CSCI 301 - Lecture 20:

A valiant attempt at regular closure; NFAs

Reminder:

Def. A language A is regular if there is a DFA M such that $L(M) = A$

Regular Operations

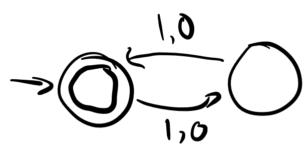
- The **union** of two languages A and B is defined as $A \cup B = \{w : w \in A \text{ or } w \in B\}$.
- The **concatenation** or **product** of two languages A and B is defined as $AB = \{ww' : w \in A \text{ and } w' \in B\}$.
- The **closure** or **star** (or Kleene closure) of a language A is defined as: $A^* = \{u_1 u_2 \ldots u_k : k \geq 0 \text{ and } u_i \in A \text{ for all } i = 1, 2, \ldots, k\}$

Theorem: Regular languages are closed under the regular operations.

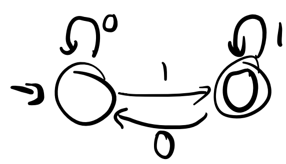
Proof: Nope! But let's do an example for union

Warmup: design a DFA accepting each of the following languages over $\Sigma = \{0, 1\}$:

$L_A = \{w : |w| \text{ is even}\}$



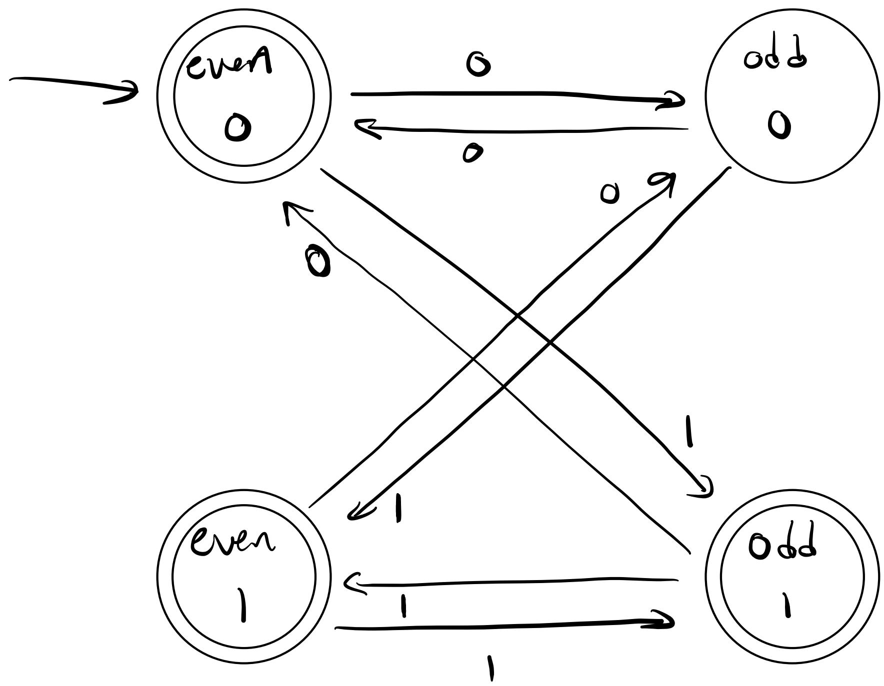
$L_B = \{w : w \text{ ends in } 1\}$



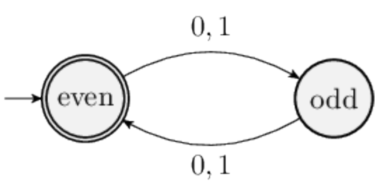
$\boxed{\text{Ex. Pt 0x40}}$

Problem: design a DFA that accepts $L_A \cup L_B$

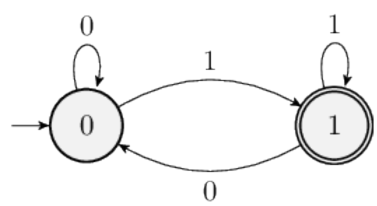
(brainstorm)



M_A :



M_B :

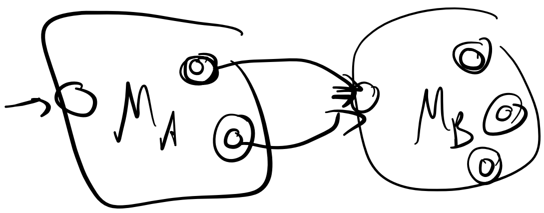


The general procedure is this:

- The set of states is $Q_A \times Q_b$ $\quad Q_A \times Q_B$
- The start state is the state (q_A, q_B) $\quad (q_A, q_B)$
- The accept states are $\{(s_A, s_B) : s_A \in F_A \text{ or } s_B \in F_B\}$
- The transition function is defined by:
  - $\delta_{A \cup B}((s_A, s_B), x) = (\delta_A(s_A, x), \delta_B(s_B, x))$

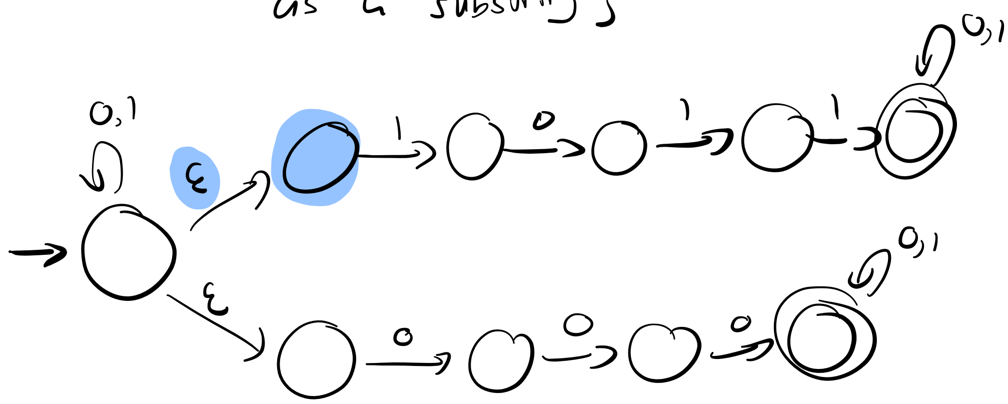
    In other words, the transition function takes you from the pair of states you're at to the pair of states the two individual machines would be in after they each saw the next symbol.

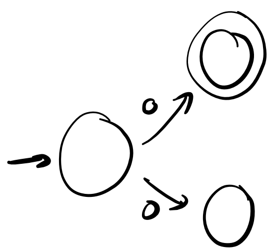
That was hard! Surely concatenation and star are easier...?



Determinism is a drag. Let's break some rules!

$L = \{ w : w$ contains either 1011 or 000 as a substring $\}$





Nondeterministic Finite Automata

DFA definition:

A non(deterministic) finite automaton is a 5-tuple: $(Q, \Sigma, \delta, q, F)$

1. Q is a finite set whose elements are states
2. Σ is an alphabet, whose elements are symbols

$$\Sigma' = \Sigma \cup \{\varepsilon\}$$

3. δ is a function $\delta : Q \times \Sigma \to \cancel{Q}$ $\to P(Q)$ called the transition function
4. q is the start state
5. F is a subset of Q whose members are called accept states.

Medium-Large Question: What languages can NFA's accept that DFA's can't?

Medium-large plot twist: nothing!

A string w is accepted by an NFA if <u>any</u> sequence of allowable state transitions ends in an accept state.