Lecture 6 :

* Handout 1 is out!
* Checkup 2 due tonight
* HW1 next Monday
* checkup 3 soon! Due Wednesday night
* Some challenge problem?

Grady: we'll use AI assisted grading

this is a pilot program
(other profs are trying to)

Our main goals are: more concistent grading

better feedback

Checkups ..... Quizz ⟶ final reassessment

HW ⟶

↑ more details to prepare you better
and for you to see how to improve!
and release grade + feedback before quizz

⚠ This is a pilot: All regrade requests handled by TAs (humans)

TAs independently grade a subset of HW, compare with AI
=> refine, give AI new instructions

It's in agreement with Columbia (privacy)

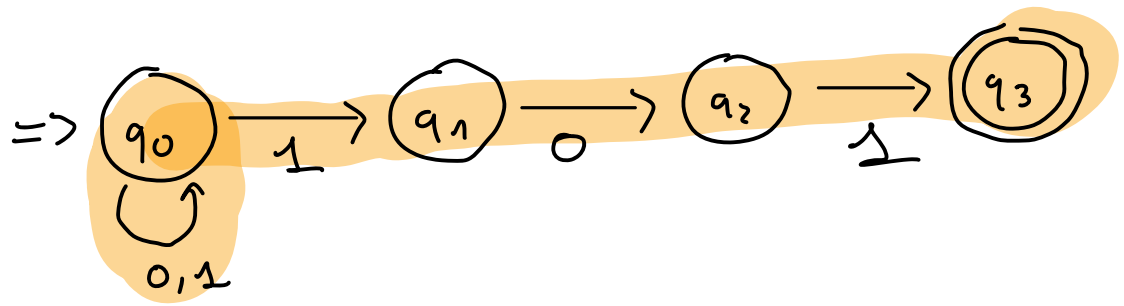
but if you have concerns feel free to talk to Tal and I.

NFAs : like DFAs but:

- for each state + symbol 0 or more transitions
- ε transitions (later)

⚠ A DFA is a special case of an NFA

We looked at this NFA, N:



On an input w: 0 or more computation

On input 101 we have several computations

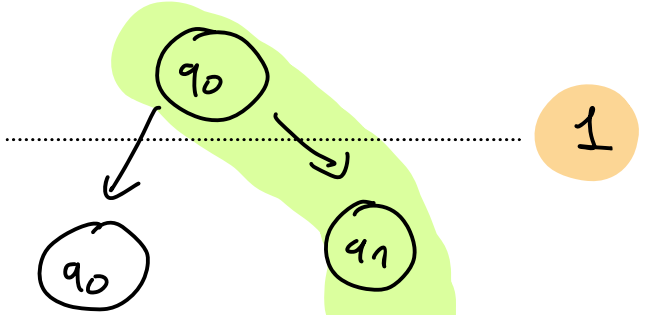
$q_0 \xrightarrow{1} q_0 \xrightarrow{0} q_0 \xrightarrow{1} q_0$

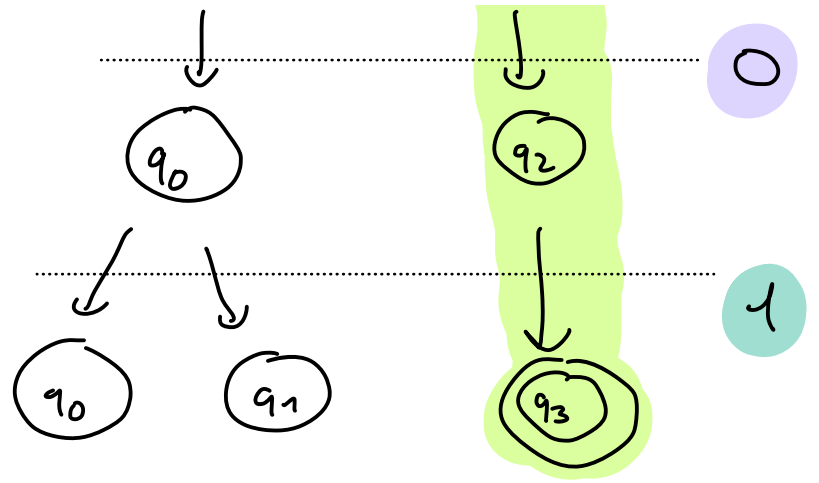
$q_0 \xrightarrow{1} q_1 \xrightarrow{0} q_2 \xrightarrow{1} q_3$

$q_0 \xrightarrow{1} q_0 \xrightarrow{0} q_0 \xrightarrow{1} q_1$

IF at least one computation accepts, the NFA accepts w

computation tree on 101

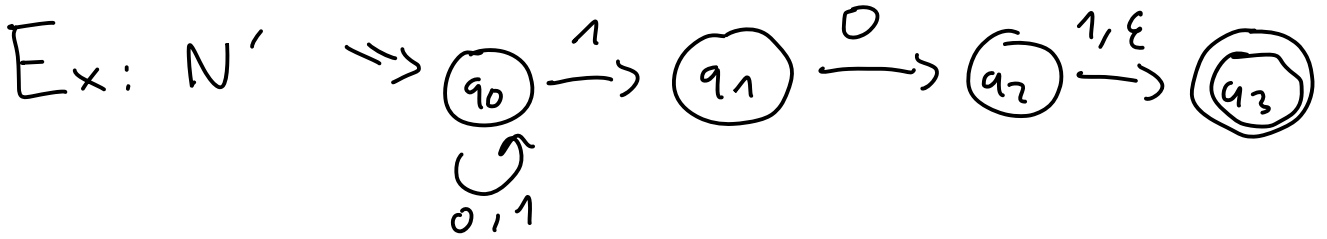




We saw $L(N) = \{w \mid w \text{ ends in } 101\}$

When there's an ε transition, computation can either take it or not without reading a symbol.

Ex: N'



101000

On input 10:

$\to q_0 \xrightarrow{1} q_0 \xrightarrow{0} q_0$

$\to q_0 \xrightarrow{1} q_1 \xrightarrow{0} q_2$

$\to q_0 \xrightarrow{1} q_1 \xrightarrow{0} q_2 \xrightarrow{\varepsilon} q_3$) this computation ends in accept

so 10 is accepted

$L(N') = \{ w \mid w \text{ contains } 10 \}$

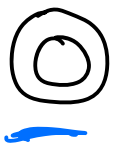
101 $q_0 \xrightarrow{1} q_0 \xrightarrow{0} q_0 \xrightarrow{1} q_0$

$q_0 \xrightarrow{1} q_1 \xrightarrow{0} q_2 \xrightarrow{1} q_3$ ✓

$L(N') = \{ w \mid w \text{ ends in } 10 \text{ OR ends in } 101 \}$

$w = x10$ $q_0 \overset{x}{\rightsquigarrow} q_0 \xrightarrow{1} q_1 \xrightarrow{0} q_2 \xrightarrow{\varepsilon} q_3$

Another example:

N'': =>  , $L(N'') = \{\varepsilon\}$

On any input other than ε, there's no transition to take (the computation aborts)
=> there's no accepting computation

2 views of NFA computation:

* try all possible computation in parallel, accept if at least one succeeds

OR

* whenever there is more than one option, guess and hope for the best (you end in accept state)

Note: this doesn't seem like a realistic model of computation... your computer doesn't do that...

Why define it?

It turns out it's super useful!

Formal definition of an NFA: (difference from DFA highlighted)

For an alphabet Σ, we let $\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$

Def of an 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
* $\delta : Q \times \Sigma_\varepsilon \to \mathcal{P}(Q)$ — this is the power set of Q, the set of all subsets of Q.
* q_0 : the start stat
* $F \subseteq Q$: the set of accept states

$\mathcal{P}(Q) = \{S \mid S \subseteq Q\}$

Def : for an NFA $N = (Q, \Sigma, \delta, q_0, F)$ and $w \in \Sigma^*$, $w = w_1 \cdots w_n$

a computation of N on w is a sequence of states $(r_0, r_1, \cdots, r_m)$ such that $w = 001$, $0 \varepsilon 01 \varepsilon$

- w can be written as $w_1' w_2' \cdots w_m'$ (w with some ε)
  where $w_i' \in \Sigma \cup \{\varepsilon\}$
- $r_0 = q_0$
- $\forall i \in \{1, \cdots m\}$, $r_i \in \delta(r_{i-1}, w_i')$ — because $\delta(\cdot,\cdot)$ outputs a set of state

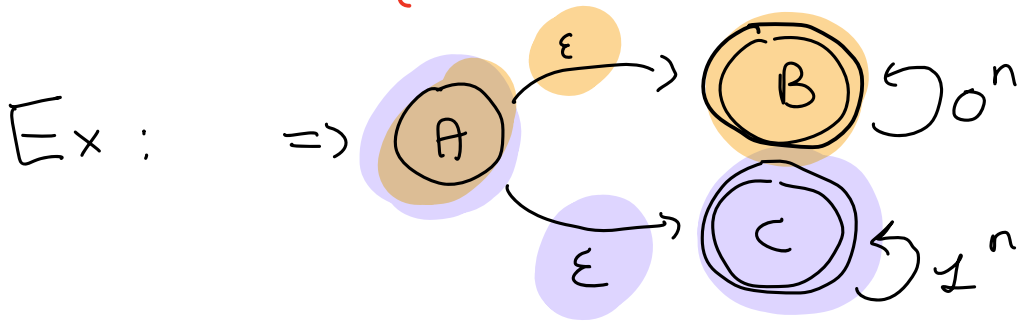
I.e follow transitions and the concatenation of symbols & ε along them gives w

A computation is accepting if $r_m \in F$

N accepts w if and only if there exists an accepting computation of N on w

(If all computations reject, then we say N rejects w)

Def: For an NFA, the language recognized by N is $L(N) = \{w \in \Sigma^* \mid N \text{ accepts } w\}$

Ex: 

N in formal question:

$$N = (\underbrace{\{A, B, C\}}_{Q}, \underbrace{\{0,1\}}_{\Sigma}, \delta, \underbrace{A}_{\text{start}}, \underbrace{\{B, C\}}_{\text{accept states}})$$

δ:

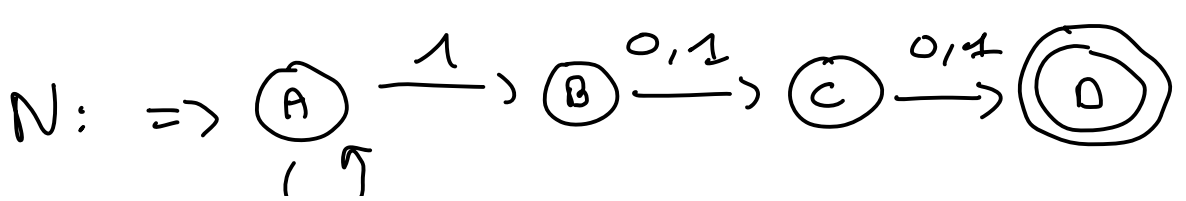
	0	1	ε
A	∅	∅	{B, C}
B	{B}	∅	∅
C	∅	{C}	∅

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

↑ no transition for that symbol!

What's $L(N)$? $L(N) = \underbrace{\{0^n \mid n \geq 0\}}_{\text{any \# of 0's}} \cup \underbrace{\{1^n \mid n \geq 0\}}_{\text{any \# of 1's}}$

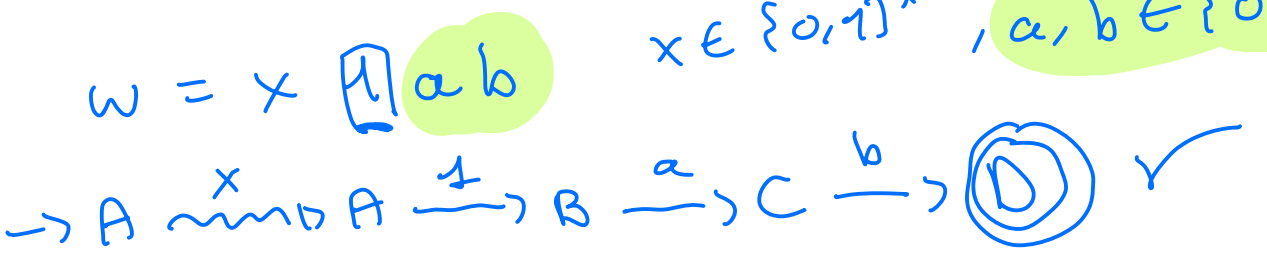
$A \xrightarrow{\varepsilon} B \xrightarrow{0} B \cdots \xrightarrow{0} B$ ✓ $\quad A \xrightarrow{\varepsilon} C \xrightarrow{1} \cdots \xrightarrow{1} C$

N: 

$\overbrace{\quad}^{0,1}$

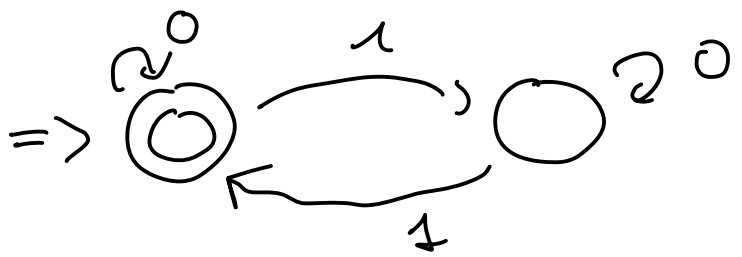
$L(N) = \{w \mid$ the third to last symbol in w is a $1\}$

$w = x\,1\,a\,b \qquad x \in \{0,1\}^*,\ a, b \in \{0,1\}$

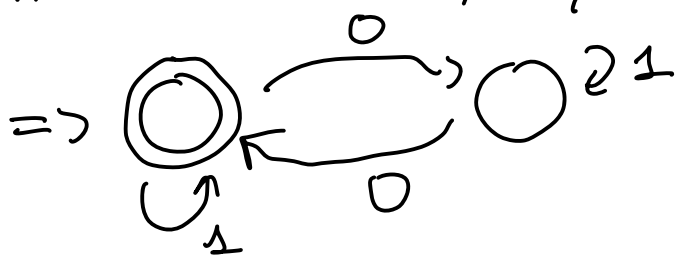


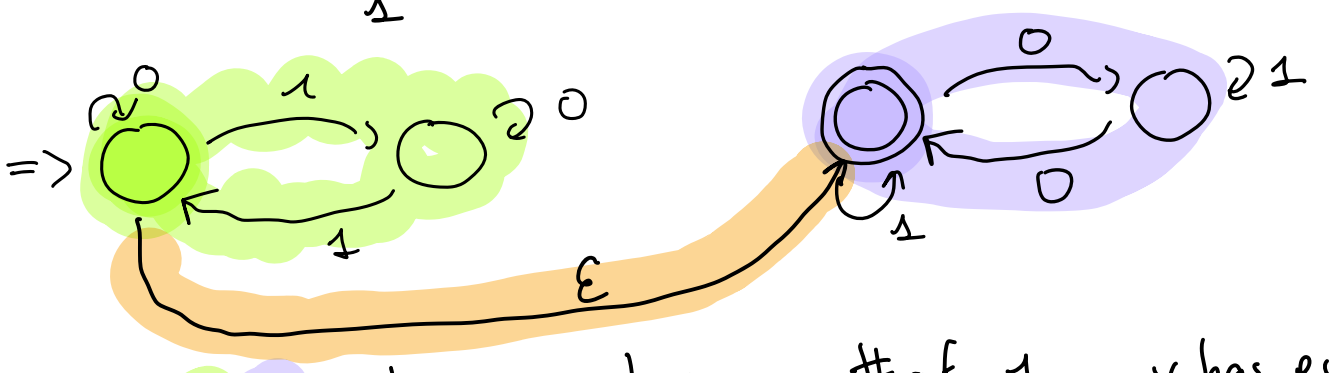
$L = \{w \mid w \in \{0,1\}^*,\ w = xy$ such that x has even # of 1, y has even # of 0$\}$

(1) An NFA for $\{x \mid x$ has an even #of 1s$\}$



(2) An NFA for $\{y \mid y$ has an even #of 0s$\}$





$w = xy$ where x has even #of 1, y has even # of 0s

$w = 1100 = \underbrace{11}_{x} \cdot \underbrace{00}_{y} = \underbrace{1 \cdot 100}_{}$

$= \underbrace{\varepsilon}_{x} \cdot \underbrace{1100}_{y}$ doesn't

$= 11001100$

Thm: L is regular (so there is a DFA recognizing L) if and only if there exists an NFA that recognizes L

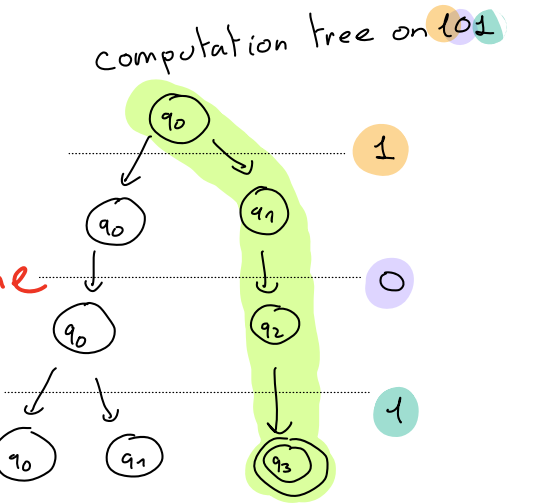
proof: $\Rightarrow$ a DFA is a special case of an NFA

$\Leftarrow$ Let $N = (Q_N, \Sigma, \delta_N, q_N^0, F_N)$ be an NFA.

We build a DFA that recognizes the same language

$D = (Q_D, \Sigma, \delta, q_D^0, F_D)$

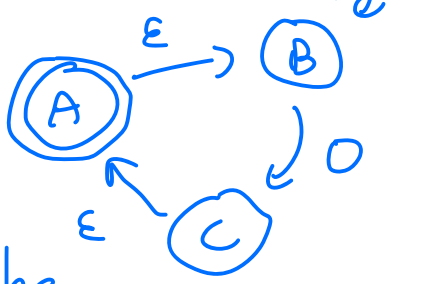
In NFA => we're in multiple state at the same time



Idea: as you process the string we'll keep track of all possible states that N could be in at the same time.

In Q_D we'll have one state for each set of states of Q_N.

* $Q_D = \mathbb{P}(Q_N)$



: states would be

$\emptyset, \{A\}, \{B\}, \ldots \{A,B\}, \ldots \{A,B,C\}$

For a set $S \subseteq Q_N$ we define (8 states)

$E(S) = S \cup \{$all states that are reachable from S by following any # of ε transition$\}$

$E(\{A\}) = \{A\} \cup \{B\} = \{A, B\}$ $\quad E(\{C\}) = \{C\} \cup \{A, B\}$

$E(\{B\}) = \{B\}$

$E(S) = S \cup \{$extra stuff$\}$

* $q_D^0 = E(\{q_N^0\})$ => start state of NFA + anything reachable by ε

* $F_D = \{S \subseteq Q_N \mid S \cap F_N \neq \emptyset\}$ | start is $E(\{A\}) = \{A, B\}$

so S is an accept state of D if it contains an accept state of N

* δ_D: In the NFA, say we're in parallel in $q_1 \cdots q_k$, then the next states after reading "a" would be $\delta_N(q_1, a) \cup \cdots \delta_N(q_k, a)$

then we can follow any # of ε transitions

$$\delta_D(S, a) = \{q' \in Q_N \mid q' \in E(\delta_N(q, a)) \text{ for some } q \in S\}$$

We can show that computation of D on w ends in $\{q \in Q_N \mid \exists$ a computation of N on w that ends in $q\}$

this a set; that's a state of the DFA