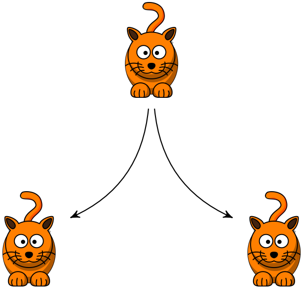
Non-regular Languages

Mika Göös

EPFL School of Computer and Communication Sciences

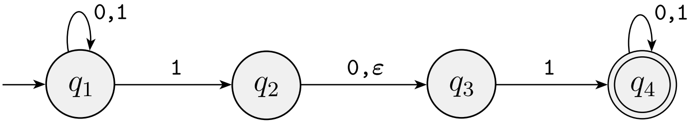
Lecture 3



Should I stay or should I go (left or right)

RECALL: NFAs, Subset Construction

Nondeterminism vs Determinism



State diagram of a **Nondeterministic Finite Automaton (NFA)**

Differences to DFAs:

1. Ability to transition to more than one state on a given symbol
   Ex: q_1 has two outgoing transitions for symbol 1
2. A state may have no transition on a particular symbol
   Ex: q_3 has no outgoing transitions for symbol 0
3. Ability to take a step without reading any input symbol (ε-transitions)
   Ex: q_2 can transition to q_3 without reading a symbol

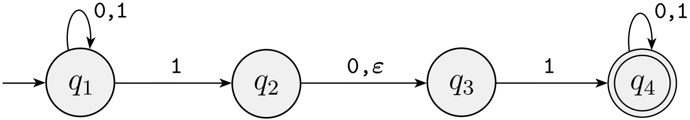
Formal definitions

A nondeterministic finite automaton (NFA) M is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where

- Q is a finite set called the states,
- Σ is a finite set called the alphabet,
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to 2^Q$ is the transition function,
- $q_0 \in Q$ is the start state, and
- $F \subseteq Q$ is the set of accept states. *(allow $F = \emptyset$)*

- Here 2^Q denotes the power set of Q. Ex:
  $Q = \{q_1, q_2\}, 2^Q = \{\emptyset, \{q_1\}, \{q_2\}, \{q_1, q_2\}\}$
- An input is accepted if there exists at least one path that ends at an accepting state
- An input is rejected if no computation path end at an accepting state

Example



The formal description of N_1 is $(Q, \Sigma, \delta, q_1, F)$, where

1. $Q = \{q_1, q_2, q_3, q_4\}$,

2. $\Sigma = \{0,1\}$,

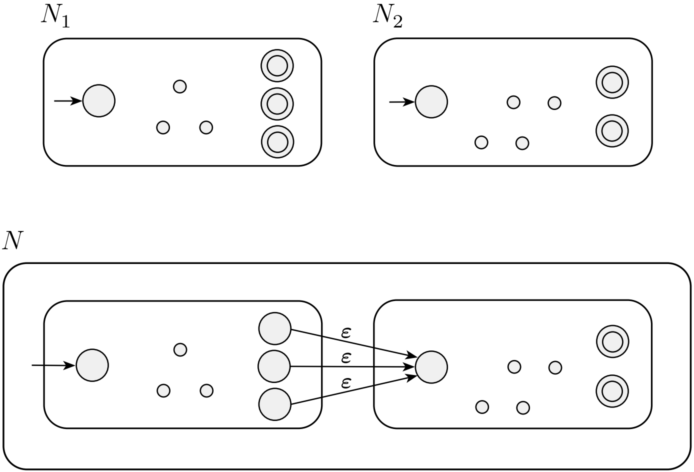
3. δ is given as

	0	1	ε
q_1	$\{q_1\}$	$\{q_1, q_2\}$	$\emptyset$
q_2	$\{q_3\}$	$\emptyset$	$\{q_3\}$
q_3	$\emptyset$	$\{q_4\}$	$\emptyset$
q_4	$\{q_4\}$	$\{q_4\}$	$\emptyset$,

4. q_1 is the start state, and

5. $F = \{q_4\}$.

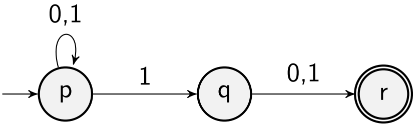
Concatenation



Theorem. Every nondeterministic finite automaton has an equivalent deterministic finite automaton.

Corollary. A language is regular if and only if some nondeterministic finite automaton recognizes it.

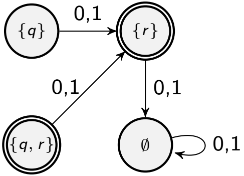
Example

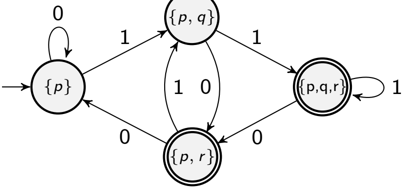


$$\tilde{\delta}\colon 2^Q \times \Sigma \to 2^Q$$

$$\tilde{\delta}(A, a) = \bigcup_{q \in A} \delta(q, a)$$

$\tilde{\delta}$	**0**	**1**
$\emptyset$	$\emptyset$	$\emptyset$
$\{p\}$	$\{p\}$	$\{p, q\}$
$\{q\}$	$\{r\}$	$\{r\}$
$\{r\}$	$\emptyset$	$\emptyset$
$\{p, q\}$	$\{p, r\}$	$\{p, q, r\}$
$\{p, r\}$	$\{p\}$	$\{p, q\}$
$\{q, r\}$	$\{r\}$	$\{r\}$
$\{p, q, r\}$	$\{p, r\}$	$\{p, q, r\}$





Is every Language Regular?

What DFAs/NFAs Can Do

- Pattern matching
  - $L = \{w \in \{0,1\}^* \mid w \text{ contains } 1011011 \text{ as a substring}\}$
- Checking parity of numbers, checking divisibility, counting "modulo some number"
  - $L = \{w \in \{0,1\}^* \mid \#\text{ones}(w) \text{ is divisible by } 5\}$
- Skipping prefixes, suffixes, . . . (e.g. 2nd last bit is 1)
  More generally – "regular" expressions
  - $L = 0^*110^* - \{0^n110^m \mid n, m \in \mathbb{N}\}$
- Closure under complements, unions, intersections, . . .

Test your intuition

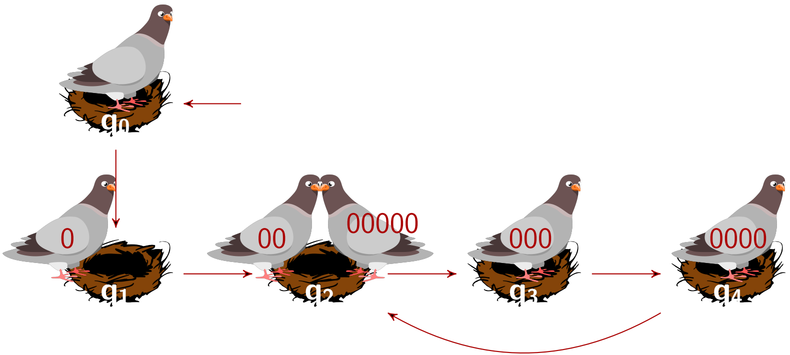
Are these languages regular?

$B = \{0^n 1^n \mid n \geq 0\} = \{\varepsilon, 01, 0011, 000111, \ldots\}$

$C = \{w \mid w \text{ has an equal number of 0's and 1's}\}$

$D = \{w \mid w \text{ has an equal number of occurrences of 01 and 10 as substrings}\}$

$B = \{0^n 1^n \mid n \geq 0\} = \{\varepsilon, 01, 0011, 000111, \ldots\}$ is **not** regular



- Suppose on the contrary that B is recognized by a DFA with p states (e.g. $p = 5$)
- Consider what happens when the input is long enough, e.g., 0^p
- Thus, for two strings $0^i, 0^j, i \neq j$ we must end up in the same state
- Since $0^i 1^i$ is accepted $0^j 1^i$ is also accepted – contradiction!

Thus B is NOT regular!

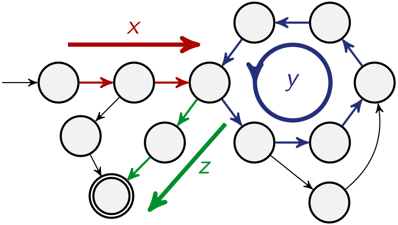
Pumping Lemma and its Proof Sketch

If A is a regular language, then there is a number p (the pumping length) such that, for every string s in A of length at least p, there exists a division of s into three pieces, $s = xyz$ s.t.

1. for each $i \geq 0$, $xy^iz \in A$
2. $|y| \geq 1$, and
3. $|xy| \leq p$.

(What if A is finite?)

- Let $M = (Q, \Sigma, \delta, q_0, F)$ s.t. $L(M) = A$
- Consider a string $s \in A$ s.t. $|s| \geq |Q| = p$
- Stop once in a state for the 2nd time, say at times j, k
- $s = xyz$ where x is the first j letters, y is letter $j + 1$ to k, z is from $k + 1$ to end



More formal proof

If A is a regular language, then there is a number p (the pumping length) such that, for every string s in A of length at least p, there exists a division of s into three pieces, $s = xyz$ s.t.

1. for each $i \geq 0$, $xy^iz \in A$
2. $|y| \geq 1$, and
3. $|xy| \leq p$.

- Let $p = |Q|$ be the number of states in a DFA $M = (Q, \Sigma, \delta, q_0, F)$ accepting A
- Let s be an accepted string of length $|s| \geq p$
- Let q be the first state that is repeated when starting at q_0 and feeding s and let x, y, z be the strings s.t. $|xy|$ is minimized and

$$\delta(q_0, x) = q \quad \text{and} \quad \delta(q_0, xy) = q \quad \text{and} \quad \delta(q, z) = r \in F$$

- Then $\delta(q_0, xy^iz) = r$ for all $i \geq 0$
- Note that $|y| \geq 1$ as there are two distinct occurences of q
- $|xy| \leq p$ by piqeonhole principle and that $|xy|$ equals the number of steps until a state is visited twice.

Applications

$$F = \{ww \mid w \in \{0,1\}^*\}$$

F is not regular!

Proof: (by contradiction)

- Assume F is regular, let p be its pumping length
- Pick $s = 0^p 1^p 0^p 1^p \in F$

  All strings don't work! Fun part is guessing which string to pick

- Pumping lemma: $s = xyz, |xy| \leq p, |y| \geq 1, xy^i z \in F$ for all $i \geq 0$

  Pumping lemma tells us there is such a decomposition – *we can't choose it!* Your reasoning should work for any decomposition

- Since $|xy| \leq p$ and $|y| \geq 1$, $y = 0^k$ for some $k > 0$
- According to pumping lemma, $xy^2z \in F$
- $xy^2z = 0^{p+k}1^p0^p1^p \notin F$
- **Contradiction!**

$$E = \{0^m 1^n \mid m > n\}$$

E is not regular!

Proof: (by contradiction)

- Assume E is regular, let p be its pumping length
- Pick $s = 0^{p+1}1^p \in E$
- Pumping lemma: $s = xyz, |xy| \leq p, |y| \geq 1, xy^iz \in E$ for all $i \geq 0$
- Since $s = 0^{p+1}1^p$, $|xy| \leq p$, $|y| \geq 1$, we have $y = 0^k$ for some $k > 0$
- Need to find a string of the type $xy^iz \notin E$ for some $i \geq 0$
- However, $xy^2z = 0^{p+k+1}1^p, xy^3z = 0^{p+2k+1}1^p, \ldots \in E$
- Try $i = 0$. According to pumping lemma, $xy^0z = xz \in E$
- But $xz = 0^{p+1-k}1^p$ and $p + 1 - k \leq p$. Thus, $xz \notin E$.
  Contradiction!

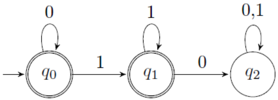
$C = \{w \mid w \text{ has equal number of 0's and 1's}\}$

C is not regular! Reduction from $B = \{0^n 1^n \mid n \geq 0\}$

- $B \subseteq C$ and B is non-regular does **not** imply C is non-regular...
- e.g. $B \subseteq \{0,1\}^*$ as well but $\{0,1\}^*$ is regular!
- However we can still use non-regularity of B

Proof: (by contradiction)

- Assume C is regular
- Let $0^*1^* = \{0^m 1^n \mid m, n \geq 0\}$
- 0^*1^* is regular
- Claim: $B = C \cap (0^*1^*)$
- Intersection of two regular languages is regular
- $\Rightarrow B$ is regular – a **contradiction!**



$C = \{w \mid w$ has equal number of 0's and 1's$\}$

C is not regular! (Proof using Pumping Lemma)

Proof: (by contradiction)

- Assume C is regular, let p be the pumping length
- We need to find a string which cannot be "pumped"
- Try $s = 010101010101 = (01)^p$
  - $s = xyz$ with $|xy| \leq p, |y| \geq 1$ and $xy^iz \in C$ for every $i \geq 0$
  - What if $x = \varepsilon$ and $y = 01$ then indeed $xy^iz \in C$?
  - Our choice of s gives no contradiction!
- Try $s = 0^p1^p$
  - $s = xyz$ with $|xy| \leq p, |y| \geq 1$ and $xy^iz \in C$ for every $i \geq 0$
  - In **every** possible case y is composed of 0s only!
  - Say $y = 0^k$ for some $k > 0$ then $xy^2z = 0^{p+k}1^p \notin C$
  - **Contradiction!**

Task: Prove that L is not regular ..

Let's try to use the Pumping Lemma

Assume *L* is regular →

← *p* : the pumping length

Fun part is guessing which string to pick!

Pick *s in L* with $|s| \geq p$ →

Your reasoning should work for any decomposition

← *s=xyz s.t.*
$|xy| \leq p$ *and* $|y| \geq 1$ *and* $xy^iz \in L \ \ \forall i \geq 0$

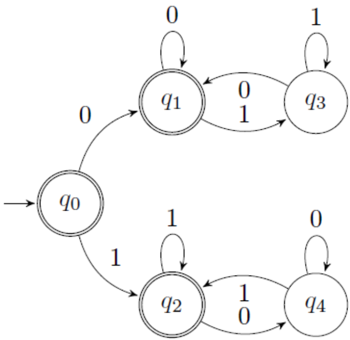
xz, xyyz,xyyyz,xyyyyz,… → If one of them NOT in *L – DONE!*

If all of them in *L – Try again!* ←

Test your intuition...

$D = \{w \mid w$ has equal nr of occurrences of 01 and 10 as substrings$\}$

D is regular!



Beyond Regular Languages
Turing Machines . . .