Today: NFA → G[N]FA → regex.

Recall: REX_Σ = set of all regexes over alphabet Σ.

Def: A GNFA is a tuple $\langle Q, \Sigma, \delta, q_0, F \rangle$ where

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

(Q, Σ, q_0, F are as usual)

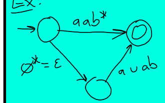
Def: Let $G := \langle Q, \Sigma, \delta, q_0, F \rangle$ be a GFA, and let $w \in \Sigma^*$ be an (input) string.

A comp path of G on input w is a sequence of states $s_0, s_1, \ldots, s_k$ (some $k \geq 0$) such that there exist strings $w_1, \ldots, w_k \in \Sigma^*$ such that:

- $s_0 = q_0$
- $w = w_1 \cdots w_k$ ← must read the entire string
- $\forall i,\ 1 \leq i \leq k,$

$$w_i \in \delta(s_{i-1}, s_i)$$

Ex:

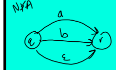


Def: G as above, w as above. G accepts w if there exists a comp path $s_0, \ldots, s_k$ on input w where $s_k \in F$

NFA to GNFA:

The GNFA's transition diagram looks the same except for edge consolidation:

NFA



means

$r \in \delta(q, a)$

$r \in \delta(q, b)$

$r \in \delta(q, \varepsilon)$

GNFA:

$q \xrightarrow{a \cup b \cup \varepsilon} r$

means

$a \cup b \cup \varepsilon = \delta(q, r)$

Recall: $q \xrightarrow{R} r$

means $R = \delta(q, r)$

$$\underbrace{N}_{\text{clean NFA}} \to \underbrace{G_0 \to G_1 \to \cdots \to G_n}_{\text{equiv GNFAs}} \to \underbrace{R}_{\text{regex}}$$

Input N:

// Assume N clean

1. Let G_0 be the GNFA obtained from N as described above.

2. $i := 1$

   while G_{i-1} has an $\underbrace{\text{intermediate state}}_{\text{not the start state, not the accept state}}$, do

   a) choose an intermediate state q of G_{i-1}

   b) bypass q

   c) remove q, resulting in G_i

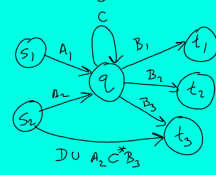
   d) i++

end-while

// G_i has no intermediate
// states, only start state q_0
// and one accept state q_f

Return $\underbrace{\delta(q_0, q_f)}_{\text{in } G_i = G_n}$

Bypass q (how-to)

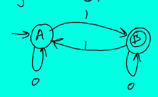


$\delta(s_2, t_3) = D$ before bypass,
$\delta(s_2, t_3) = D \cup A_2 C^* B_3$ after the bypass

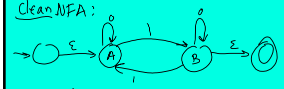
Do this for every combination of s_i & t_j

Example: $\Sigma = \{0, 1\}$,
$L = \{w \in \Sigma^* : w$ has an odd # of 1's$\}$

rec by this DFA:

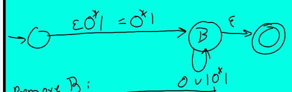


Clean NFA:

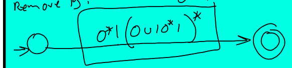


same diagram for G_0

Remove A (chosen arbitrarily)



Remove B:



Thm: A language is regular if and only if there is a regex denoting it.

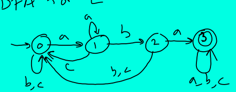
Proof: By previous constructions. ∎

Ex: $\Sigma = \{a, b, c\}$
$L = \{w \in \Sigma^* : w$ has aba as a substring$\}$

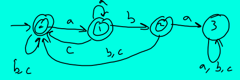
regex:
$(a \cup b \cup c)^* aba (a \cup b \cup c)^*$

Regex for $\overline{L}$?

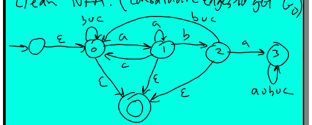
DFA for L:



DFA for $\overline{L}$:

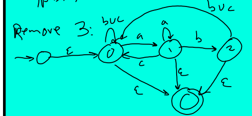


Clean NFA: (consolidate edges to get G_0)

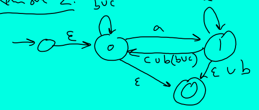


Strategy: choose state with fewest bypasses needed to remove next.

Remove 3:



Remove 2:



Finish next time