NFA $\xrightarrow[\text{GNFAs}]{}$ regex

**Recall:** A GNFA (GFA) is a tuple $\langle Q, \Sigma, \delta, q_0, F \rangle$ where $Q, \Sigma, q_0, F$ are as with an NFA and

$\delta : Q \times Q \to REX_\Sigma$ (set of all regexes over $\Sigma$)

$q \xrightarrow{s} r$ $\quad SEREX_\Sigma$

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

$q \xrightarrow{\varepsilon} r$

means $q \xrightarrow{\varepsilon^*} r$

$q$ (no arrow) $r$

means $\delta(q, r) = \emptyset$.

**Def** (Semantics of GNFAs):
Let $G := \langle Q, \Sigma, \delta, q_0, F \rangle$ be a GNFA. A computation of $G$ on input $w \in \Sigma^*$ is a sequence of states $s_0, s_1, \ldots, s_n \in Q$ (for $n \geq 0$) such that there exists strings $w_1, \ldots, w_n \in \Sigma^*$ such that

1) $w = w_1 \cdots w_n$
2) $s_0 = q_0$
3) For all $i$, $1 \leq i \leq n$, $w_i \in \delta(s_{i-1}, s_i)$

Ex:

$s_{i-1} \xrightarrow{R} s_i$

$R$ contains $w_i$
$R$ matches $w_i$
$w_i$ matches $R$

$s_n$ is the end state of this computation.
$G$ accepts $w$ if there exists a computation of $G$ on input $w$ that ends in an accepting state of $G$.

$L(G)$ defined just as before.

**Plan:** Given an NFA $A$:

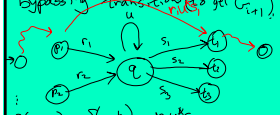
$A \mapsto G_0$ :
leave edge labels as is, except $\emptyset, q_0$


clean

$A \mapsto G_0 \mapsto G_1 \mapsto G_2 \mapsto \cdots \mapsto G_k$ } remove a state at each step → 2 states: start state & accepting state

How a single step works:

Let $q$ be a state of $G_i$ $(i < k)$. Remove $q$ and add bypassing transitions (to get $G_{i+1}$).




$\delta(p_i, t_j) := \delta(p_i, t_j) \cup r_i u^* s_j$

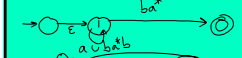
At end:


$L(G_k) = L(A) = R$

Ex:


clean NFA for even # of b's ($\Sigma = \{a, b\}$)

Remove (2):




Remove (1):