Regular Languages : Recap

- FA $A = (Q, \Sigma, \delta, q_0, F)$

  DFA $\delta : Q \times \Sigma \to Q$

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

- Regular expressions : $\emptyset, \varepsilon, a \ (a \in \Sigma)$

  $E_1 \cup E_2, \ E_1 \circ E_2, \ (E)^*$

- $\underbrace{\text{DFA} \equiv \text{NFA} \equiv \text{Reg. expressions}}_{\text{Regular Languages}}$

- Pumping Lemma : (used to show some languages are <u>not</u> regular)

  $\forall$ regular language L

  $\exists \ p > 0$

  $\forall \ w \in L, \ |w| \geq p \Rightarrow \exists x, y, z \in \Sigma^*$

  $w = x \circ y \circ z$

  and

  (1) $y \neq \varepsilon$

  (2) $|xy| \leq p$

  (3) $\forall i \geq 0, \ x y^i z \in L$.

Decidable properties of DFAs

1. $A_{DFA} = \{ \langle M, w \rangle \mid \text{DFA } M \text{ accepts } w \}$

2. $E_{DFA} = \{ \langle M \rangle \mid \text{DFA } M \text{ has } L(M) = \emptyset \}$

3. $EQ_{DFA} = \{ (M_1, M_2) \mid \text{DFA } M_1 \text{ \& } M_2 \text{ have } L(M_1) = L(M_2) \}$

<u>Claim</u>: A_{DFA}, E_{DFA}, EQ_{DFA} are decidable (in polytime).

<u>Proof</u>: We'll give an (efficient) algo. for each of these decision problems (languages).

3. M_1, M_2 : DFA

$$L(M_1) = L(M_2) \ ?$$

$$\Updownarrow$$

$$\underbrace{L(M_1)}_{A} \,\Delta\, \underbrace{L(M_2)}_{B} = \emptyset$$

$$A \triangle B = (A - B) \cup (B - A)$$

strings x s.t. $x \in A$ & $x \notin B$

$$x = x_1 x_2 \ldots x_n \in \Sigma^*$$

$$A: M_1 \quad \rightarrow q_1 \xrightarrow{x_1} q_2 \xrightarrow{x_2} q_? \rightarrow \ldots \xrightarrow{x_n} q'$$

$$B: M_2 \quad \rightarrow p_1 \xrightarrow{x_1} p_2 \xrightarrow{x_2} p_? \rightarrow \ldots \xrightarrow{x_n} p'$$

accept iff $q' \in F_A$

$p' \notin F_B$

Define a new DFA M for $C = A - B$

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

$$q_0 = \begin{pmatrix} q_0^A \\ p_0^B \end{pmatrix}, \quad \delta\begin{pmatrix} q \\ p \end{pmatrix} = \begin{pmatrix} \delta_A(q) \\ \delta_B(p) \end{pmatrix}$$

$$Q = Q_A \times Q_B$$

$$F = \left\{ \binom{q}{p} \;\middle|\; q \in F_A , p \notin F_B \right\}$$

— Get a DFA for $A - B$
Get a DFA for $B - A$
Need to check if both $A - B = \emptyset$
and $B - A = \emptyset$

Minimization of DFA, NFA, Reg Expr.

DFA A on k states
for L

$\Downarrow$ is in poly time

DFA A_{opt} for L
on the fewest number of states

NFA, Reg Express. Minimization are PSPACE-hard
so unlikely to have poly-time algo.

Turing Machines

TM $\underline{M} = (Q, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej})$

Q : finite set of states Q

Σ : input alphabet (eg, $\Sigma = \{0,1\}$)

Γ : tape alphabet $\Gamma \supseteq \Sigma \cup \{\sqcup\}$

blank

$q_0 \in Q$: init state

$q_{acc} \in Q$: accepting state

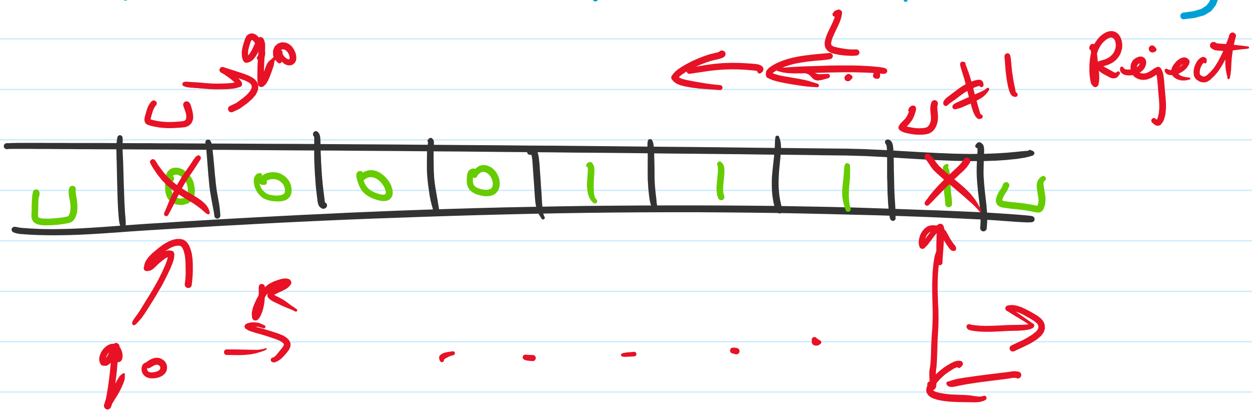
$q_{rej} \in Q$: rejecting state

DFA $\delta : Q \times \Sigma \to Q$

$\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$

$$L(M) = \{ w \mid w \in \Sigma^* \text{ s.t. } q_0 \xrightarrow{w} q_{acc} \}$$

Example: $L = \{0^n 1^n \mid n \geq 0\}$



$$\delta(q_0, \sqcup) = q_{acc}$$