Theorem: For every finite automaton (DFA, NFA, GNFA...) $M$, we can compute a regular expression $R$ such that $L(R) = L(M)$.

Key Idea:

We can eliminate any nonaccepting state $q$ different from the start state by bypassing every incoming edge $(p, \beta, q)$ to every outgoing edge $(q, \gamma, r)$, thereby updating the entry $(p, r)$.

Eliminate $q$:

For each incoming $(p, \beta, q)$ { delete $q$;  
  For each outgoing $(q, \gamma, r)$ {  
    $T(p, r) = \beta \gamma^* \eta$  
  }  
  Update: $\alpha' = \alpha + \beta \gamma^* \eta$  
}

Since $q \notin F$, any processing that comes in to $q$ must come out at $r$ or $r'$...

Hence we can provide the same processing capability directly. Then delete edge $(q, r)$.
Example: Answer to HW2 1(g). First DFA \( M \): 
\[ L = \{ x \in \{ a, b \}^* : |x| \text{ is even} \land \#a(x) \text{ is odd} \} \]

Even \(|w|\):

Odd \(|w|\):

Eliminate 4:

\[
\begin{array}{c|c|c|c}
\text{Incoming state} & \text{Regex Char} & \text{Self} & \text{Outgoing state} \\
\hline
1 & \alpha & \varepsilon & 2 \\
2 & \beta & \varepsilon & 1 \\
\end{array}
\]

Eliminate 3:

Outgoing \(|w|\):

Only one acc. state 9 (\( \neq 5 \))
Hence no need to add \( \varepsilon \) arcs.

Eliminate 4
Eliminate 3
\[
R = L(M) = L_{12}.
\]

Since we will use \( \varepsilon^* \), and
\( \emptyset^* = \varepsilon^* = \varepsilon \)
It doesn’t matter if an empty “self” is called \( \emptyset \) or \( \varepsilon \).

Update \( T(1,1), T(1,2), T(2,1) \) and \( T(2,2) \).

Old \( T(1,1) = \varepsilon \) New \( T(1,1) = \varepsilon^* \cdot \varepsilon \cdot \alpha = \varepsilon^* a \)

Old \( T(1,2) = \emptyset \) New: \( T(1,2) = \varepsilon + \alpha \cdot \beta = ab \)

Old \( T(2,1) = \emptyset \) New: \( \beta \cdot \varepsilon \cdot \alpha = ba \)

Old \( T(2,2) = \varepsilon \) New: \( T(2,2) = \varepsilon + \beta \cdot \varepsilon \cdot b = \varepsilon^* + bb \)

Eliminate 3: Outgoing \(|w|\):

\[
T(1,1) = T(1,3) T(3,3)^* T(3,1) \\
T(1,2) = T(1,3) T(3,3)^* T(3,2) \\
T(2,1) = T(2,3) T(3,3)^* T(3,1) \\
T(2,2) = T(2,3) T(3,3)^* T(3,2)
\]

\[
\begin{align*}
\text{new } T(1,1) &= \alpha + bb = \varepsilon + a + bb \\
\text{new } T(1,2) &= ab + ba
\end{align*}
\]
From the Two-Stage Base Case

\[ L(M) = L_{12}^* = (\alpha + \beta \gamma \eta \eta^* \beta^* \gamma^*)^{*} \]

All laps until last time through 0

\[ \text{New } T(1,2) = \frac{T(1,1) + T(1,2) T(2,2)^* T(2,1)}{T(2,1) T(2,2)} \]

\[ T(2,2)^* = (\varepsilon + \alpha + \beta + b b)^* \]

\[ (\alpha + b b)^* \]

\[ \left[ \varepsilon + \alpha + b b + (a b + b a)(\alpha + b b)^* (a b + b a) \right]^{*} \]

There is no obligation to simplify these monstrosities

\[ = (\alpha + b b + (a b + b a)(\alpha + b b)^* (a b + b a))^{*} \]

\[ (a b + b a)(\alpha + b b)^* \]

\[ = L_{11} U L_{12} \]

\[ \text{ON HW, you may stop here} \]

\[ \text{Sketch Example 2} \]

After Elim 3 & 4

\[ L(\Lambda) = \text{final } L_{11} = \]

\[ d^* = \left( T(1,1) + T(1,2) T(2,2)^* T(2,1) \right)^* \]

Answer = \( (d^*)^{*} \)
§1.4: Now we say things that only work for DFAs.

Key Idea: Suppose we have a DFA $M$ that processes $x$ from $s$ to some state $p$, and processes $y+x$ from $s$ to some state $q$. (When) can $p = q$?

![Diagram of DFA M]

Suppose there is some string $z \in \Sigma^*$ such that $x \in L(M)$ but $y \in L(M)$ or vice versa.

Then $p \neq q$.

I.e. Suppose $L$ is a language and strings $x$ and $y$ ($x \neq y$) are such that for some $z \in \Sigma^*$:

$$\left( x \in L \land y \notin L \right) \lor \left( x \notin L \land y \in L \right)$$

Then any DFA $M$ such that $L(M) = L$ (if $M$ exists at all) must process $x$ and $y$ to different states $p$ and $q$.

I.e. If we have a set $S$ of $K$ strings such that for all distinct $x \in S$ there is a $z$ s.t. $L(xz) \neq L(zy)$, then any DFA $M$ such that $L(M) = L$ must process them all to different states. i.e. $M$ must have at least $K$ states. Thus: $K \geq \infty$.
Extra: One More Example.

Same arcs as PS2 (9) answer from lecture, but different accepting states.

Since there is only one accepting state different from start, we can get away with avoiding the texts step of adding a new final state with ε-arcs from old ones. Eliminating 2 & 3 will be enough.

Elim(3): Inc. Self Out
1 \ a \ ε
2 \ b \ ε
3 ++aa
4 \ a \ ε + bb

T(1,1)_{new} = T(1,1)_{old} + a \ ε \ a = ε + aa
T(1,2)_{new} = T(1,2)_{old} + a \ ε \ b = \emptyset + ab = ab
T(2,1)_{new} = T(2,1)_{old} + b \ ε \ a = \emptyset + ba = ba
T(2,2)_{new} = ε + bb

Why not ε + ba? Because 1 \rightarrow 2 is not a self-loop.

Note: Self* = (ε + bb)* = (bb)*

That's why OK to omit "ε*" in self loops

T(1,1)_{new} = T(1,1)_{old} + T(1,2) T(2,2) T(2,1)
= \emptyset + aa + ab bb \* ba

T(1,4)_{new} = T(1,4)_{old} + T(1,2) T(2,4)
= b + ab (bb)* a

T(4,4)_{new} = T(4,1)_{old} + T(4,2) T(2,4)
= \emptyset + a (bb)* ba

L(M') = L_{11} U L_{14} = \epsilon + T(1,1) T(4,4)

Written out in full, the answer is sort long. The joke is on us, because in fact \epsilon \cdot L_{11} \subseteq L_{11}: \epsilon \text{ is even } = (bb)*.