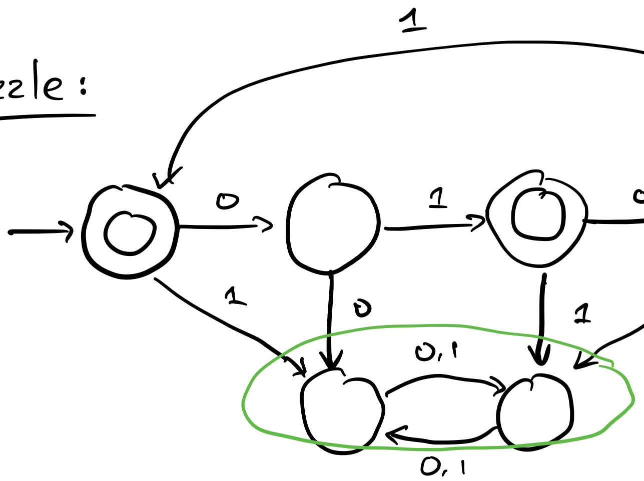
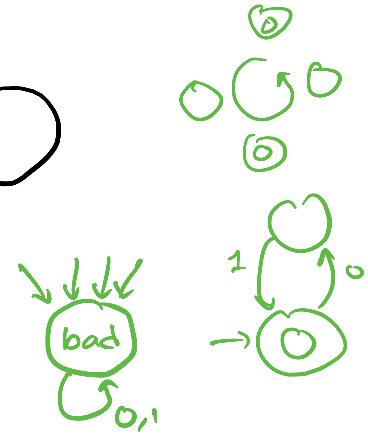
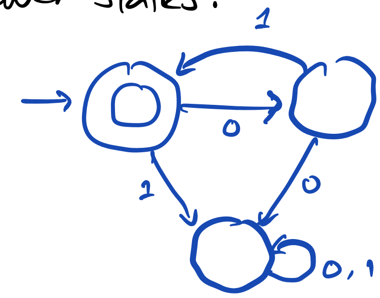
Puzzle:





- can you rewrite w/ fewer states?
- ≤ 3 states?

ε
01
0101
010101
⋮



Today:

1. Regular languages are closed under complement ($^{-}$) and union ($\cup$)
2. NFAs
3. NFA-recognizable languages are closed under union ($\cup$), concatenation ($\circ$), and star ($*$)

<u>Languages</u> are sets of <u>strings</u>, which are finite sequences of charcters from finite alphabets.

DFAs take input strings from a fixed alphabet, follow ($\to$) one transition per character, and accept/reject.

DFAs can be written as 5-tuple:

$$(\underbrace{Q}_{\text{states}}, \underbrace{\Sigma}_{\text{alphabet}}, \underbrace{\delta}_{\text{transition function}}, \underbrace{q_0}_{\text{start state}}, \underbrace{F}_{\text{accept state(s)}})$$

Given a DFA D, $L(D)$ denotes the set of strings D accepts, otherwise known as the language of/recognized by D

Regular = rec. by DFA

non-regular?

1. Regular Operations

Q: If A is a regular language, is the complement $\overline{A}$ regular?

(some DFA D_A)

with respect to all strings over Σ

Prop. If A is regular, $\overline{A}$ is regular.

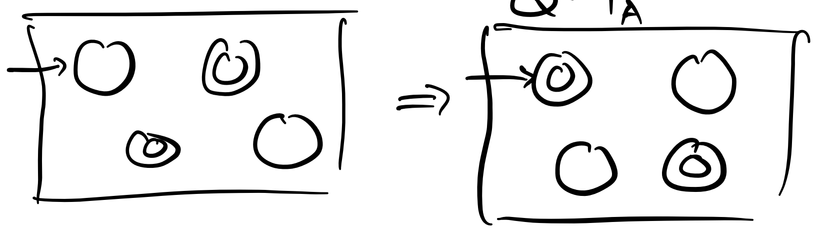
Proof. Let D_A be a DFA that recognizes A.

$\hookrightarrow = (Q, \Sigma, \delta, q_0, F_A)$

Define $D_{\bar{A}}$ by turning accept states into reject states and vice versa. $F_{\bar{A}} = Q \setminus F_A$

$Q - F_A$

$F_{\bar{A}}$ = $D_{\bar{A}}$'s accept states



Equivalent statement: The regular languages are closed under complement.

A set is closed under an operation if applying that operation to set elements gives something in the set.

Ex. $\mathbb{Z}$ closed under $+, -,$ but not under $\div$ ($1/2$)

$+(2,3) \approx 2+3$

Regular operations: operations under which the regular languages are closed.
(* assume A, B over same Σ)

- ✓ Complement: $\bar{A} = \{x \mid x \notin A\}$
- Union: $A \cup B = \{x \mid x \in A \text{ OR } x \in B\}$
- Intersection: $A \cap B = \{x \mid x \in A \text{ AND } x \in B\}$
  (difference, XOR, etc.)
- Concatenation: $A \circ B$, or AB
  $= \{xy \mid x \in A, y \in B\}$
- (Kleene) Star:
  $A^* = \{x_1 x_2 x_3 \cdots x_k \mid \text{all } x_i\text{'s} \in A,\ k \geq 0\}$

$$\{0,1\}^* = \{\varepsilon, 0, 1, 00, 01, 10, 11, \underbrace{000, \dots\}}_{\text{infinite}}$$

$$\{\}^* = \emptyset^* = \{\varepsilon\}$$

$$\{\varepsilon\} = \{\varepsilon\}$$

Know: A, B regular

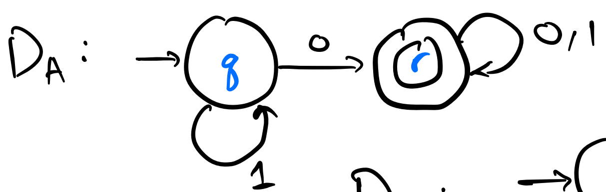
Will follow: $(A \cap B)^* B$

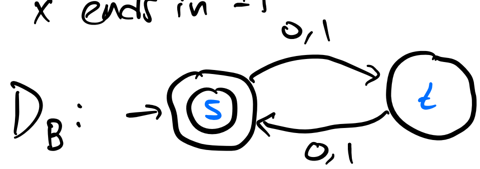
Proposition. If A, B regular languages, $A \cup B$ is regular.

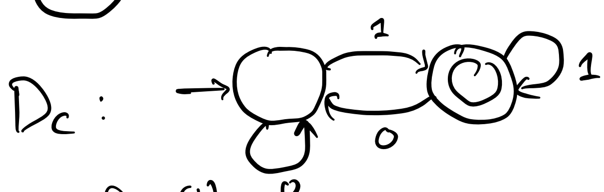
Puzzle. $A = \{x \in \{0,1\}^* \mid x \text{ has at least one } 0\}$

$B = \{x \in \{0,1\}^* \mid |x| \text{ is even}\}$

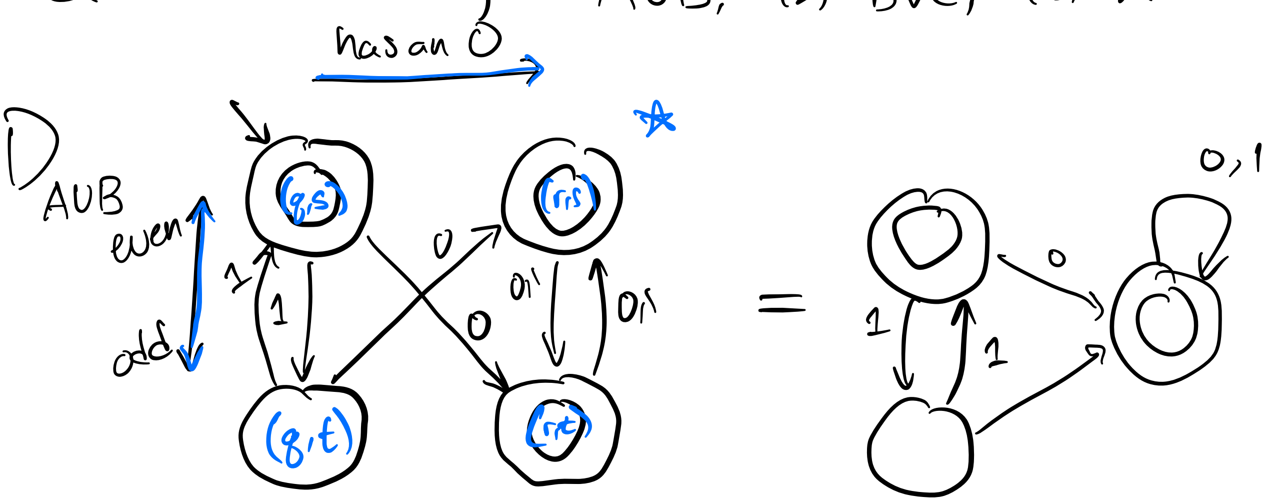
$C = \{x \in \{0,1\}^* \mid x \text{ ends in } 1\}$



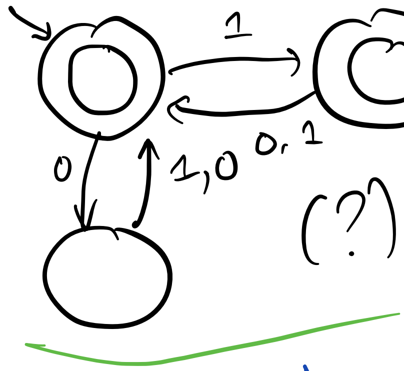


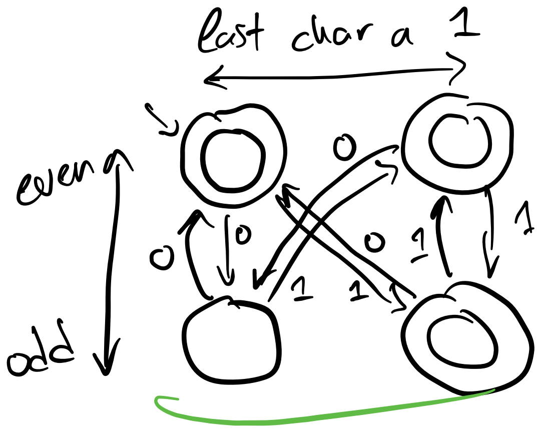


Q: Find a DFA for (1) $A \cup B$, (2) $B \cup C$, (3)* $A \cup C$.



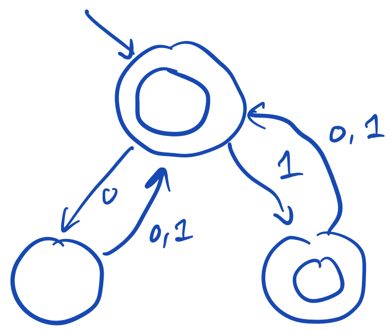
D_{BUC}





||

ε — ✗
0 — ✗
1 — ✓
00 — ✓
01 — ✓
10 — ✓
11 — ✓
000 — ✗



Back at 2:22

Proof. If A, B regular languages, then $A \cup B$ regular.

Let A, B be regular languages w/ DFAs

$$D_A = (Q_1, \Sigma, \delta_1, q_1, F_1)$$

$$D_B = (Q_2, \Sigma, \delta_2, q_2, F_2)$$

We'll build a new DFA,

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

Main idea: One state for each pair in $Q_1 \times Q_2$.

⊛ $Q = Q_1 \times Q_2 = \{(r_1, r_2) \mid r_1 \in Q_1, r_2 \in Q_2\}$

Σ same

⊛ $q_0 = (q_1, q_2)$

for $\cap$: change to AND

⊛ $F = \{ (r_1, r_2) \mid r_1 \in F_1, \cancel{\text{or}}\ r_2 \in F_2 \}$

For $(r_1, r_2) \in Q$, $a \in \Sigma$: $\delta((r_1, r_2), a) = (\delta_1(r_1, a), \delta_2(r_2, a))$.

Claim: If $w \in A \cup B$, then $D_{A \cup B}$ accepts w.

Say $w \in A$ w.l.o.g.

then, the first coordinate of the sequence of states computed when $D_{A \cup B}$ runs on w is the same as the sequence of states when D_A runs on w.

Since D_A accepts w, $D_{A \cup B}$ ends on pair whose first element is an accept state for D_A.

Claim: If $w \notin A \cup B$, $D_{A \cup B}$ rejects w.

- $w \notin A$, so running $D_{A \cup B}$ on w ends at some state (r_1, r_2) with $r_1 \notin F_1$.
- $w \notin B$, ——— " ——— (r_1, r_2) with $r_2 \notin F_2$.

so $D_{A \cup B}$ on w ends in state (r_1, r_2) with $r_1 \notin F_1$, $r_2 \notin F_2$,

$(r_1, r_2) \notin F$. ∎

Corollary for $\cap$. By substituting

$F = \{ (r_1, r_2) \mid r_1 \in F_1\ \cancel{\text{or}}\ \text{AND}\ r_2 \in F_2 \}$.

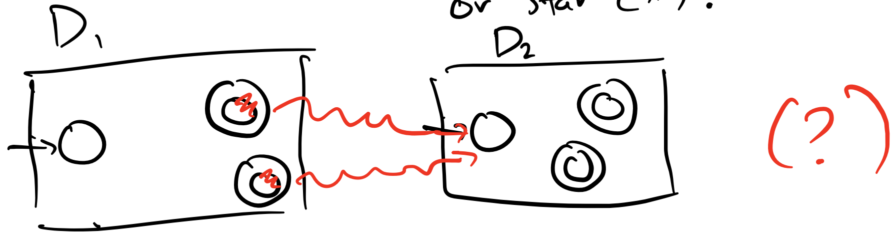
Now, we accept if and only if D_A, D_B both accept, equivalently, $w \in A \cap B$.

Conclusion: The regular languages are closed under $\cup$, $\cap$.

——— Break until 3:10 ———

Closure under concatenation $(\circ)$?

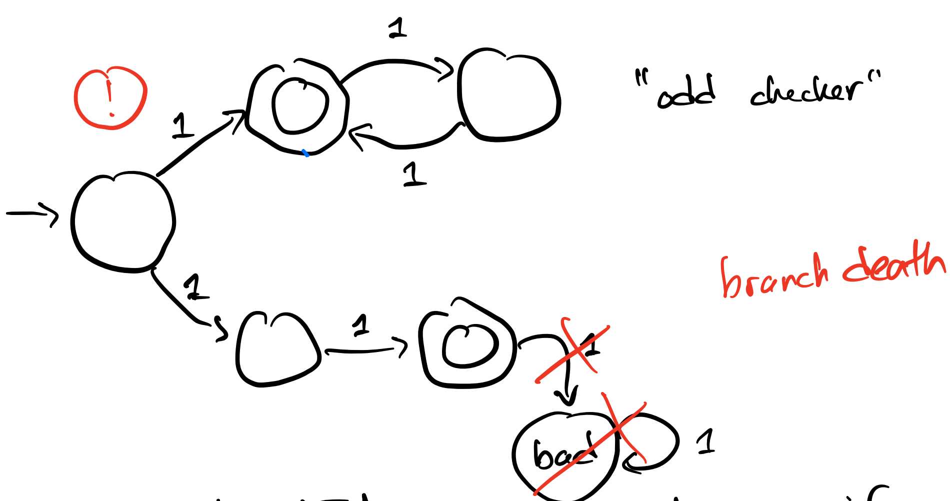
or star $(*)$?



Nondeterminism (NFAs.)

think: "making lucky guesses"
"try multiple options at once, accept if anything works."

$C = \{ x \mid x \in \{1\}^*, \ |x| \text{ is odd OR } |x| = 2 \}$



NFA acceptance: An NFA accepts a string w if there exists <u>some</u> valid sequence of transitions that ends in an accept state.

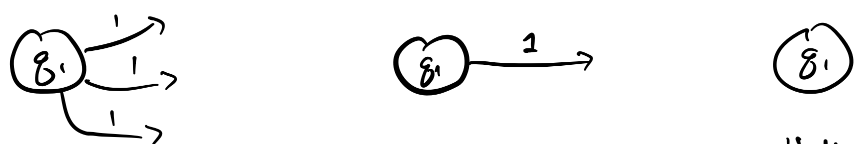
Formally: we'll change $\delta : \underset{(q_1,\,0)}{Q \times \Sigma} \to \underset{q_3}{Q}$ $\quad \{q_1, q_3\}$

to $\delta : Q \times \Sigma \to \mathcal{P}(Q)$

$\mathcal{P}(Q)$ is the "power set" of Q: the set of all subsets of Q. $\mathcal{P}(Q) = \{ R \mid R \subseteq Q \}$

\mathcal{P}

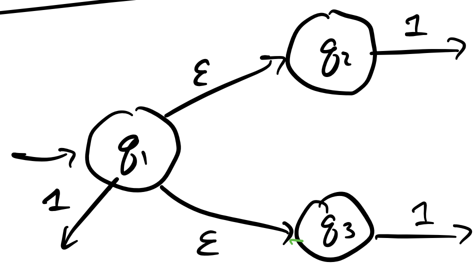
$\mathcal{P}(\{a, b\}) = \{ \{a\}, \{b\}, \{a, b\}, \emptyset \}$



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

"the branch dies"

Free transitions (ε-transitions)



Branches "guess"

q_1, q_2, and q_3 before continuing

or "split"

Def. (NFA). An NFA is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where

Q is a finite set of states,
Σ is an alphabet
q_0 is the start state,
F is the set of accept states

same as DFA

101
1εε01
10ε1ε

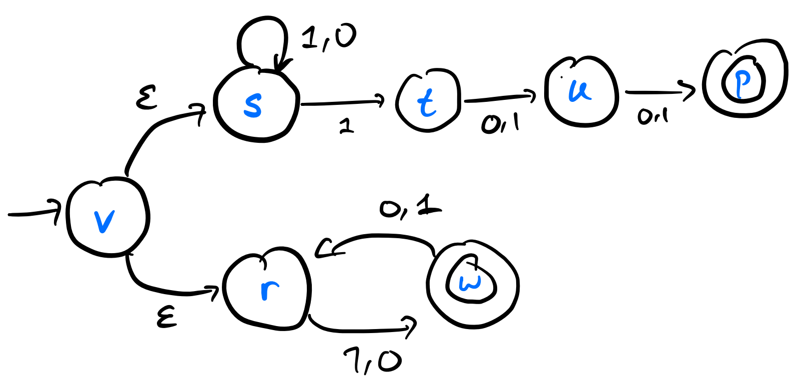
$$\delta : \underbrace{Q \times (\Sigma \cup \{\varepsilon\})}_{\text{read: a state, and 1 character OR } \varepsilon} \to \mathcal{P}(Q)$$

read: a state, and
1 character OR ε
go to a set of states.

Def. (NFA acceptance). An NFA N accepts a string $w = w_0 w_1 \cdots w_{n-1}$, with each $w_i \in (\Sigma \cup \{\varepsilon\})$, if there is a sequence of states $r_0, r_1, r_2, \ldots r_n$ such that

$r_0 = q_0$,
$r_{i+1} \in \delta(w_i; r_i)$ for $0 \le i \le n$
and $r_n \in F$.

N:



$\delta(p, 1) = \emptyset$
$\delta(p, 0) = \emptyset$

$\delta(s, 1) = \{s, t\}$

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

$Q = \{v, s, t, u, p, r, w\}$

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

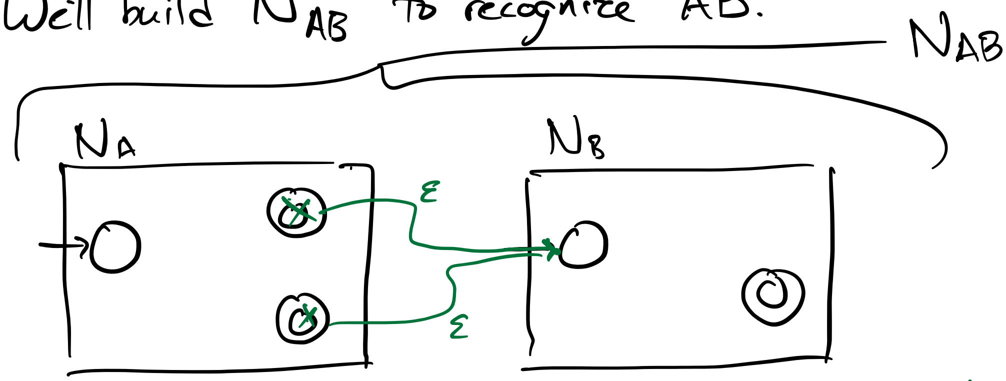
$F = \{w, p\}$

δ	v	s	t	u	p	r	w
0	$\emptyset$	$\{s\}$	$\{u\}$	$\{p\}$	$\emptyset$	$\{w\}$	$\{r\}$
1	$\emptyset$	$\{s,t\}$	$\{u\}$	$\{p\}$	$\emptyset$	$\{u\}$	$\{r\}$
ε	$\{s,r\}$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$

Theorem: If A, B are NFA-recognizable, $AB = \{xy \mid x \in A, y \in B\}$ is NFA-recognizable.

Proof: Let $A = L(N_A)$, $B = L(N_B)$.

We'll build N_{AB} to recognize AB.



(1) Turn N_A's accept states into regular states.

(2) Add ε-transitions from N_A's accept states to N_B's start state.

Claim: If $x \in A$, $y \in B$, N_{AB} accepts xy.

Proof: Some branch reaches N_A's accept state on x; then, some branch takes the ε-transition and reaches N_B's accept state on y.

Claim: If N_{AB} accepts some string w, $w \in AB$.

By definition, $w = w_0 w_1 \cdots w_i w_{i+1} \cdots w_n$, where

$w_0 \cdots w_i$ reaches an accept state in N_A

and $w_{i+1} \cdots w_n$ reaches an accept state in N_B

By def, $w_0 \cdots w_i \in A$, $w_{i+1} \cdots w_n \in B$, so $w \in AB$. $\blacksquare$

Summary:

1. Reg. languages closed under complement ($\overline{}$) and union ($\cup$)
2. NFAs: "lucky guessers" or "branching programs"
3. NFA-recognizable languages closed under concat ($\circ$). Next time: union ($\cup$) and star ($*$).

Reminders:

- My O hours @ 5:30 6:30, Zoom
- HW 1 Due Mon @ 11:59 PM (Tues)

For $(r_1, r_2) \in Q$ and $a \in \Sigma$: $\delta(\underbrace{(r_1, r_2), a}_{Q \times \Sigma}) = (\underline{\delta_1(r_1, a)}, \underline{\delta_2(r_2, a)})$