MATH 2301

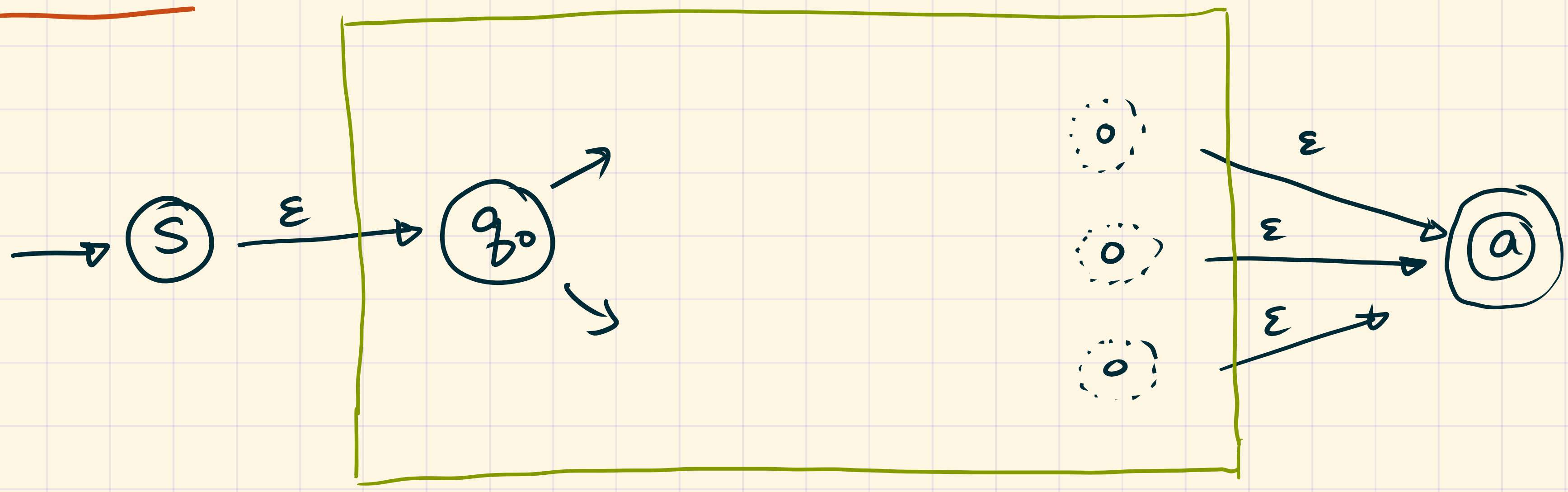
* NFA → Regex

** Let M be an NFA. We begin by adding a new start state s & a new accept state a

- Connect $s \xrightarrow{\varepsilon} q_0$

- For every old accept state q of M, connect $q \xrightarrow{\varepsilon} a$ and make q rejecting.

Result:



The innards of M, without any accept states.

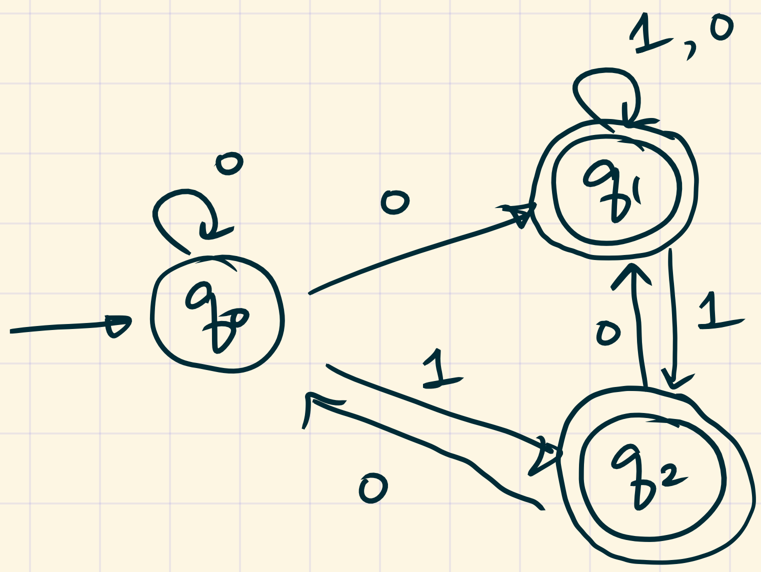
⇝ The new machine has the same language as M

** Process

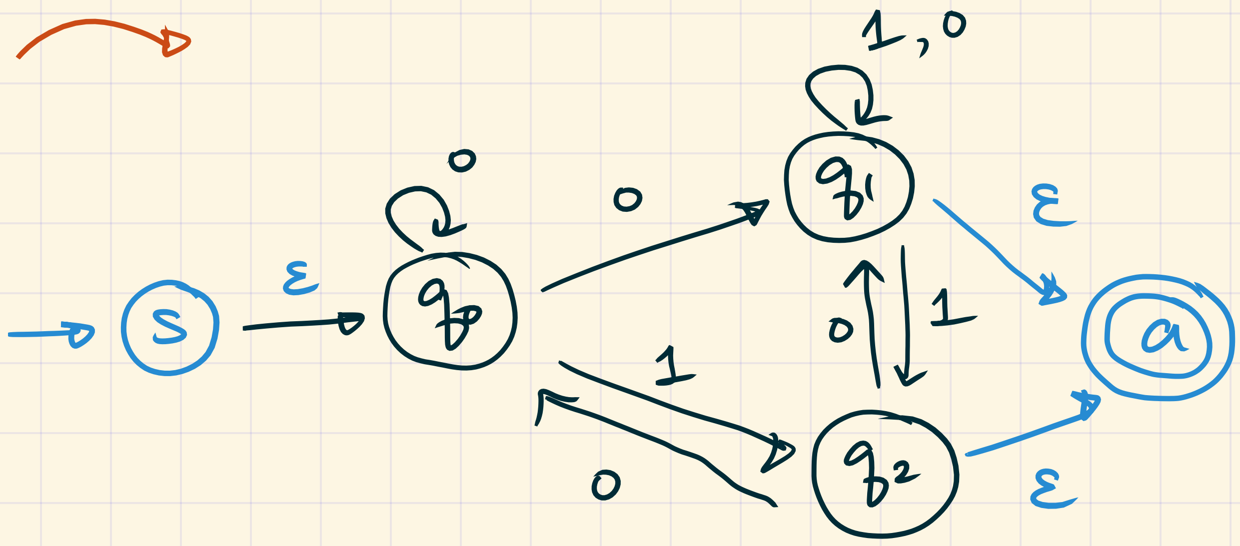
- Eliminate one state at a time from the green portion, replacing edge labels by regexes
- Each step should produce an equivalent machine

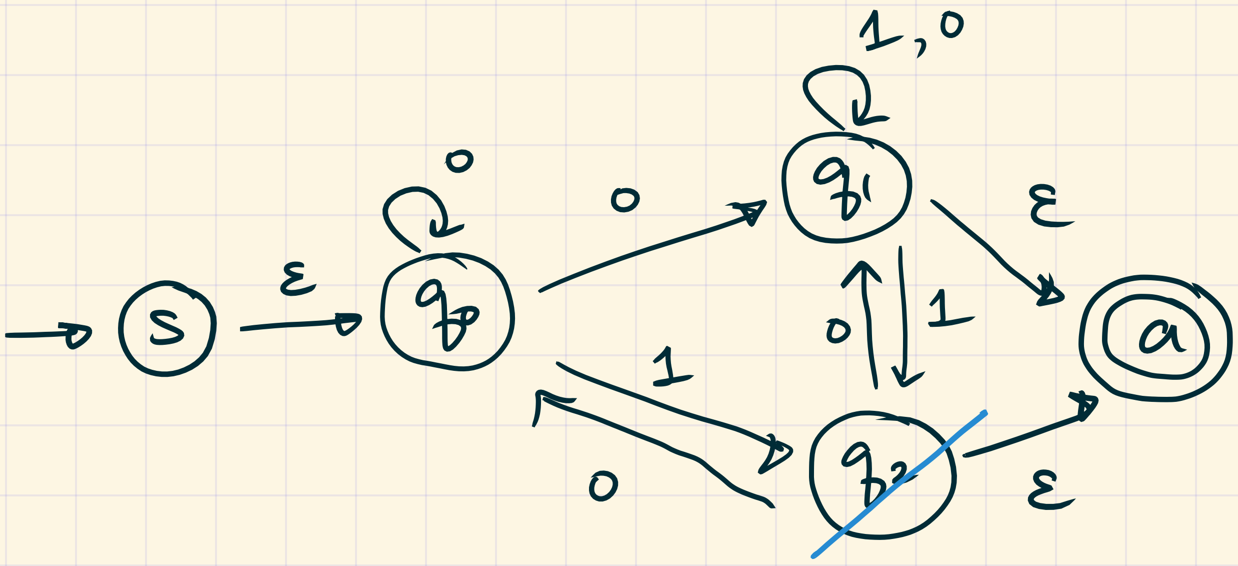
- At the end, obtain $\to s \xrightarrow{r} a$ such that $L(r) = L(M)$.

** Example



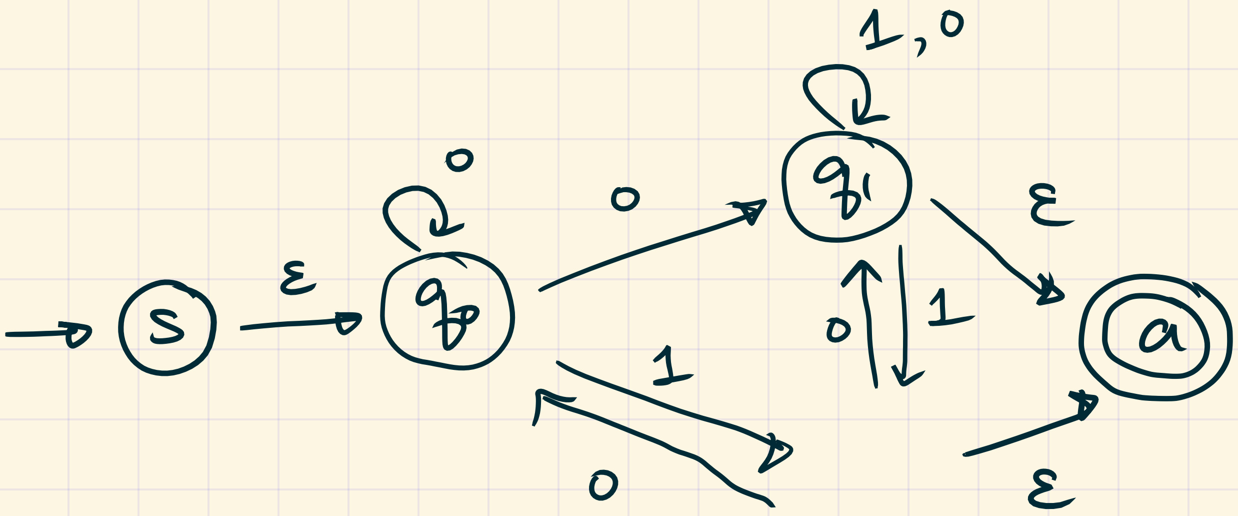
initial reduction





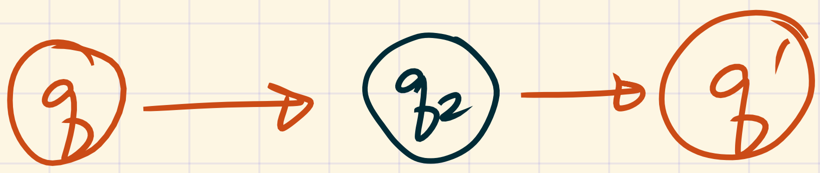
** Eliminate internal states (i.e. except s and a) one by one, as follows.

** The order doesn't matter. Say we start with q_2.



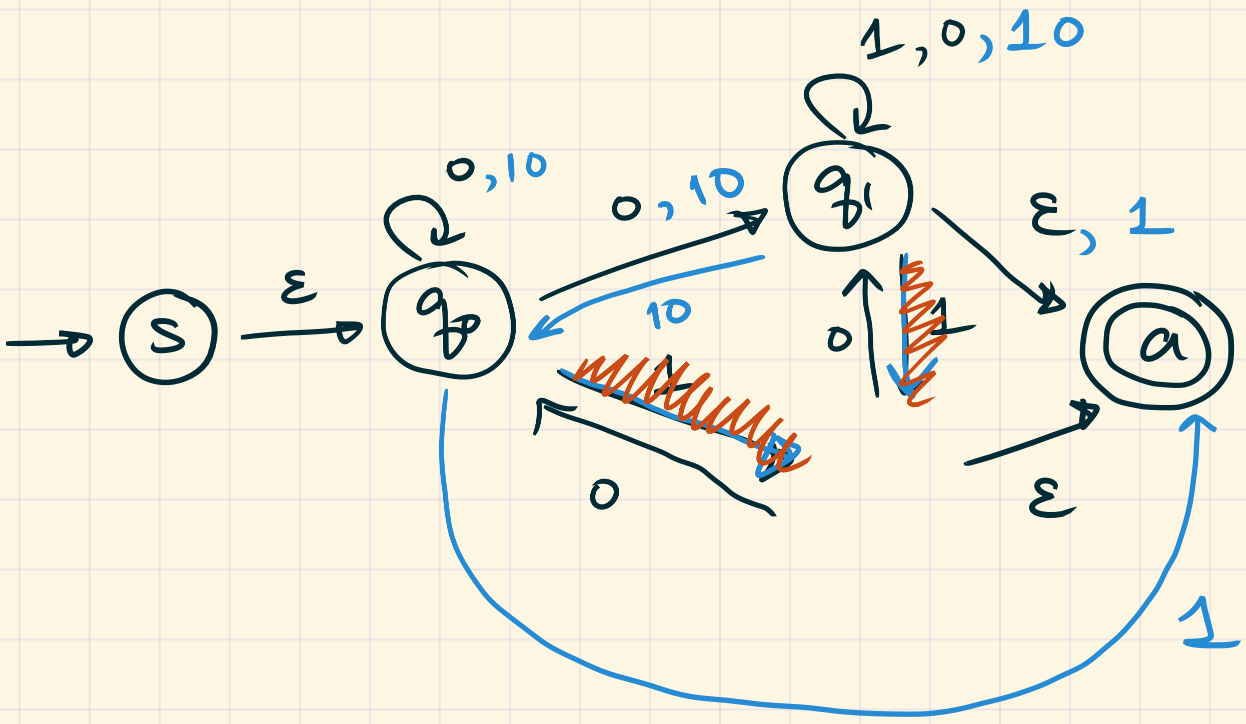
1) Look at all length-2 paths through q_2 (excluding self-loops)

2) Update labels to short-cut these length-2 paths by reading along them.



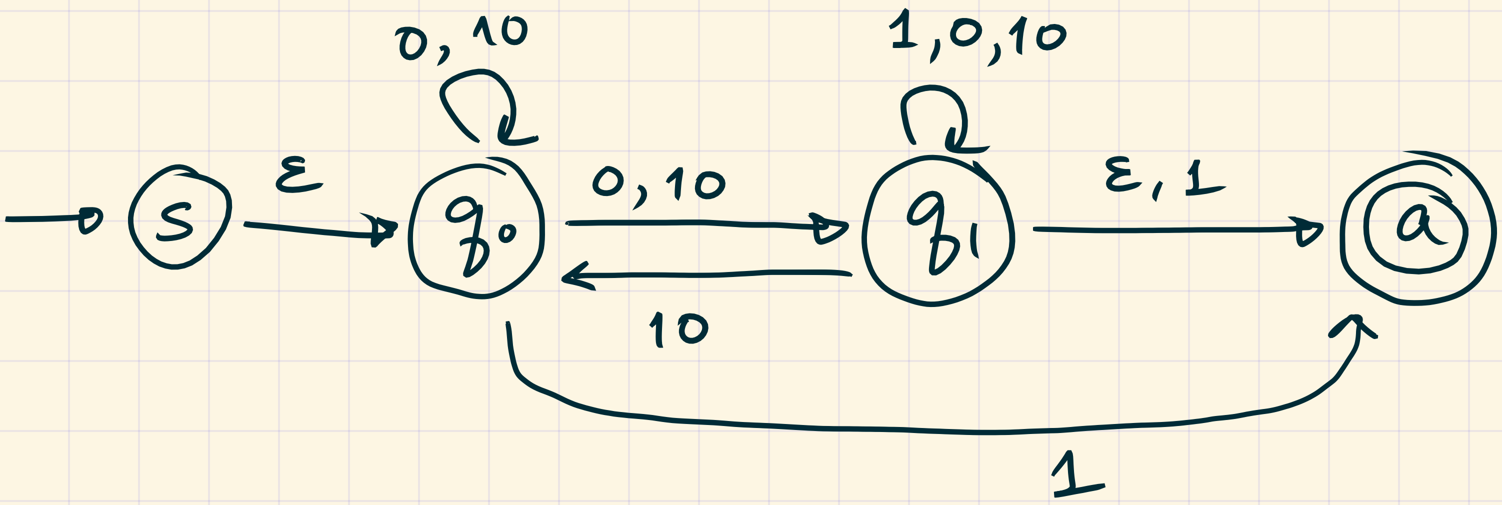
($q \neq q_2$, $q' \neq q_2$)

3) If there are self-loops, deal with them. (Explained later).



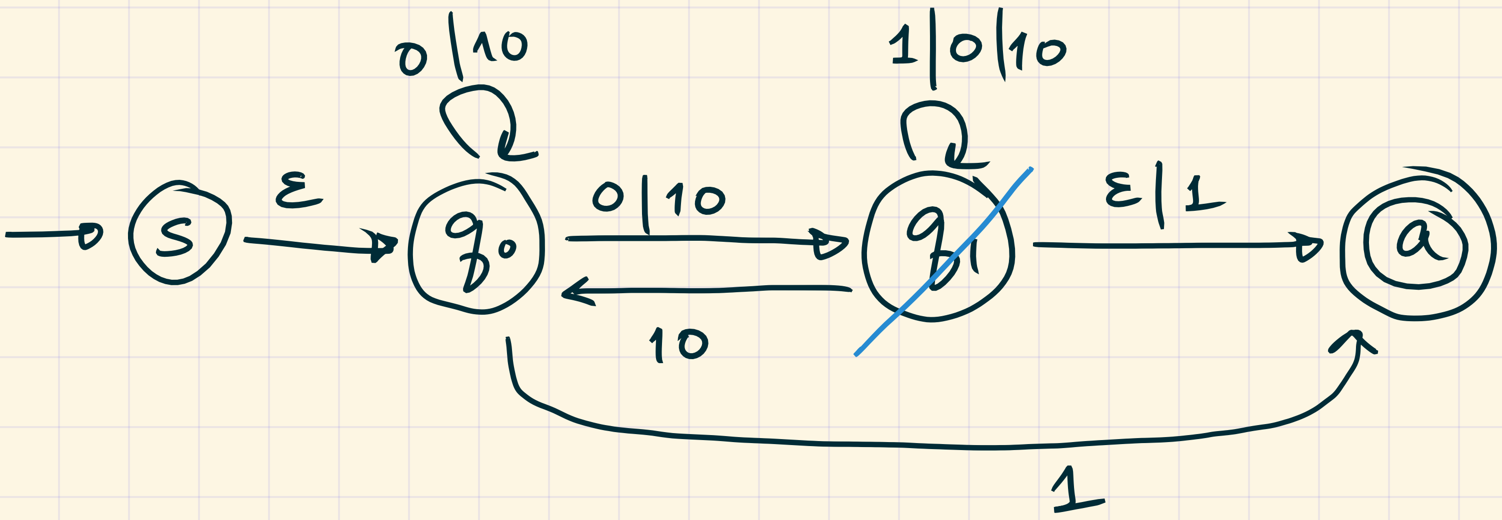
** Remarks

- This process was for a state (q_2) that had no self-loops



- Labels are now regular expressions

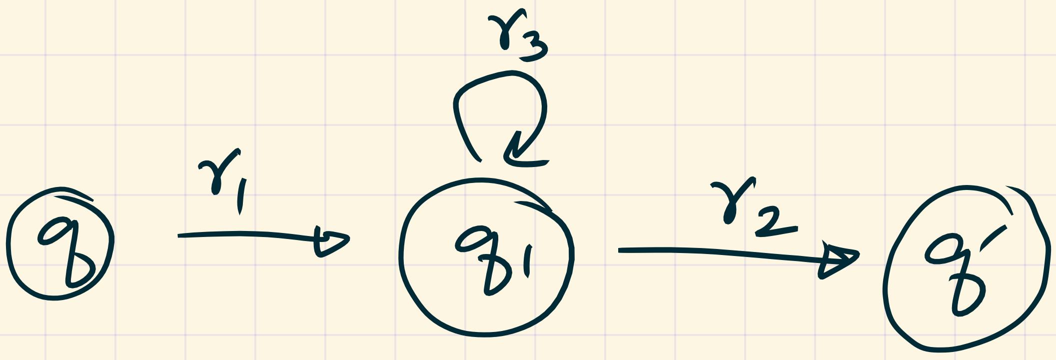
- Commas are the same as "or"s "|".



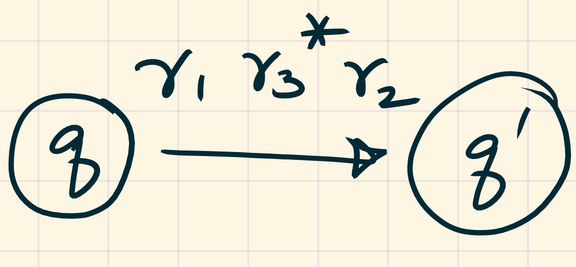
** Try to eliminate q_1 next (say).

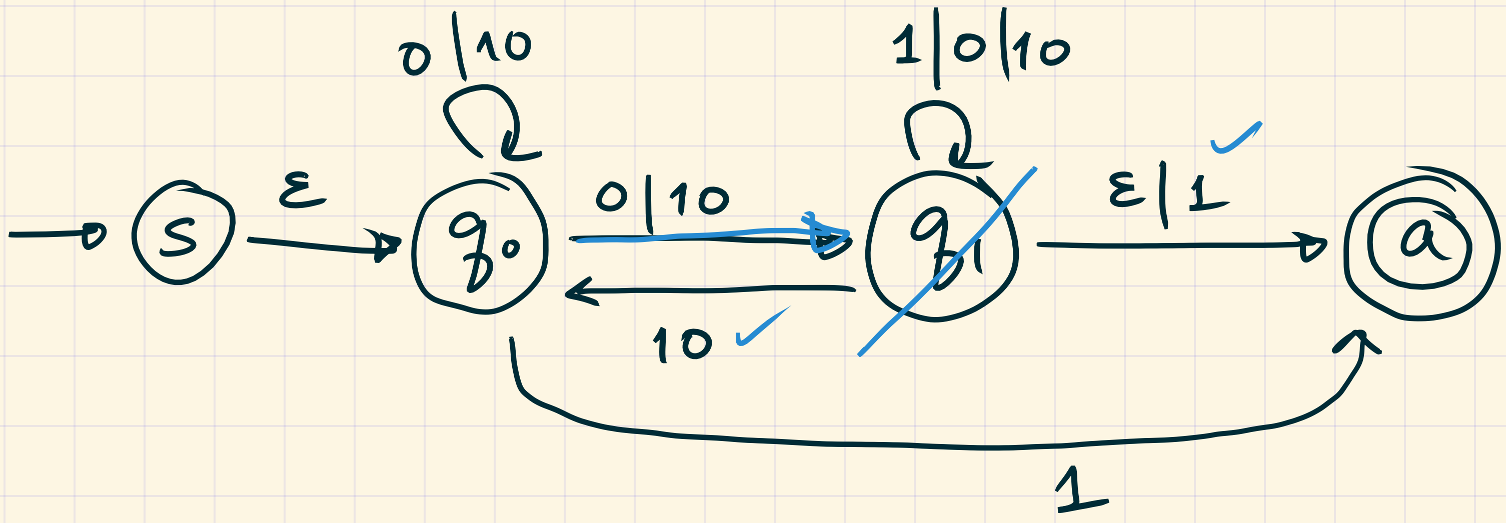
- one incoming arrow : (0|10)
- Two outgoing arrows : (10) and (ε|1)
- Self-loop : (1|0|10)

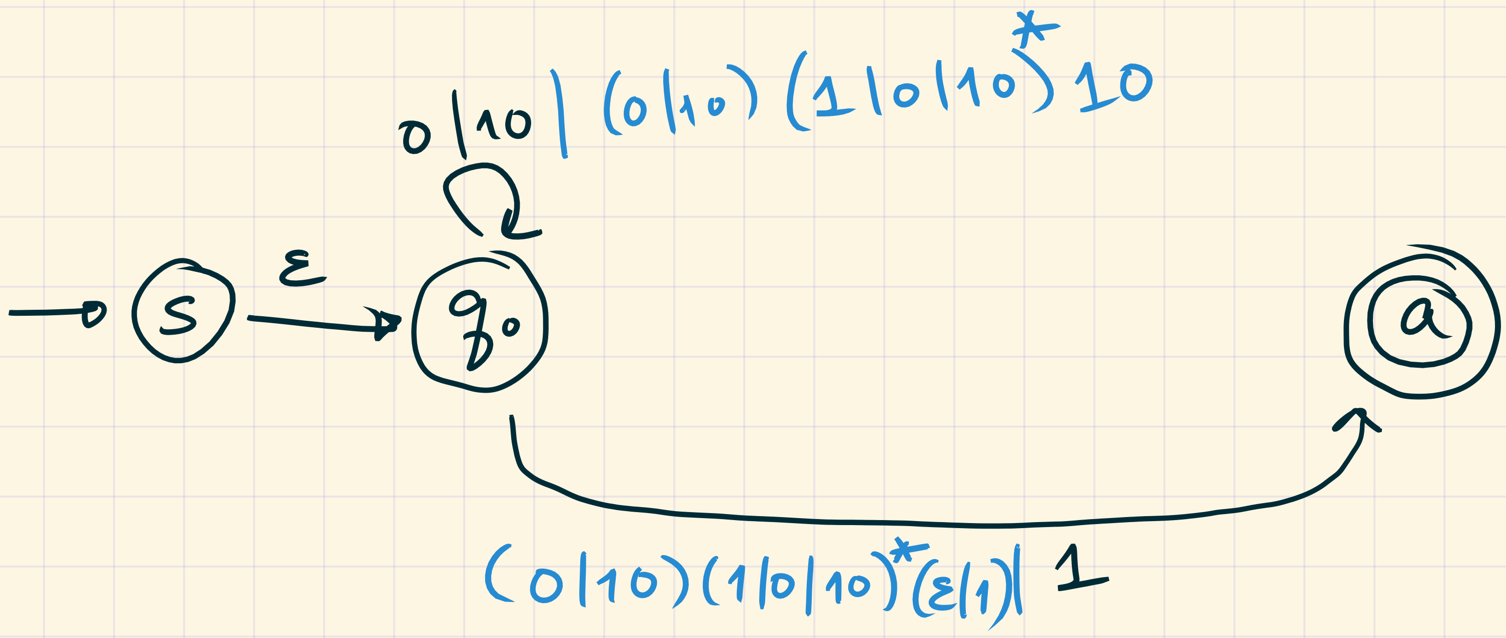
For every instance of :



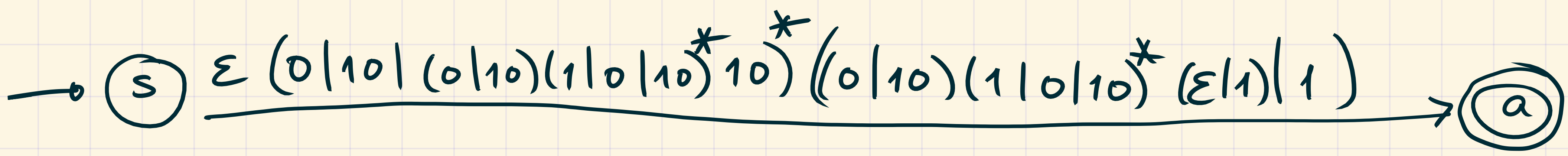
, draw







** Finally, eliminate q_0



We've (basically) proved the following:

** Theorem : For any NFA M, there is a regular expression r such that
$L(r) = L(M)$.

** Process :

1) Add (s) & (a) as explained.

2) For every internal state q r_3, and every

length-2 path

$$p_1 \xrightarrow{r_1} q \xrightarrow{r_2} p_2 \quad (q \xrightarrow{r_3} q)$$

, do the following:

- Add an edge $p_1 \longrightarrow p_2$ (if not already there)
- Add (via an "or" construction) the label

$r_1 r_3^* r_2$ to the existing label on $p_1 \longrightarrow p_2$.

3) Erase q.

4) Proceed until you only have s & a.

** Definition : A language L is regular if any of the following equivalent conditions hold:

1) there is some regex r such that $L = L(r)$
2) there is some NFA M such that $L = L(M)$
3) there is some DFA M' such that $L = L(M')$

** Fact : Not all languages are regular!

Example : $\{0^n 1^n \mid n \geq 0\}$ is not regular.