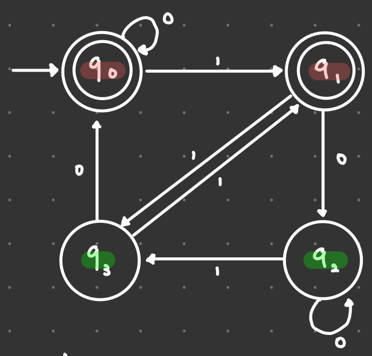
Pre-lecture Video

Deterministic Finite State Automata (DFSA)

Ex 1: of a DFSA



Accepting states: q_0, q_1

Not accepting states: q_3, q_2

Example input: 110100

State	q_0	q_1	q_3	q_0	q_1	q_2	q_2
Input symbol		1	1	0	1	0	0

The string is rejected since q_2 is not an accepting state.

Definition:

A DFSA M is a 5-tuple $M = (Q, \Sigma, \delta, s, F)$, where

- Q is a finite set of states
- Σ is the input alphabet
- δ is the transition function $\longrightarrow$ $\delta : \underbrace{Q \times \Sigma}_{\text{ordered pair}} \longrightarrow Q$ e.g. $\delta(q, c) = q'$
- $s \in Q$ is the initial state
- $F \subseteq Q$ is the set of accepting state.

Extended Transition Function

$\delta^* : Q \times \Sigma^* \rightarrow Q$

$\delta^*(q, x) = q'$ means if we start at q and read x (a string), then we end at q'.

$$\delta^*(q, x) = \begin{cases} q & \text{if } x = \varepsilon \\ \delta(\delta^*(q, y), a) & \text{if } x = ya, \text{ where } y \in \Sigma^*, a \in \Sigma \end{cases}$$

Definition:

We say a DFSA $M = (Q, \Sigma, \delta, s, F)$ accepts a string $x \in \Sigma^*$ iff $\delta^*(s, x) \in F$ (M accepts F)

The language of a DFSA M is

$$\mathcal{L}(M) = \{ x \in \Sigma^* : M \text{ accepts } x \}$$

Ex 2: What is $\mathcal{L}(M)$ from Ex 1?

$$\mathcal{L}(M) = \{ x \in \Sigma^* : x \text{ has an odd \# of 1s iff } x \text{ ends with } 1 \}$$

How to prove Ex 2?

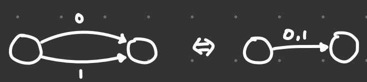
We use a state invariant

$$\delta^*(s, x) = \begin{cases} q_0 & \text{, if } \_\_\_\_ \\ q_1 & \text{, if } \_\_\_\_ \\ \vdots \end{cases} \qquad \text{(SI)}$$

Ex 3: Find SI of Ex 2

$$\delta^*(q_0, x) = \begin{cases} q_0 & \text{, if has an even \# of 1s and } x \text{ doesn't end with } 1 \\ q_1 & \text{, if has an odd \# of 1s and } x \text{ ends with } 1 \\ q_2 & \text{, if has an odd \# of 1s and } x \text{ doesn't end with } 1 \\ q_3 & \text{, if has an even \# of 1s and } x \text{ ends with } 1 \end{cases}$$

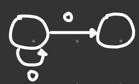
Conventions

Combining transitions 

Omitting dead states 

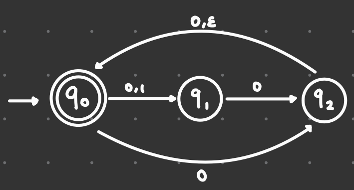
Nondeterministic Finite State Automata (NFSA)

A DFSA with 2 features.

(i) Multiple choices when reading a symbol 

(ii) Epsilon transitions. Change state without reading any input. 

Ex 4: of a NFSA



Input: 00

Trace 1: $q_0 \xrightarrow{0} q_1 \xrightarrow{0} q_2$ reject

Trace 2: $q_0 \xrightarrow{0} q_1 \xrightarrow{0} q_2 \xrightarrow{\varepsilon} q_0$ accept

Trace 3: $q_0 \xrightarrow{0} q_2 \xrightarrow{0} q_0$ accept

Trace 4: $q_0 \xrightarrow{0} q_2 \xrightarrow{\varepsilon} q_0 \xrightarrow{0} q_1$ reject

The traces are called computations.

Definition:

An NFSA M accepts an input x iff there's an accepting computation of M on x.
iff $\delta^*(q_0, x) \cap F \neq \varnothing$

The language of an NFSA M is

$$\mathcal{L}(M) = \{ x \in \Sigma^* : M \text{ accepts } x \}$$

Ex 5: Find $\mathcal{L}(M)$ of Ex 4.

$$\mathcal{L}(M) = \mathcal{L}\left(\left((\varepsilon + 0 + 1)\,0\,(0 + \varepsilon)\right)^*\right)$$

Definition:

An NFSA M is a 5-tuple $M = (Q, \Sigma, \delta, s, F)$, where

- ↳ Q, Σ, s, F are as in a DFSA.
- ↳ δ is the transition function.

subset of Q

$$\delta : Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow P(Q)$$

e.g. in Ex 4, $\delta(q_0, 0) = \{q_1, q_2\}$

Extended Transition Function

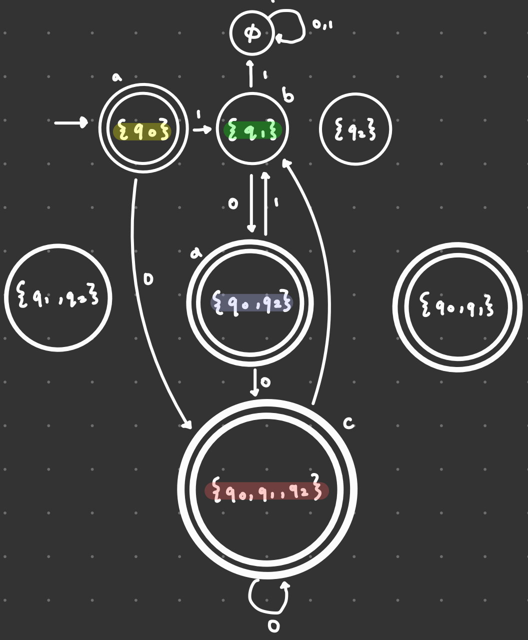
$$\delta^* : Q \times \Sigma^* \rightarrow P(Q)$$

$\delta^*(q, x)$ is the set of states that are readable starting at a state q and reading all of x.

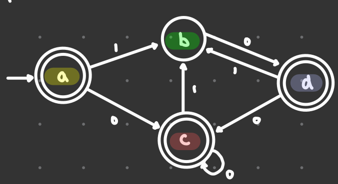
Ex 6: δ^* of Ex 4.

$\delta^*(q_0, 0) = \{q_0, q_1, q_2\}$
$\delta^*(q_0, 1) = \{q_1\}$
$\delta^*(q_0, 11) = \phi$

Ex 7: Find an equivalent DFSA M' (Subset construction)



Simplified DFSA:



Theorem: The BIG Result

Let L be a language. Then the following are equivalent.

(i) $L = \mathcal{L}(R)$ for some regex R

(ii) $L = \mathcal{L}(M)$ for some DFSA M

(iii) $L = \mathcal{L}(M)$ for some NFSA M

B36 Oct 20 Lec 1 Notes

```
Week 6 (October 20):
  * For convenience, we'll type:
    - 'e' for the empty string symbol (\epsilon),
    - 'E' for the empty set symbol (\emptyset).

    Let Sigma = {0,1}.
    For each n in N, we define the language L_n as follows.
     L_n = {x in Sigma*: x has a palindromic substring of length n or more}.

    a) For each n in N, is L_n finite?  Justify your answer.
       This should be relatively easy.
    b) For each n in N, is L_n cofinite?  I.e., is L_n's complement finite?
       Justify your answer.
    c) For each n in {0,1,2,3,4,5}, find a regex that denotes L_n.
       Explain why you regexes are correct.
    d) For each n in {0,1,2,3,4,5}, find a regex that denotes L_n's complement. Explain why your regexes are
correct.
```

(a) Infinite

(b) finite

(c) $n=0$: Σ^* $\qquad L_0 =$

$n=1$: $(0+1)(0+1)^*$

$n=2$: $(0+1)^2$

$L_0 = \{ All \} \Rightarrow L_0^c = \{\}$

$L_1 = \{ All - \varepsilon \} \Rightarrow L_1 = (1+0)(1+0)^*$

$L_1^c = \{\varepsilon\}$

$L_2 = \{ All - \varepsilon - 1, 0, \varepsilon, 01, 10 \} = \{ (1+0)^3 (1+0)^*$

$L_2^c = \{ 1, 0, \varepsilon, 01, 10 \}$

$\underbrace{010}_{3}1 \qquad (1+0)^3(1+0)^* + 11 + 00$

$L_3^c = \{ 1, 0, \varepsilon, 01, 10, 11, 00, 011, 110, 001, 100$

$L_3 = \{ All - 1, 0, \varepsilon, 01, 10, 11, 00, 011, 110, 001, 100 \}$