L3. 2022-09-12

Deterministic vs Nondeterministic automata

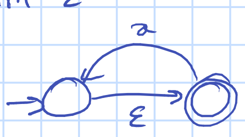
Formal def of NDA: like DFA, but

$$\delta: Q \times (\Sigma \cup \{\varepsilon\}) \to Q$$

and not necessarily a function (one input can have 0, 1, or more outputs).

We write $((q,a),r) \in \delta$ or $r \in \delta(q,a)$ to mean r reachable from q on input a.

e.g. with ε:



def computation: still the same, but not uniquely defined.

def M accepts s if $\exists s'$ st s' without $\varepsilon \equiv s$ and $\exists$ accepting computation of (M, s').

Informal trick: use fingers/pebbles/... to keep track of all possible states at every character in input.
We'll use it to simulate NFAs.

Q: are NDAs more powerful than DFAs?

A: they recognize the same languages, but NDAs can be exponentially smaller than DFAs.

thm: Given NDA, there exists a DFA that recognizes exactly the same language.

proof idea: have one state for every possible position of fingers.

proof: Let $N = (Q, \Sigma, \delta, q_0, F)$ be an NFA.

Assume for simplicity no ε-transitions.

We build a DFA $M = (Q', \Sigma', \delta', q_0', F')$ as follows.

$Q' = \mathcal{P}(Q)$ // i.e. all subsets of Q

$\Sigma' = \Sigma$

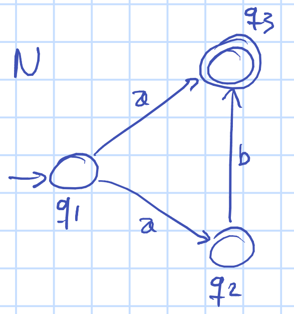
$\delta'(R, a) = \bigcup_{q \in R} \delta(q, a)$ // i.e. all states reachable from some state in current subset

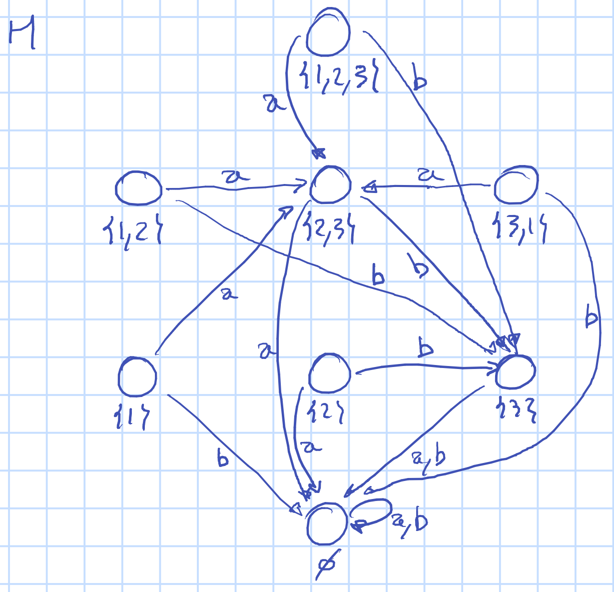
$q_0' = \{q_0\}$

$F' = \{R : R \cap F \neq \emptyset\}$ // i.e. sets that contain some accepting state

Easy to check that s accepted by N iff accepted by M.

e.g.





To finish proof, let us also consider ε-transitions.

Let $E(q)$ be the states reachable from q by only following ε-transitions.
$E(R) = \bigcup_{q \in R} E(q)$.

$$\delta'(R, a) = \bigcup_{q \in R} E(\delta(q, a))$$

$$q_0' = E(q_0)$$

Easy to check that $\varepsilon^{k_0} s_1 \varepsilon^{k_1} s_2 \cdots s_n \varepsilon^{k_n}$ accepted by N iff $s_1 \cdots s_n$ accepted by M.

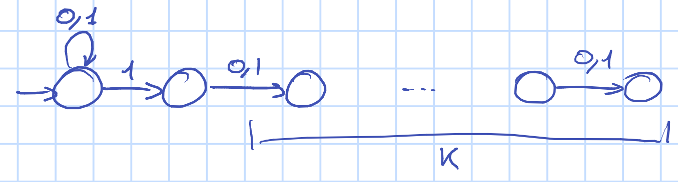
※

So DFAs as powerful as NFAs... but let's show less efficient.

th There is a language recognizable by a NFA with $k+2$ states that requires $\geq 2^k$ DFA states.

proof. consider the language of strings st $s_{n-k} = 1$ (i.e. $k+1$-th character from end is a 1).

NFA:



Next time we'll see no small DFA.