MATH 2301 08 October 2020

* Last time, we:

(1) Defined NFAs & how to compute with them & what it means for an NFA to accept a string. (decision tree)

(2) Proved that for each regex r, there is an NFA whose language is exactly $L(r)$

* Related rule: Every DFA is also an NFA.

* Today: We'll sketch the proof of the following theorem

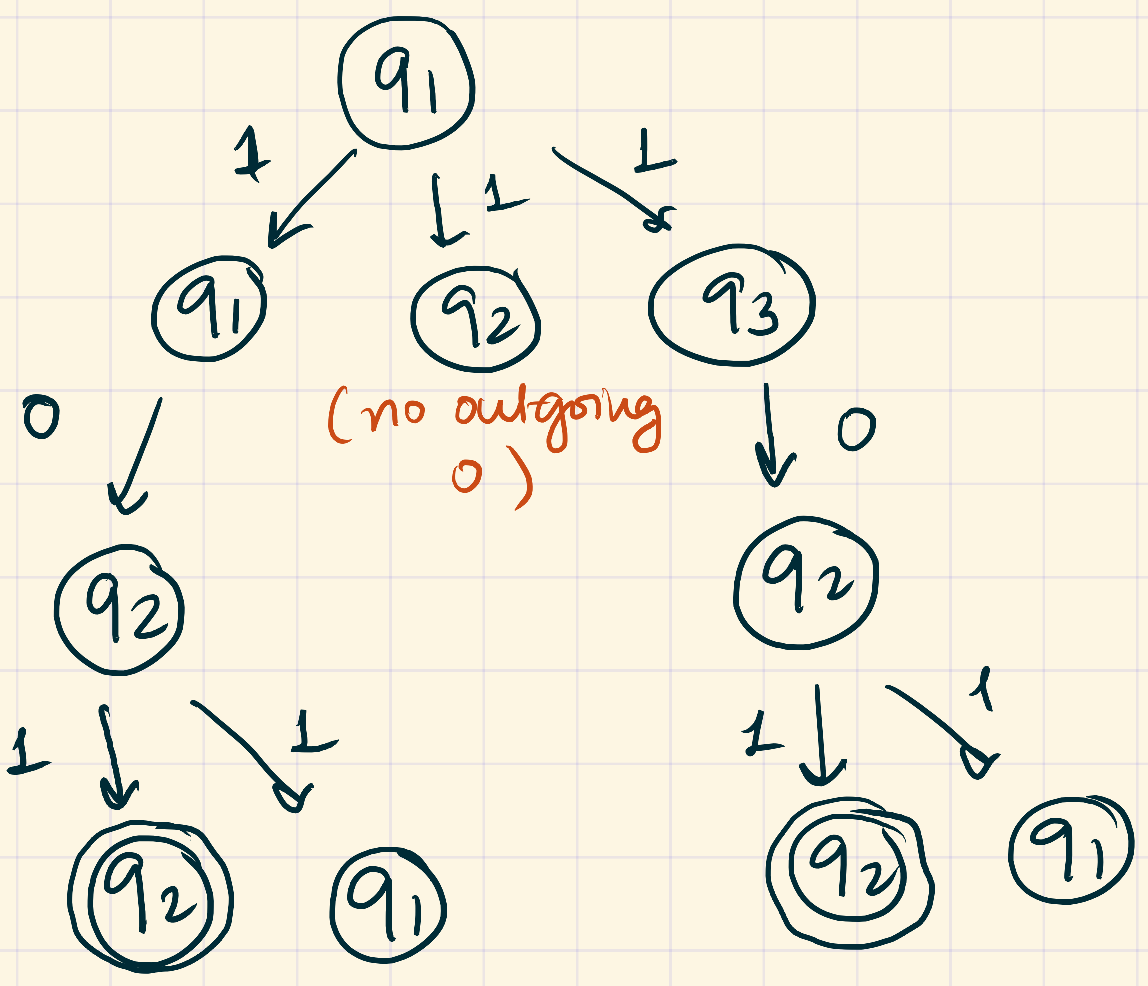
Thm: Given any NFA M, there is always an equivalent DFA M_1; that is, $L(M) = L(M_1)$.

Proof sketch:

Let M be an NFA: states Q, start state q_1, accept states A, transition fn $\Delta : Q \times (\Sigma \cup \{\varepsilon\}) \longrightarrow P(Q)$

(*) For simplicity: assume for now that we don't have any ε-arrows

Any computation with M gives you a decision tree:



$A = \{q_2\}$

(Example)

Each level of the tree is a bunch of states of M.

E.g. $\{q_1\} \xrightarrow{1} \{q_1, q_2, q_3\} \xrightarrow{0} \{q_2\} \xrightarrow{1} \{q_1, q_2\}$

rewriting of decision tree as subsets of Q & transitions

* We'll construct M_1 as follows:

set of states : $\mathcal{P}(Q)$; a state of M' is a set of states of M.

initial state : $\{q_1\}$

set of accepting states : $\{B \subseteq Q \mid$ there is at least some $q \in B$ such that $q \in A$, i.e. q is an accept state of $M\}$.

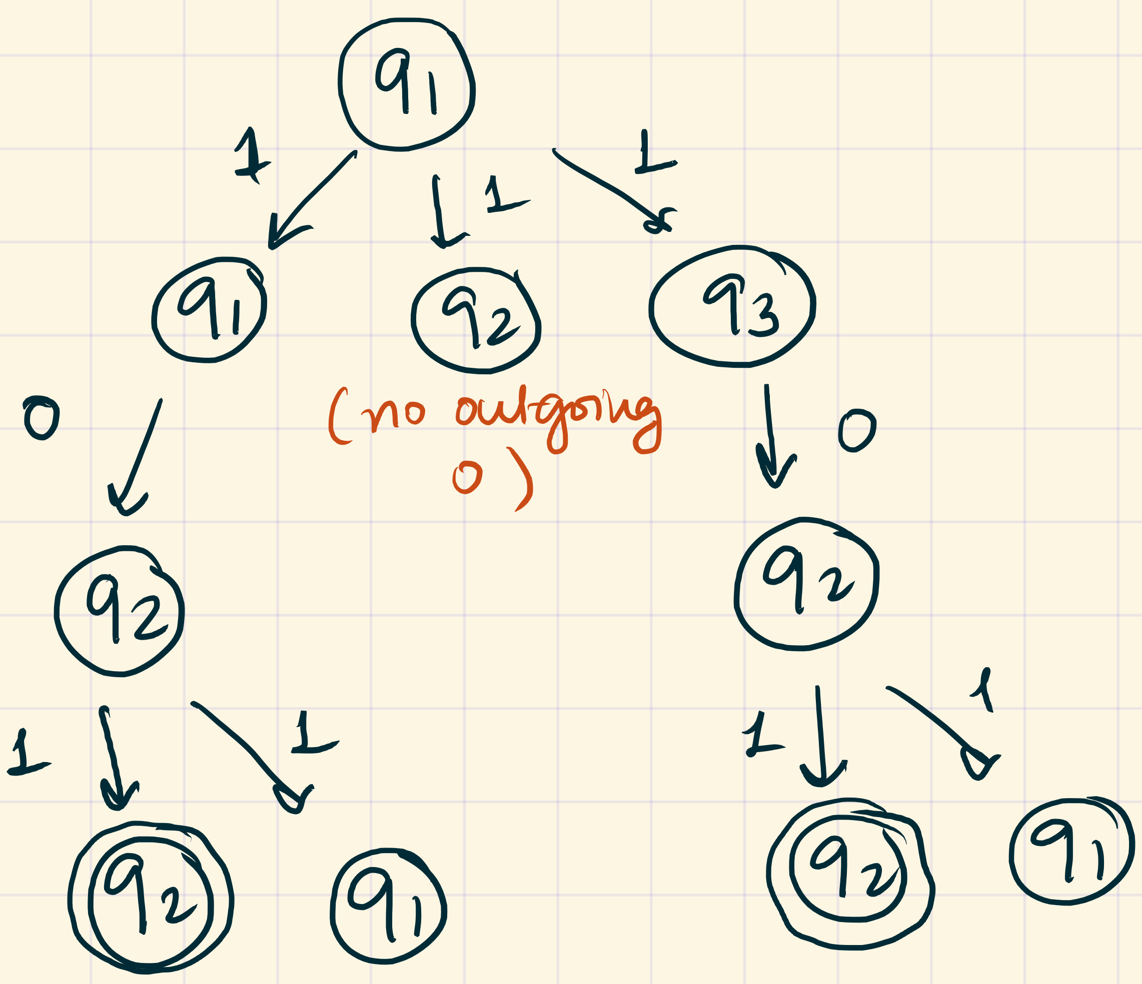
[E.g. in previous example, $\{\{q_2\}, \{q_1, q_2\}, \{q_2, q_3\}, \{q_1, q_2, q_3\}\}$]

transition function:

$$\delta : \underset{\substack{\cup\!\shortmid \\ B}}{\mathcal{P}(Q)} \times \underset{\substack{\cup\!\shortmid \\ a}}{\Sigma} \to \mathcal{P}(Q)$$

$$B, a \mapsto \bigcup_{q \in B} \Delta(q, a)$$

(set of states of M) (symbol)



Take every element of B (a state in M), apply $\Delta(q, a)$ to get a subset of states in M, and for each such choice, combine all outputs.

E.g. (Second level → third level):

$\{q_1, q_2, q_3\}$, read 0.

① $\Delta(q_1, 0) = \{q_2\}$

② $\Delta(q_2, 0) = \phi$

③ $\Delta(q_3, 0) = \{q_2\}$

union together to get $\{q_2\}$

Result: M_1 is a DFA but behaves exactly like the NFA M.

* However : you get an exponential blowup in the size: if M has k states, M_1 has 2^k states

* Remark : This was assuming we didn't have ε-arrows. If you do have ε-arrows, you can first convert to an equivalent NFAs with no ε-arrows, then use procedure above. (details omitted)

* Result:

* Thm: DFAs and NFAs have equivalent computational power.

  That is, suppose L is a language.

  There is a DFA recognising L iff there is an NFA recognising L.

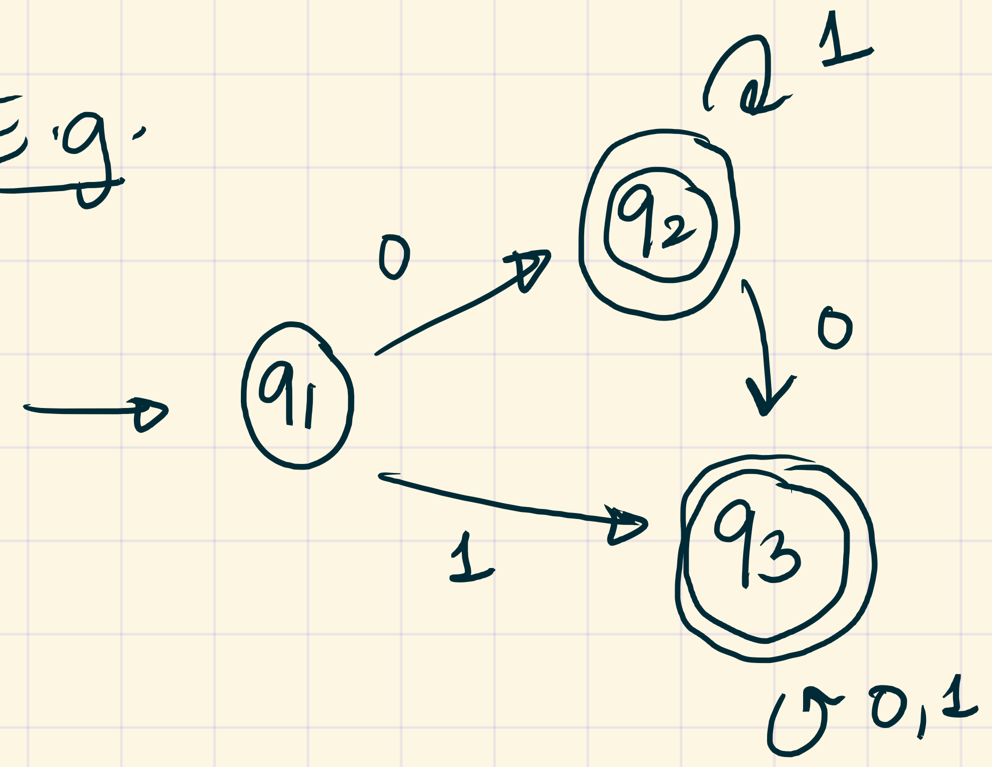
* Thm: If r is any regular expression, there is a DFA M such that $L(M) = L(r)$.

---

* Next goal: other direction?

  Given a DFA, construct an equivalent regex?

E.g.



To go "from q_1 to q_3",

- the direct edge should correspond to the regular expression

  $1(0|1)^*$

- the other option is to go through q_2.

  $01^*0(0|1)^*$

$q_1 \to q_2$ should correspond to 01^*

All together:

$(1(0|1)^*) | (01^*0(0|1)^*)$

should correspond to "$q_1 \to q_3$"