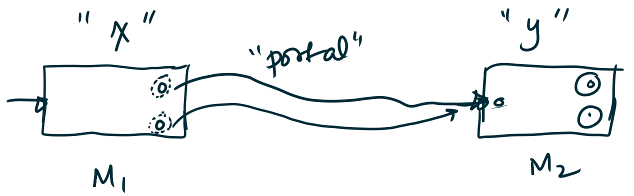
MATH 2301

** Converting regexes to DFAs

We have finished the cases of $r=\phi$, $r=\varepsilon$ and $r=a$ for $a\in\Sigma$

4) $r=r_1 r_2$ where r_1, r_2 (smaller) regexes.

Suppose we have M_1, M_2 such that
$L(M_1)=L(r_1)$ and $L(M_2)=L(r_2)$
Want to construct M such that $L(M)=L(r_1)\circ L(r_2)$
If $w=xy$, such that $x\in L(r_1)$, $y\in L(r_2)$



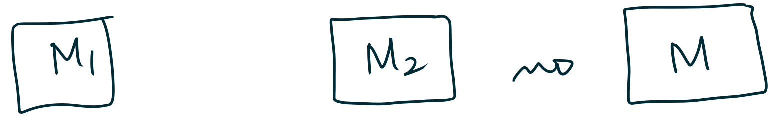
We'd like to do this...

Ideally: We want a method to "teleport" from the accept states of M_1 to the start state of M_2
This construction would exactly accept what we want, namely $L(M_1)\circ L(M_2) = L(r_1)\circ L(r_2)$
But at the moment it's not allowed.

We've stuck... with our current definition.

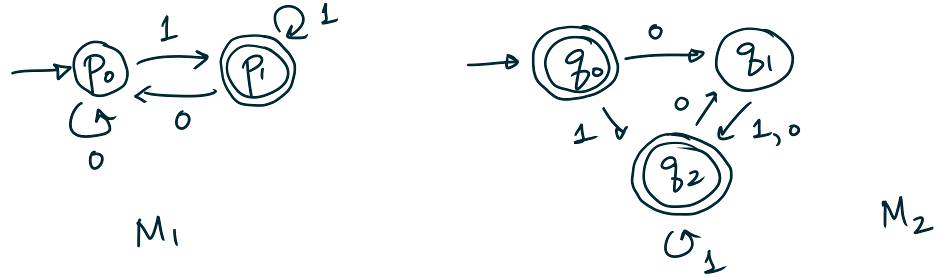
5) $r=r_1 | r_2$; $L(r)=L(r_1)\cup L(r_2)$

Suppose we have M_1 & M_2 such that
$L(M_1)=L(r_1)$ and $L(M_2)=L(r_2)$
Want to construct M such that
$L(M)=L(r_1)\cup L(r_2)$
In other words, $w\in L(M)$ if either it is in $L(M_1)$ or it is in $L(M_2)$.



We want to "simultaneously" run M_1 & M_2 on any given word, and accept if any one accepts.

** Example



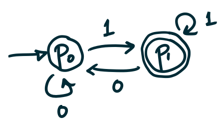
$w = 11011$

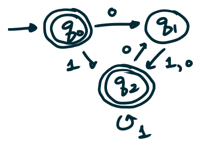
To simulate M_1 & M_2 simultaneously, we need "two pointers". We'll do this using a product DFA

** Product automaton

*** **Def** : Let M_1 be a DFA with P as its set of states, p_0 the start state, A the set of accept states, and $\delta_1 : P \times \Sigma \to P$

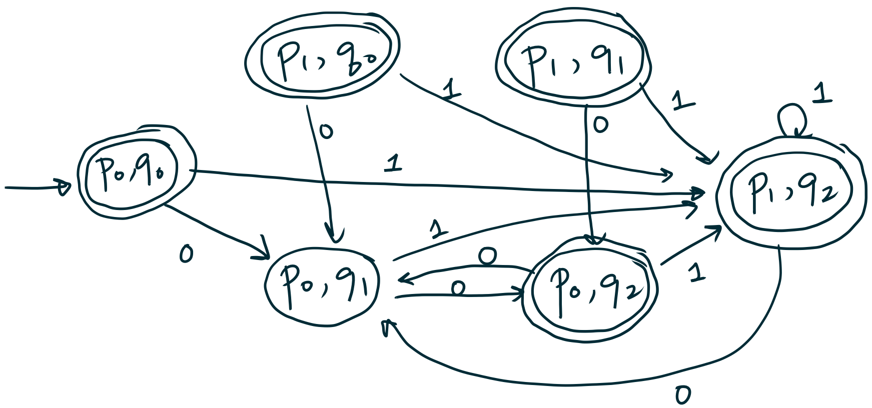
Similarly M_2 has Q as the state set, q_0 the start state, B the accept states, and $\delta_2 : Q \times \Sigma \to Q$





M = product automaton for $L(M_1) \cup L(M_2)$

- set of states : $P \times Q$
- start state : (p_0, q_0)



Transition function $\delta : (P \times Q) \times \Sigma \to P \times Q$

$$\delta((p,q), a) = (\delta_1(p,a), \delta_2(q,a))$$

Accept states (for $L(M_1) \cup L(M_2)$)

We say that $(p,q) \in P \times Q$ is an accept state for M, if :

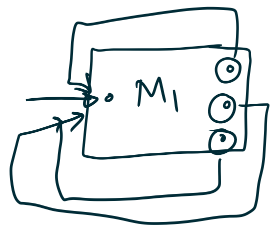
either $p \in A$ or $q \in B$ (or both)

accepting states of M_1 & M_2 resp.

This finishes the case $r = r_1 | r_2$

6) $r = (r_1)^*$.

Given M_1 s.t. $L(M_1) = L(r_1)$, we want M, such that $L(M) = L(r_1)^*$.



Want to teleport from the accept states of M_1, to the start state.

This will essentially give us what we want, but we also need to accept ε ... simply making the current start state an accepting state is not correct

** **Remark** : It is possible to write down DFAs that work for the $r_1 r_2$ & $(r_1)^*$, but it's not at all obvious.

** <u>Preview</u> : We'll introduce non-deterministic finite automata (NFAs), i.e. DFAs with choices.