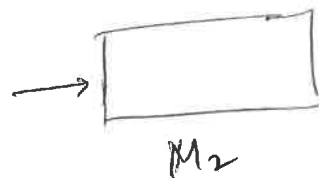
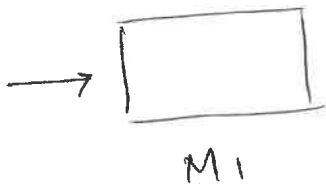


- * Admin: Next lecture Tuesday 03 October
2pm-3pm in Robertson (same room as Wed lecture)
- * Last time: Tried to construct DFAs to mimic regex constructors (we mostly failed for ④, ⑤)
- * Today: Alternate way to (partially) succeed ~~for~~, and a new flavour of machines.

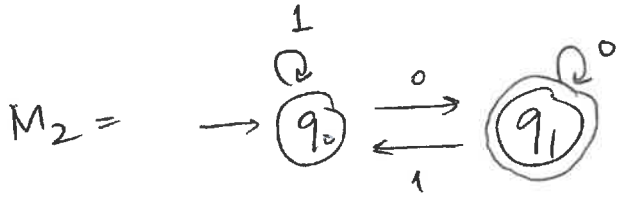
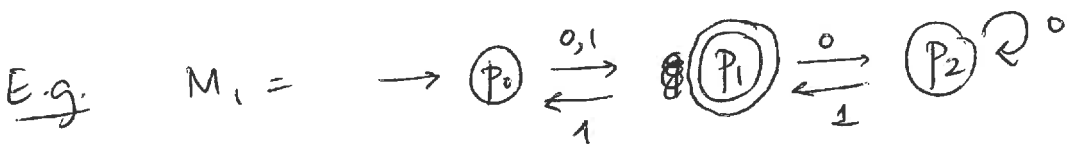
$$\textcircled{4} \quad r = r_1 | r_2 ; \quad L(r) = L(r_1) \cup L(r_2).$$

Given M_1, M_2 with $L(M_i) = L(r_i)$,
construct M s.t. $L(M) = L(r_1) \cup L(r_2)$
 $= L(M_1) \cup L(M_2)$

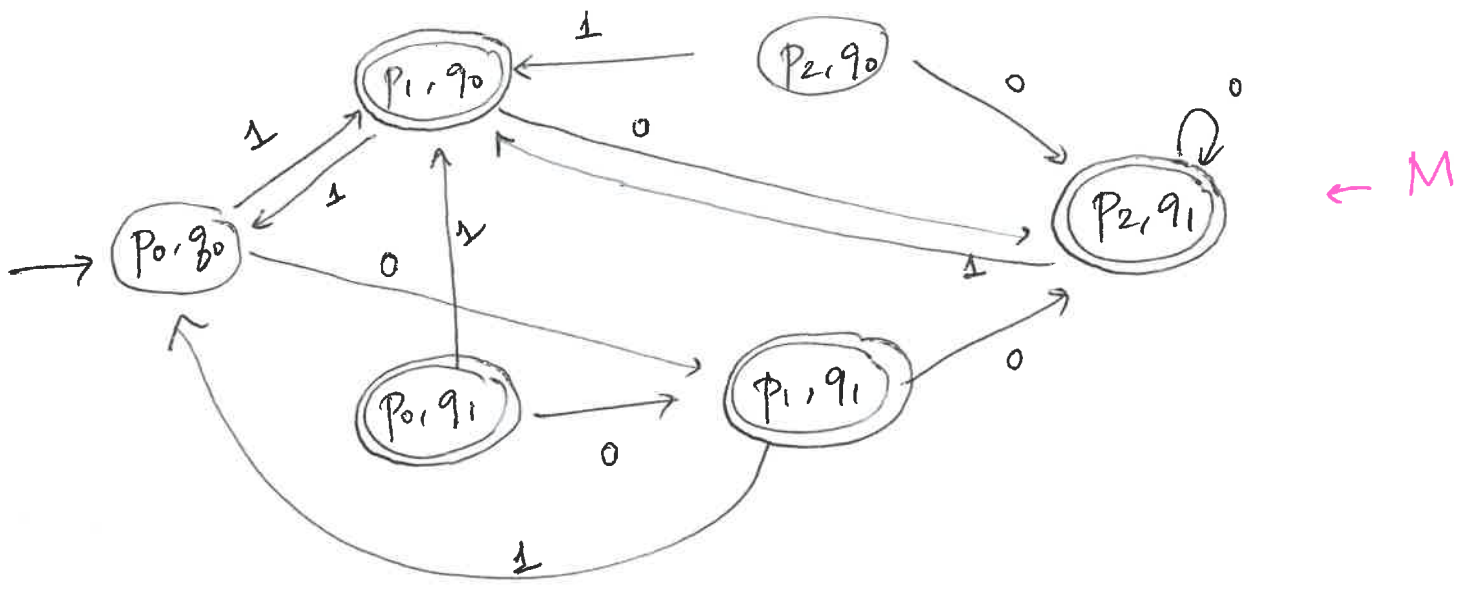


Given w , simulate M_1 and M_2 on w simultaneously?

* Solution : Take the "product" DFA!



Product automaton : (Keeps track of states in M_1 & M_2 simultaneously)



$L(M) = L(M_1) \cup L(M_2)$

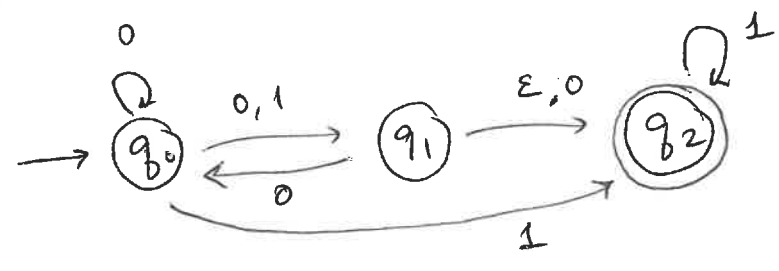
Accept states are (x,y) where either x is accepting in M_1 or y is accepting in M_2 (or both).

* What about concatenation and star??

* The other constructors are hard to construct DFAs for! We'll introduce an upgrade to our machines to make it easier.

* Non-deterministic finite automata (NFAs)

Example



An NFA state diagram resembles a DFA state diagram, with the following changes:

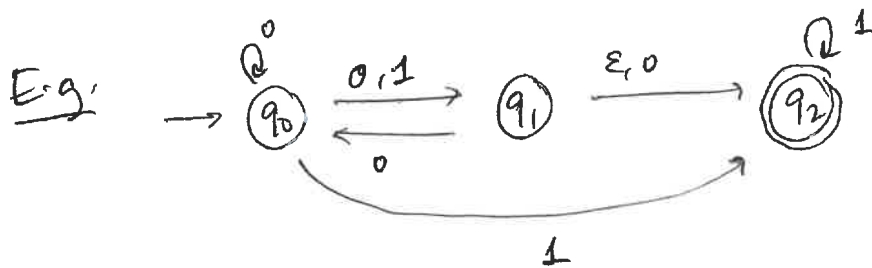
- (1) You can have states with no outgoing arrows labelled by one or more letters.
- (2) You can have multiple outgoing arrows labelled by the same letter.
- (3) You can have ϵ as edge labels.

Def: An NFA consists of the following:

- (1) An alphabet language Σ
- (2) A set of states Q
- (3) A start state $q_0 \in Q$
- (4) A set of accept states $A \subseteq Q$
- (5) A transition function

$$\Delta: Q \times (\Sigma \cup \{\epsilon\}) \longrightarrow \mathcal{P}(Q)$$

↑
state you're at
↑
what you read
↑
where you go
(a set of states)



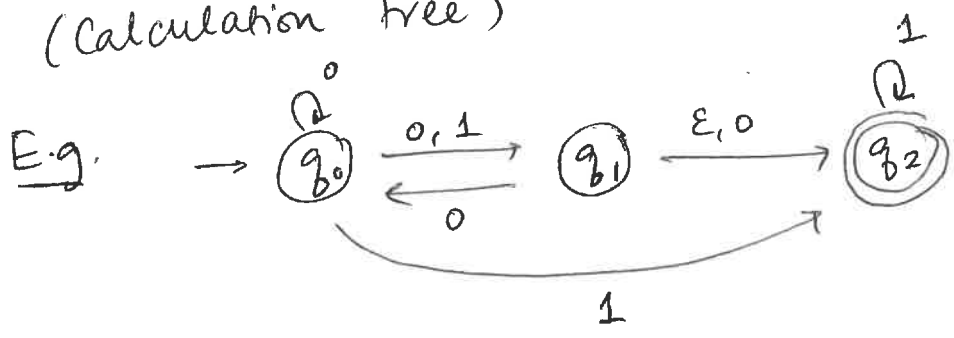
$$\Delta(q_0, 1) = \{q_1, q_2\}$$

$$\Delta(q_0, \epsilon) = \emptyset$$

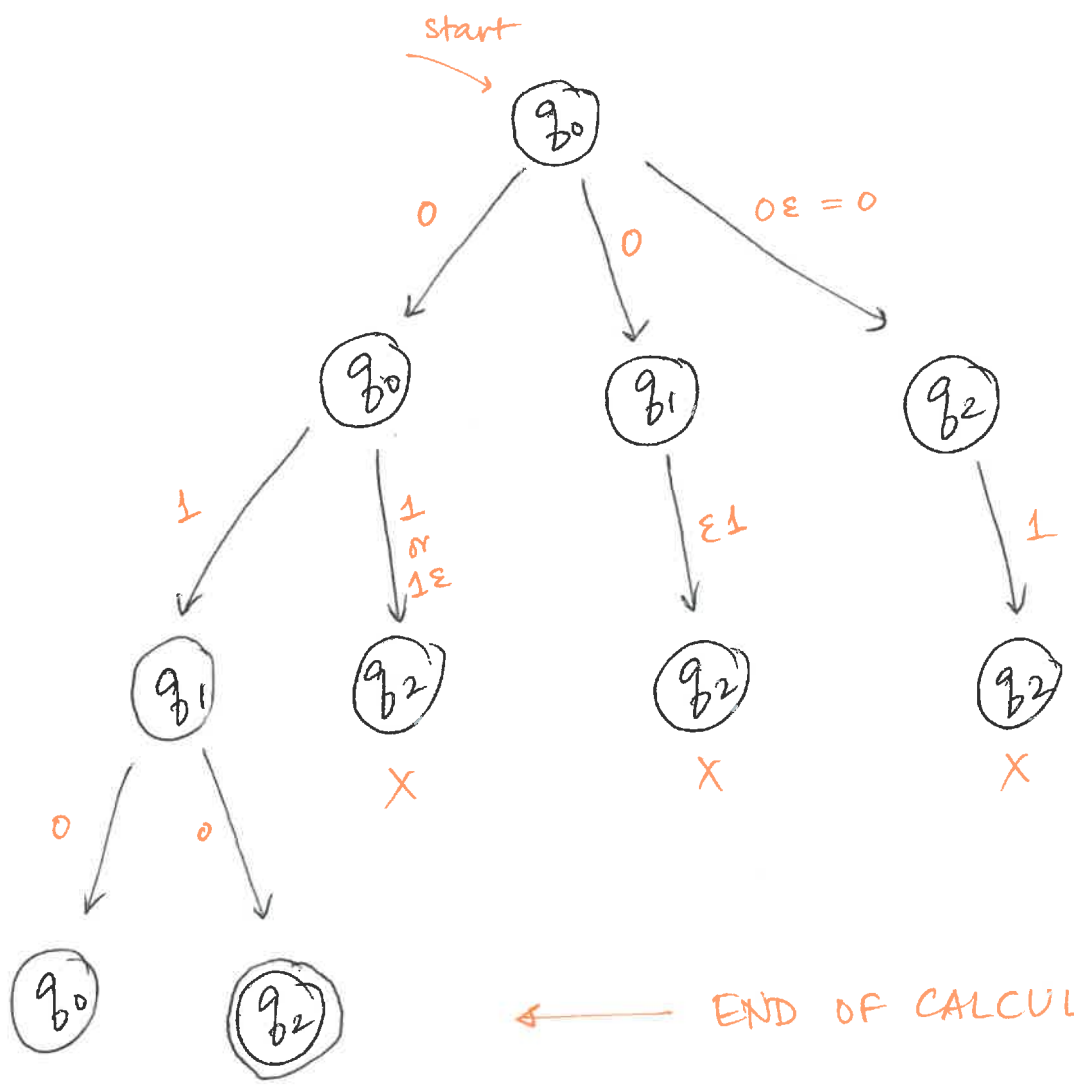
$$\Delta(q_0, 0) = \{q_0, q_1\}$$

$$\Delta(q_1, 1) = \emptyset$$

* How to compute with an NFA?
(Calculation tree)



E.g string
W = 010



Final set of states is $\{q_0, q_2\}$; q_2 accepts
so we accept.

(One of the final set is accepting!)