MATH 2301

29 Sep 2022

* Yesterday: Introduced non-deterministic finite automata (NFAs)

* Today: NFAs, formally

Def: An NFA consists of:

(1) An alphabet Σ

(2) A set of states Q (finite)

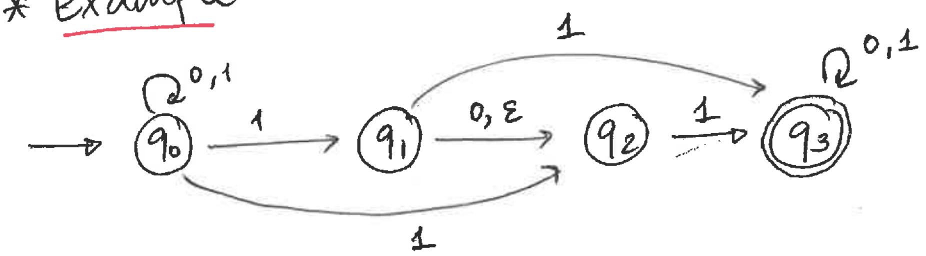
(3) A start state $q_0 \in Q$.

(4) A set $A \subseteq Q$ of accept states

* (5) A transition function

$$\delta : \underset{\substack{\uparrow \\ \text{state you're at}}}{Q} \times (\underset{\substack{\uparrow \\ \text{thing you read}}}{\Sigma \cup \{\varepsilon\}}) \to \underset{\substack{\uparrow \\ \text{possible states you can go to}}}{\mathcal{P}(Q)}$$

* Example



What does δ do? Some examples

* $\delta(q_0, 1) = \{q_0, q_1, q_2\}$
* $\delta(q_0, 0) = \{q_0\}$
* $\delta(q_0, \varepsilon) = \phi$
* $\delta(q_1, \varepsilon) = \{q_2\}$
* $\delta(q_2, 0) = \phi$

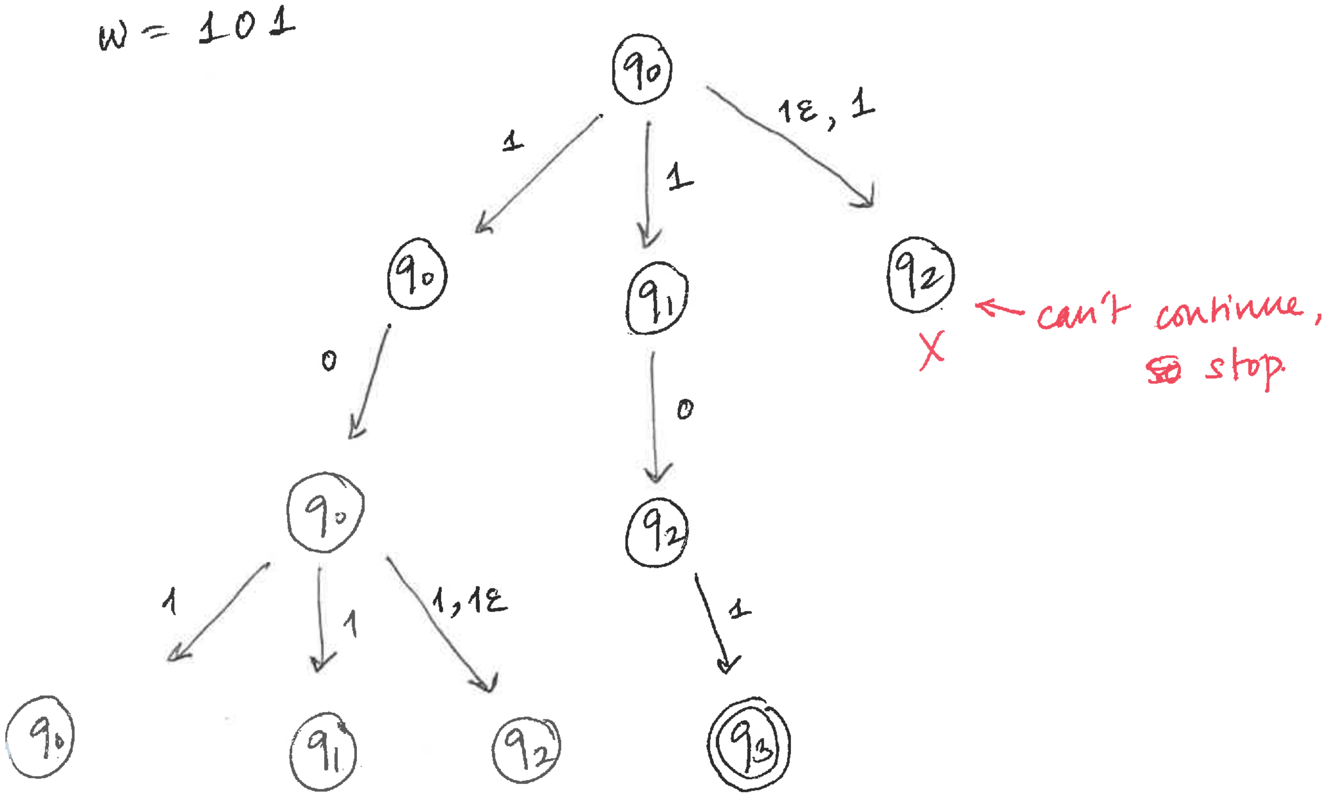
* Calculation tree of an NFA

(Use previous example)

Eg. $w = 101$.

- Start at q_0
- At each step, read a single letter, and any "ε" that come before or after.
- Draw all possibilities, and continue reading the next letter from all of the possibilities

$w = 101$

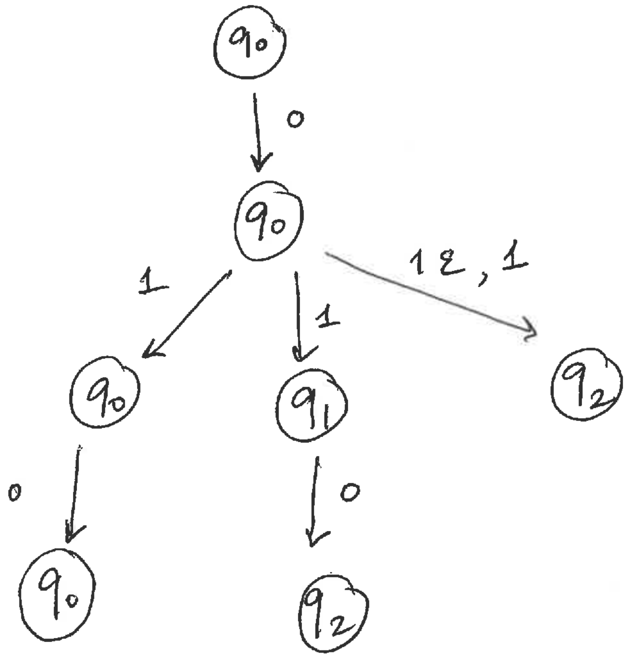


This process is guaranteed to stop because you consume a letter at each step.

ACCEPT the string if at least one of the states at the bottom-most level is an accept state

REJECT if all paths that go through the entire string end up at non-accepting states

$w = 0\,1\,0$



REJECT

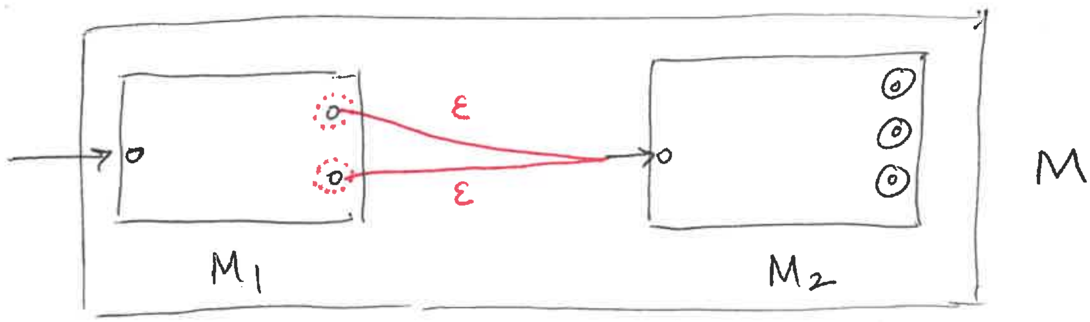
<u>Rule</u>: Any NFA that has arrows labelled by ε, can be converted to an equivalent NFA without any ε-labels. [come back to this later]

---

* <u>Back to trying to convert regexes → NFAs</u>

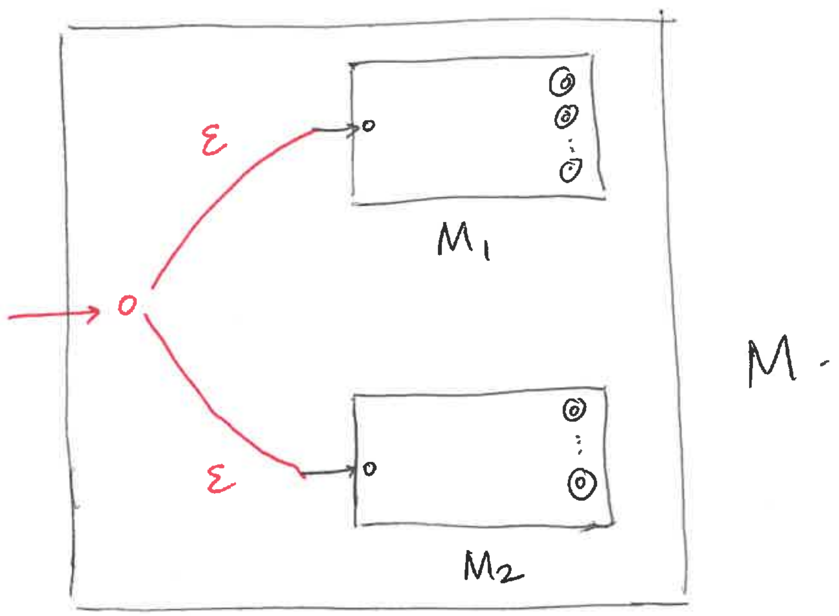
$r = \phi$, $r = \varepsilon$, $r = a$ go through as before
(Note a DFA is also an NFA.)

(4) $r = r_1 r_2$



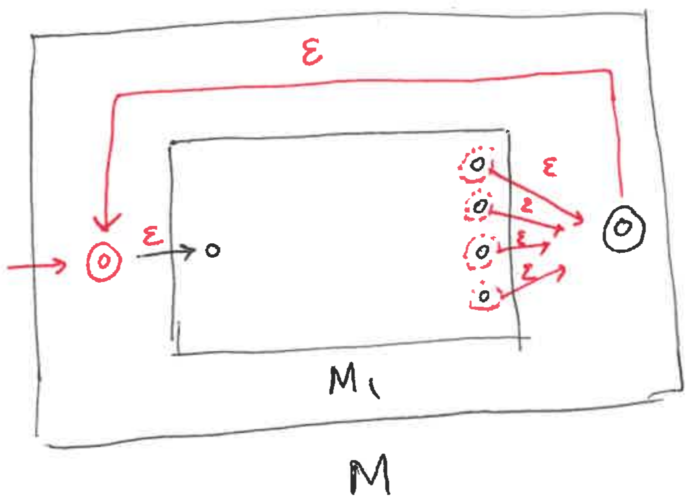
(Add ε-arrows from each accept state of M_1 to the start state of M_2; make them non-accepting.)

(5) $r = r_1 \mid r_2$



(Alternative to the product construction)

(6) $r = r_1^*$



(Explain again on Wed.)