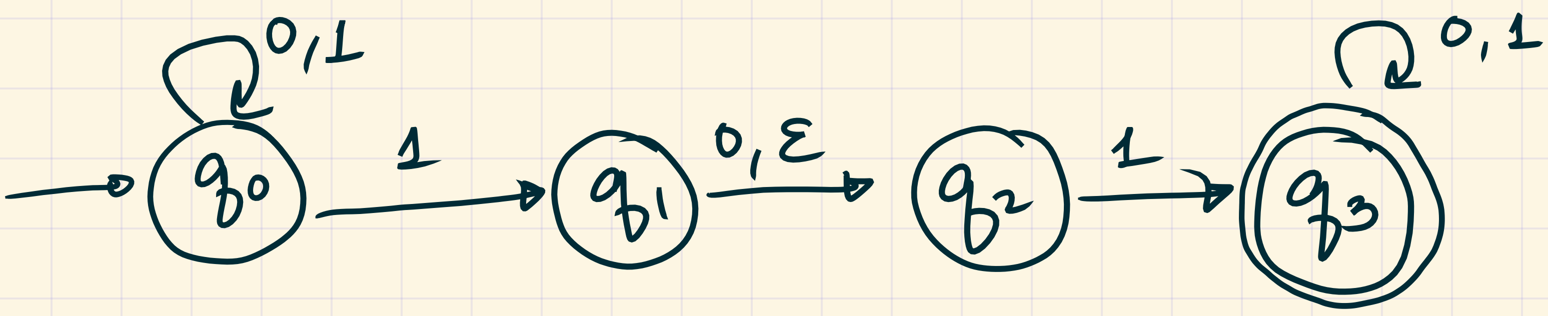
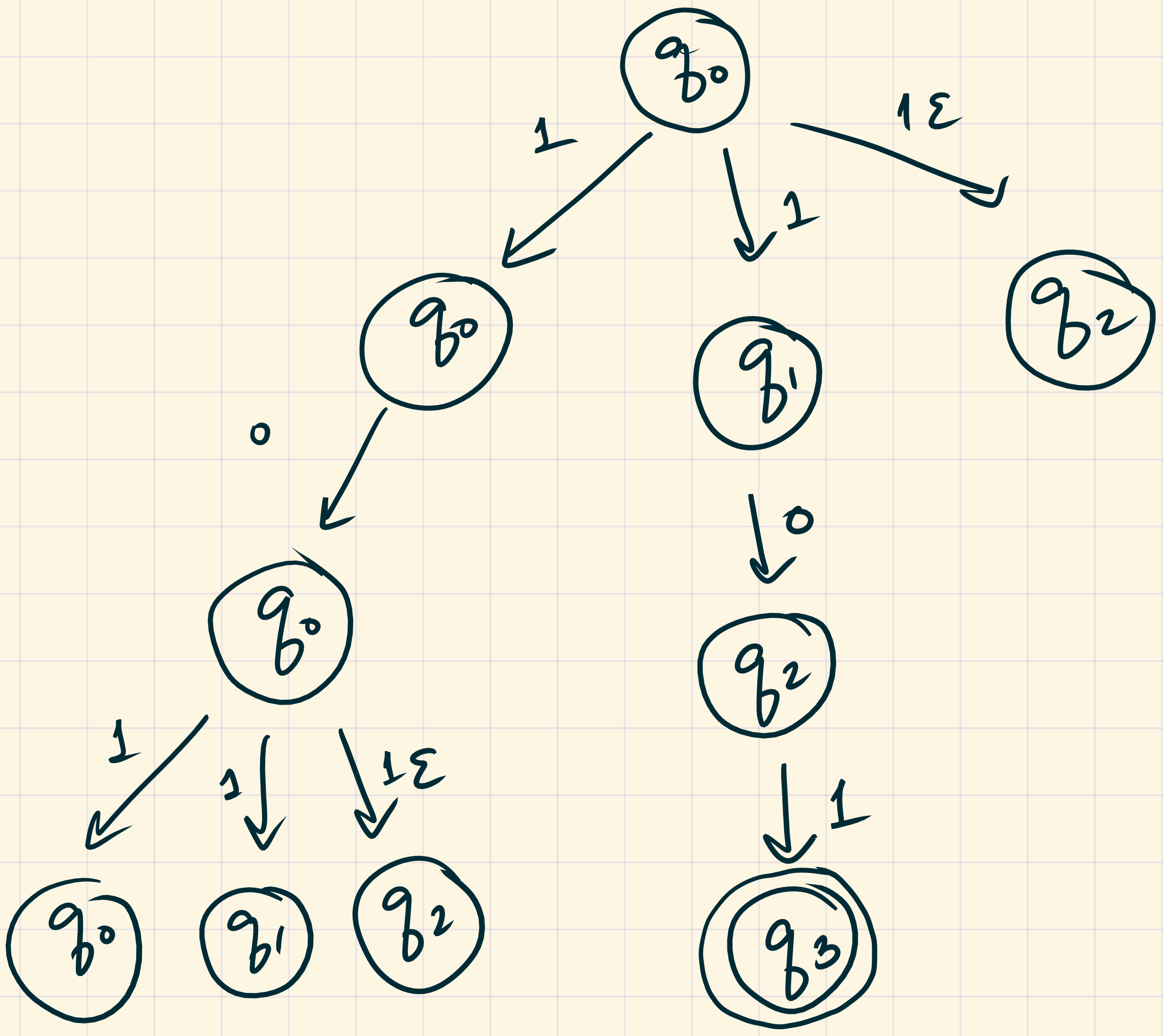
MATH 2301

* Goal : Convert any NFA into an equivalent DFA (one that recognises the same language).

** Motivating example



$w = 101$



The calculation tree is a deterministic process.

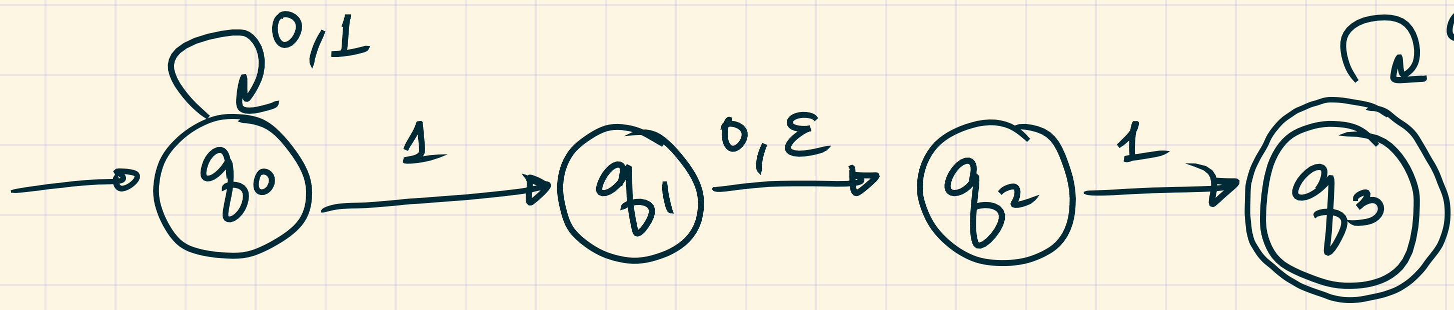
After each step, we get a set of states.

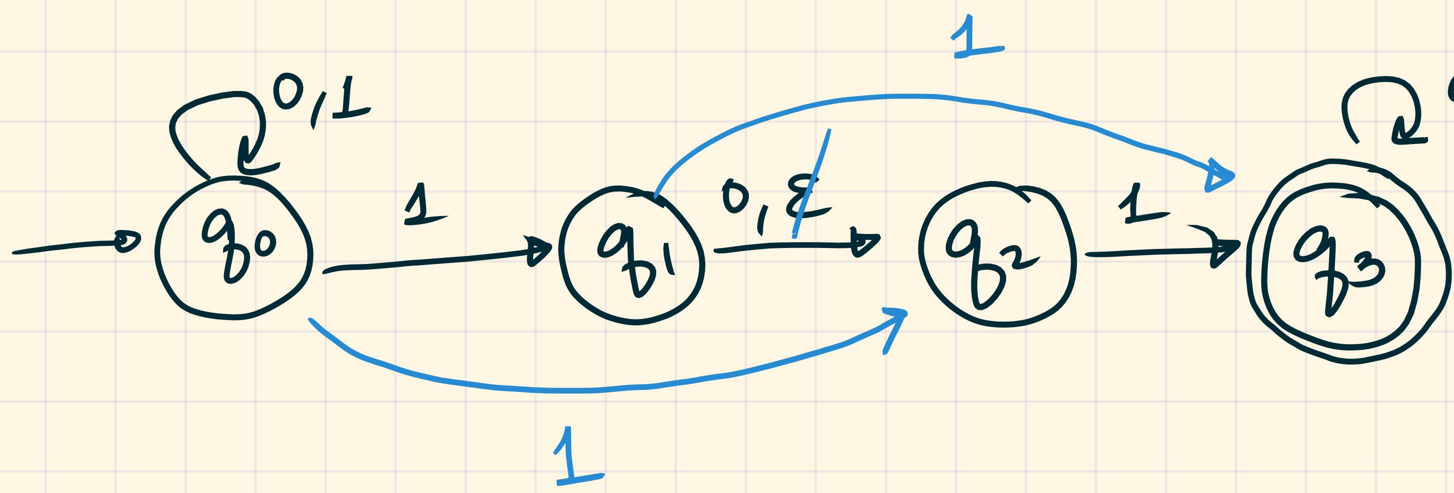
* But how to fit it into the definition of a DFA?

** Removing ε-arrows

Prop: Any NFA can be converted to an equivalent one without any ε-arrows.

*** Example





** We won't prove this formally, but the idea is to add direct arrows from

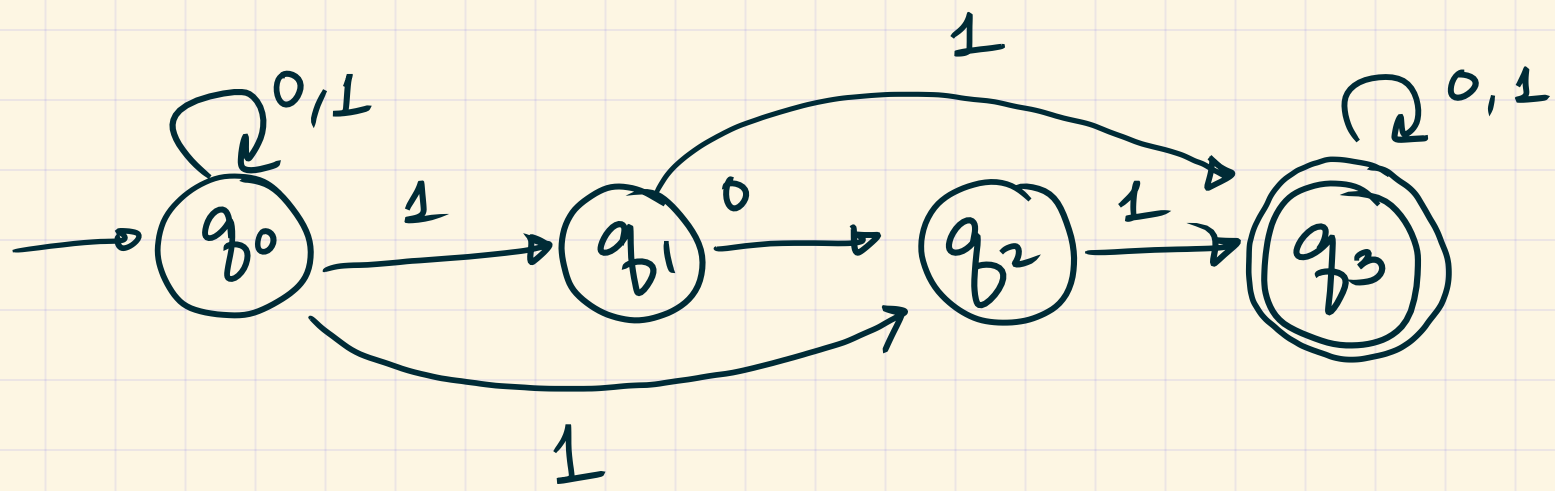
$A \xrightarrow{a} C$, any time there are arrows

of the form $A \xrightarrow{\varepsilon} B \xrightarrow{a} C$ or

$A \xrightarrow{a} B \xrightarrow{\varepsilon} C$

* Now assume that we have an NFA that has no ε-arrows. We'll try to convert this to an equivalent DFA.

** NFA → DFA



Say we have an NFA M w/ states Q, start state q_0, accept states $A \subseteq Q$, and transition function $\Delta : Q \times \Sigma \to P(Q)$

we've removed ε-arrows!

We construct a new DFA M' as follows:

- Set of states of M' is $P(Q)$ ⇝ power set of Q.
- Start state of M' is: $\{q_0\}$ ⇝ the calculation tree of M always begins here!
- Accept states of M' are:

  $\{X \subseteq P(Q) \mid X$ contains at least one element of $A\}$

  $= \{X \subseteq P(Q) \mid X \cap A \neq \phi\}$ ⇝ at least one branch succeeds.

- Transition function

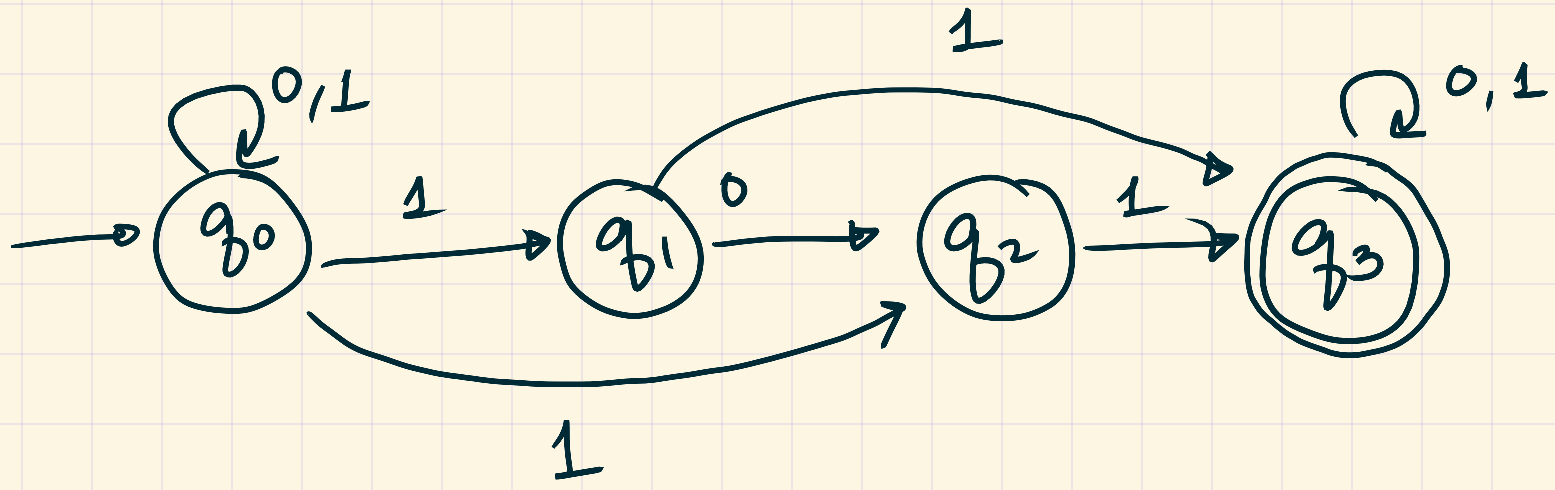
  $\delta : P(Q) \times \Sigma \longrightarrow P(Q)$

  (state, symbol) $\mapsto$ state.

Let $X \subseteq P(Q)$, and let $a \in \Sigma$

$$\delta(X, a) = \bigcup_{q \in X} \Delta(q, a)$$

apply transition fn of M to (q, a), ranging over all $q \in X$.
Then take union over all possible q.



The corresponding DFA has states $P(Q) = P(\{q_0, q_1, q_2, q_3\})$ ← 16 states

(In general, it'll have 2^n states.)

Start state is $\{q_0\}$

Accept states: All subsets containing q_3. ← 8 states

Examples of δ

$$\delta(\{q_0, q_1\}, 1) = \{q_0, q_1, q_2, q_3\}$$

$$\delta(\{q_2\}, 0) = \phi$$

$$\delta(\{q_2, q_3\}, 1) = \{q_3\}$$

** Recap : Every NFA has an equivalent DFA.
(ie, recognising the same language.)

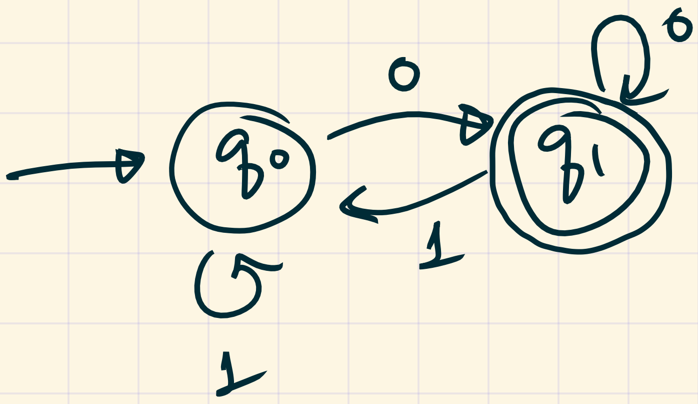
$\Rightarrow$ Every regex has an equivalent DFA!

Unfortunately, this causes an exponential space blowup (n states become 2^n.)

---

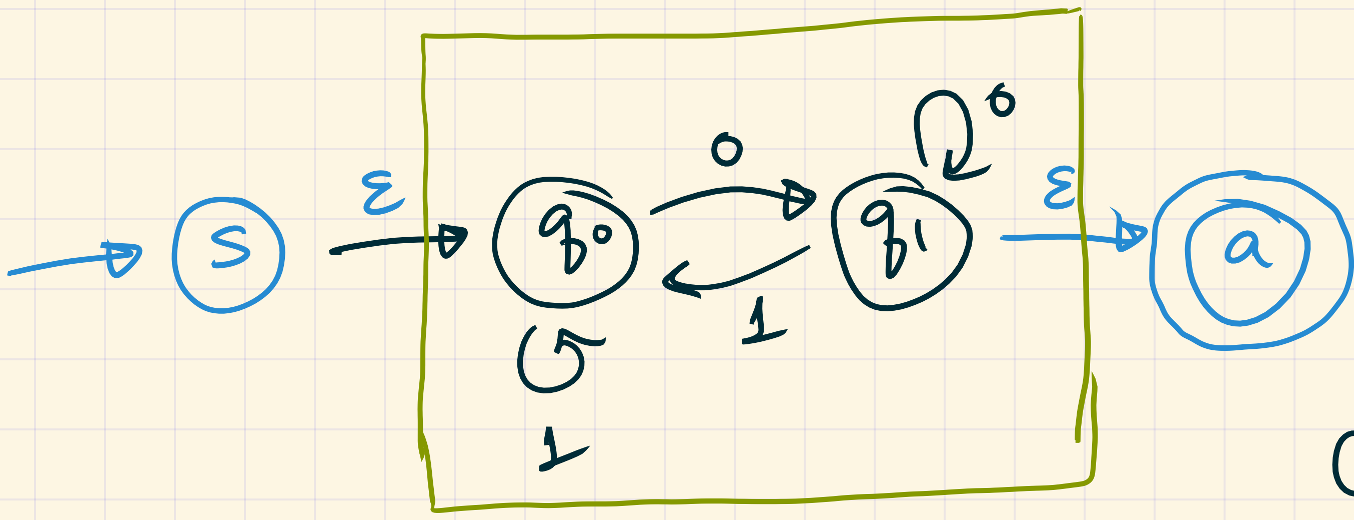
* What we'll attempt next : convert any NFA/DFA into an equivalent regular expression.

** Example



Idea : Go state-by-state and try to eliminate states.

Simplification #1



Add a new start state ⓢ, and a new accept state ⓐ

Connect ⓢ to q_0 with an ε-arrow. Connect all old accepting states to ⓐ by ε arrows, and make them reject.