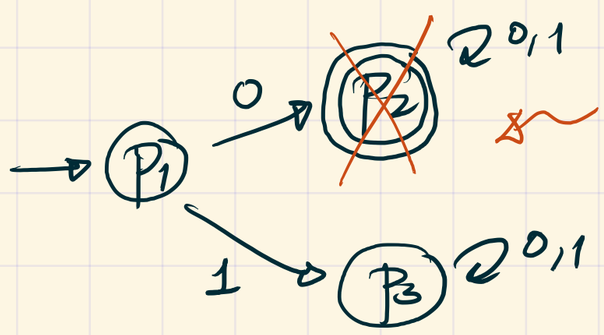
MATH 2301 7 Oct 2020

* Last time: we tried (but failed) to produce a DFA recognising $L(r_1 r_2)$, given M_i recognising $L(r_i)$.

Attempt:

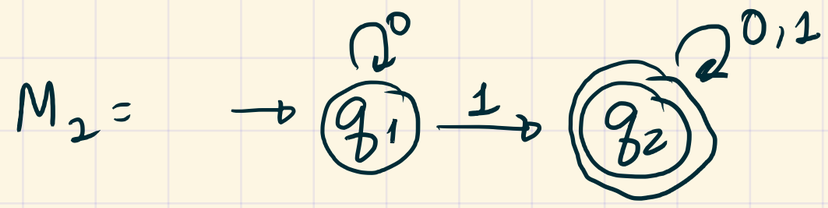
E.g. $M_1 =$



replace with M_2 (try)

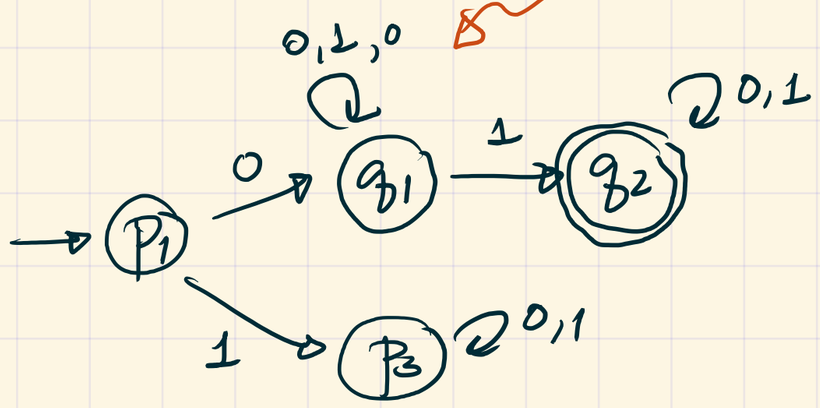
(also NFAs)

$M_2 =$



too many arrows!!

$M_3 =$



(secretly an NFA)

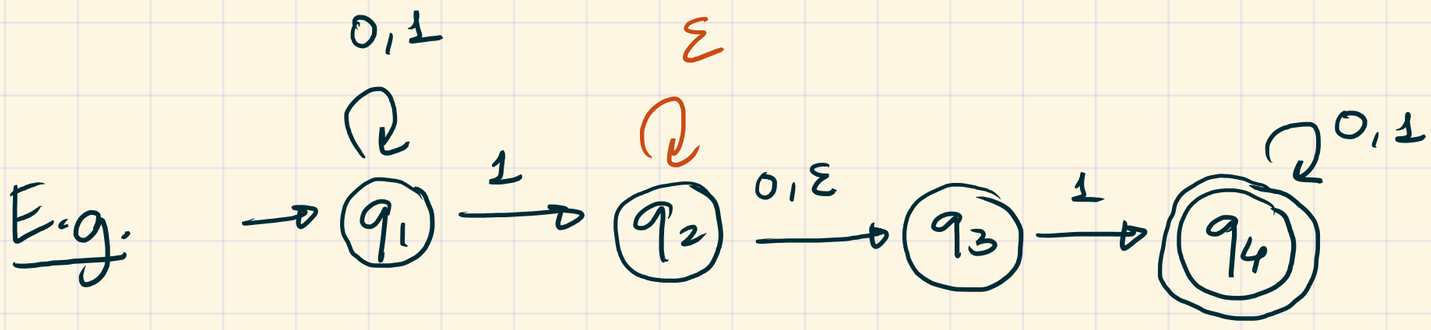
Solution: Change the rules! (Temporarily)

* Non-deterministic finite automata (NFAs)

A directed graph like a DFA, with one start state, some accept states (possibly 0 accept states) & labelled arrows between states. Differences:

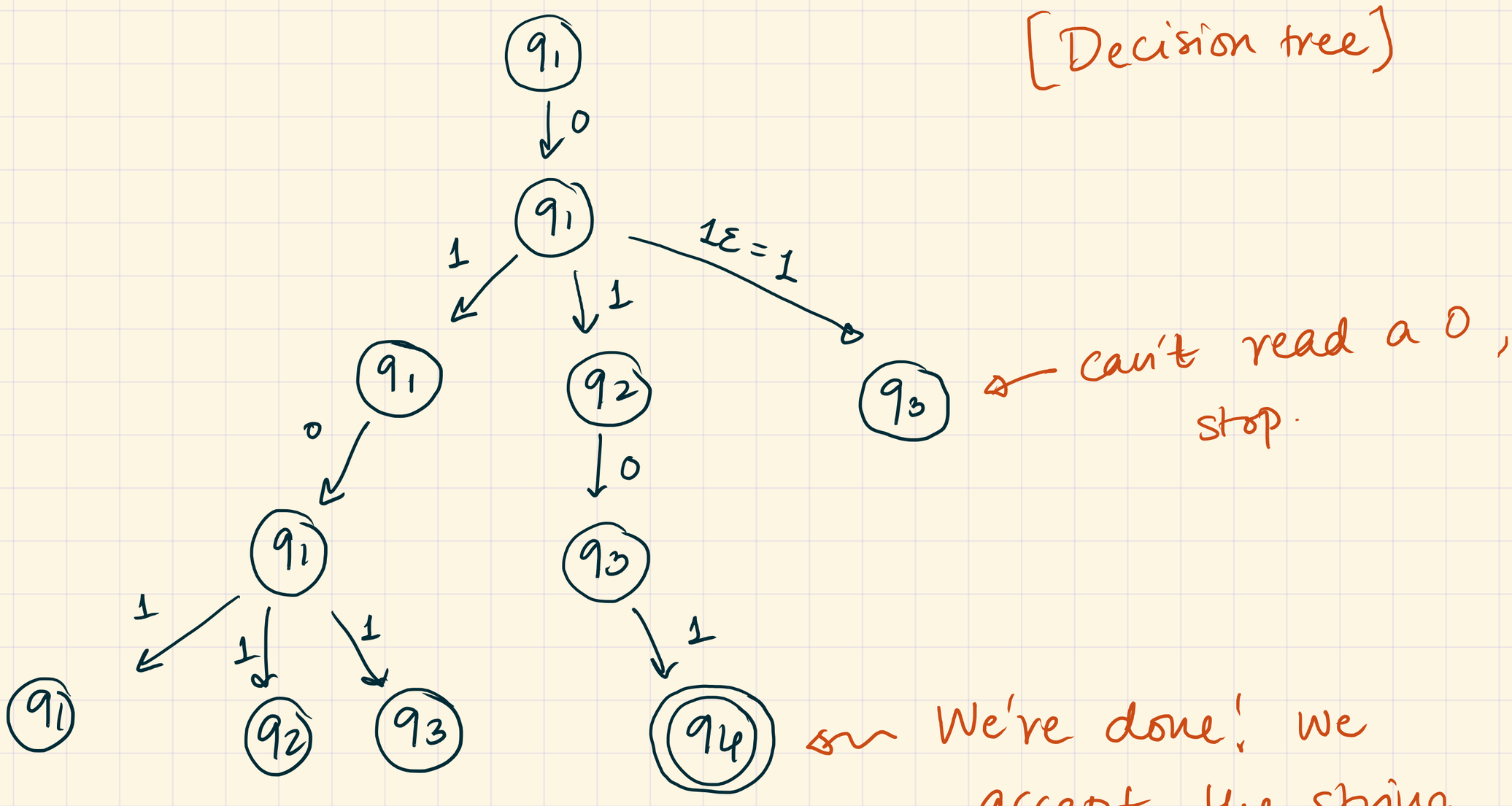
① For each state $q \in Q$, we allow zero or more outgoing arrows labelled by any letter of Σ.

② Some arrows can be labelled by "ε".



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

[Decision tree]



← can't read a 0, stop.

← We're done! we accept the string.

* 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 ε-arrows as "portals", so that in the decision tree, when we read a symbol, we also read any ε-arrows before & after the symbol. This ensures a finite decision tree.

* Note that ε-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: $\Delta : Q \times (\Sigma \cup \{\varepsilon\}) \to P(Q)$

↑ power set of Q

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

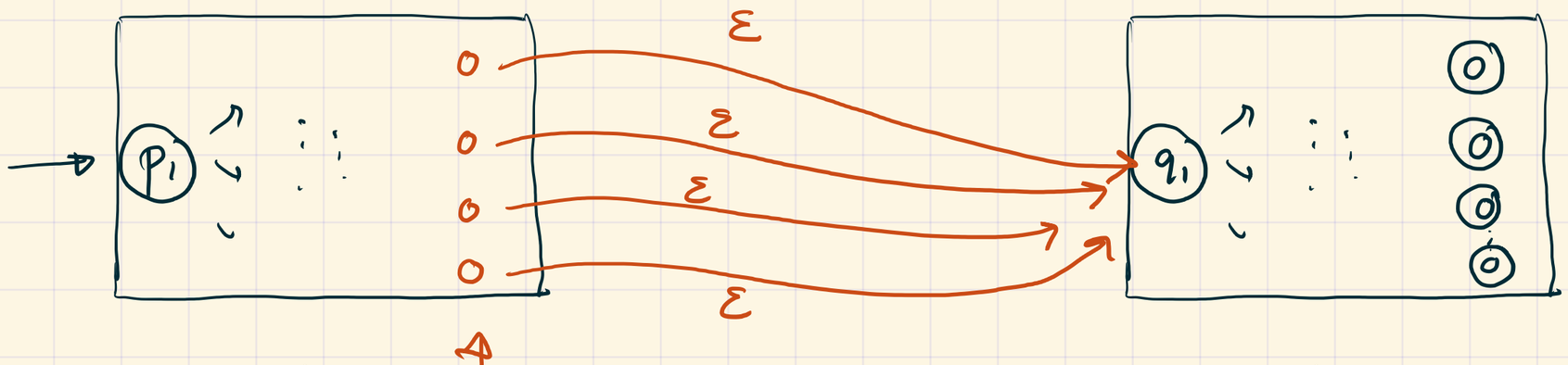
Later we'll see that any NFA can be converted to a DFA! (But for now, let's just use them.)

---

(5)

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

We already have M_1 & M_2



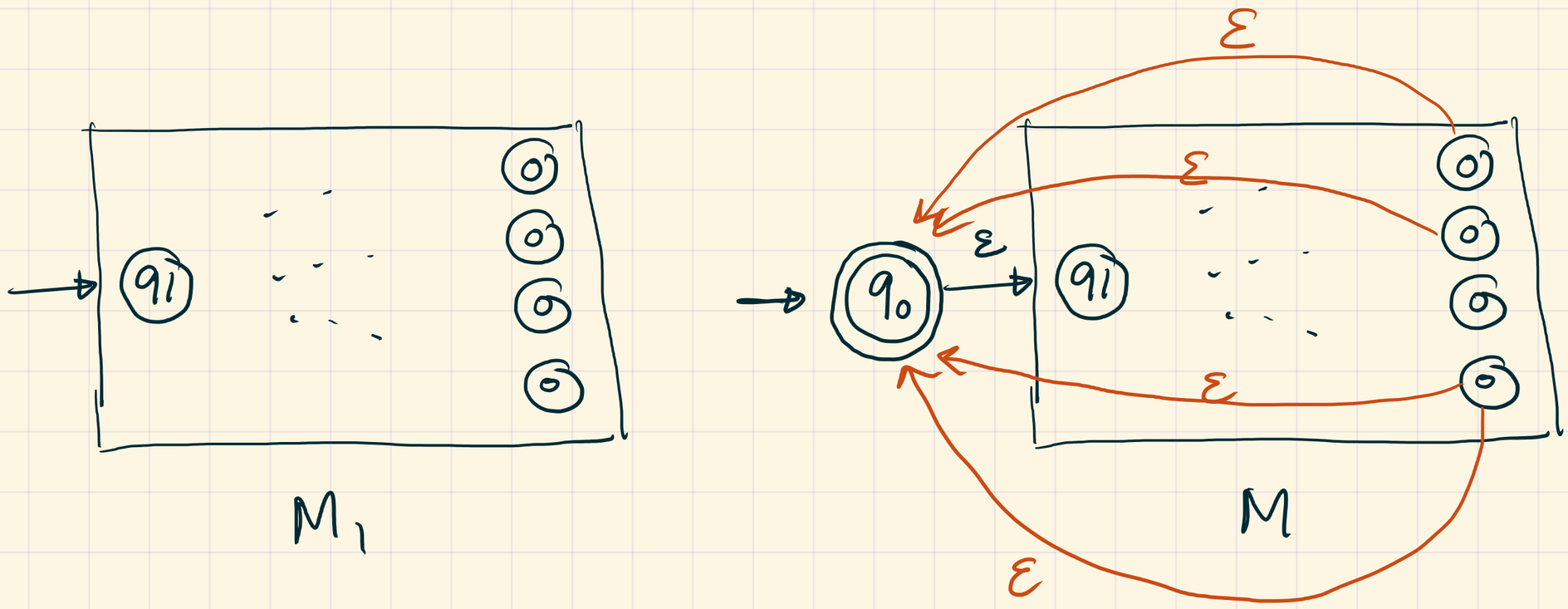
M_1 ↑ M_2

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 ε

② Accepts any number of concats of elements of $L(r) = L(M)$.

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

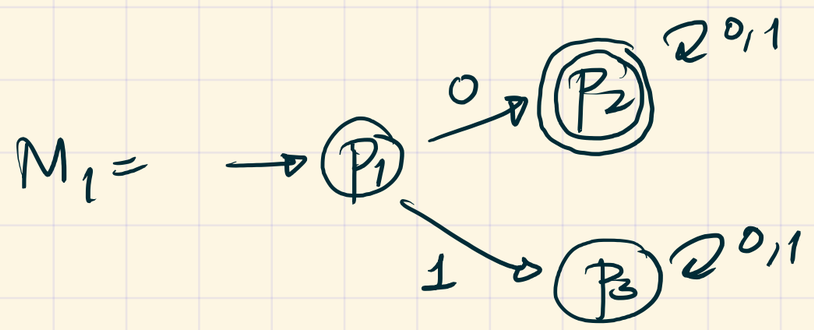
Exercise (Maybe in worksheet):

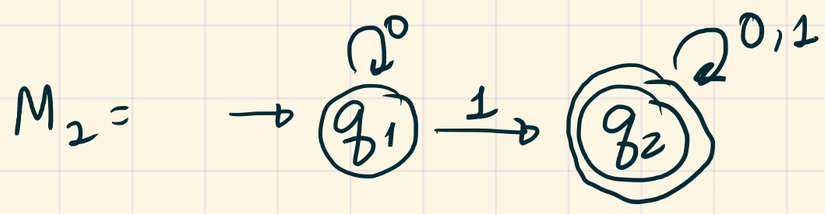
Give a simpler construction of an NFA that recognises $L(M_1) \cup L(M_2)$.

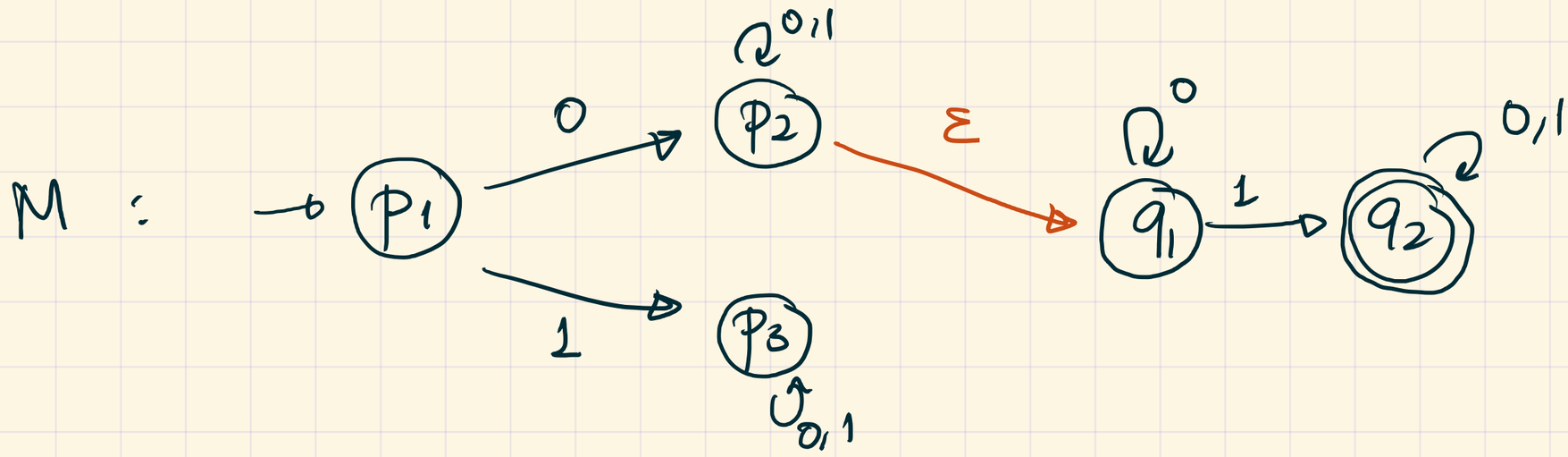
Exercise: Do some sample calculations using the NFAs we've constructed

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

E.g.

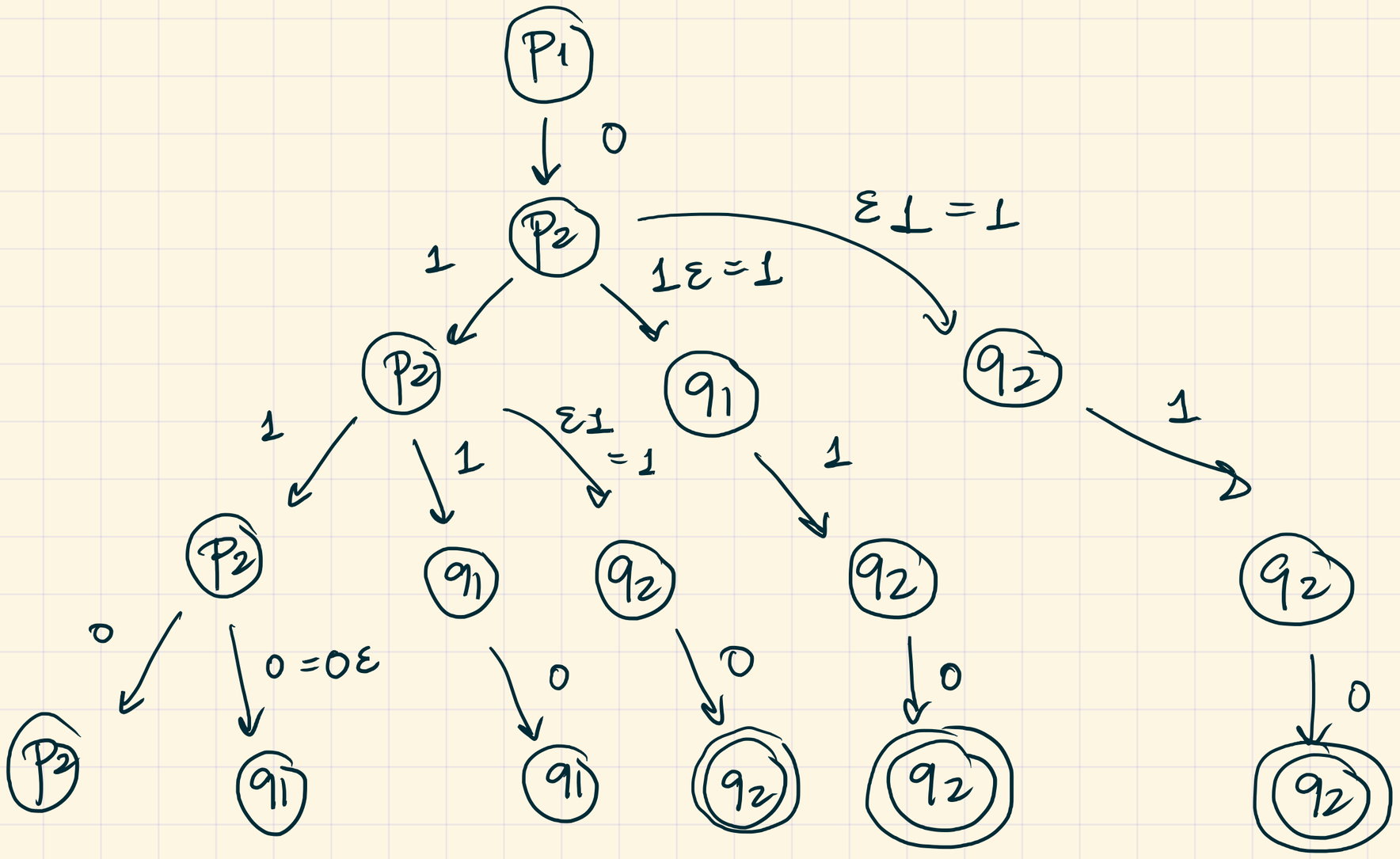






Supposed to recognise $L(M_1) \circ L(M_2)$

E.g. $w = 0110$



we accept because we can reach q_2!

* Next time : Equivalence of NFAs & DFAs, then

NFA → regular expression