.9**	21	**0:46.75**

M. Waga (Kyoto U.)

Summary of Experiment

$(\lvert L\rvert, \lvert\Sigma\rvert)$	$\mathrm{AL}^*_{\mathrm{RTA}}$				$\mathrm{NL}^*_{\mathrm{RTA}}$			
	# EqQ	# MemQ	$\lvert L_{\mathcal{A}_{\mathrm{hyp}}}\rvert$	Total Time	# EqQ	# MemQ	$\lvert L_{\mathcal{A}_{\mathrm{hyp}}}\rvert$	Total Time
(3,2)	13.1	975.05	**3**	**0:00.01**	**10.9**	**303.75**	4	0:00.07
(4,2)	17.15	2449.9	**4.05**	**0:00.02**	**15.65**	**810.4**	5.05	0:00.12
(5,2)	22.6	6280.5	**5**	**0:00.06**	**20.7**	**1279.1**	6	0:00.17
(6,2)	**22.3**	10440.1	**6.1**	**0:00.18**	22.4	**1737.5**	7.2	0:00.31
(8,2)	31.3	28058.3	**8**	0:00.77	**27.6**	**3093.6**	9.1	**0:00.45**
(8,4)	48.4	55362.2	**8**	**0:00.66**	**46.4**	**8449.5**	9.1	0:02.05
(10,2)	38.2	50558.9	**10**	0:01.61	**36.4**	**5665.3**	11.1	**0:01.02**
(10,4	[illegible]	[illegible]	**10**	**0:02.28**	**49.2**	**9664.4**	11.1	0:02.42
(10,	[illegible]	[illegible]	**10**	**0:02.92**	**62.8**	**17346.3**	11	0:06.16
(10,8)	97	445801.5	**10**	0:20.75	**83.1**	**20298.7**	11	**0:07.55**
(10,10)	114.2	723024.4	**10**	0:33.37	**92**	**26558.5**	11	**0:13.48**
(12,2)	56	144836.1	**12.3**	0:08.06	**48.4**	**9245.6**	13.3	**0:02.13**
(12,4)	70	295512.5	**12**	0:06.59	**66**	**14968.3**	13	**0:04.39**
(14,4)	81.7	590253.4	**14.1**	0:40.19	**74**	**20697.2**	15.1	**0:07.69**
(16,4)	91.4	812425.1	**16**	1:39.27	**85.5**	**41004.4**	17.1	**0:28.44**
(18,4)	100.7	1136254	**18.3**	2:08.25	**97.5**	**35404**	19.4	**0:18.07**
(20,4)	105.7	1122431.9	**19.8**	1:01.14	**103.9**	**56806.9**	21	**0:46.75**

Smaller automata

Summary of Experiment

$(\lvert L\rvert, \lvert\Sigma\rvert)$	$\mathrm{AL}^*_{\mathrm{RTA}}$				$\mathrm{NL}^*_{\mathrm{RTA}}$			
	# EqQ	# MemQ	$\lvert L_{\mathcal{A}_{\mathrm{hyp}}}\rvert$	Total Time	# EqQ	# MemQ	$\lvert L_{\mathcal{A}_{\mathrm{hyp}}}\rvert$	Total Time
(3,2)	13.1	975.05	**3**	**0:00.01**	**10.9**	**303.75**	[illegible]	[illegible]
(4,2)	17.15	2449.9	**4.05**	**0:00.02**	**15.65**	**810.4**	[illegible]	[illegible]
(5,2)	22.6	6280.5	**5**	**0:00.06**	**20.7**	**1279.1**	[illegible]	[illegible]
(6,2)	**22.3**	10440.1	**6.1**	**0:00.18**	22.4	**1737.5**	[illegible]	[illegible]
(8,2)	31.3	28058.3	**8**	0:00.77	**27.6**	**3093.6**	[illegible]	[illegible]
(8,4)	48.4	55362.2	**8**	**0:00.66**	**46.4**	**8449.5**	[illegible]	[illegible]
(10,2)	38.2	50558.9	**10**	0:01.61	**36.4**	**5665.3**	[illegible]	[illegible]
(10,4)	[illegible]	[illegible]	**10**	**0:02.28**	**49.2**	**9664.4**	11.1	0:02.42
(10,6)	[illegible]	[illegible]	**10**	**0:02.92**	**62.8**	**17346.3**	11	0:06.16
(10,8)	97	[illegible]	**10**	0:20.75	**83.1**	**20298.7**	11	**0:07.55**
(10,10)	114.2	723024.4	**10**	0:33.37	**92**	**26558.5**	11	**0:13.48**
(12,2)	56	144836.1	**12.3**	0:08.06	**48.4**	**9245.6**	13.3	**0:02.13**
(12,4)	70	295512.5	**12**	0:06.59	**66**	**14968.3**	13	**0:04.39**
(14,4)	81.7	590253.4	**14.1**	0:40.19	**74**	**20697.2**	15.1	**0:07.69**
(16,4)	91.4	812425.1	**16**	1:39.27	**85.5**	**41004.4**	17.1	**0:28.44**
(18,4)	100.7	1136254	**18.3**	2:08.25	**97.5**	**35404**	19.4	**0:18.07**
(20,4)	105.7	1122431.9	**19.8**	1:01.14	**103.9**	**56806.9**	21	**0:46.75**

More efficient partly because of smarter cex handling à la Rivest–Schapire

Smaller automata

Conclusions

- Show the succinctness theorem of ARTAs wrt DRTAs
  - Similar to the property of AFAs
- L*-style active learning algorithm for ARTAs
  - 1. Learning AFA over subset of $\Sigma \times \mathbb{R}_{\geq 0}$
  - 2. Generalize delays to intervals
- Implementation + experiments
  → Tends to learn smaller automata than $\mathrm{NL}^*_{\mathrm{RTA}}$

Conclusions

- Show the succinctness theorem of ARTAs wrt DRTAs
  - Similar to the property of AFAs
- L*-style active learning algorithm for ARTAs
  - 1. Learning AFA over subset of $\Sigma \times \mathbb{R}_{\geq 0}$
  - 2. Generalize delays to intervals
- Implementation + experiments
  → Tends to learn smaller automata than NL^*_{RTA}

Learning alternating branching and timing constraints seems orthogonal

Future Directions

- Improve the efficiency of AL^*_{RTA}, e.g. by
  - Rivest–Schapire-style counterexample handling
  - Smarter extension of successors
- Generalize the idea to other automata
  - e.g., symbolic automata, event-recording automata

Generalization of RTAs for other predicates

“Learnable” subclass of timed automata

Appendix

Summary of NL*$_{\text{RTA}}$

- Row equivalence is generalized to representability by disjunctions
  - More restrictive than AL*$_{\text{RTA}}$
- $P \subseteq S$ to represent the rows in $S \cup R$ is easily obtained
  - Idea: take smaller rows in the elementwise order with $\bot < \top$
  - This works by the monotonicity of disjunctions

Counterexample Handling

cex : counterexample normalized in delay

L*$_{RTA}$

- L*-style cex handling, i.e., add $prefixes(cex)$ to R

NL*$_{RTA}$

Smarter

- Rivest–Schapire-style cex handling, i.e., binary-search of cex for a relevant suffix and add its suffixes to E
- They also add $prefixes(cex)$ to R

But this makes observation table large

AL*$_{RTA}$

- Maler-Pnueli-style cex handling, i.e., add $suffixes(cex)$ to E

Successors in NL*$_{\text{RTA}}$ and AL*$_{\text{RTA}}$

$\Phi(X) \subseteq \Sigma \times \mathbb{R}_{\geq 0}$: events with delay in X

NL*$_{\text{RTA}}$

- For any $s \in S, a \in \Sigma, s \cdot (a,0) \in S \cup R$
- For any $s \in S, e \in E, s \cdot e \in S \cup R$
- For any $s \cdot a \in S \cup R, s' \in S,$ if $\mathrm{row}(s') \leq \mathrm{row}(s)$, $s' \cdot a \in S \cup R$

E can be small

AL*$_{\text{RTA}}$

$\Phi(S \cup E)$ can be large, partly because E can be much larger

- For any $p \in P, a \in \Phi(S \cup E), p \cdot a \in S$

Observation Table Size

$(\lvert L\rvert, \lvert\Sigma\rvert)$	AL^*_{RTA} Rows	AL^*_{RTA} Columns	NL^*_{RTA} Rows	NL^*_{RTA} Columns
(3,2)	45.8	21.75	44.55	6.5
(4,2)	80.75	30.6	73.95	10.65
(5,2)	123.6	51.6	83.5	14.9
(6,2)	181.3	58.7	116.9	14.3
(8,2)	315.3	86.5	144.9	20.6
(8,4)	586.4	101	299	28.1
(10,2)	507.1	103.7	222.7	25.1
(10,4)	829.7	133.2	298.7	31.7
(10,6)	1063.9	164.6	455.2	37.6
(10,8)	1856.4	237.2	476.7	42.8
(10,10)	2903.2	247.6	579.3	44.4
(12,2)	892.7	159.7	272.8	34.1
(12,4)	1644	184.8	390.2	37.9
(14,4)	2413	216.3	457.4	44
(16,4)	3133.3	266.5	722.7	54.6
(18,4)	3584.1	312.9	580.5	59.5
(20,4)	3388.6	351.2	840.3	65.4

Observation Table Size

$(\lvert L\rvert, \lvert\Sigma\rvert)$	AL^*_{RTA} Rows	AL^*_{RTA} Columns	NL^*_{RTA} Rows	NL^*_{RTA} Columns
(3,2)	45.8	21.75	44.55	6.5
(4,2)	80.75	30.6	73.95	10.65
(5,2)	123.6	51.6	[illegible]	14.9
(6,2)	181.3	58.7	[illegible]	14.3
(8,2)	315.3	86.5	[illegible]	20.6
(8,4)	586.4	101	299	28.1
(10,2)	507.1	103.7	222.7	25.1
(10,4)	829.7	[illegible]	298.7	31.7
(10,6)	1063.9	[illegible]	455.2	37.6
(10,8)	1856.4	[illegible]	476.7	42.8
(10,10)	2903.2	247.6	579.3	44.4
(12,2)	892.7	159.7	272.8	34.1
(12,4)	1644	184.8	390.2	37.9
(14,4)	2413	216.3	457.4	44
(16,4)	3133.3	266.5	722.7	54.6
(18,4)	3584.1	312.9	580.5	59.5
(20,4)	3388.6	351.2	840.3	65.4

Likely due to simpler cex analysis

Likely due to the successor extension