Automata learning algorithms obtained from Myhill-Nerode-style theorems

The free Monoid and the right congruence

Definition (Free Monoid)

Given an alphabet Σ, let " $\cdot$ " denote the string concatenation operator.
A sequence $s_1 \cdot s_2 \cdots s_{n-1} \cdot s_n$ is called a word and Σ^* denotes the set of all words over Σ, including the empty word λ.
$(\Sigma^*, \cdot)$ is called the "Free Monoid".

Definition (Right equivalence)

Given a language $L \subseteq \Sigma^*$ we can define an equivalence relation on Σ^*:

$u \equiv_L w$ iff $\forall r \in \Sigma^*: u \cdot r \in L \iff w \cdot r \in L$

Let $[u]_L$ denote the equivalence class of u w.r.t. this relation.
Note that $[u]_L \cdot w = [u \cdot w]_L$

The Myhill-Nerode theorem

Theorem (Myhill-Nerode 1957)

1. A language L is regular if and only if the number of equivalence classes of the right equivalence is finite.

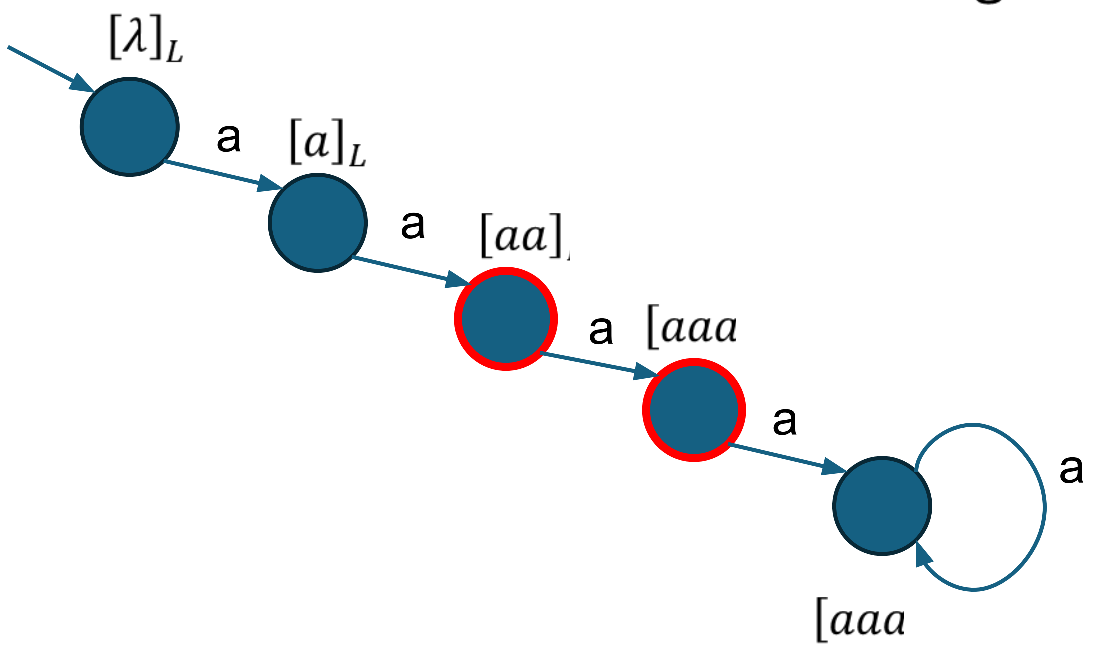
2. The number of states in the minimal deterministic automata recognising L is equivalent to the number of right equivalence classes.

3. The set Σ is a generating set for $[\Sigma^*]_L$ using "$\cdot$". The graph generated by Σ over $[\Sigma^*]_L$ is isomorphic to every minimal DFA recognising L (It is the canonical minimal DFA).

The canonical DFA

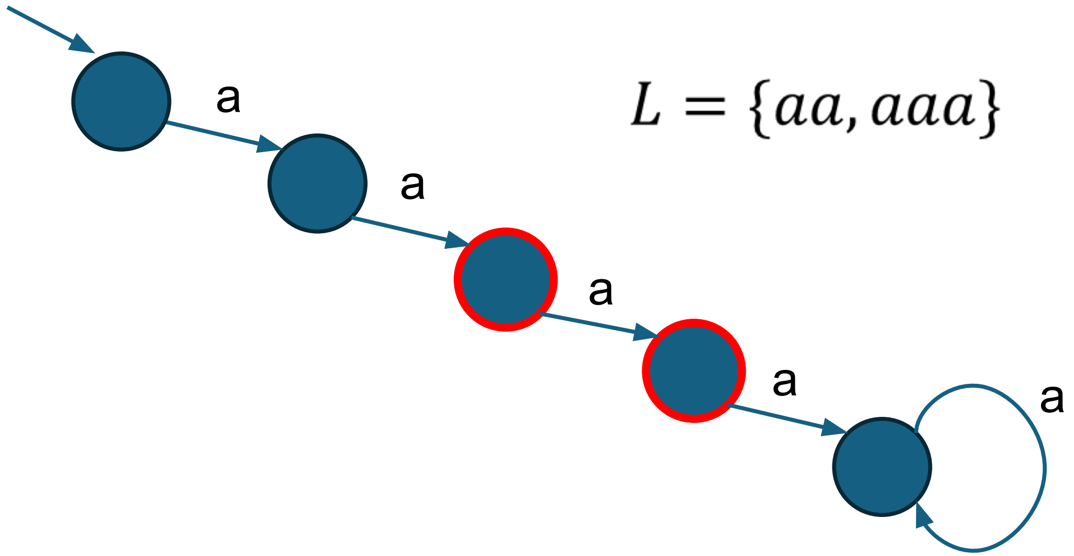
$L = \{aa, aaa\}$

Generating set: Σ



Active learning

Membership oracle

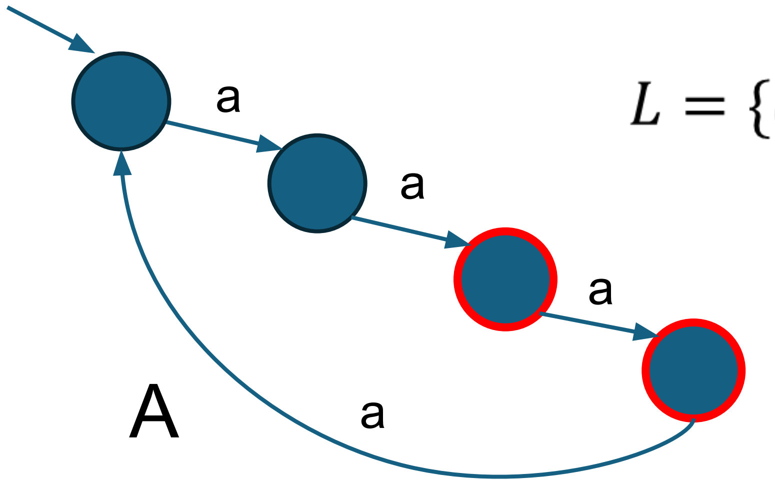


$L = \{aa, aaa\}$

Is $w \in \Sigma^*$ in L?

a	-	No
aaa	-	Yes

Equivalence oracle



$L = \{aa, aaa\}$

Does A recognise L?

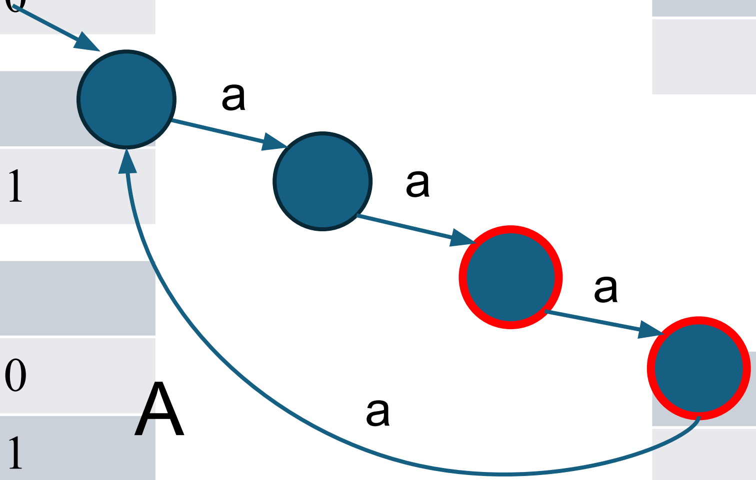
The L*-algorithm (Dana Angluin)

	0	0

	0	1

	0	0
	0	1
	1	1
	1	0

	0	0



	0	0
	0	1

	1	1

	0	0
	0	1
	1	1

	1	0

The L*-algorithm – Counterexample processing (Rivest and Schapire)

Counterexample from Oracle:

$aaaaaa$

Scan through counterexample by substituting access word whenever possible.

$aaaaaa \to \lambda aa$

After every substitution query oracle on both words:

$aaaaaa \to 0$

$\lambda aa \to 1$

aa witnesses the mistake of our model

	0	0	1
	0	1	1
	1	1	0
	1	0	0

	0	0	0

Correctness of the L*-algorithm

- Clearly the automata B found by L* accepts L, since otherwise the equivalence query would have failed
- Assume B it is not minimal:
  - Let L_q denote all words that end in state $Q \in Q_B$
  - By PHP there exists an equivalence class $[u]_L$ such that two distinct states $q, q' \in Q_B$ intersect it ($L_q \cap [u]_L \neq \emptyset$ and $L_{q'} \cap [u]_L \neq \emptyset$)
  - Since an extensions distinguishing q and q' exists there must also exist some equivalence classes that are distinct and intersected by $L_q$and $L_{q'}$
    - $[w]_L \neq [v]_L$ with $[w]_L \cap L_q \neq \emptyset$ and $[v]_L \cap L_{q'} \neq \emptyset$
    - Since $[w]_L \neq [v]_L$ at least one of them is distinct from $[u]_L$, say w.l.o.g. $[w]_L$
  - Since L_q intersects two distinct equivalence classes ($[u]_L, [w]_L$) there exists and extension $r \in \Sigma^*$ s.t. there exist two words $u, w \in L_q$ with

    $u \cdot r \in L \iff w \cdot r \notin L$, however, B will either accept or reject both.

Running time of the L* algorithm

- Each equivalence query results in a new state. So, at most n equivalence queries are necessary.
- The number of membership queries depends on the size of the table and the cost of processing a counter-example
  - Size of the table:
    - At most $n + \Sigma n$ rows, since it returns the canonical DFA
    - At most n columns since each new column leads to a new state
    - At most $n^2 + \Sigma n^2$ membership queries to fill the table
  - Processing counterexamples:
    - 2 membership queries for every substitution made while processing. Can be done with binary search
    - $2\log(c)$ membership queries, where c is the length of the counterexample

What was used in the proofs?

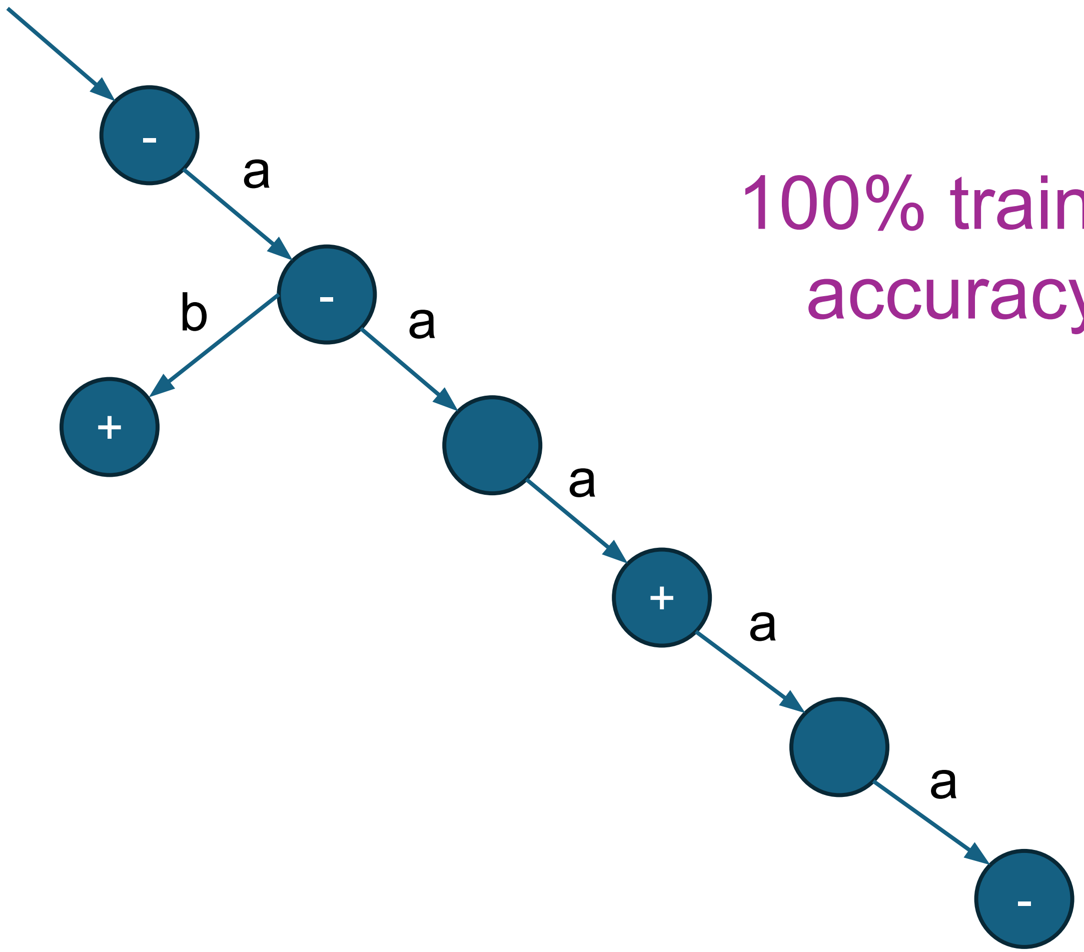
- The right equivalence classes correspond to the states of the minimal DFA

- Each equivalence class can be generated from the class that contains the empty string through Σ

- A counterexample leads to a proof of non-equivalence by stepping through the generating process for the counter-example and the states in parallel

Passive learning

Augmented Prefix Tree Acceptor (APTA)

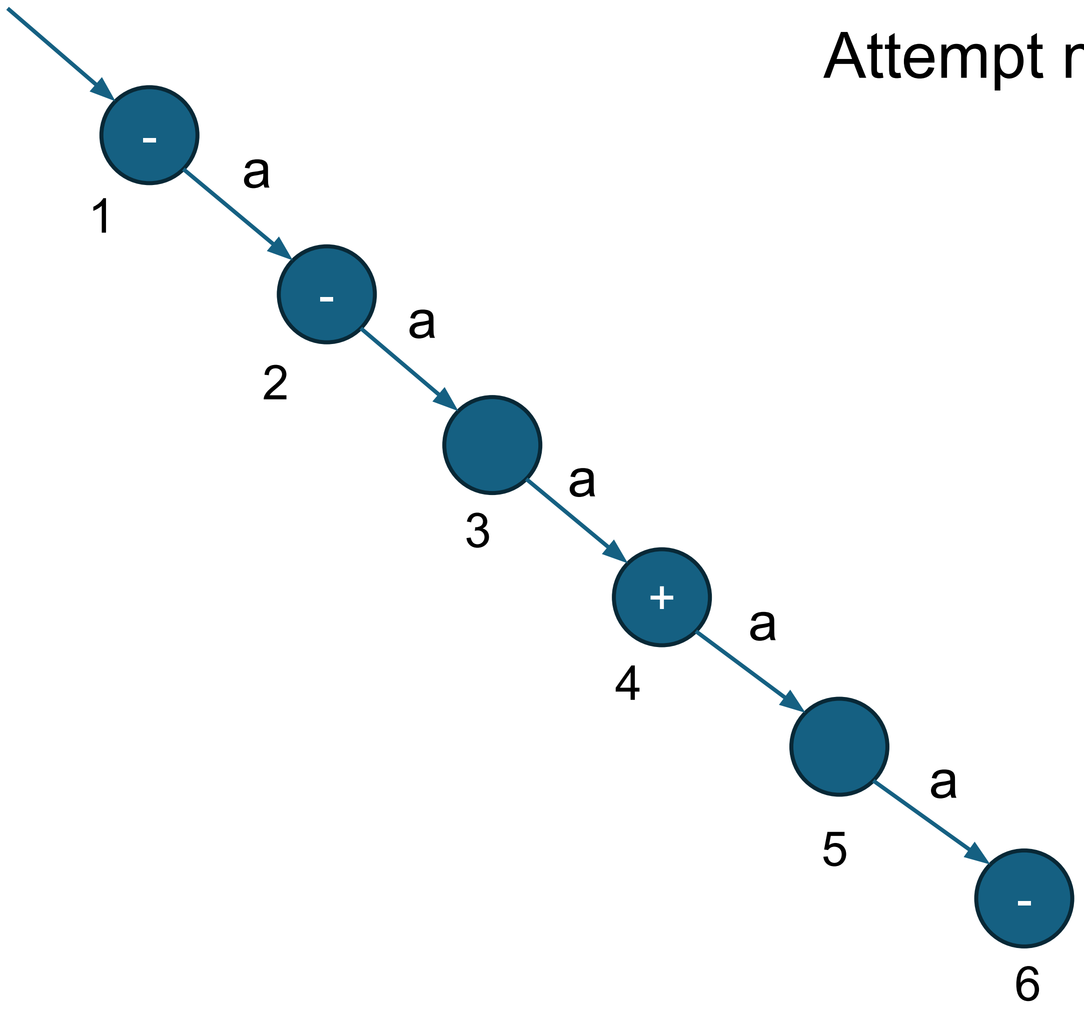


100% training accuracy!

Set of Samples

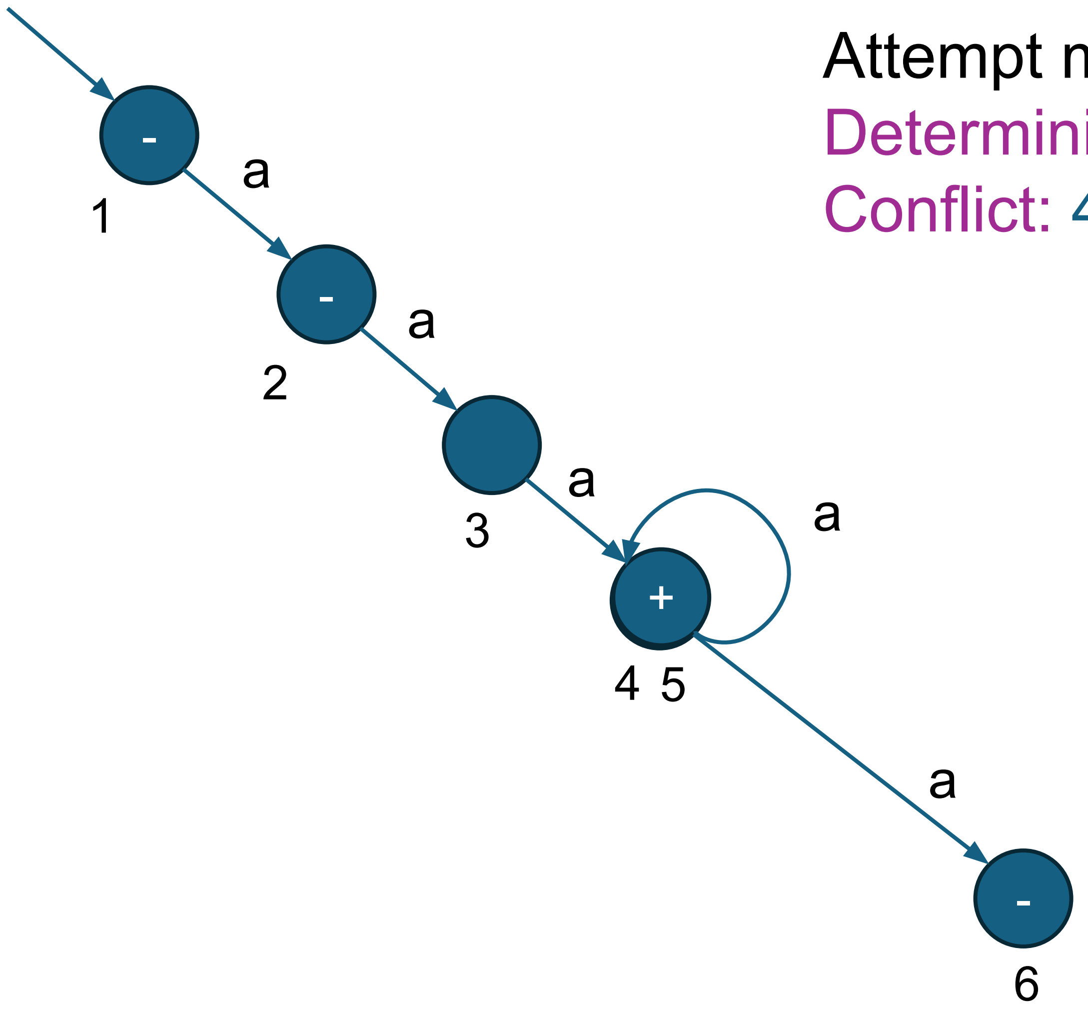
a	-
aaa	+
aaaaa	-
_	-
ab	+

Merging (and determinisation)



Attempt merge: 4 + 5

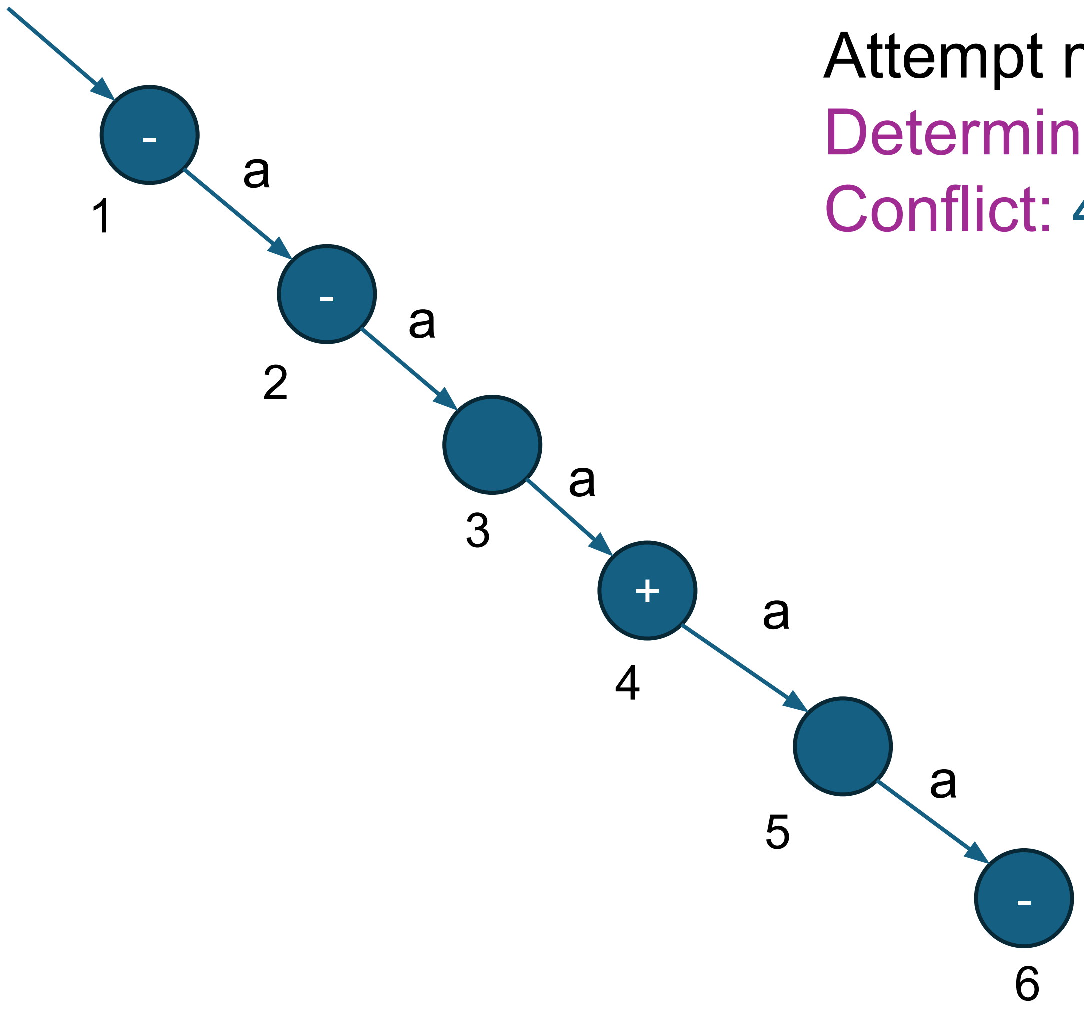
Merging (and determinisation)



Attempt merge: 4 + 5
Determinisation: (4,5) + 6
Conflict: 4 ! 6

Possible Merges:
5 + 6
6 + 1
6 + 2
6 + 3
5 + 1
5 + 2

Merging (and determinisation)

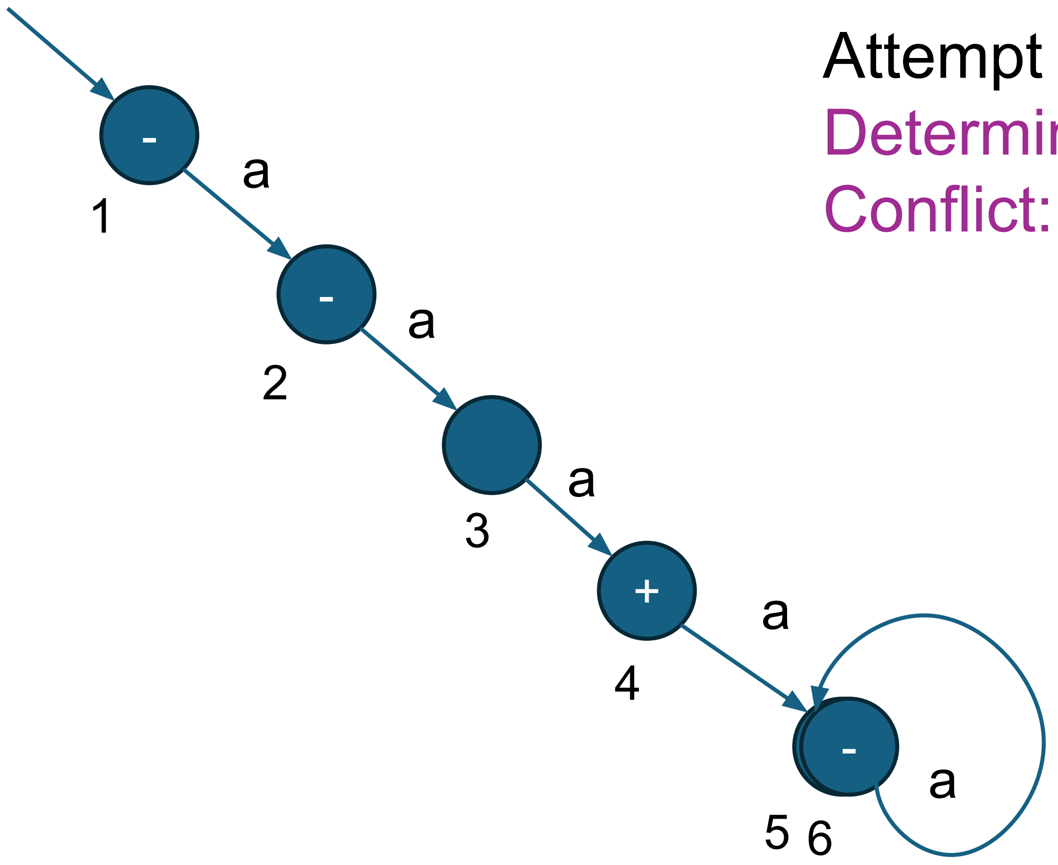


Attempt merge: 4 + 5
Determinisation: (4,5) + 6
Conflict: 4 ! 6

Possible Merges:

5 + 6
6 + 1
6 + 2
6 + 3
5 + 1
5 + 2

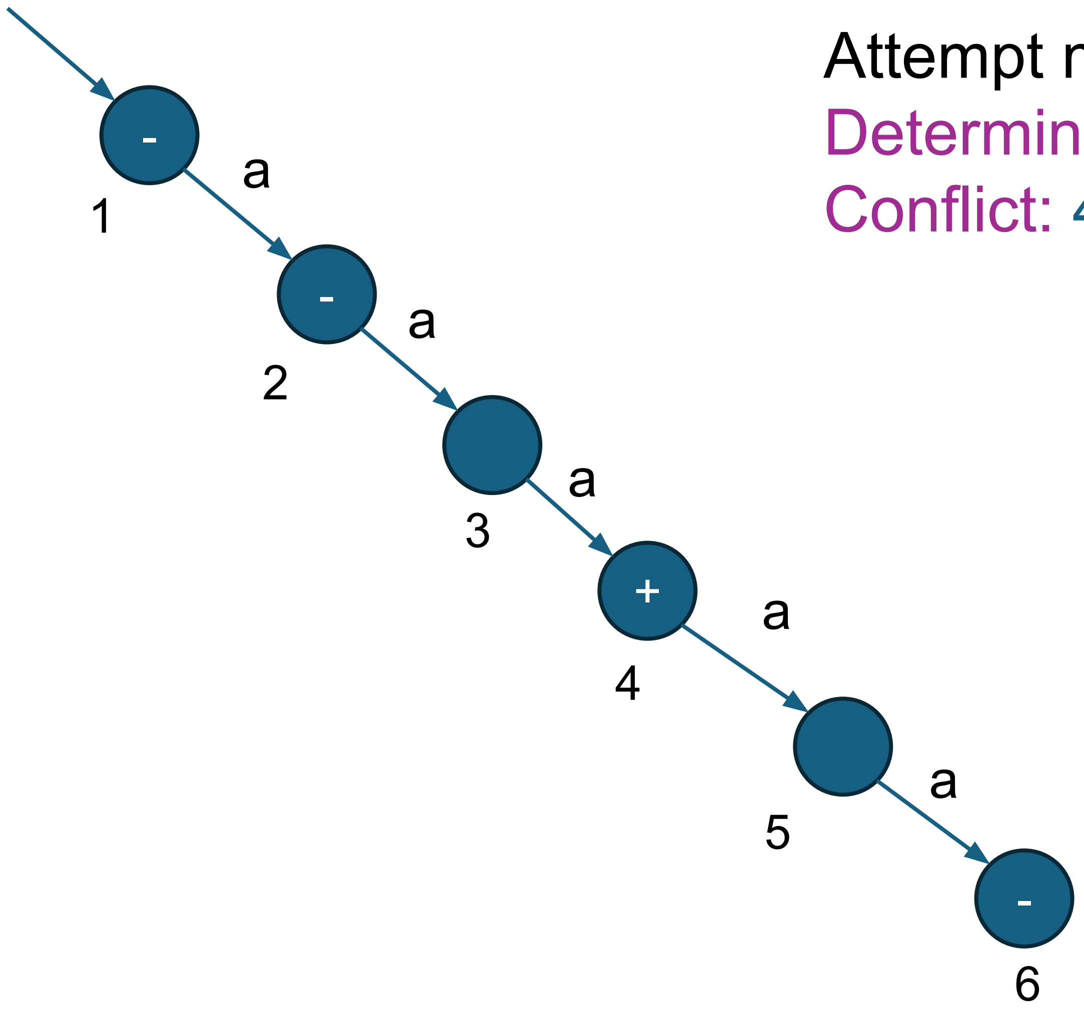
Merging (and determinisation)



Attempt merge: 4 + 5
Determinisation: (4,5) + 6
Conflict: 4 ! 6

Possible Merges:
5 + 6
6 + 1
6 + 2
6 + 3
5 + 1
5 + 2

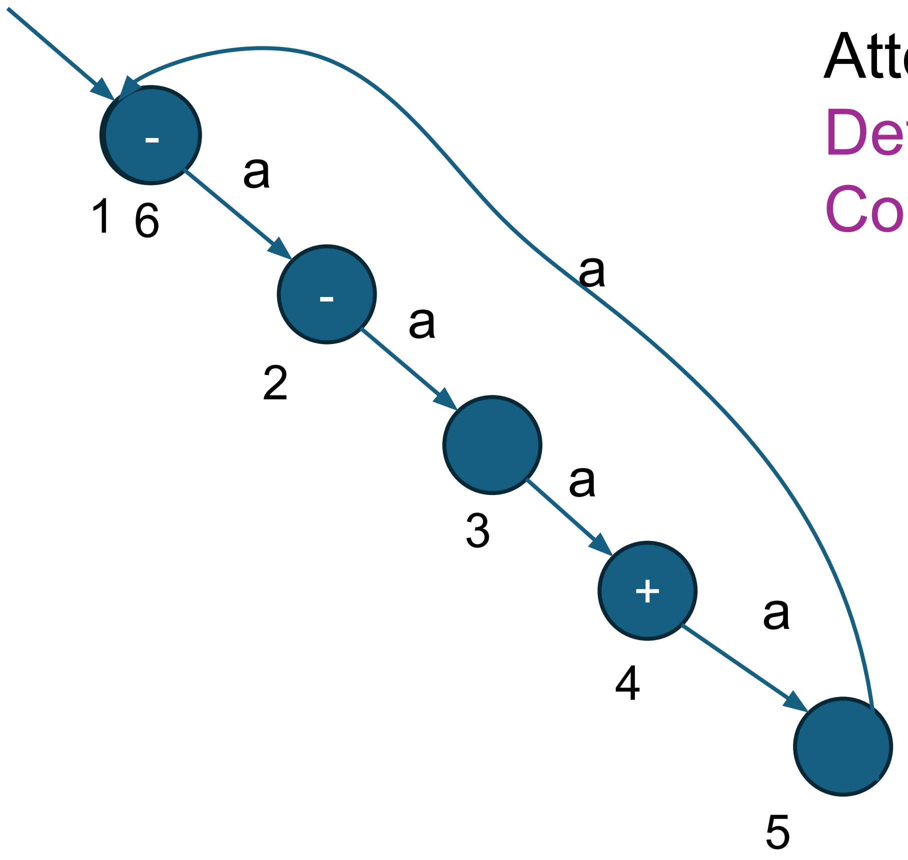
Merging (and determinisation)



Attempt merge: 4 + 5
Determinisation: (4,5) + 6
Conflict: 4 ! 6

Possible Merges:
5 + 6
6 + 1
6 + 2
6 + 3
5 + 1
5 + 2

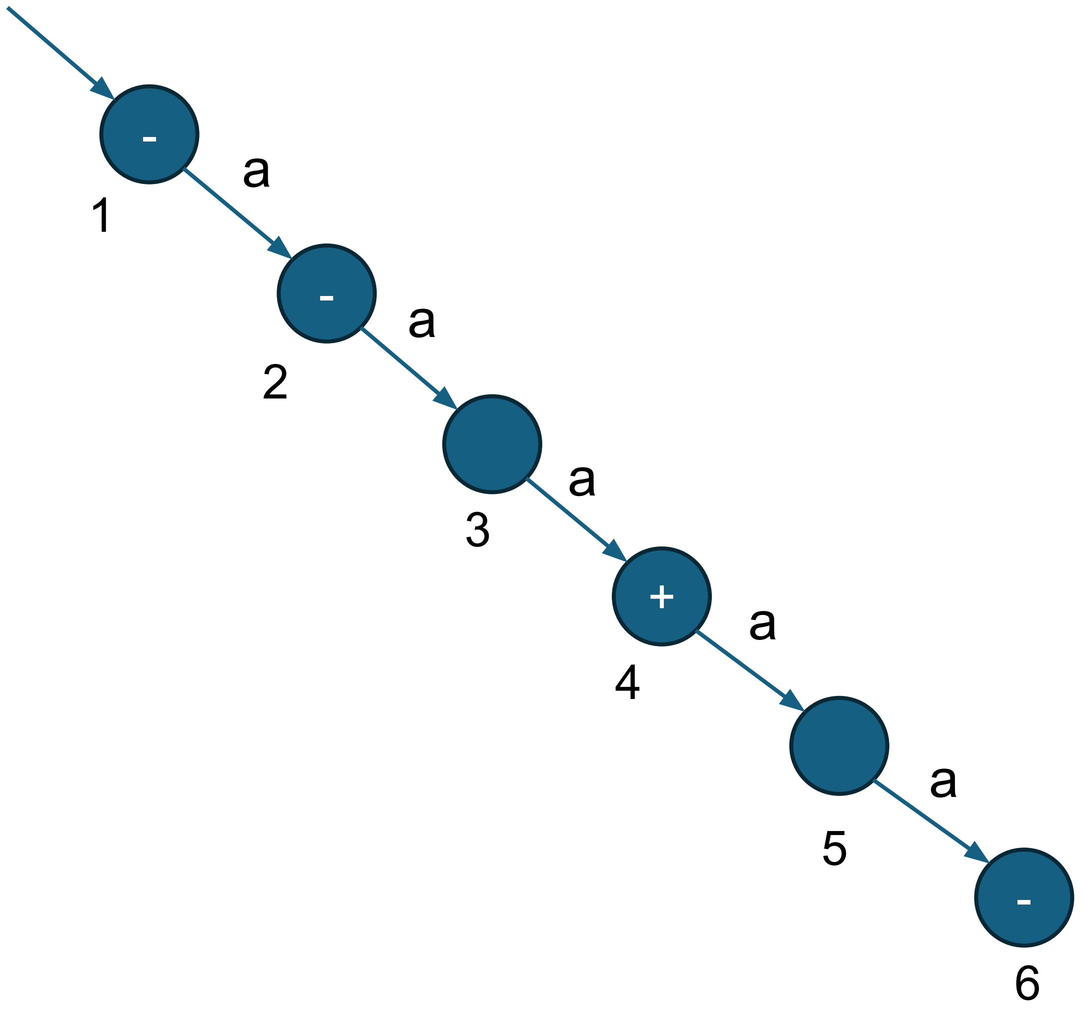
Merging (and determinisation)

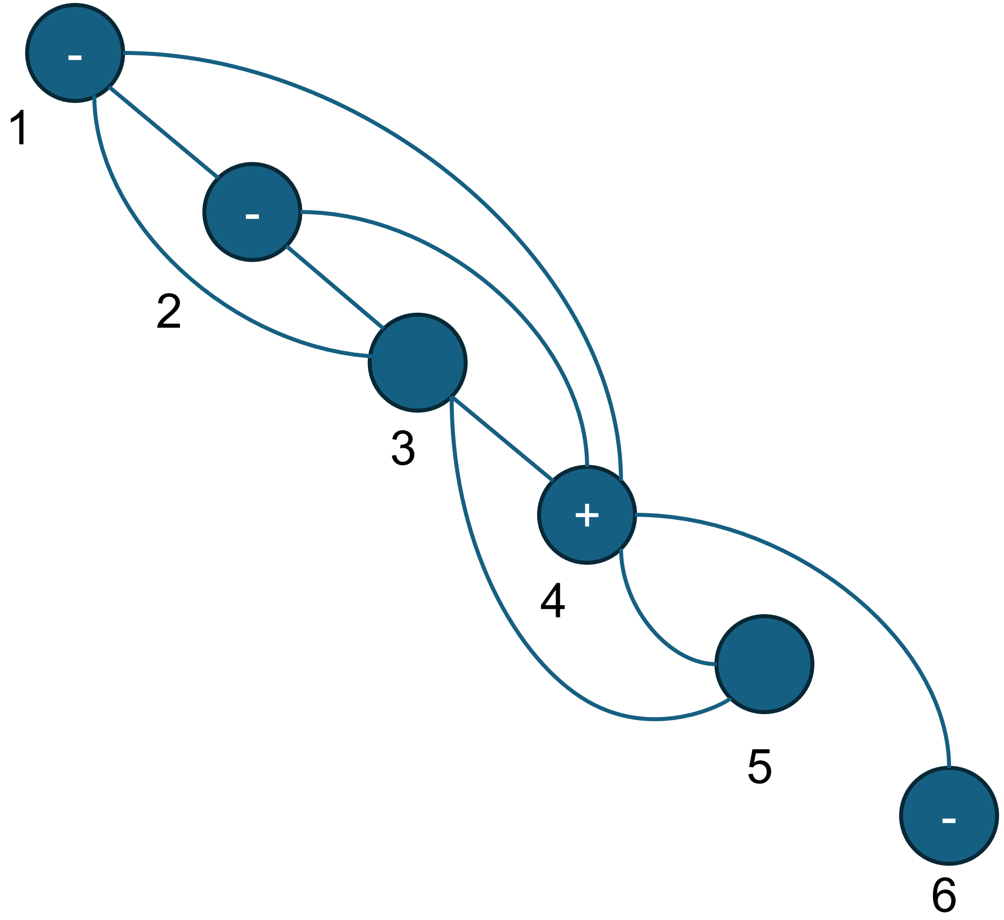


Attempt merge: 4 + 5
Determinisation: (4,5) + 6
Conflict: 4 ! 6

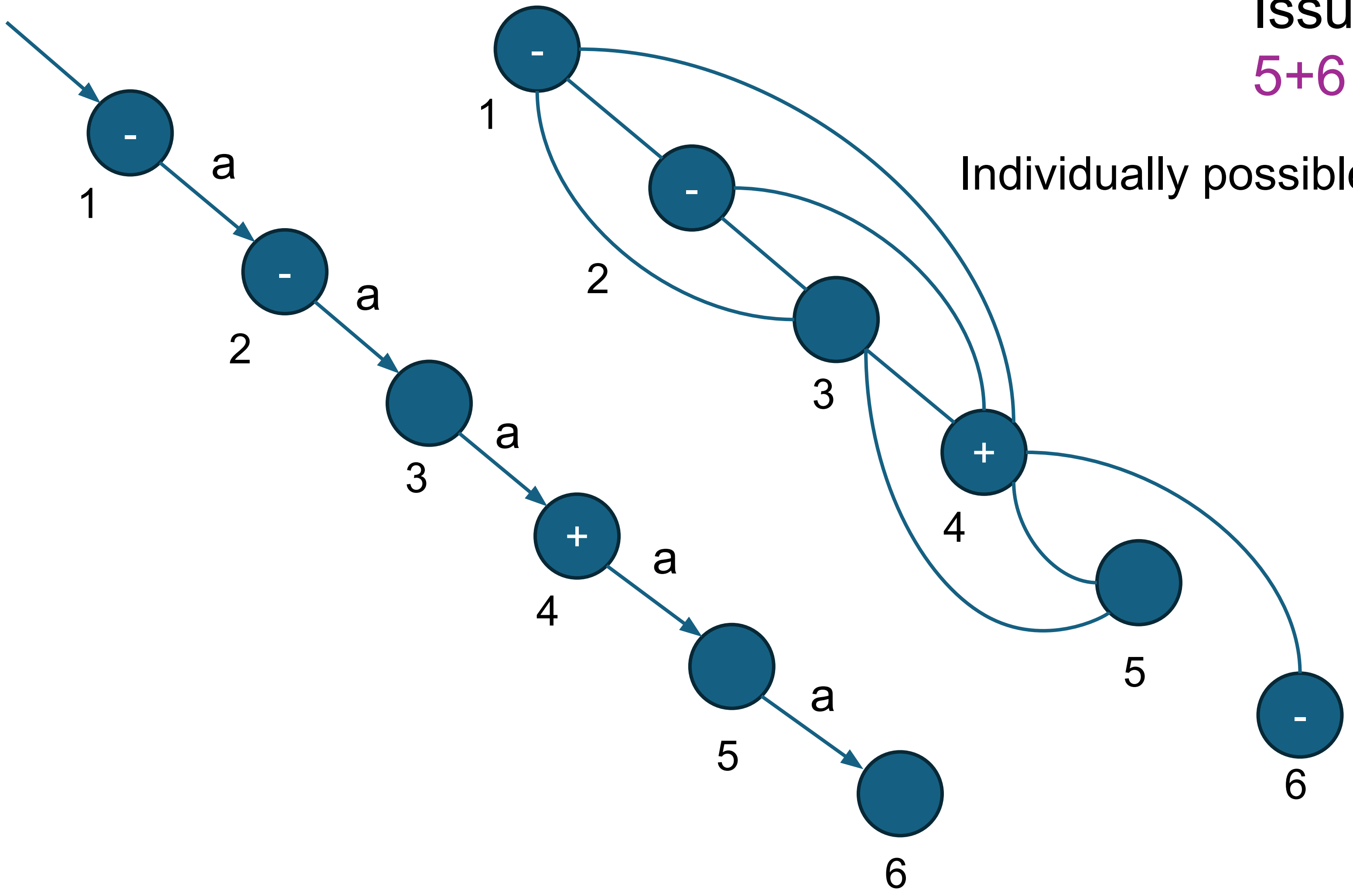
Possible Merges:
5 + 6
6 + 1
6 + 2
6 + 3
5 + 1
5 + 2

Conflict Graph





Colouring the Conflict Graph



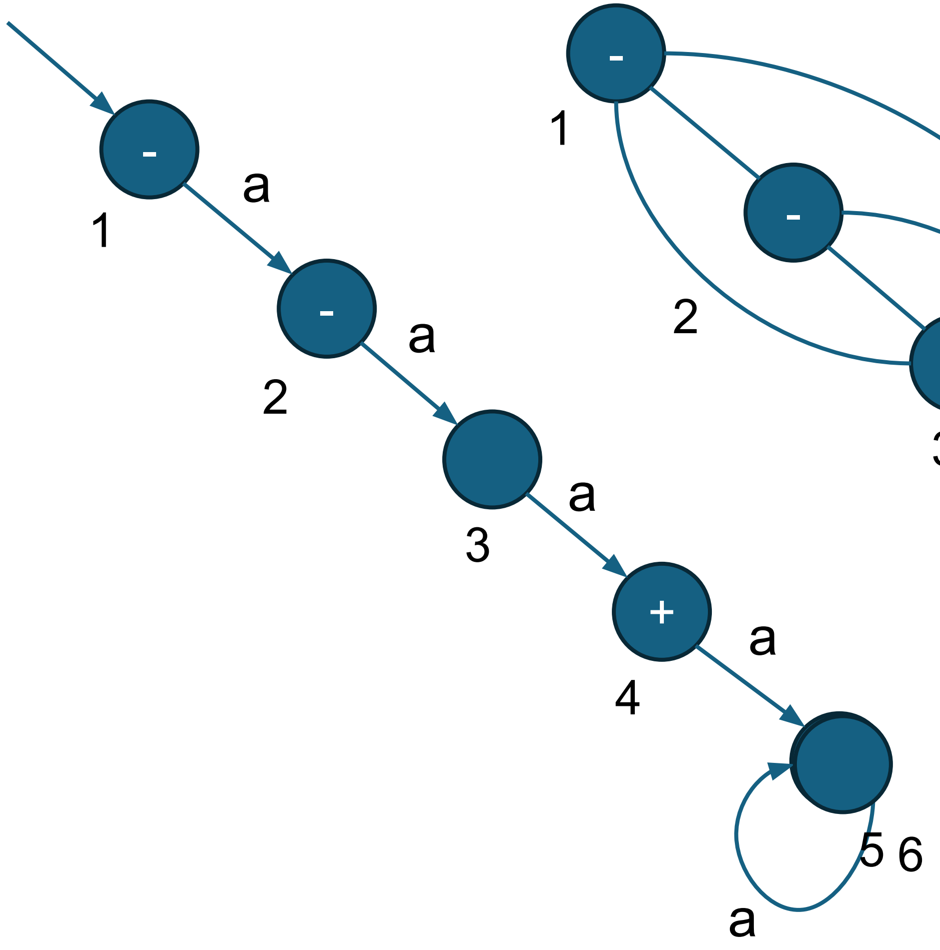
Issue:

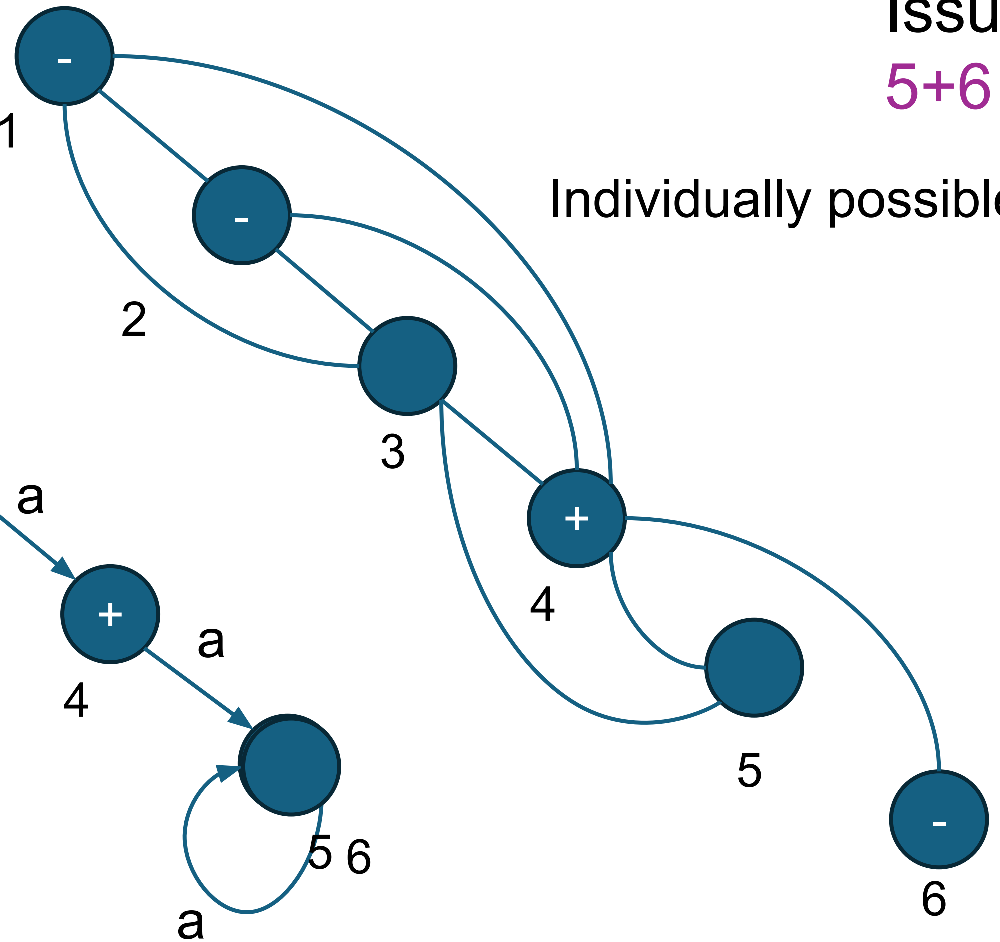
5+6 5+1 6+1

Individually possible

Not possible in combination!

Colouring the Conflict Graph



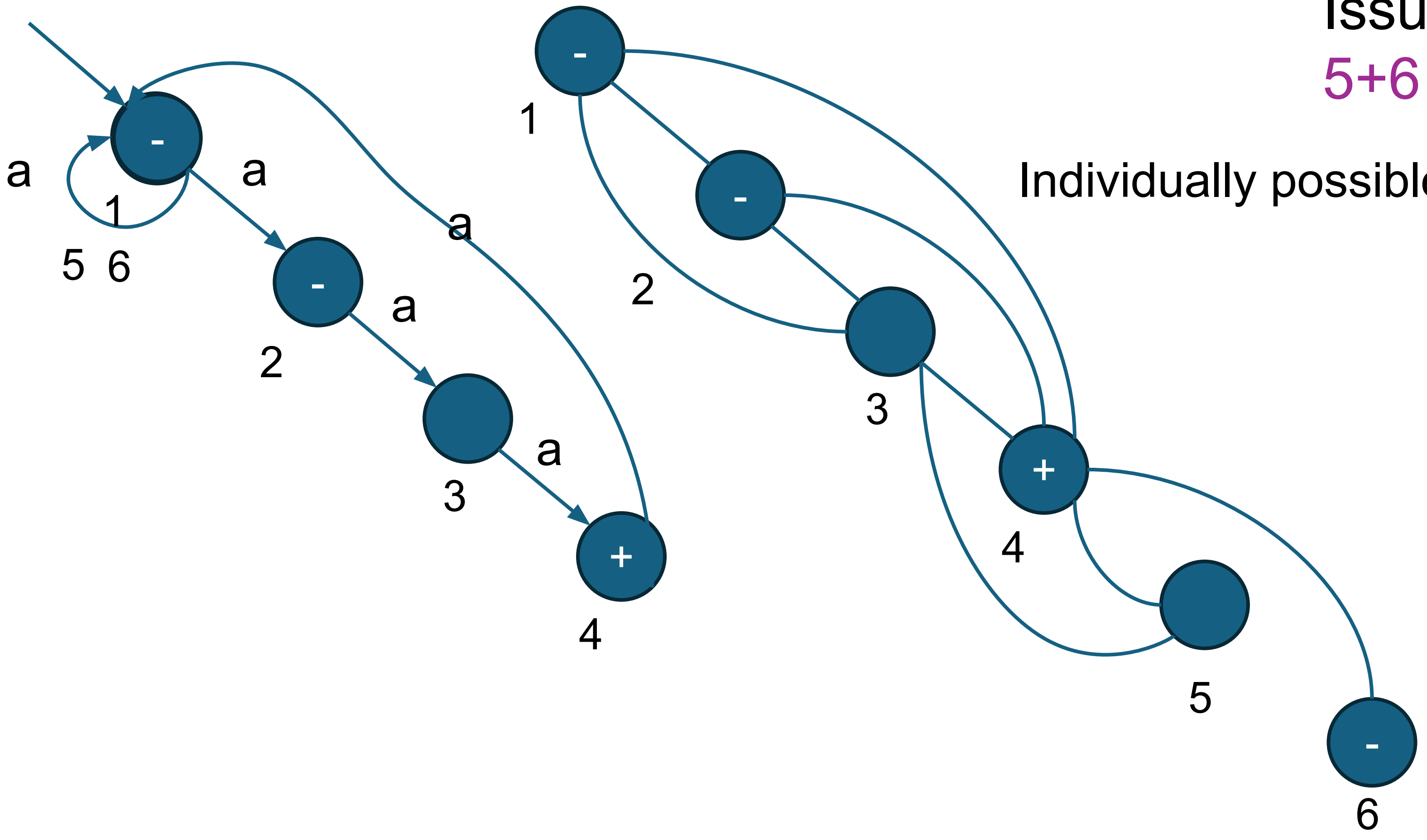


Issue:

5+6   5+1   6+1

Individually possible   Not possible in combination!

Colouring the Conflict Graph

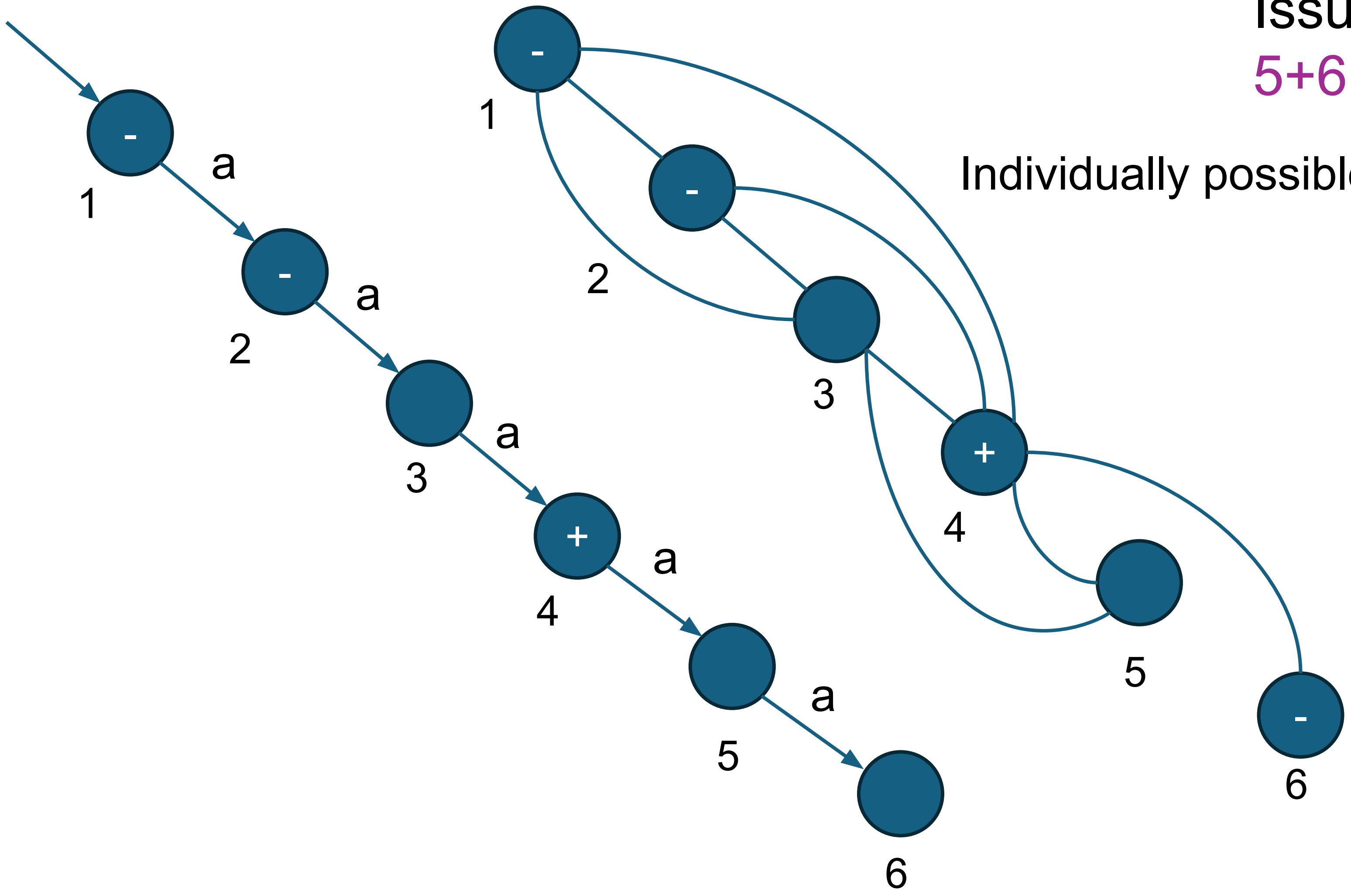


Issue:

5+6 5+1 6+1

Individually possible Not possible in combination!

Colouring the Conflict Graph



Issue:

5+6 5+1 6+1

Individually possible Not possible in combination!

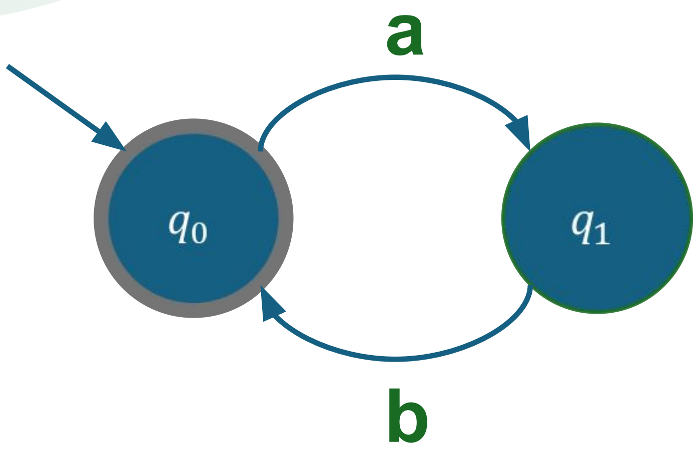
Solution: Determinisation also enforced for colouring. (Heule, Verwer 2010)

For probabilistic models: Spectral learning (Balle et al 2014)

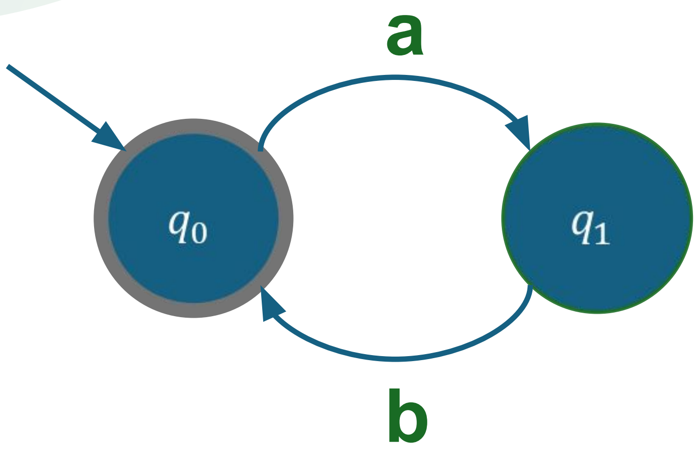
Learning in the limit (Gold)

- A language is learnable in the limit if:
  - There exists an ordering of words in a language such that there exists a learner that will, after seeing some finite sequence in that order, return the correct solution and will never change its output again upon seeing further words.
- All regular languages are learnable in the limit
- Context-free languages are not learnable in the limit

Learning bidirectional DFA

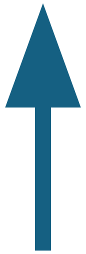


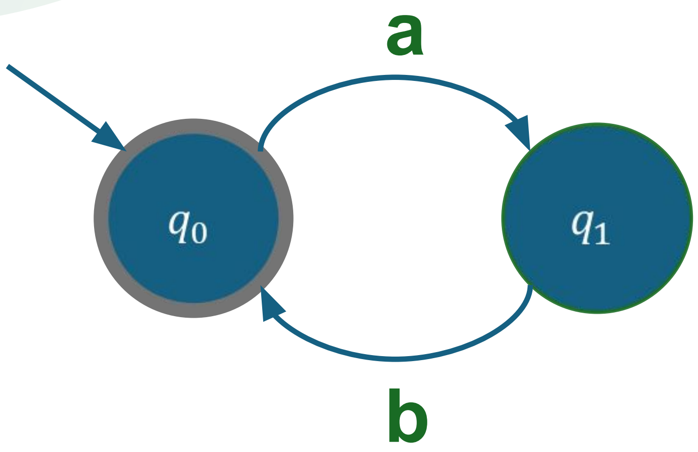
$$L_1 = (ab)^n$$



$$L_1 = (ab)^n$$

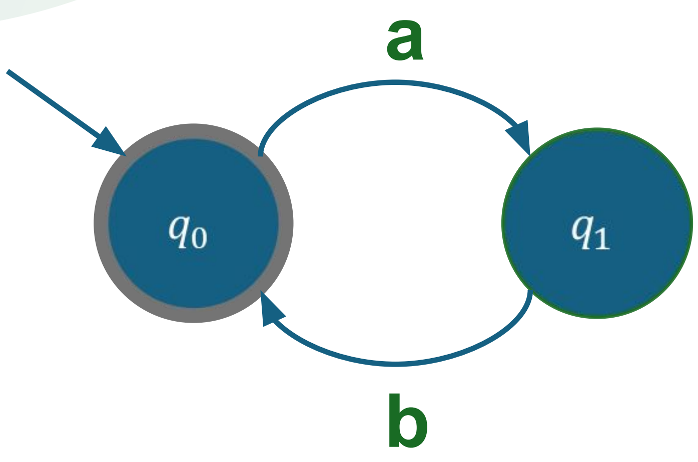
a b a b a b a b a b a b a b





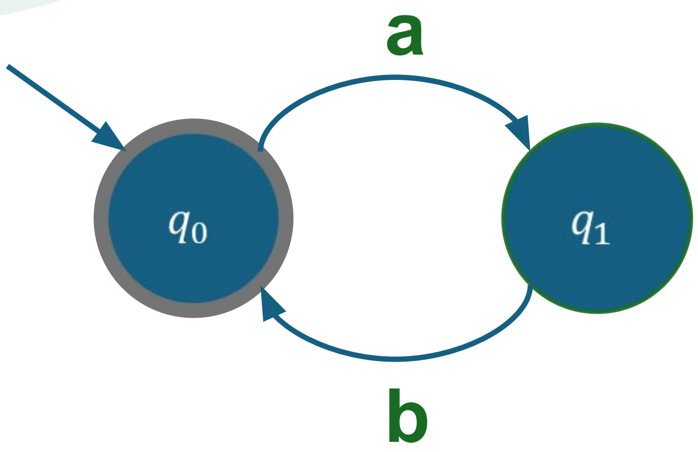
$$L_1 = (ab)^n$$

a b a b a b a b a b a b a b



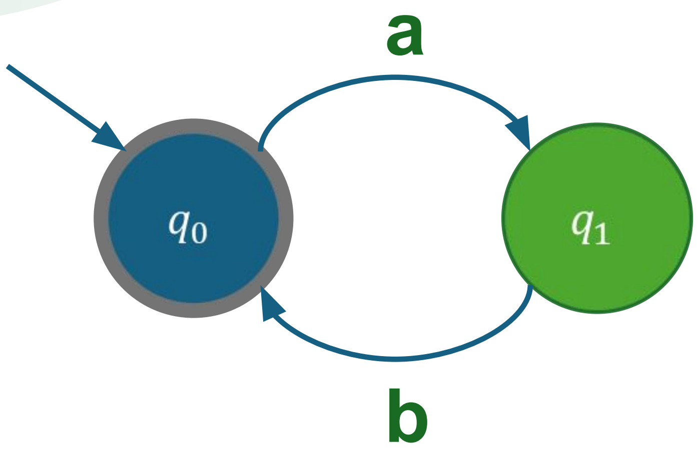
$$L_1 = (ab)^n$$

a b a b a b a b a b a b a b

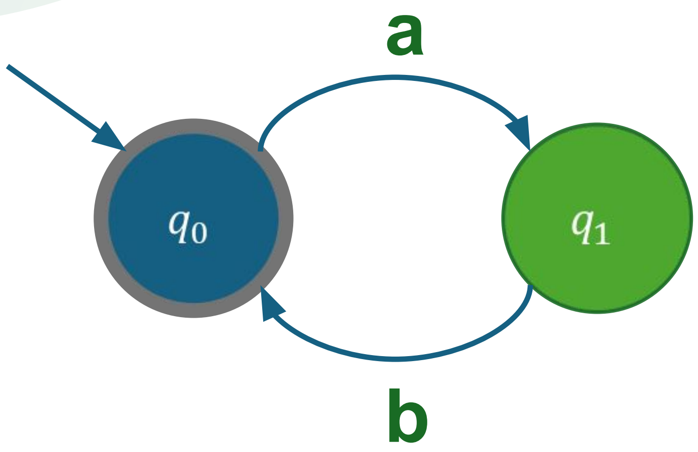


$$L_1 = (ab)^n$$

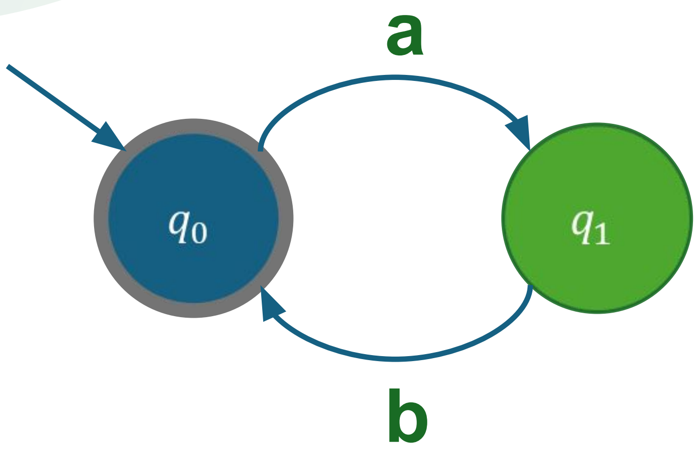
a b a b a b a b a b a b a b



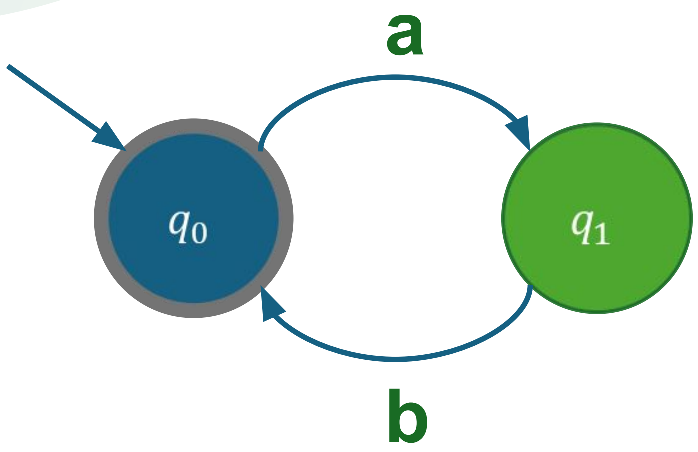
a b a b a b a b a b a b a b



a b a b a b a b a b a b a b

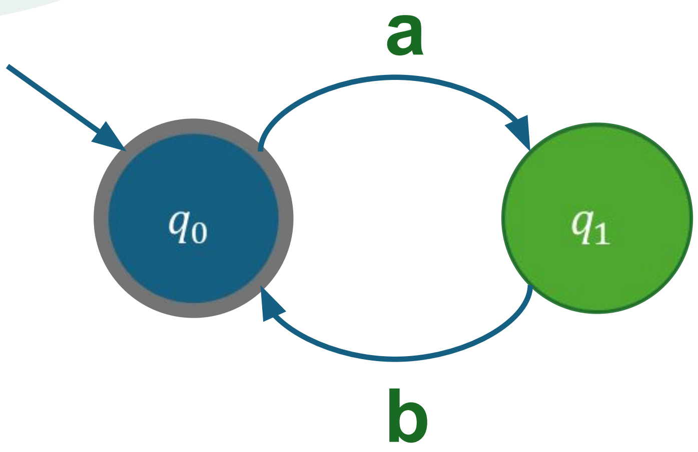


a b a b a b a b a b a b a b



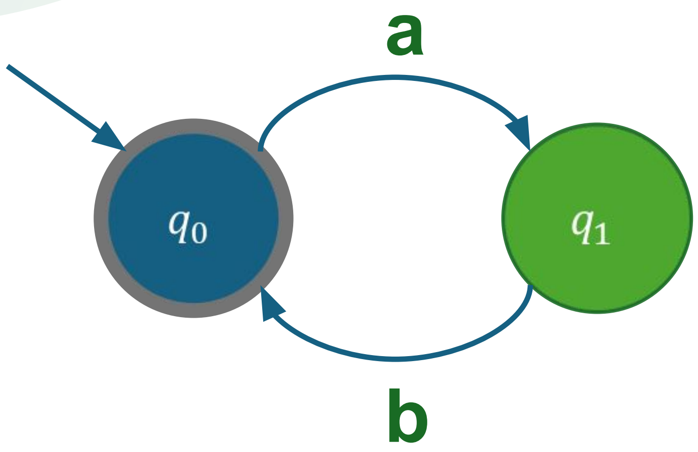
$$L_1 = (ab)^n$$

a b a b a b a b a b a b a b

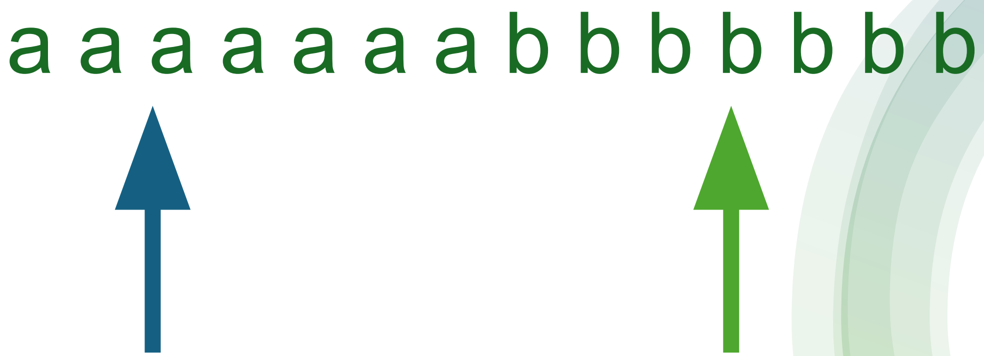


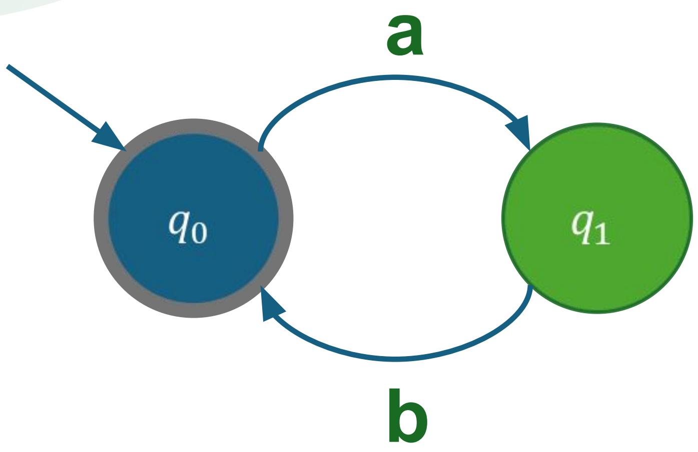
$$L_2 = a^n b^n$$

a a a a a a a b b b b b b b



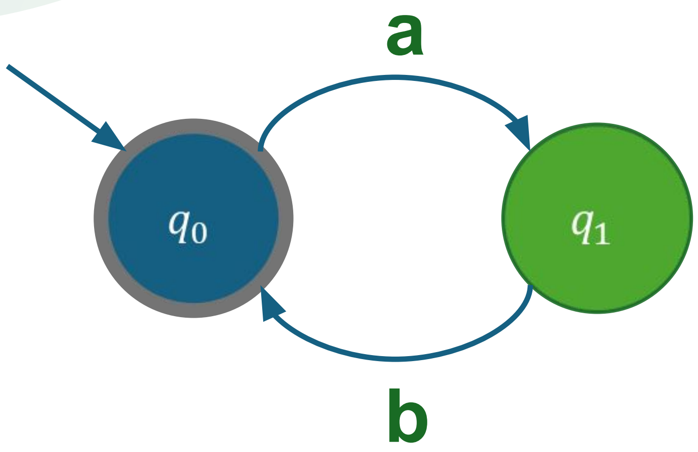
$$L_2 = a^n b^n$$



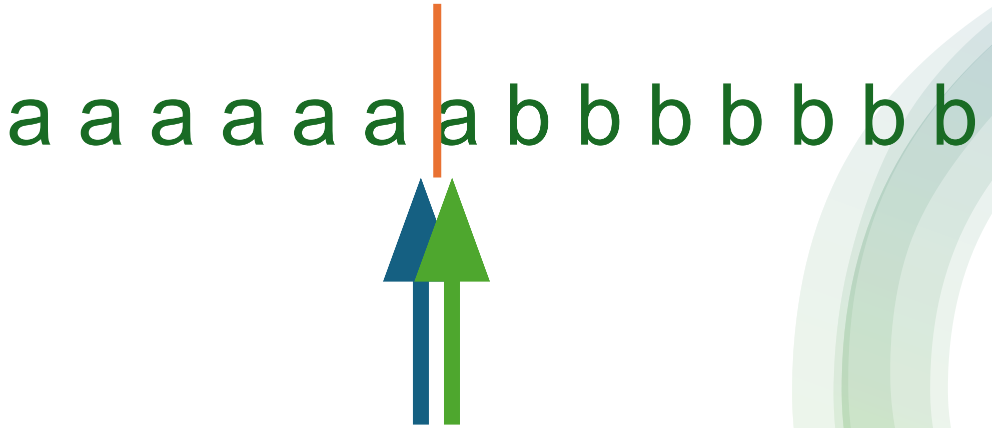


$$L_2 = a^n b^n$$

a a a a a a a b b b b b b b

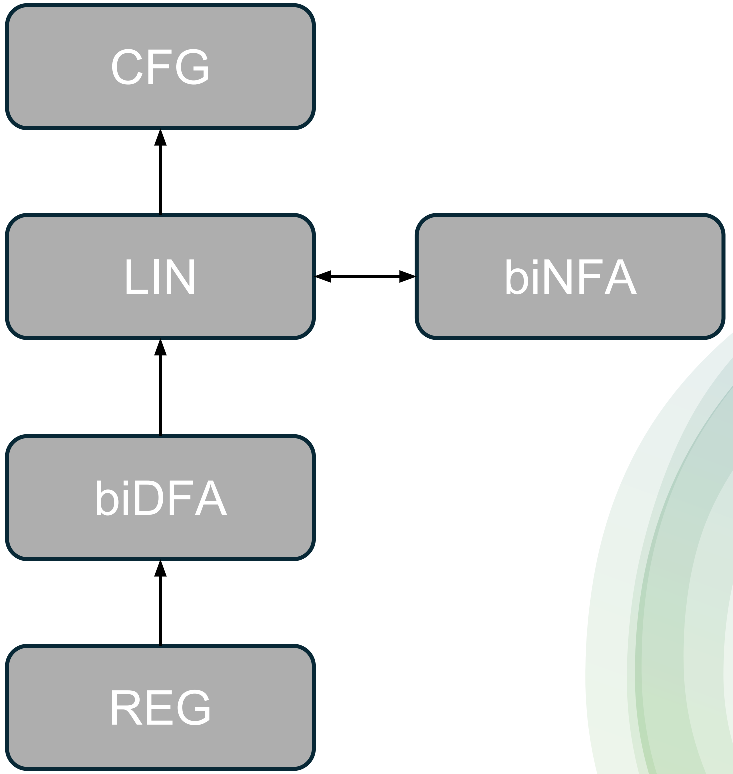


$$L_2 = a^n b^n$$

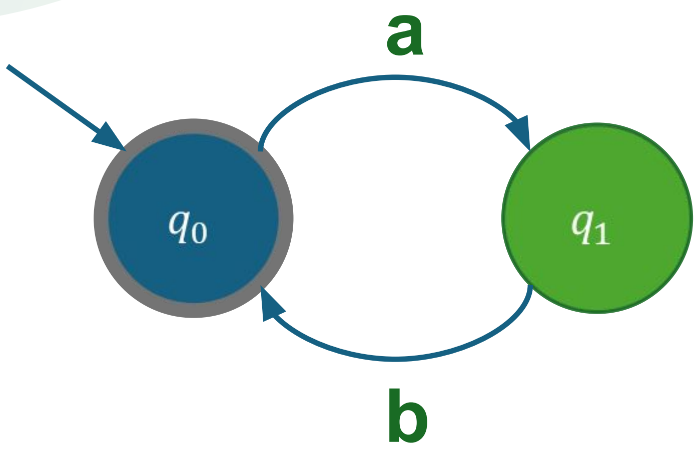


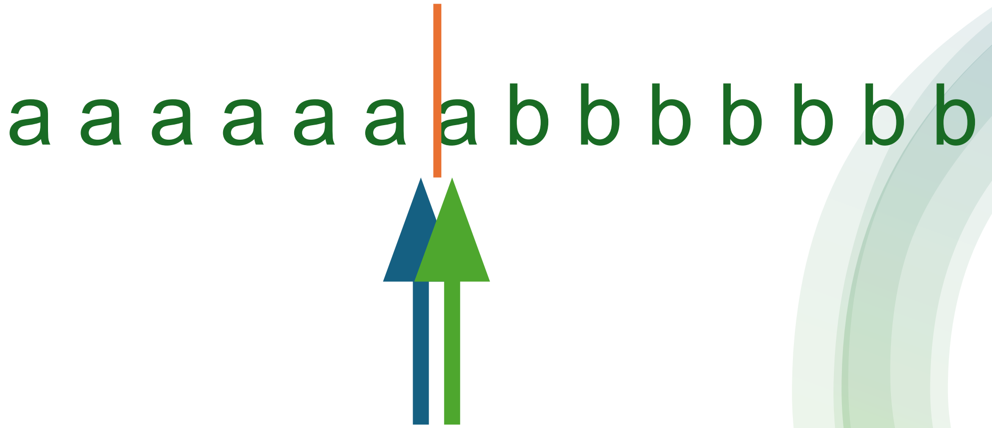
Expressiveness (Jirásková and Klíma)

biDFA recognise a proper subset of the linear languages and a proper superset of the regular languages

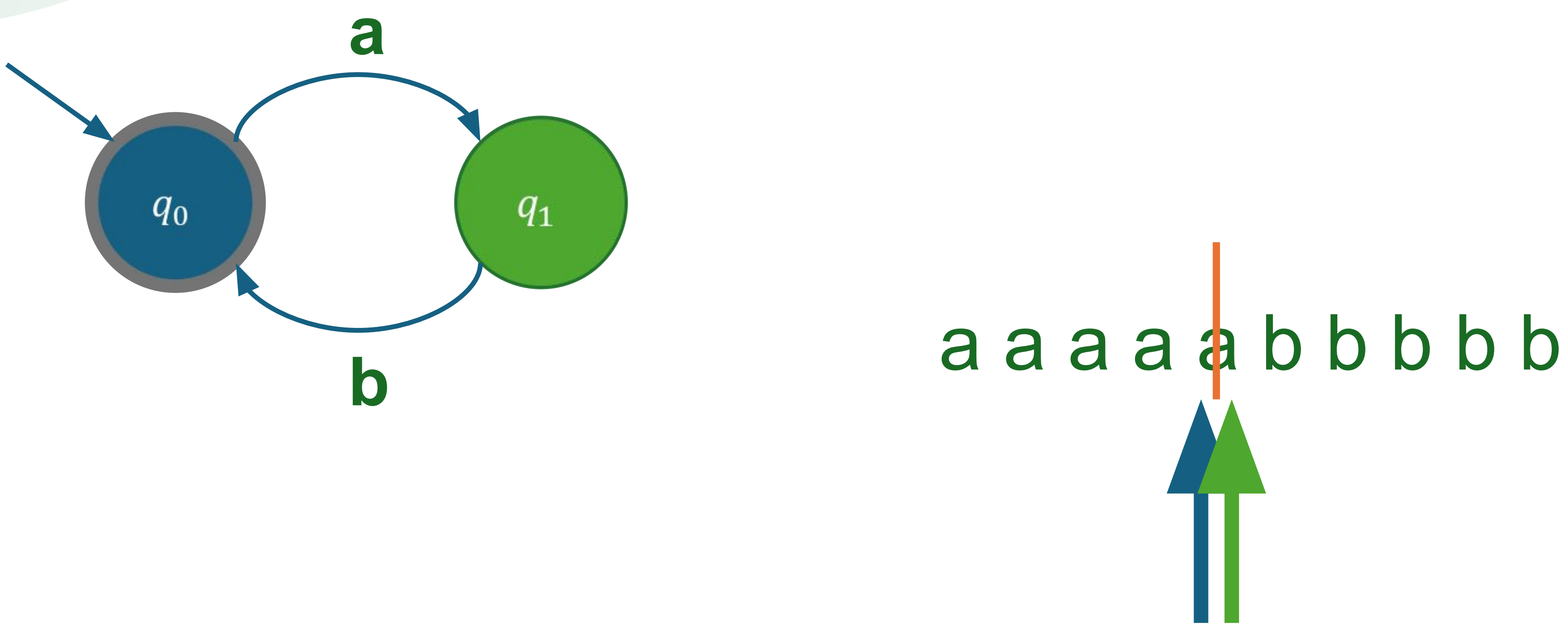


A Myhill-Nerode style theorem for biDFA?

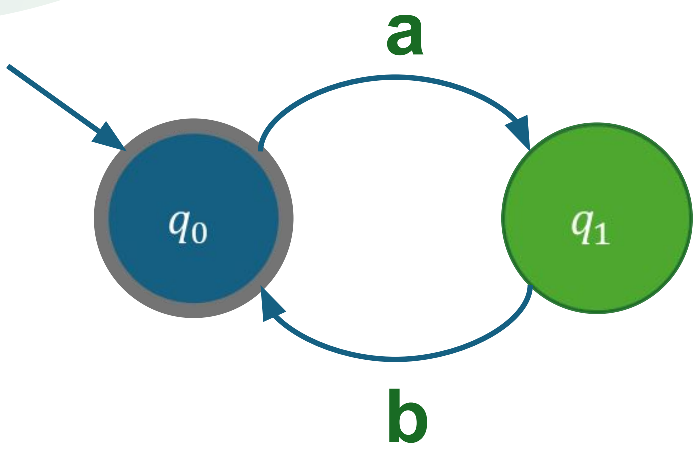




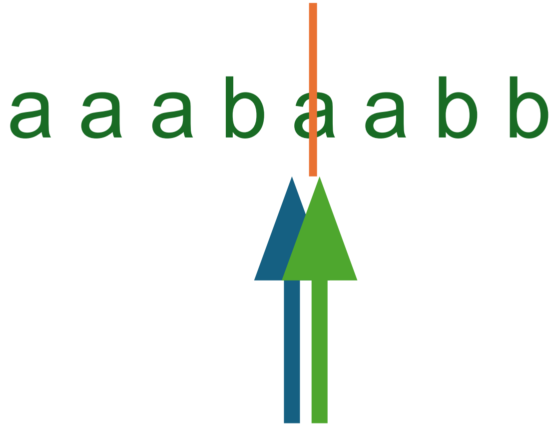
A Myhill-Nerode style theorem for biDFA?



A Myhill-Nerode style theorem for biDFA?



If one is given a function $c: \Sigma^* \to \mathbb{N}$ that assigns to every word a centre, does this together with a language L define a biDFA?



The centre function for biDFA

- Given a centre function $c: \Sigma^* \to \mathbb{N}$ one can define a c-insertion operation:

- $u \cdot_c v = u_1 u_2 \dots u_{c(u)} \cdot v \cdot u_{c(u)+1} \dots u_n$. The word v is inserted after the c(u)-th character

- Intuitively, this is what a biDFA does on a transition

The centre function for biDFA

- In order to obtain a biDFA from a centre function it needs to fulfil two conditions:

  - Stability: If $u \cdot_c s = v$, for $u \in \Sigma^*, s \in \Sigma$, then $\mathrm{c(v)} - \mathrm{c(u)} \leq 1$

  - Strict stability:
    - Given a language L we can define an equivalence relation:

      $u \equiv_{c,L} v$ iff $\forall w \in \Sigma^*\colon\ u \cdot_c w \in L \ \Leftrightarrow v \cdot_c w \in L$
    - c is strictly table w.r.t. a language L if

      $u \equiv_{c,L} v \ \Rightarrow \forall s, s' \in \Sigma\colon c(u) = c(u \cdot_c s) \Leftrightarrow c(v) = c(v \cdot_c s')$

The Myhill-Nerode theorem for biDFA

Theorem (Dieck 2023)

1. A language L is recognisable by a biDFA if there exists a strictly stable centre function c s.t. the number of the equivalence classes w.r.t c and L is finite.

2. The number of states in the biDFA which is minimal w.r.t. c and L is equal to the number of equivalence classes w.r.t. c and L.

3. The set Σ is a generating set for $[\Sigma^*]_{c,L}$ using "$\cdot_c$". The graph generated by Σ over $[\Sigma^*]_{c,L}$ is isomorphic to every minimal DFA recognising L (It is the canonical minimal DFA).

Active Learning for biDFA

Inference problem for a biDFA recognisable language L

Given:

- A membership oracle for L
- An equivalence oracle for L
- An orientation oracle

Result:

- A biDFA recognising L that is minimal w.r.t. the state orientations given by the

	o		a	b
_	left	1	0	0
a	right	0	0	1
b	left	0	0	0

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

Active Learning for biDFA

Inference problem for a biDFA recognisable language L

Given:
- A memb
- An equiv
- An orien

Result:
- A biDFA recognising L that is minimal w.r.t. the state orientations given by the

As long as every row in the top table is unique the orientations in the extended table are not required to fill it.

Counterexample processing in the style of Rivest and Schapire (1993) can maintain this property

	o		a	b
_	left	1	0	0
				1
				0

				b
				0
ab				0
ba		0	0	0
bb		0	0	0

Theoretical Corollaries

			No restrictions
	NL-complete	**NL-complete**	Undecidable
	NL-complete	**In NL**	
	NP-complete	**In P**	NP-hard

Problems

- Generally, we are not given a strictly stable centre function
  - We need to use a heuristic to guess it in practice
- We do not even know if equivalence is decidable
  - (For arbitrary centre functions)
- Is the lack of a given centre function an issue for learning in the limit when doing passive learning?

- If you have a Myhill-Nerode style theorem, you can likely also obtain a learning algorithm
- There are several automata types out there that would be interesting for learning
  - E.g. Translucent letter automata
- The learning algorithm also often translates into a minimisation algorithm

Thank you for listening!

Heuristics

- Only membership oracle is normally available in practice
- Equivalence is an open problem even if a biDFA is given as teacher
- Finding a good state orientation is the goal and as such not given

Heuristics

- Only membership oracle is normally available in practice
- Equivalence is an open problem even if a biDFA is given as teacher
- Finding a good state orientation is the goal and as such not given

→

For DFA inference (original L*) an equivalence oracle constitutes an NP oracle.

Heuristics for DFA equivalence exist.

W-method

Chow (1978)

Given a set A of all access words for every state of a DFA B and a set W of all words up to length k:

If B has the same output on all words aw, where $a \in A, w \in W$ as an automata C, then B and C are either equivalent or

$$|C| > |B| + k$$

W-method

Chow (1978)

Given a set A of all access words for every state of a DFA B and a set W of all words up to length k:

If B has the same output on all words aw, where $a \in A, w \in W$ as an automata C, then B and C are either equivalent or

$$|C| > |B| + k$$

- Original proof introduces a relation between the two automata
- This does not work with partially defined state orientations
- Proof needs to be independent of how states are oriented after parsing a

W-method for biDFA

Dieck and Verwer (2025)

Given a set A of all access words for every state of a biDFA B and a set W of all words up to length k:

If B has the same output on all words aw, where $a \in A, w \in W$ as an automata C, then assuming $c_B(a) = c_C(a)$ for all $a \in A$ B and C are either equivalent or

$$|C| > |B| + k$$

Orientation heuristic

Goal: Minimise the number of states

Idea: Be locally greedy

Orientation heuristic

	o		a	b
_	left	1	0	0
a				
b	left	0	0	0

	o		a	b
_	left	1	0	0
a				
b	left	0	0	0

			a	b
aa				
ab				
ba		0	0	0
bb		0	0	0

			a	b
aa				
ab				
ba		0	0	0
bb		0	0	0

Orientation heuristic

	o		a	b
_	left	1	0	0
a	left	0	0	1
b	left	0	0	0

	o		a	b
_	left	1	0	0
a	right	0	0	1
b	left	0	0	0

			a	b
aa				
ab				
ba		0	0	0
bb		0	0	0

			a	b
aa				
ab				
ba		0	0	0
bb		0	0	0

Orientation heuristic

	o		a	b
_	left	1	0	0
a	left	0	0	1
b	left	0	0	0

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

	o		a	b
_	left	1	0	0
a	right	0	0	1
b	left	0	0	0

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

Orientation heuristic

	o		a	b	ab
_	left	1	0	0	1
a	left	0	0	1	0
b	left	0	0	0	0

	o		a	b	ab
_	left	1	0	0	1
a	right	0	0	1	0
b	left	0	0	0	0

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

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

Orientation heuristic

	o		a	b	ab
_	left	1	0	0	1
a	left	0	0	1	0
b	left	0	0	0	0

	o		a	b	ab
_	left	1	0	0	1
a	right	0	0	1	0
b	left	0	0	0	0

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

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

Orientation heuristic

	o		a	b	ab
_	left	1	0	0	1
a	left	0	0	1	0
b	left	0	0	0	0

	o		a	b	ab
_	left	1	0	0	1
a	right	0	0	1	0
b	left	0	0	0	0

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

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

Orientation heuristic

	o		a	b	ab
_	left	1	0	0	1
a					
b					

	o		a	b	ab
_	left	1	0	0	1
a					0
b					0

- If an orientation decision does not have an immediate effect locally the heuristic can not make a good judgement (Longer loops are an issue)
- If not enough future traces are tested the heuristic might base its decision on wrong information

aa					
ab					
ba		0	0	0	0
bb		0	0	0	0

aa					0
ab					1
ba		0	0	0	0
bb		0	0	0	0

Orientation heuristic

Language		Without prefilling columns	With prefilled columns
	#Minimum Discovered	6/10	10/10
	#100+ states	0/10	0/10
	Average biDFA size	5.8	3
	#Minimum Discovered	5/10	10/10
	#100+ states	0/10	0/10
	Average biDFA size	7.1	5
	#Minimum Discovered	6/10	8/10
	#100+ states	0/10	0/10
	Average biDFA size	7.3	4.5

Orientation heuristic

Language		Without prefilling columns	With prefilled columns
	#Minimum Discovered	1/10	5/10
	#100+ states	0/10	0/10
	Average biDFA size	8.8	6.3
	#Minimum Discovered	2/10	10/10
	#100+ states	8/10	0/10
	Average biDFA size	5*	5
	#Minimum Discovered	0/10	2/10
	#100+ states	10/10	5/10
	Average biDFA size	_*	10.8

*Only runs that did not get forcibly terminated counted towards the average

Thank you for listening!