Announcements

- HW1 - due last night; if using late hours: no later than tonight
  - solutions posted soon after (we'll email)
- checkup 3 - first attempt by tomorrow night
- checkup 2 - last attempt by Thur night
- quiz 1: in class next Tue, 30 minute
  - closed book, no notes, no materials
  - have your ID in case we check
  - Come on time
  - what's on it? see previous lectures...

Last time:

L is regular $\Leftrightarrow$ $\exists$ DFA recognizing L $\Leftrightarrow$ $\exists$ NFA recognizing L

- $\Leftrightarrow$: by definition
- $\Leftrightarrow$: $\Rightarrow$ immediate; $\Leftarrow$ subset construction

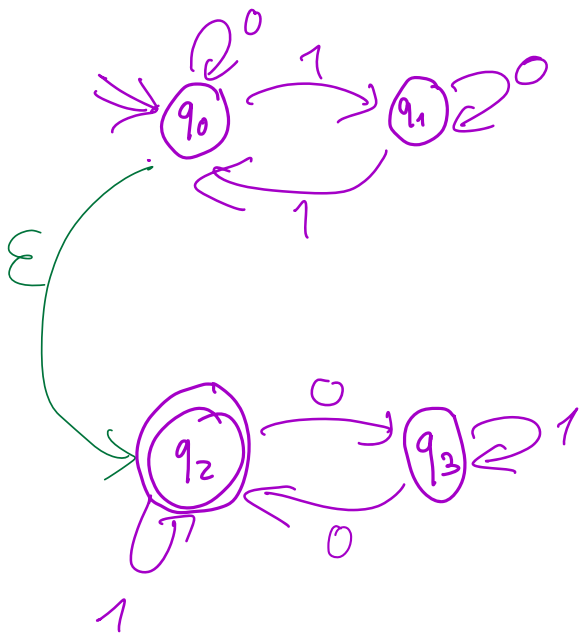
Subset Construction: If NFA states are Q_N

DFA states are $Q_D = P(Q_N)$

(can draw all states, then add transitions, like in checkup 3, or can start from start state and keep adding transitions till no new state is added, thus avoiding unreachable state)

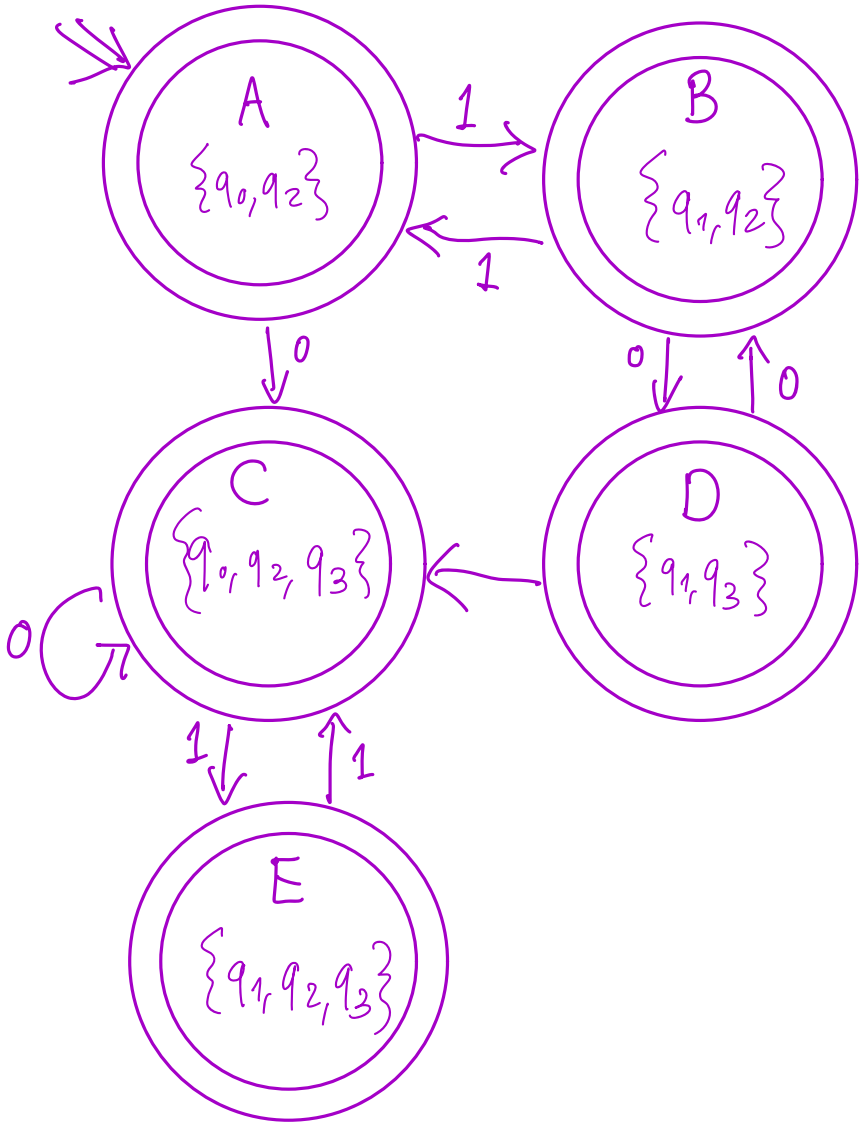
Example: $L = \{ w \in \{0,1\}^* \mid w = x \cdot y$ x has even # of 1s, y has even # of 0s$\}$

NFA:
↓
we saw last time



DFA:
↓
using the subset construction

(in class we showed how we get states A, B, C - continuing in the same way till no new state added yields this DFA)



Note: subset construction blows # states exponentially ($|Q_N| = n \Rightarrow |D_N| = 2^n$).

Maybe there's a better construction of NFA ⇒ DFA?

→ Not in general!

For some languages, exponential gap unavoidable.

$L_n = \{w /$ n-th to last char. in w is 1 $\}$:

minimal DFA has 2^n states → see MN lecture notes

But ∃ NFA with $n+1$ states → exercise (see last lecture for L3)

Today:

(Some more) closure properties for regular languages

(Previously we saw closure under complement, union, intersection)

Def: Let $L_1, L_2, L \subseteq \Sigma^*$ be languages. Define:

- **Concatenation:** $L_1 \circ L_2 = \{xy \mid x \in L_1, y \in L_2\}$ (xy: $x \cdot y$)

  $\neq L_2 \circ L_1$ in general

Ex: $L_1 = \{\varepsilon, 001, 1\}$ $L_2 = \{11, 1\}$

$L_1 \circ L_2 = \{11, 1, 00111, 0011, 111\}$

both $\varepsilon \circ 11$, $1 \circ 1$

- **power (L^k):** (for $k \geq 0$)

$$\begin{cases} L^0 = \{\varepsilon\} \\ L^{k+1} = L^k \circ L \end{cases}$$

That is, $L^k = \underbrace{L \circ \ldots \circ L}_{k \text{ times}}$ i.e all strings you can get by choosing k strings from L and concatenating them to each other.

Eg: $L_2 = \{11, 1\}$

$L_2^0 = \{\varepsilon\}$ $L_2^1 = \{11, 1\}$ $L_2^2 = \{1111, 111, 11\}$

$L_2^3 =$ exercise.

- **Kleene Star:** $L^* = \bigcup_{k=0}^{\infty} L^k = \{\varepsilon\} \cup L \cup L^2 \cup \ldots$

$$= \{ w \in \Sigma^* \mid \exists\, k \geq 0,\ w_1, \ldots, w_k \in L,\ w = w_1 \circ \ldots \circ w_k \} \cup \{\varepsilon\}$$

← already included here when $k=0$ but adding it for clarity.

= set of all strings that can be decomposed into a concatenation of 0 or more strings from L

Theorem: The class of regular languages is closed under concatenation, power, star.

i.e:
- L_1, L_2 regular $\Rightarrow$ $L_1 \circ L_2$ regular
- L regular $\Rightarrow$ $\forall k \geq 0$ L^k is regular
- L regular $\Rightarrow$ L^* is regular.

Proof:

- Concatenation: Let L_1, L_2 be regular, so there are

DFAs $D_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ recognizing L_1 $(L(D_1)=L_1)$

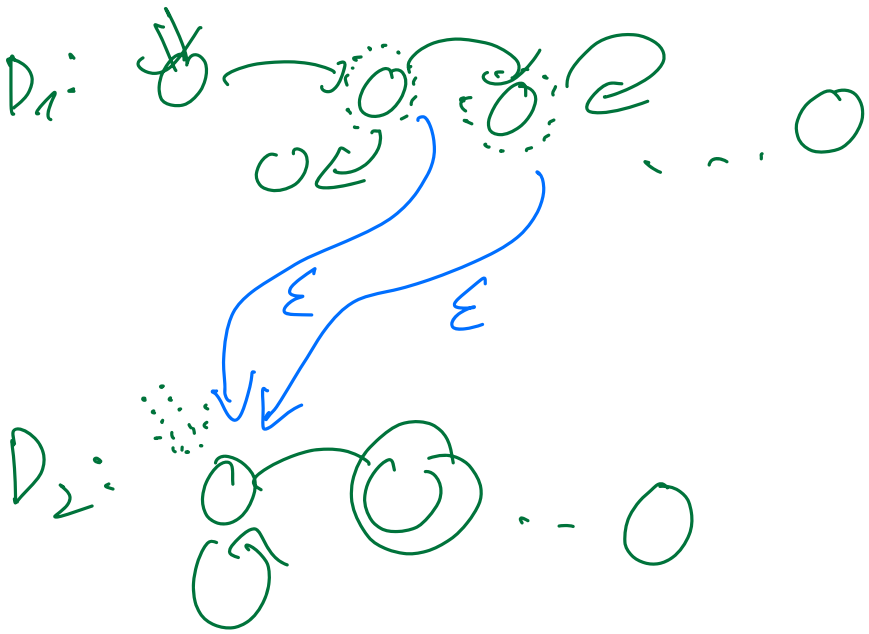
$D_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ recognizing L_2 $(L(D_2)=L_2)$

We will construct an NFA recognizing $L_1 \circ L_2$:

Idea: add ε-transitions from accept states of D_1 to start state of D_2

start state is q_1 (start state of D_1)

accept states are F_2 (accept states of D_2)



Now if $w = \underbrace{x}_{\in L_1} \cdot \underbrace{y}_{\in L_2}$, we can compute on the D_1 part until we guess that it's time to split, then use ε-transition and compute on y part. (and any w accepted must have taken ε-transition...)

Formally:

We define $N = (\underbrace{Q_1 \cup Q_2}_{Q}, \Sigma, \delta, q_1, F_2)$

where $\delta : Q \times \Sigma_\varepsilon \to \mathcal{P}(Q)$ is defined as follows:

D_1 transitions: $\{ \forall q \in Q_1, \forall a \in \Sigma, \quad \delta(q,a) = \{\delta_1(q,a)\}$

ε-transitions we added $\{ \forall q \in F_1, \quad \delta(q,\varepsilon) = \{q_2\}$

D_2 transitions $\{ \forall q \in Q_2, \forall a \in \Sigma, \quad \delta(q,a) = \{\delta_2(q,a)\}$

all other transitions: δ returns $\emptyset$

N accepts $w \iff w = x \cdot y$ where D_1 accepts x
D_2 accepts y

$\iff w = xy, \; x \in L_1, \; y \in L_2 \iff w \in L_1 \circ L_2$

✓

- power: Let L be regular, $k \geq 0$

want to show L^k is also regular.

$$\underbrace{L \circ L \circ L \ldots \circ L}_{k \text{ times}}$$

proof by induction on k:

$k=0$: $L^0 = \{\varepsilon\}$ is regular.

(eg  is an NFA for ε)

suppose true for k, prove for $k+1$:

$$L^{k+1} = \underbrace{\overbrace{L^k}^{L_1}}_{\text{regular by induction hypothesis}} \circ \underbrace{\overbrace{L}^{L_2}}_{\text{regular}}$$

and we just proved above that concat. of two regular lang. is regular $\Rightarrow L^{k+1}$ also regular.

✓.

- **Kleene star:** Let L be regular.

want to prove that L^* is regular.

(failed) **Attempt:** $L^* = \bigcup_{k=0}^{\infty} L^k$.

- We proved that the class of regular lang is closed under power (L^k is regular $\forall k \geq 0$)
- previously we proved the class of reg lang is closed under union

$\not\Rightarrow$ $L^* = \bigcup_{k=0}^{\infty} L^k$ is regular.

Doesn't work!

We proved that union of 2 regular languages is regular.

Can use induction to prove that the union of any finite number of languages is regular.

But this does not imply the same for infinite union.

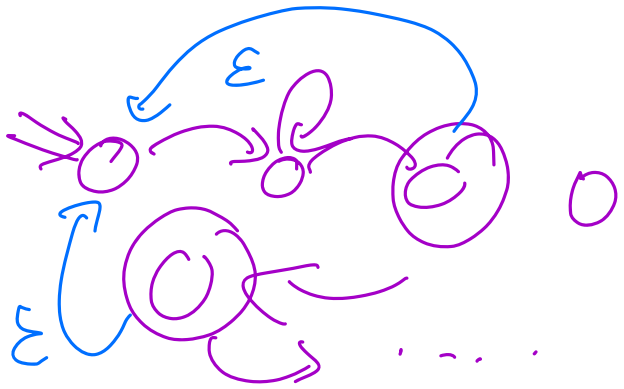
In fact, the class of regular languages is not closed under infinite union. Can you show this?

Find infinitely many regular languages s.t. their union non-regular?

What about infinite intersection? ← (good question by a student after class)

(almost successful):

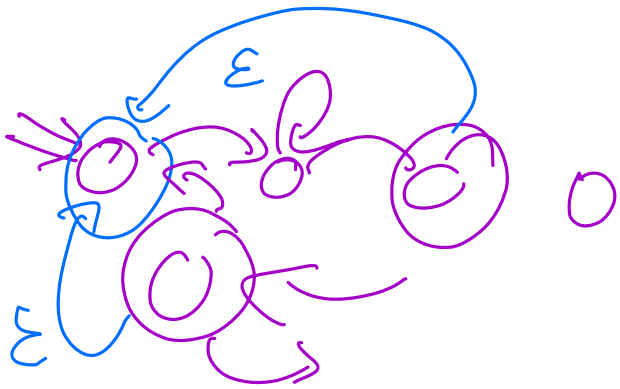
Attempt: Given a DFA for L, make an NFA for L^* by adding ε-transitions from all accepting states to the starting state.



problem: does not necessarily accept ε (but $\varepsilon \in L^*$ always!)

actually good NFA for $\bigcup_{k=1}^{\infty} L^k$

can try to fix by making start state accepting:

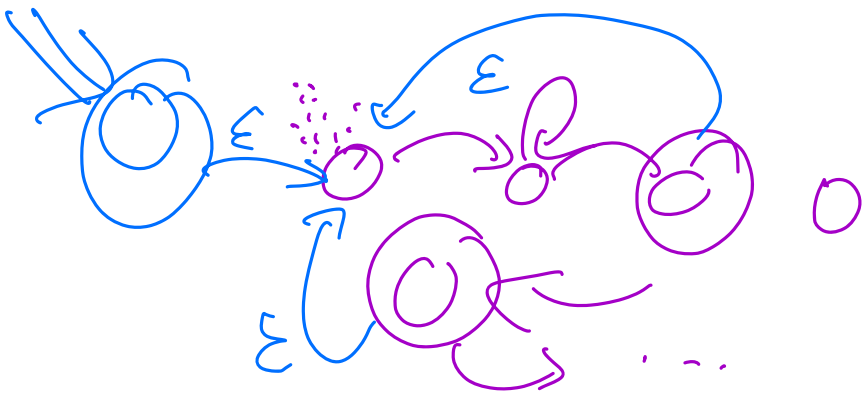


still does not work:
accepts all strings in L^* but may accept too many strings (can you show counter example?)

Construction (that works!):

Add ε-transitions from accept states to the (old) starting state.

Additionally, add a new start state with ε-transitions to old start state, and make new start state accepting.



Doing this formally left as exercise (see solution on next page).

For this N we can prove that $L(N) = L^*$ (i.e. w accepted $\Leftrightarrow$ w is a concat. of 0 or more strings in L).

Formal description of above construction:

Let L be regular, and let $D = (Q, \Sigma, \delta, q_0, F)$ be a DFA s.t. $L(D) = L$. Construct NFA

$N = (Q \cup \{q^*\}, \Sigma, \delta', q^*, F \cup \{q^*\})$ where

δ' is defined as follows:

ϵ transition from new start state to old one: $\delta'(q^*, \epsilon) = q_0$

D transitions: $\forall q \in Q, \forall a \in \Sigma: \delta'(q,a) = \{\delta(q,a)\}$

ϵ-transitions from accept states to q_0: $\forall q \in F, \quad \delta'(q, \epsilon) = \{q_0\}$

For all other transitions, δ' returns $\emptyset$.

At this point, we've seen that the class of regular languages is closed under:

union $\cup$, intersection $\cap$, Complement $\overline{}$

Concatenation $\circ$, power, Kleene star *.

preview into what is coming up!
(we'll do this in the next lecture):

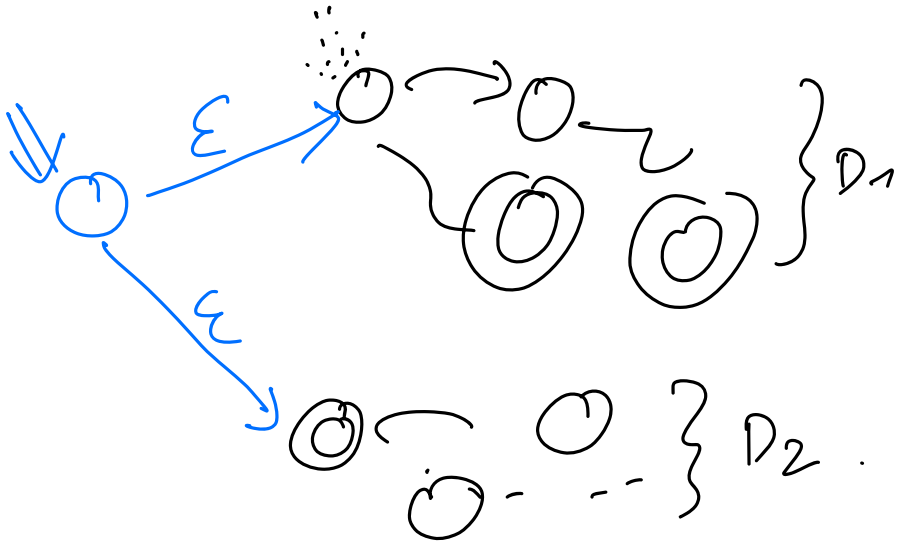
- Another proof for closure under union:

L_1, L_2 regular $\Rightarrow$ $L_1 \cup L_2$ regular

(we saw a proof by De-Morgan $L_1 \cup L_2 = \overline{\overline{L_1} \cap \overline{L_2}}$ and a construction of DFA for $L_1 \cup L_2$ given DFAs for L_1, L_2. Now we'll see a simpler construction of NFA for $L_1 \cup L_2$:)

Suppose we have DFAs D_1, D_2 s.t. $L(D_1) = L_1$
$L(D_2) = L_2$

Can construct an NFA for $\cup$:
add a new start state with ε transition to each of the start states of D_1, D_2.



(Formal description: exercise)

Def: The operations $\cup, \circ, *$ are called the regular operations.

We will see: L is regular $\iff$

L can be written as a finite combination of characters in Σ, $\emptyset$, and regular operations.

→ "regular expressions"