Today:

0. Review

1. Regular Ops on Languages
   - 1.1 Regular languages closed under $\cup$, $\cap$, $\overline{}$

2. Nondeterministic FA's
   - 2.1 NFAs reduce to DFAs
   - 2.2 Reg langs closed under $\circ$, $*$

3. Regular Expressions

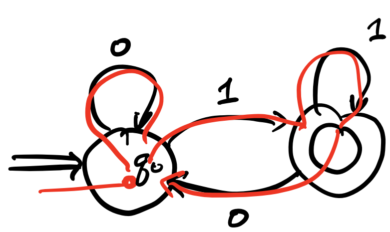
Review

Languages (like $\{0, 11, 010\}$, $\{0,1\}^*$) are sets of strings.

DFAs are math machines that read in strings (left to right, character by character) and say YES/NO.

$L(D)$, the language of a DFA D, is the set of strings D accepts.

(Regular languages are those recognized by DFAs)



$\Sigma = \{0, 1\}$

0110

Formally, a DFA can be summarized by a 5-tuple

$(Q, \Sigma, q_0, F, \delta)$ $\quad \delta : Q, \Sigma \to Q$

~~state~~ set — alphabet $\{0, 1\}$ — start state — accept states

Ex. Formal → state diagram for DFA

Let $D = (Q, \Sigma, q_0, F, \delta)$ be a DFA.

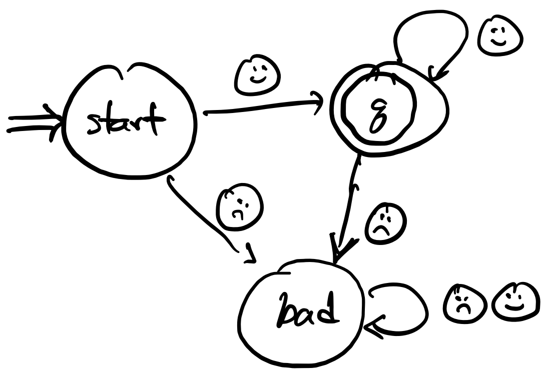
$Q = \{\text{start}, q, \text{bad}\}$

$\Sigma = \{☺, ☹\}$

$q_0 = \text{start}$

$F = \{q\}$

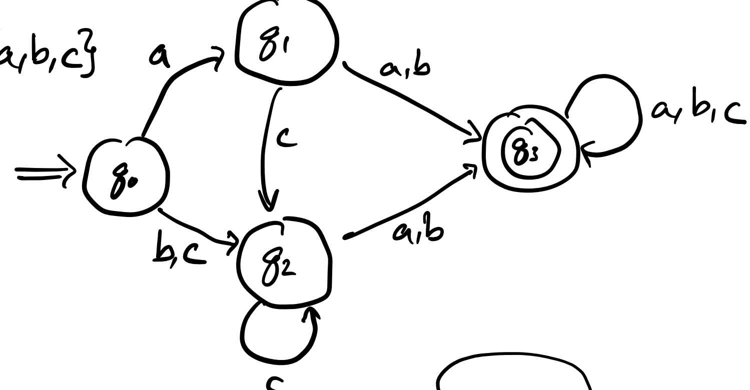
δ	☹	☺
start	bad	q
q	bad	q
bad	bad	bad



$L(D)$ = strings where everyone is happy.

DFA → formal def.

$\Sigma = \{a, b, c\}$



$D = (Q, \Sigma, q_0, F, \delta)$

↳ $\{a, b, c\}$ $\quad F = \{q_3\}$

$Q = \{q_0, q_1, q_2, q_3\}$

δ:

	a	b	c
q_0	q_1	q_2	q_2
q_1	q_3	q_3	q_2
q_2	q_3	q_3	q_2
q_3	q_3	q_3	q_3

---

1. Regular Operations.

L_1 is regular $\longleftrightarrow$ some DFA D recognizes it.

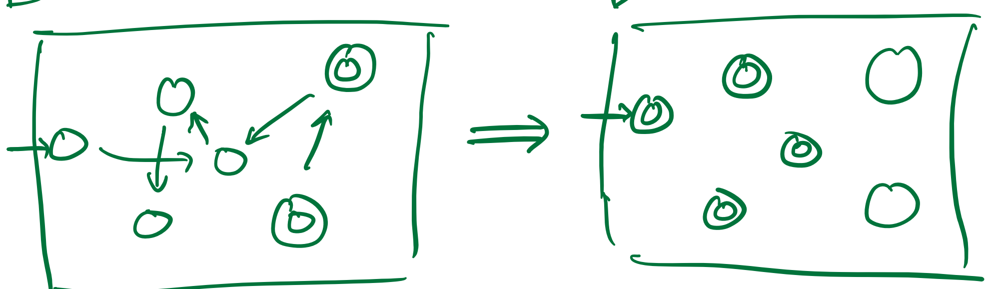
Can we define 'macros' that make it easier to prove languages are regular?

Q: If L_1 is regular, is $\overline{L_1}$ also regular?

every string over the same alphabet that's not in L.

Some DFA D has $L(D) = L_1$

D $\Longrightarrow$ D'



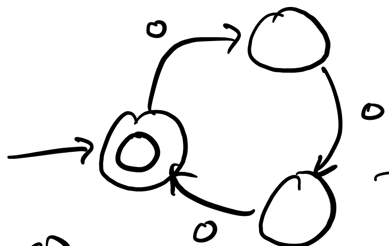
- In D, a string w ends at an ◎ if and only if $w \in L_1$.
- $\therefore$ In D', we accept w if and only if $w \notin L_1$.

(relation of D and D'? maybe "complement"?)

"Is L_2 regular?" "$\overline{L_2}$ is regular, so yes!"

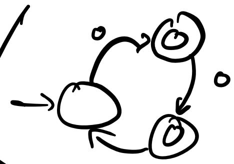
$L_2 = \{x \mid x$ is a string of 0's with length **not** divisible by 3.$\}$

$\Sigma = \{0\}$



accepts if $|w|$ is divisible by 3.

recognizes $\overline{L_2}$

( recognizes L_2)

Def. When a regular operation is applied to regular languages, the result is regular.

A, B regular $\rightarrow$ $A \circ B$ regular

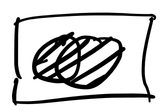
$A \circ B$ regular $\not\rightarrow$ A, B regular

List of regular operations:

✓ – Complement. $\overline{A} := \{x \mid x \notin A\}$

(If A regular, $\overline{A}$ is regular) $(\overline{A})$

todo

– Union. $A \cup B := \{x \mid x \in A$ or $x \in B\}$ 

– Intersection $A \cap B := \{x \mid x \in A$ and $x \in B\}$ 

– Concatenation $A \circ B := \{wx \mid w \in A, x \in B\}$

$\neq B \circ A$

- (Kleene) Star: $A^* := A \circ A \circ A \circ \cdots \circ A$
$= \{x_1 x_2 \cdots x_k \mid x_i\text{'s} \in A,\ k \geq 0\}$

$\{0,1\}^* = \{\varepsilon, 0, 1, 00, 01, 10, 11,$
$000, 001, \ldots \}$

Puzzle: A such that A^* is finite?

$A = \{\},= \emptyset, \quad A^* = \emptyset^* = \{\varepsilon\}$

$A = \{\{\}\} \rightarrow$ (not quite defined.)

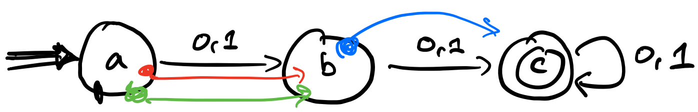
$A^* = \{\varepsilon\}^* = \{\varepsilon, \varepsilon\varepsilon, \varepsilon\varepsilon\varepsilon, \ldots\} = \{\varepsilon\}$

$\{x \mid x \in \{0,1\}^*,\ x = 0^k 1^k \text{ for } k \geq 0\}$ - nonregular

If we can prove operation regularity, we can use the operation to show languages are regular!

1.1 Simulating two DFAs at once.

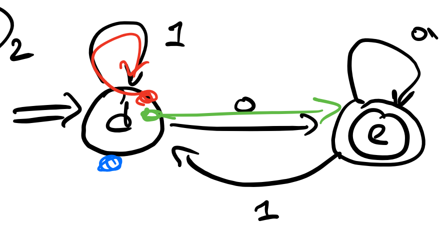
D_1



00 ✓

$L(D_1) = x \in \{0,1\}^*$ with ≥ 2 characters.

D_2

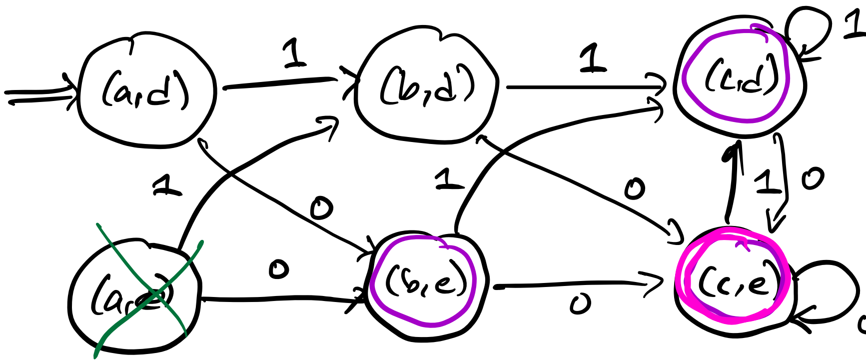


$L(D_2) =$ strings that end in 0 (again over $\{0,1\}$).

What about $L(D_1) \cup L(D_2)$? (strings w/ length ≥ 2 OR strings that end in 0)

$L(D_1) \cap L(D_2)$? (length ≥ 2 AND end in 0).

Create a DFA with a state for every pair of states in D_1, D_2:



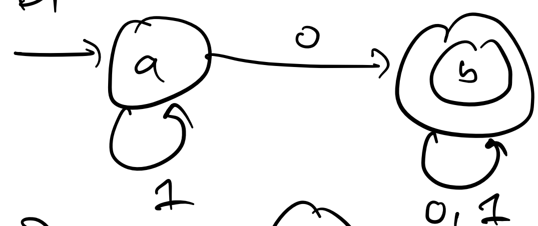
Transition as if running D_1, D_2 simultaneously.

— Resume 2:23 —

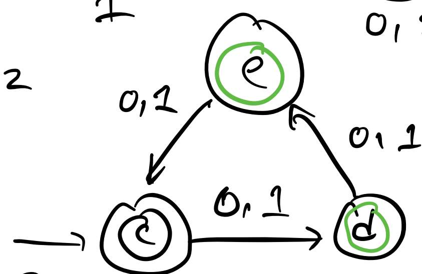
Challenge ex —

Build a DFA for $\underline{L(D_1) \cap L(D_2)}$.

D_1



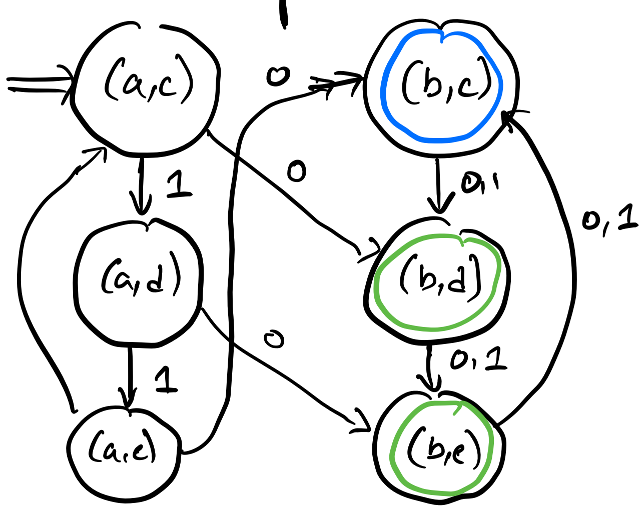
D_2



(*) DFA for $\underline{L(D_1) \cap \overline{L(D_2)}}$

(**) Can you do this with < 6 states?

not yet 0 | seen 0



Theorem. Regular languages are closed under union ($\cup$).
(If A regular, B regular, then $A \cup B$ regular)
($\cup$ is a regular operation.)

Idea: Simulate two DFAs for A, B, and accept if either accepts.

Proof. Let A, B be regular languages.
Let D_A, D_B be DFAs that recognize A and B.

Let $D_A = (Q_1, \Sigma, \delta_1, s_1, F_1)$.
$D_B = (Q_2, \Sigma, \delta_2, s_2, F_2)$.

Build $D_{A \cup B}$ as follows:

Let $D_{A\cup B} = (Q, \Sigma, \delta, q_0, F)$.

$Q = \{(r_1, r_2),\ \underbrace{r_1 \in Q_1,\ r_2 \in Q_2}_{= Q_1 \times Q_2}\}$

Σ = same.

$q_0 = (s_1, s_2)$

$F = \{(r_1, r_2) \mid \underbrace{r_1 \in F_1 \text{ OR } r_2 \in F_2}_{\text{union}}\}$

δ: on every pair $(r_1, r_2) \in Q$, $a \in \Sigma$,
$\delta((r_1, r_2), a) = (\delta_1(r_1, a), \delta_2(r_2, a))$

Claim: end state of $D_{A\cup B}$ on any input w
= (end state of D_A, end state of D_B) on w.

By defn of F, if $D_{A\cup B}$ accepts w, either $D_A \cup D_B$ accepts w. $\square$

(To modify for $\cap$: $F: \{(r_1, r_2) \mid r_1 \in F_1 \text{ AND } r_2 \in F_2\}$.

How prove? A, B regular $\longrightarrow$ $A \circ B$ regular

---

2. Nondeterminism

(Deterministic) Finite Automaton

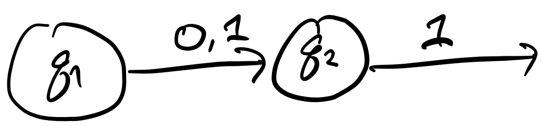
$$\delta : Q \times \Sigma \longrightarrow Q$$

- Break the rule of going to exactly one state.

1. Multiple transitions on one char

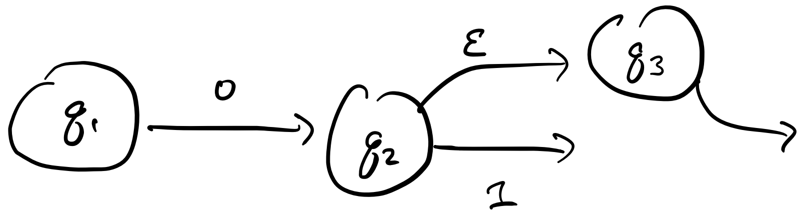
$$\delta(q_1, 0) = \{q_1, q_2, q_3\}$$

2. 0 transitions on a char: branch of computation "dies"



$$\delta(q_2, 0) = \emptyset$$

3. ε-transition is "free branch"



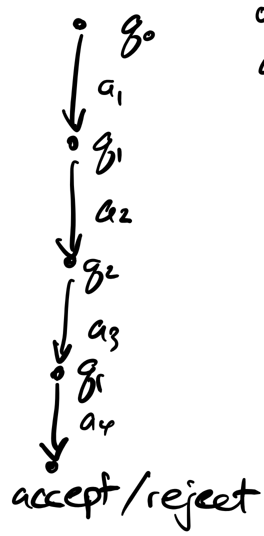
$$\delta(q_2, \varepsilon) = \{q_3\}$$

on a state with an ε-transition,
we split into two computational branches:
One takes the ε-edge and one stays put.

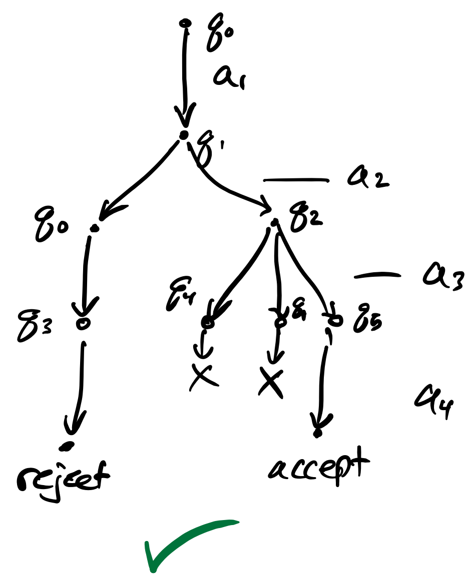
4. Resolving many branches? Accept if <u>any</u> branch of computation is in an accept state at the end of the input string.

Deterministic comp. (DFA)

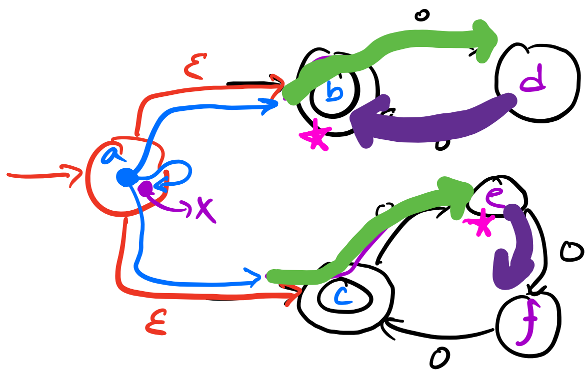
on $a_1 a_2 a_3 a_4$



NFA



Example NFA state diagram. $\Sigma = \{0\}$
$L = \{ w \mid |w| \text{ is divisible by } 2 \text{ or } 3 \}$.



example:

(ε)0000

read in	states occupied
	a
ε	{a, b, c}
0	{d, e}
0	{b, f}
0	{d, c}
0	{b, e}

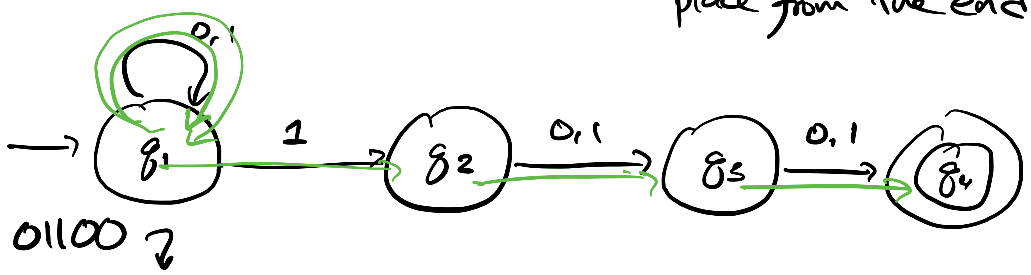
end of evaluation: in {b, e}

at least 1 accept state ⟹ accept.

Example 2.

$\Sigma = \{0, 1\}$

L: {w | w has a 1 in the third place from the end.}



01100 ?

read in	live states
	$\{q_1\}$
0	$\{q_1\}$
1	$\{q_1, q_2\}$
1	$\{q_1, q_2, q_3\}$
0	$\{q_1, q_3, q_4\}$
0	$\{q_1, q_4\}$ x

✓

Def. The Power Set $\mathcal{P}(S)$ is the set of all subsets of S.

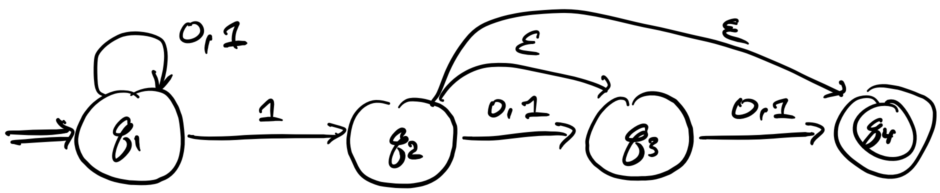
$\{a, b, c\}$. $\mathcal{P}(\{a,b,c\}) = \{\emptyset, \{a\}, \{b\}, \{c\}, \{a,b\}, \{a,c\}, \{b,c\}, \{a,b,c\}\}$

Def. An NFA is a 5-tuple

$$(Q, \Sigma, q_0, F, \delta)$$

set of states · alphabet · start state · accept states · transition function

$$\delta : \underbrace{Q}_{\text{a state}} \times \underbrace{(\Sigma \cup \{\varepsilon\})}_{\text{character (or } \varepsilon)} \longrightarrow \underbrace{\mathcal{P}(Q)}_{\text{a subset of states.}}$$



$N = (Q, \Sigma, q_0, F, \delta)$

$Q = \{q_1, q_2, q_3, q_4\}$ $\quad \Sigma = \{0,1\}$ $\quad q_0 = q_1$ $\quad F = \{q_4\}$

δ:

	0	1	ε
q_1	$\{q_1\}$	$\{q_1, q_2\}$	$\emptyset$
q_2	$\{q_3\}$	$\{q_3\}$	$\{q_2, q_3, q_4\}$
q_3	$\{q_4\}$	$\{q_4\}$	$\emptyset$
q_4	$\emptyset$	$\emptyset$	$\emptyset$

Prop. Any language recognized by a DFA is recognized by an NFA. (✓)

Prop. Any language recognized by an NFA is recognized by a DFA.
[take as true, will prove next time.]

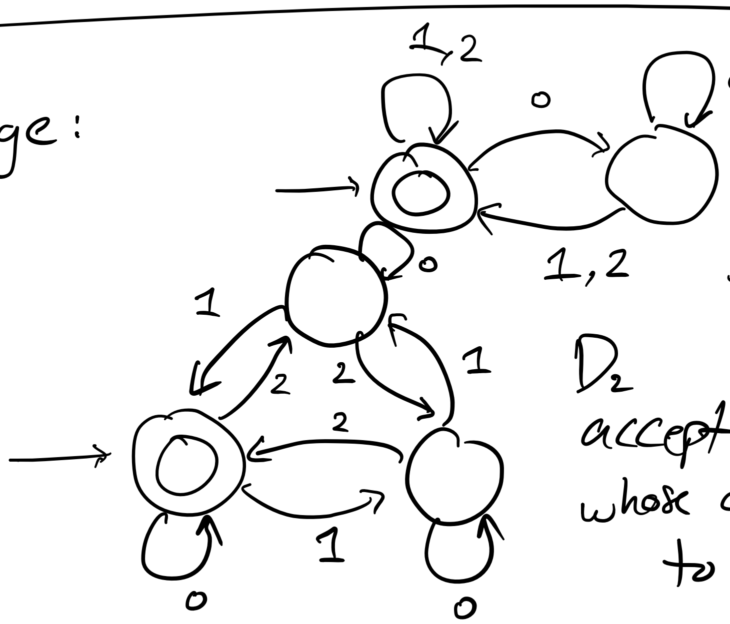
=

True statements:

- NFAs, DFAs both recognize reg. languages.
- Regular languages are closed under the operations $\overline{}$, $\cup$, $\cap$, $\circ$, $*$.

---

challenge:



D_1 accepts strings ending in 1, 2

D_2 accept strings whose characters sum to 0 mod 3.

1: Use NFA edges to build $L(D_1) \cup L(D_2)$

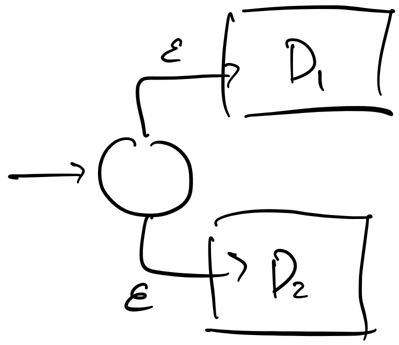
(**) Build an NFA for $L(D_1) \circ L(D_2)$

(***) Build an NFA for $L(D_1)^*$.

$$A \circ B = \{ wx \mid w \in A, x \in B \}$$

$$A^* = \{ w_1 w_2 \cdots w_k \mid \text{all } w_i\text{'s in } A,\ k \geq 0 \}.$$

(1)



(**)