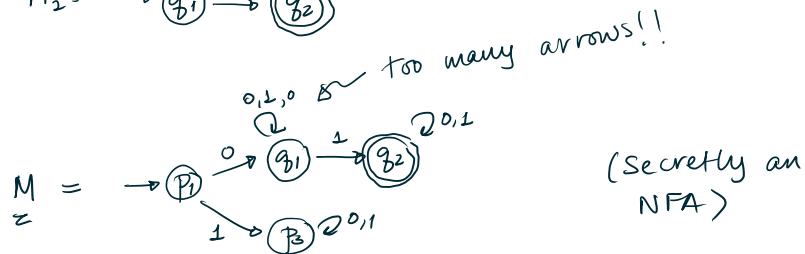
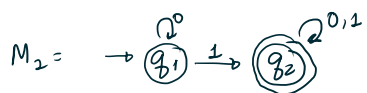
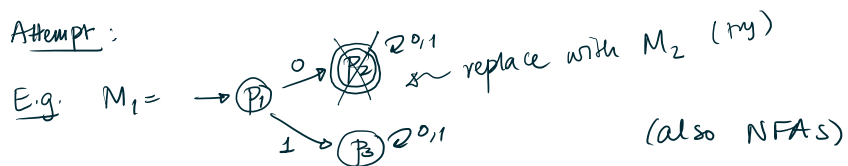


\* Last time: we tried (but failed) to produce a DFA recognising  $L(R_1 R_2)$ , given  $M_i$  recognising  $L(R_i)$ .

Attempt:



Solution: Change the rules! (Temporarily)

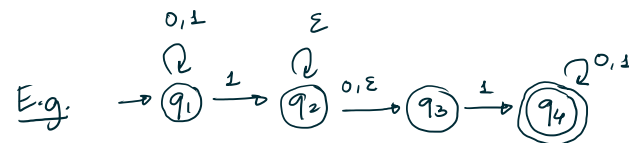
\* Non-deterministic finite automata (NFAs)

A directed graph like a DFA, with one start state, some accept states (possibly 0 accept states) labelled

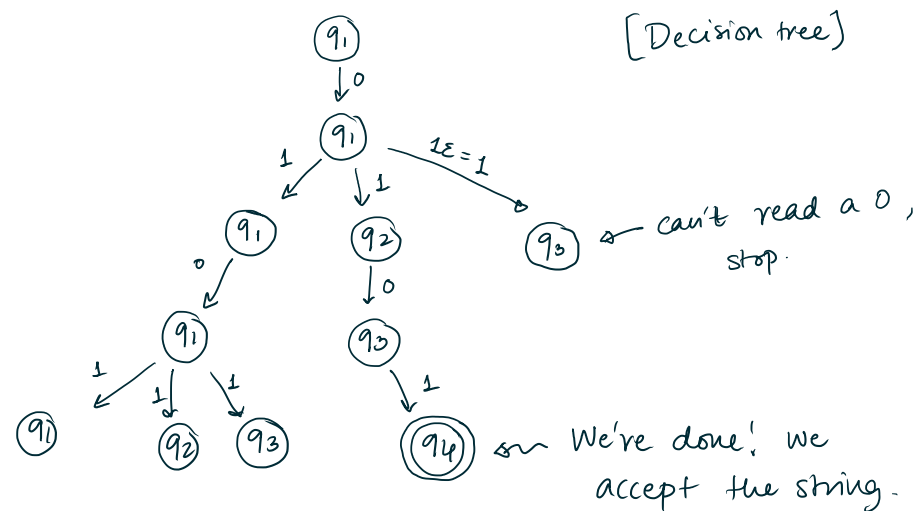
&  $\wedge$  arrows between states. Differences:

① For each state  $q \in Q$ , we allow zero or more outgoing arrows labelled by any letter of  $\Sigma$ .

② Some arrows can be labelled by " $\epsilon$ ".



Let's try to run a calculation:  $w = 0101$



\* We say that an NFA accepts a string  $w$  if at least one of the branches of its decision tree ends up at an accept state when  $w$  ends.

\* We treat all  $\epsilon$ -arrows as "portals", so that in the decision tree, when we read a symbol, we also read any  $\epsilon$ -arrow before & after the symbol. This ensures a finite decision tree.

\* Note that  $\epsilon$ -self-loops don't do anything. We can ignore them.

Formally: An NFA consists of:

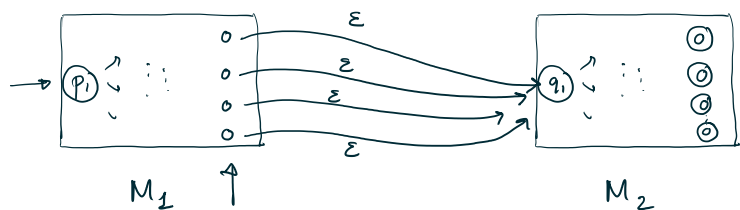
- ① A set of states  $Q$
- ② A start state  $q_1 \in Q$
- ③ Accept states  $A \subseteq Q$
- ④ Transition function:  $\Delta: Q \times (\Sigma \cup \{\epsilon\}) \rightarrow \mathcal{P}(Q)$   
↑  
power set of  $Q$

Given a state  $q$  & a symbol  $\epsilon$ , you go to multiple possible states, which can be represented by a subset of  $Q$ .

Later we'll see that any NFA can be converted to a DFA! (But for now, let's just use them.)

Recall: We want  $M$  such that  $L(M) = L(r_1 r_2)$ .  
⑤

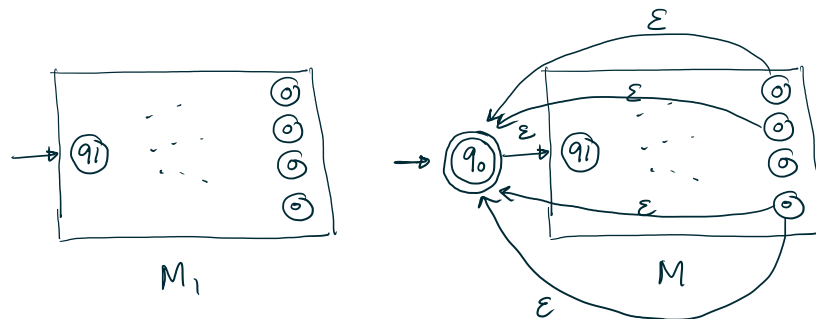
We already have  $M_1$  &  $M_2$



↑  
previously, these were accept states for  $M_1$   
Now we don't make them accepting.

This NFA exactly recognises  $L(r_1 r_2)$ .

⑥ Now we need a machine  $M$  which recognises  $L(r^*)$ , given  $M_1$  such that  $L(M_1) = L(r)$



This machine

- ① Accepts  $\epsilon$
- ② Accepts any number of concatenations of elements of  $L(r) = L(M_1)$

$$\Rightarrow L(M) = L(r^*)$$

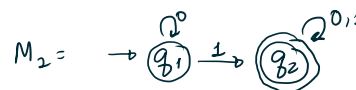
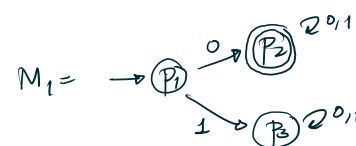
Exercise (Maybe in worksheet):

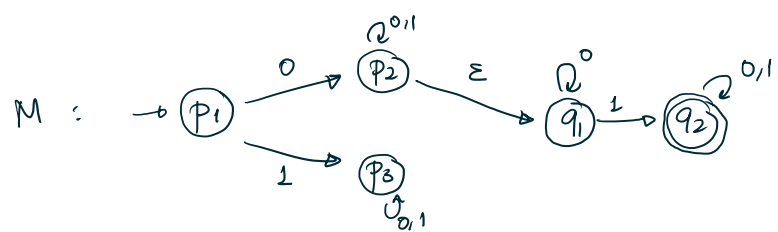
Give a simpler construction of an NFA that recognises  $L(M_1) \cup L(M_2)$ .

Exercise: Do some sample calculations using the NFAs we've constructed

\* Remember: An NFA accepts if at least one branch accepts, otherwise it rejects.

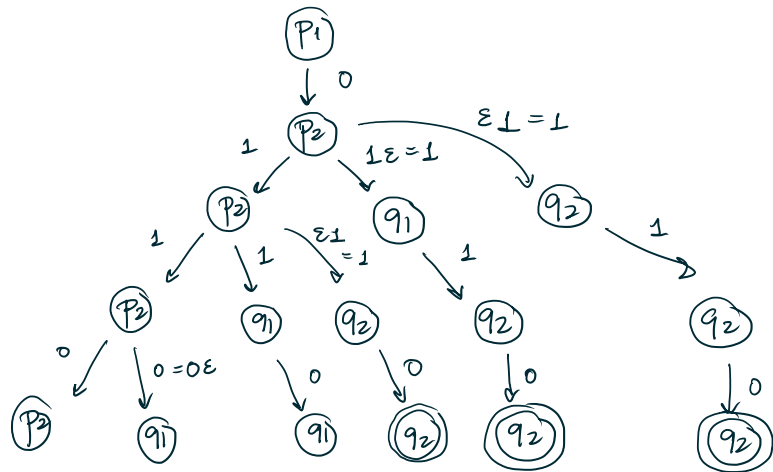
E.g.





Supposed to recognise  $L(M_1) \cap L(M_2)$

E.g.  $w = 0110$



We accept because we can reach  $q_2$  !

\* Next time : Equivalence of NFAs & DFAs, then  
NFA  $\rightarrow$  regular expressions.