MATH 2301 06/10/2023

* Recap from last time:

- Given any NFA M, we can construct an equivalent NFA M' which has no ε-labels.

- Given any NFA M without ε-labels, we can construct its "power set automaton" whose states are subsets of the original state set Q, and transitions "follow the calculation tree".

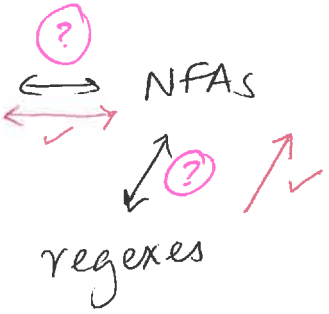
- The new power set automaton is equivalent (recognises the same language) to the old one, and moreover, it is a DFA!

** Theorem: Given any NFA M, there is an equivalent DFA M', that is, such that L(M) = L(M').

** Rmk: If M (the NFA) has n states, the power set automaton (the DFA) M' has 2^n states; this gives you an exponential blow-up in size

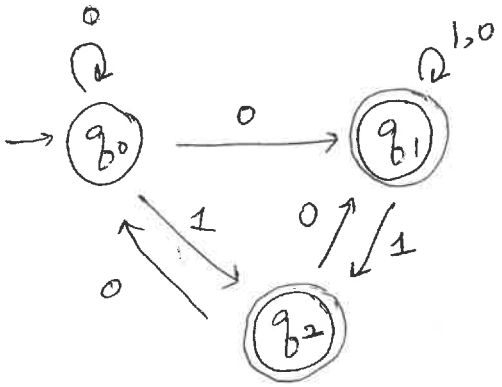
So far, we converted any regex into an equivalent NFA and then any NFA to an equivalent DFA.

* Today : More on DFAs $\overset{?}{\longleftrightarrow}$ NFAs

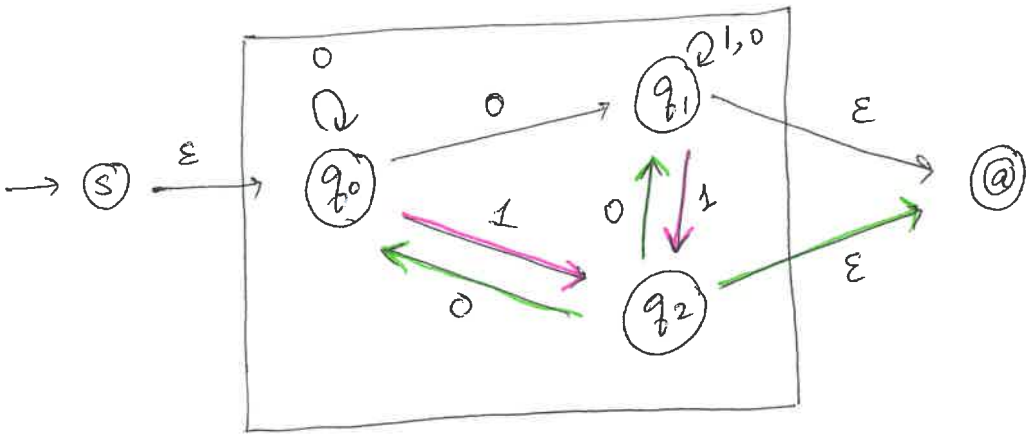


Q : Given an NFA M, (how) can we convert it to an equivalent regex?

E.g.

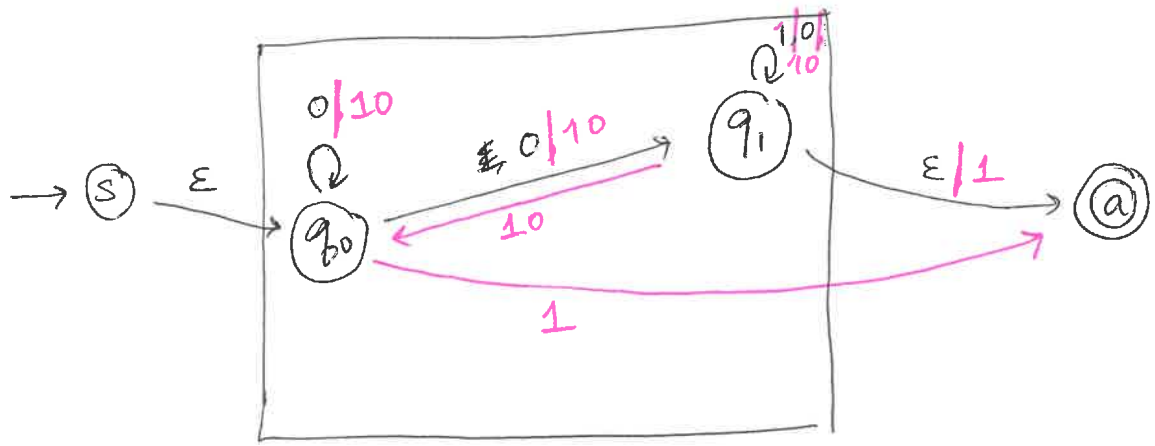


Let us start by converting this to a "sanitised copy"



Key idea : Transform the machine in the box step-by-step, removing states one at a time, adding in extra labels of regular expressions.

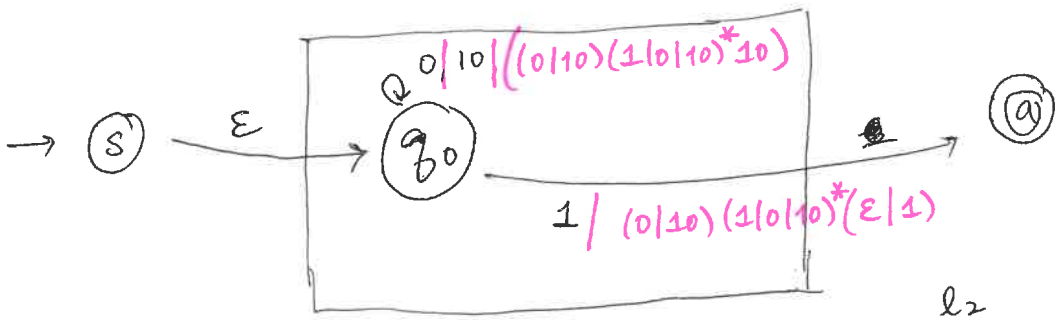
Let's start by removing one state, namely (say) q_2.



- Remove q_2.
- For any sequence $x \xrightarrow{l_1} q_2 \xrightarrow{l_2} y$, $x \neq q_2$ & $y \neq q_2$
  I added a label/arrow $x \xrightarrow{l_1 l_2} y$

- Notice that this produces something that is not an NFA; instead we get "GNFAs", i.e. generalised NFAs

* Next: let us remove q_1



- For any sequence $x \xrightarrow{l_1} q_1 \xrightarrow{l_3} y$ and $x \neq q_1$, $y \neq q_1$, we add:

$x \xrightarrow{l_1 (l_2^*) l_3} y$