Finite-State Automata: Recap

Deepak D'Souza

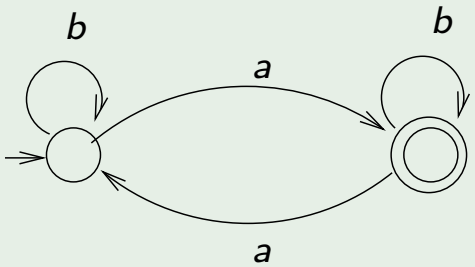
Department of Computer Science and Automation
Indian Institute of Science, Bangalore.

07 August 2019

Outline

Example DFA 1

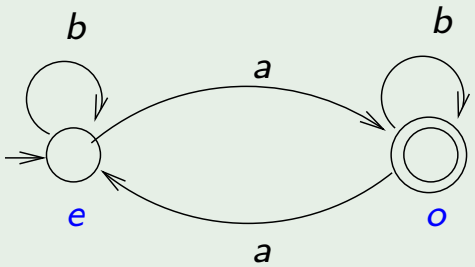
DFA for "Odd number of a's"



- How a DFA works.

Example DFA 1

DFA for "Odd number of a's"



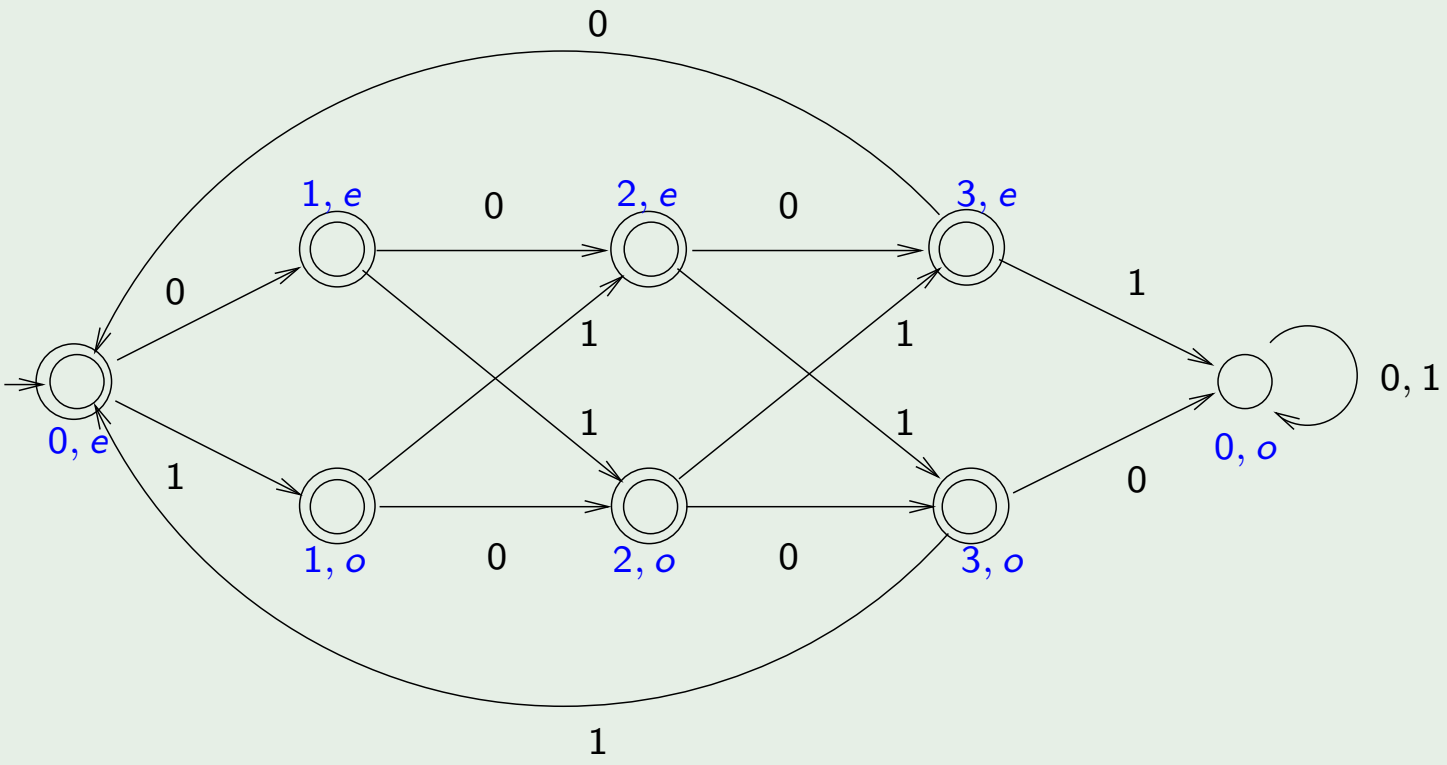
- How a DFA works.
- Each state represents a property of the input string read so far:
  - State e: Number of a's seen is even.
  - State o: Number of a's seen is odd.

Example DFA 2

Accept strings over $\{0, 1\}$ which have even parity in each length 4 block.

- Accept "$0101 \cdot 1010$"
- Reject "$0101 \cdot 1011$"

DFA for "Even parity checker"

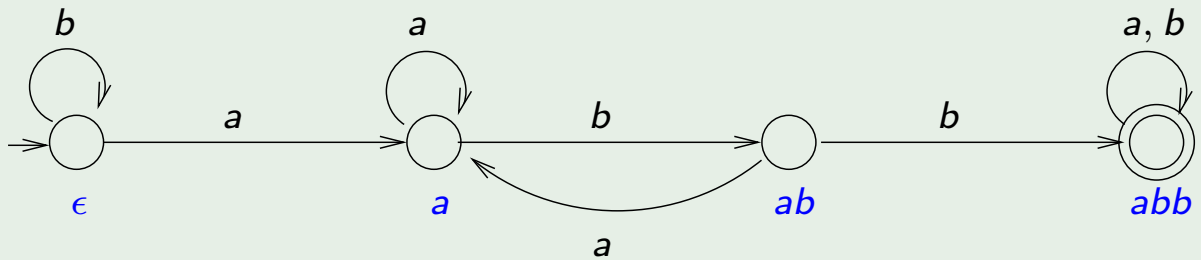


Example DFA 3

DFA for "Strings over $\{a, b\}$ that contain the substring abb"

Example DFA 3

DFA for "Strings over $\{a, b\}$ that contain the substring abb"



Each state represents a property of the input string read so far:

- State ϵ: Not seen abb and no suffix in a or ab.
- State a: Not seen abb and has suffix a.
- State ab: Not seen abb and has suffix ab.
- State abb: Seen abb.

Definitions and notation

- An *alphabet* is a finite set of symbols or "letters". Eg. $A = \{a, b, c\}$ or $\Sigma = \{0, 1\}$.
- A *string* or *word* over an alphabet A is a finite sequence of letters from A. Eg. *aaba* is string over $\{a, b, c\}$.
- Empty string denoted by ϵ.
- Set of all strings over A denoted by A^*.
  - What is the "size" or "cardinality" of A^*?

Definitions and notation

- An *alphabet* is a finite set of symbols or "letters". Eg. $A = \{a, b, c\}$ or $\Sigma = \{0, 1\}$.
- A *string* or *word* over an alphabet A is a finite sequence of letters from A. Eg. $aaba$ is string over $\{a, b, c\}$.
- Empty string denoted by ϵ.
- Set of all strings over A denoted by A^*.
  - What is the "size" or "cardinality" of A^*?
  - Infinite but Countable: Can enumerate in lexicographic order:

$$\epsilon,\ a,\ b,\ c,\ aa,\ ab, \ldots$$

.

Definitions and notation

- An *alphabet* is a finite set of symbols or "letters". Eg. $A = \{a, b, c\}$ or $\Sigma = \{0, 1\}$.
- A *string* or *word* over an alphabet A is a finite sequence of letters from A. Eg. $aaba$ is string over $\{a, b, c\}$.
- Empty string denoted by ϵ.
- Set of all strings over A denoted by A^*.
  - What is the "size" or "cardinality" of A^*?
  - Infinite but Countable: Can enumerate in lexicographic order:

$$\epsilon,\ a,\ b,\ c,\ aa,\ ab, \ldots$$

.

- Operation of *concatenation* on words: String u followed by string v: written $u \cdot v$ or simply uv.
  - Eg. $aabb \cdot aaa = aabbaaa$.

Definitions and notation: Languages

- A *language* over an alphabet A is a set of strings over A. Eg. for $A = \{a, b, c\}$:
  - $L = \{abc, aaba\}$.
  - $L_1 = \{\epsilon, b, aa, bb, aab, aba, baa, bbb, \ldots\}$.
  - $L_2 = \{\}$.
  - $L_3 = \{\epsilon\}$.
- How many languages are there over a given alphabet A?

Definitions and notation: Languages

- A *language* over an alphabet A is a set of strings over A. Eg. for $A = \{a, b, c\}$:
  - $L = \{abc, aaba\}$.
  - $L_1 = \{\epsilon, b, aa, bb, aab, aba, baa, bbb, \ldots\}$.
  - $L_2 = \{\}$.
  - $L_3 = \{\epsilon\}$.
- How many languages are there over a given alphabet A?
  - Uncountably infinite
  - Use a diagonalization argument:

	ϵ	a	b	aa	ab	ba	bb	aaa	aab	aba	abb	bbb	$\cdots$
L_0	0	1	0	0	0	1	1	0	0	0	0	0	$\cdots$
L_1	0	0	0	0	0	0	0	0	0	0	0	0	$\cdots$
L_2	1	1	0	1	0	1	1	0	0	1	0	1	$\cdots$
L_3	0	0	0	0	0	0	0	0	0	0	0	0	$\cdots$
L_4	0	1	0	0	0	1	1	0	0	0	0	0	$\cdots$
L_5	1	1	0	1	0	1	1	0	0	1	0	1	$\cdots$
L_6	0	1	0	0	0	1	1	0	0	0	0	0	$\cdots$
L_7	0	0	0	0	0	0	1	0	0	0	1	0	$\cdots$
$\vdots$													

Definitions and notation: Languages

- Concatenation of languages:

$$L_1 \cdot L_2 = \{u \cdot v \mid u \in L_1,\ v \in L_2\}.$$

- Eg. $\{abc, aaba\} \cdot \{\epsilon, a, bb\} =$
$\{abc, aaba, abca, aabaa, abcbb, aababb\}$.

Definitions and notation: DFA

A *Deterministic Finite-State Automaton* $\mathcal{A}$ over an alphabet A is a structure of the form

$$(Q, s, \delta, F)$$

where

- Q is a finite set of "states"
- $s \in Q$ is the "start" state
- $\delta : Q \times A \to Q$ is the "transition function."
- $F \subseteq Q$ is the set of "final" states.

Definitions and notation: DFA

A *Deterministic Finite-State Automaton* $\mathcal{A}$ over an alphabet A is a structure of the form

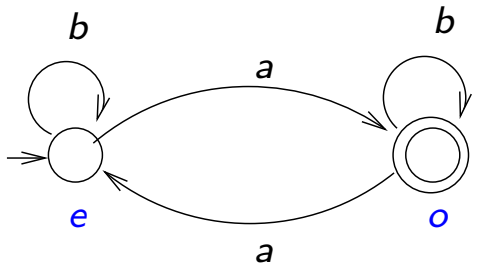
$$(Q, s, \delta, F)$$

where

- Q is a finite set of "states"
- $s \in Q$ is the "start" state
- $\delta : Q \times A \to Q$ is the "transition function."
- $F \subseteq Q$ is the set of "final" states.

Example of "Odd a's" DFA:
Here: $Q = \{e, o\}$, $s = e$, $F = \{o\}$,
and δ is given by:

$$\begin{aligned} \delta(e, a) &= o, \\ \delta(e, b) &= e, \\ \delta(o, a) &= e, \\ \delta(o, b) &= o. \end{aligned}$$



Definitions and notation: Language accepted by a DFA

- $\widehat{\delta}$ tells us how the DFA $\mathcal{A}$ behaves on a given word u.
- Define $\widehat{\delta} : Q \times A^* \to Q$ as
  - $\widehat{\delta}(q, \epsilon) = q$
  - $\widehat{\delta}(q, w \cdot a) = \delta(\widehat{\delta}(q, w), a)$.
- Language *accepted* by $\mathcal{A}$, denoted $L(\mathcal{A})$, is defined as:

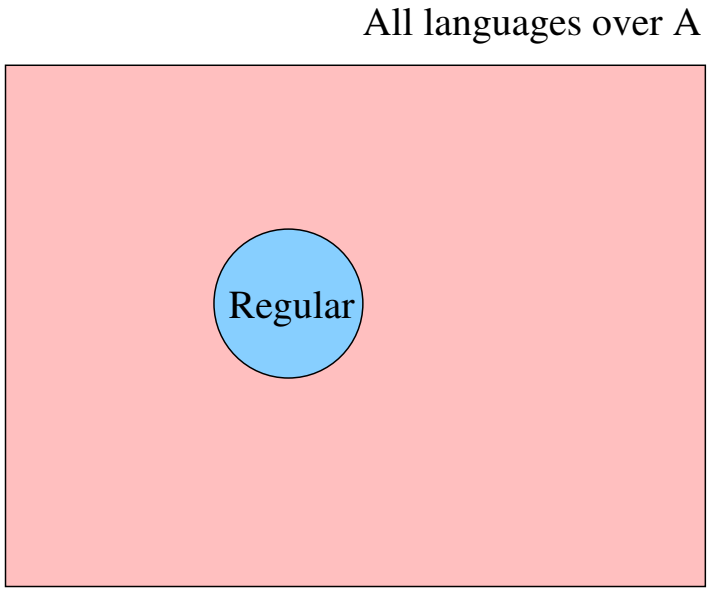
$$L(\mathcal{A}) = \{w \in A^* \mid \widehat{\delta}(s, w) \in F\}.$$

- Eg. For $\mathcal{A}$ = DFA for "Odd a's",

$$L(\mathcal{A}) = \{a, ab, ba, aaa, abb, bab, bba, \ldots\}.$$

Regular Languages

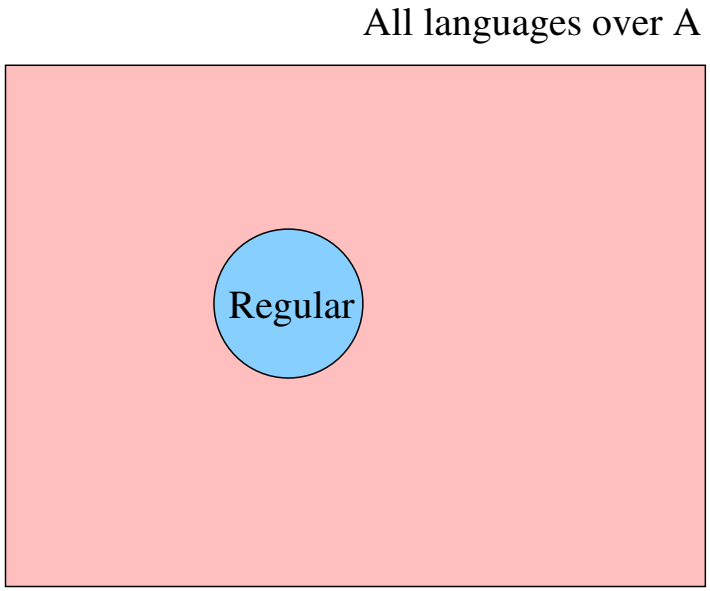
- A language $L \subseteq A^*$ is called *regular* if there is a DFA $\mathcal{A}$ over A such that $L(\mathcal{A}) = L$.
- Examples of regular languages: "Odd a's", "strings that don't end inside a C-style comment", $\{\}$, any finite language.



- Are there non-regular languages?

Regular Languages

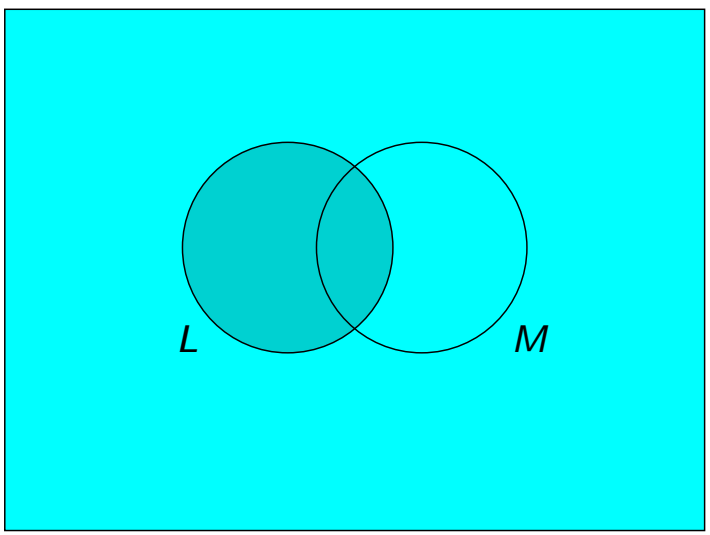
- A language $L \subseteq A^*$ is called *regular* if there is a DFA $\mathcal{A}$ over A such that $L(\mathcal{A}) = L$.
- Examples of regular languages: "Odd a's", "strings that don't end inside a C-style comment", $\{\}$, any finite language.

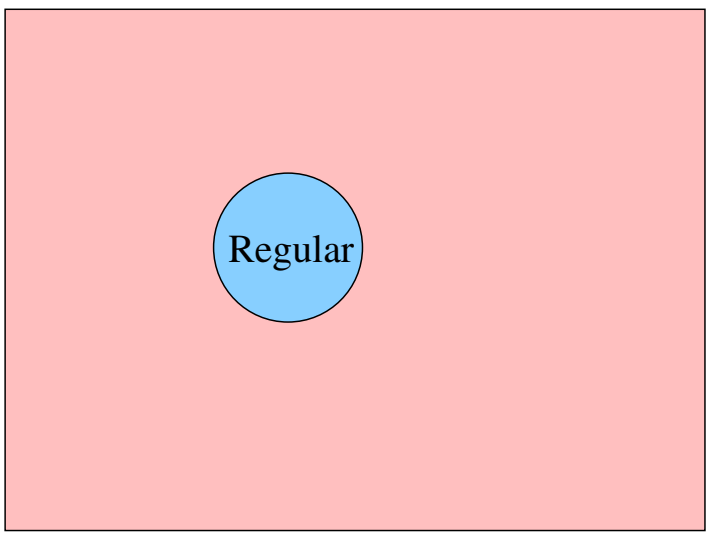


- Are there non-regular languages?
  - Yes, uncountably many, since Reg is only countable while class of all languages is uncountable.

Closure properties

- Class of Regular languages is closed under
  - Complement, intersection, and union.
  - Concatenation, Kleene iteration.
- Non-deterministic Finite-state Automata (NFA) = DFA.





Closure under complementation

- Idea: Flip final states.
- Formal construction:
  - Let $\mathcal{A} = (Q, s, \delta, F)$ be a DFA over alpahet A.
  - Define $\mathcal{B} = (Q, s, \delta, Q - F)$.
  - Claim: $L(\mathcal{B}) = A^* - L(A)$.

Proof of claim

- $L(\mathcal{B}) \subseteq A^* - L(\mathcal{A})$.

$$\begin{aligned} w \in L(\mathcal{B}) &\implies \widehat{\delta}(s, w) \in (Q - F). \\ &\implies \widehat{\delta}(s, w) \notin F \\ &\implies w \notin L(\mathcal{A}) \\ &\implies w \in A^* - L(\mathcal{A}). \end{aligned}$$

- $L(\mathcal{B}) \supseteq A^* - L(\mathcal{A})$.

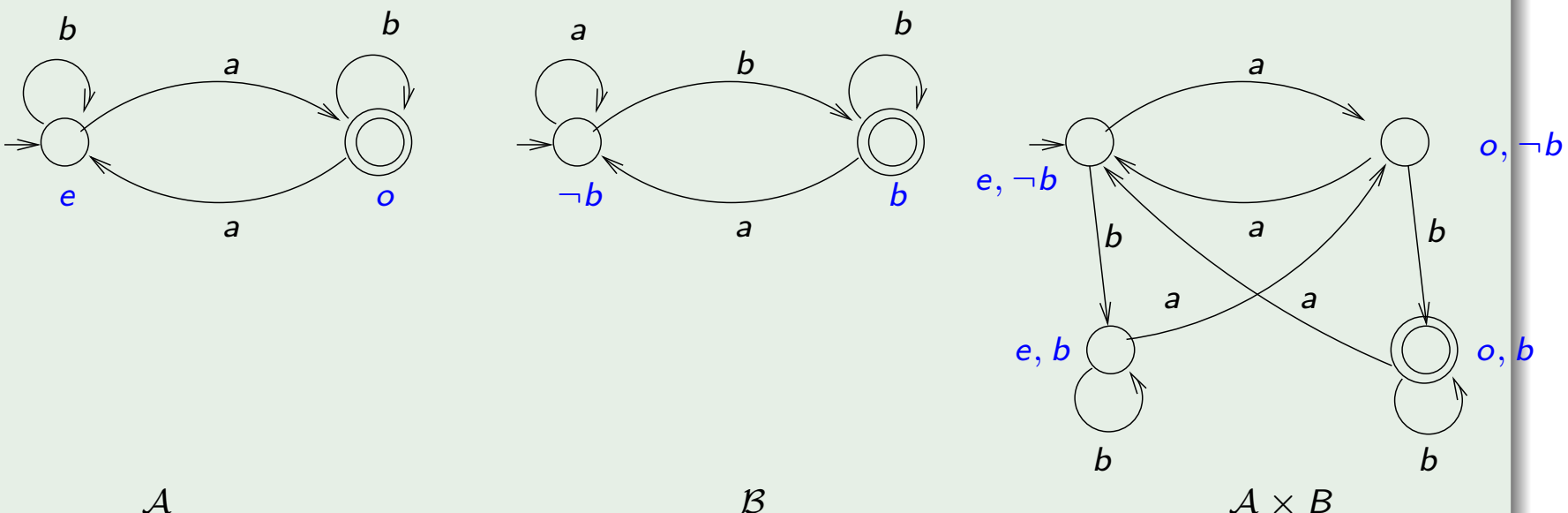
Closure under intersection

Product construction. Given DFA's $\mathcal{A} = (Q, s, \delta, F)$, $\mathcal{B} = (Q', s', \delta', F')$, define product $\mathcal{C}$ of A and B:

$$\mathcal{C} = (Q \times Q', (s, s'), \delta'', F \times F'),$$

where $\delta''((p, p'), a) = (\delta(p, a), \delta'(p', a))$.

Product construction example



Correctness of product construction

Claim: $L(\mathcal{C}) = L(\mathcal{A}) \cap L(\mathcal{B})$.

Proof of claim $L(\mathcal{C}) = L(\mathcal{A}) \cap L(\mathcal{B})$.

- $L(\mathcal{C}) \subseteq L(\mathcal{A}) \cap L(\mathcal{B})$.

$$
\begin{aligned}
w \in L(\mathcal{C}) &\implies \widehat{\delta''}((s,s'),w) \in F \times F'. \\
&\implies (\widehat{\delta}(s,w), \widehat{\delta'}(s',w)) \in F \times F' \text{ (by subclaim)} \\
&\implies \widehat{\delta}(s,w) \in F \text{ and } \widehat{\delta'}(s',w) \in F' \\
&\implies w \in L(\mathcal{A}) \text{ and } w \in L(\mathcal{B}) \\
&\implies w \in L(\mathcal{A}) \cap L(\mathcal{B}).
\end{aligned}
$$

- $L(\mathcal{C}) \supseteq L(\mathcal{A}) \cap L(\mathcal{B})$.

Subclaim: $\widehat{\delta''}((s,s'),w) = (\widehat{\delta}(s,w), \widehat{\delta'}(s',w))$.

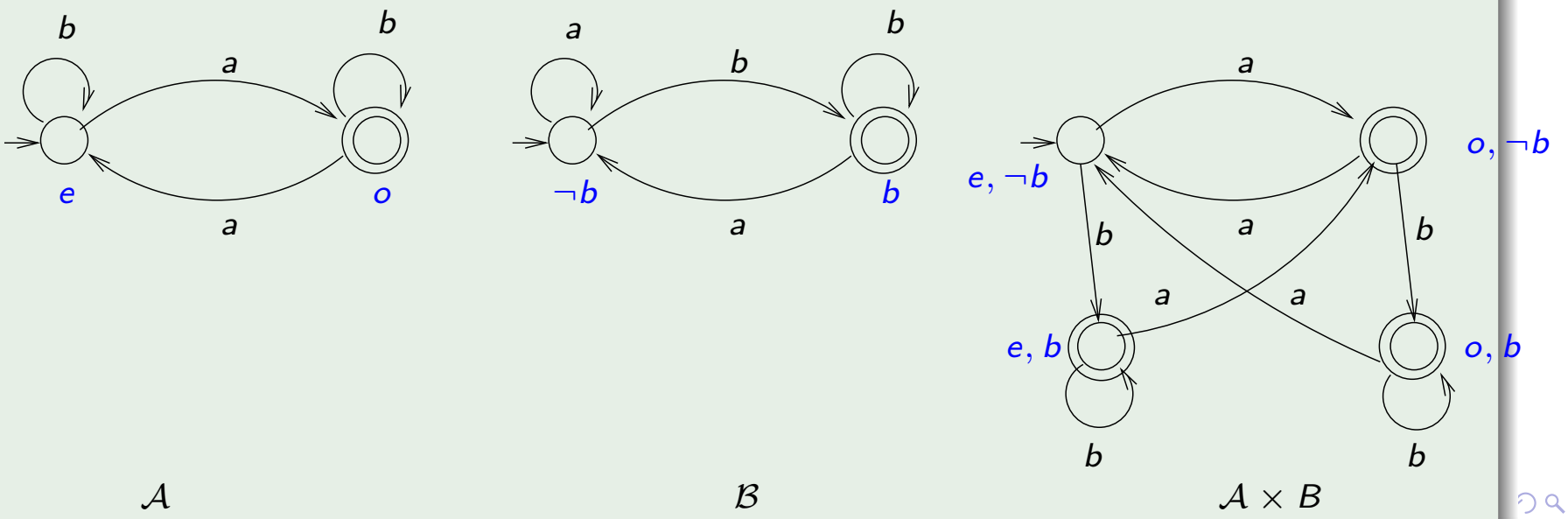
Closure under union

- Follows from closure under complement and intersection since $L_1 \cup L_2 = \overline{\overline{L_1} \cap \overline{L_2}}$.

Closure under union

- Follows from closure under complement and intersection since $L_1 \cup L_2 = \overline{\overline{L_1} \cap \overline{L_2}}$.
- Can also do directly by product construction: Given DFA's $\mathcal{A} = (Q, s, \delta, F)$, $\mathcal{B} = (Q', s', \delta', F')$, define $\mathcal{C}$: $\mathcal{C} = (Q \times Q', (s, s'), \delta'', (F \times Q') \cup (Q \times F'))$, where $\delta''((p, p'), a) = (\delta(p, a), \delta(p', a))$.

Union construction



Principle of Mathematical Induction

- $\mathbb{N} = \{0, 1, 2 \ldots\}$
- $P(n)$: A statement P about a natural number n.
- Example:
  - $P(n) =$ "n is even."
  - $P_1(n) =$ "Sum of the numbers $1 \ldots n$ equals $n(n+1)/2$."
  - $P_2(n) =$ "For all $w \in A^*$, if length of w is n then $\widehat{\delta}''((s, s'), w) = (\widehat{\delta}(s, w), \widehat{\delta}'(s', w))$."

Principle of Induction

If a statement P about natural numbers

- is true for 0 (i.e $P(0)$ is true), and,
- is true for $n+1$ whenever it is true for n (i.e. $P(n) \implies P(n+1)$)

then P is true of all natural numbers (i.e. "For all n, $P(n)$" is true).

Proof of subclaim

Exercise: Prove the Subclaim:

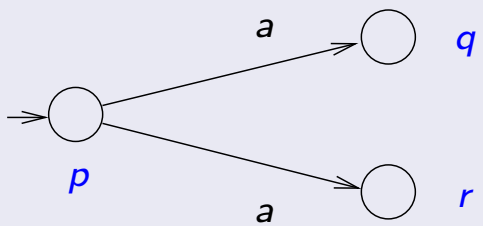
$$\widehat{\delta}''((s, s'), w) = (\widehat{\delta}(s, w), \widehat{\delta}'(s', w)).$$

using induction.

Nondeterministic Finite-state Automata (NFA)

- Allows multiple start states.
- Allows more than one transition from a state on a given letter.

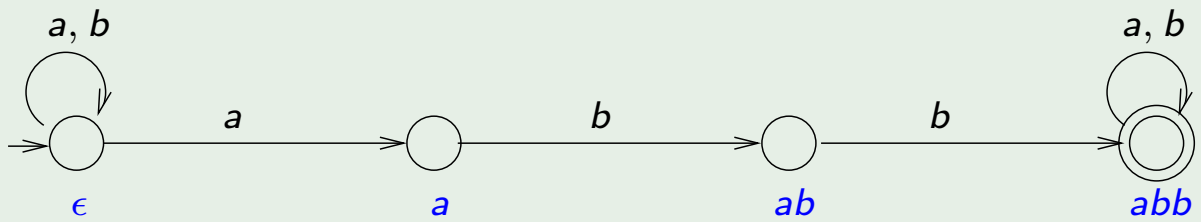
Non-deterministic transitions



- A word is accepted if there is some path on it from a start to a final state.

Example NFA's

NFA for "contains *abb* as a subword"



NFA definition

- Mathematical representation of NFA
  - $\mathcal{A} = (Q, S, \Delta, F)$, where $S \subseteq Q$, and $\Delta : Q \times A \to 2^Q$.
  - Define relation $p \xrightarrow{w} q$ which says there is a path from state p to state q labelled w.
    - $p \xrightarrow{\epsilon} p$
    - $p \xrightarrow{ua} q$ iff there exists $r \in Q$ such that $p \xrightarrow{u} r$ and $q \in \Delta(r, a)$.
  - Define $L(\mathcal{A}) = \{w \in A^* \mid \exists s \in S, f \in F : \ s \xrightarrow{w} f\}$.
- NFA $\to$ DFA: Subset construction
  - Example: determinize NFA for "contains abb."
  - Formal construction
  - Correctness

Closure under concatenation and Kleene iteration

- Concatenation of languages:

$$L \cdot M = \{u \cdot v \mid u \in L,\ v \in M\}.$$

- Kleene iteration of a language:

$$L^* = \{\epsilon\} \cup L \cup L^2 \cup L^3 \cup \cdots,$$

where

$$\begin{aligned} L^n &= L \cdot L \cdots L \ (n \text{ times}). \\ &= \{w_1 \cdots w_n \mid \text{each } w_i \in L\}. \end{aligned}$$

Required Self-Reading

- Chapters 6 (NFAs and subset construction), 9 (Regular Expressions and DFAs), and 12 (Pumping Lemma for regular languages) from Kozen's Automata and Computability book.
- Quiz on
  - Constructing DFAs
  - Closure constructions and proofs