Regular Expressions & Automata
Regular Language Equivalence Thm

Languages captured by DFAs, NFAs, and regular expressions are the same.

Proof:
If \( r \) is a regular expression with \( L(r) = L \), then there is an NFA \( N_r \) that recognizes \( L \).
Prove by induction:

If $r$ is a regular expression denoting language $L$, then there is an NFA with one accepting state that recognizes $L$

Requiring one accepting state is easy, and it simplifies the proof.

Proof is by induction. But on what??
Induction on # of operations

Base case: If $r$ contains none of $+$, concat, $*$, then $r$ is one of

- $\emptyset$: recognizes the empty set
- $\varepsilon$: recognizes the set $\{\varepsilon\}$
- $a$: recognizes the set $\{a\}$
**Inductive Step**

• Assume the theorem holds for all regular expressions $r$ formed using less than $n$ operations.
• Let $r$ be an arbitrary regular expression formed using exactly $n$ operations.
• Then $r$ is one of the following:

$$ r = r_1 + r_2 \quad \text{or} \quad r = r_1 r_2 \quad \text{or} \quad r = r_1^* $$

Where $r_1$ and $r_2$ are each formed using $< n$ operations and inductively have NFAs with one accepting state.
Case $r = r_1 + r_2$

Assume that the NFAs for $r_1$ and $r_2$ have initial states $q_1$ and $q_2$ and accept states $f_1$ and $f_2$.
Case $r = r_1 r_2$

Assume that the NFAs for $r_1$ and $r_2$ have initial states $q_1$ and $q_2$ and accept states $f_1$ and $f_2$. 
Case $r = r_1^*$

Assume that the NFA for $r_1$ has initial state $q_1$ and accept state $f_1$.
Case $r = (r_1)^*$

Assume that the NFA for $r_1$ has initial state $q_1$ and accept state $f_1$. 

![Diagram of NFA]

- Initial state $q_0$ transitions to state $q_1$ on $\varepsilon$.
- State $q_1$ is part of an NFA $N_1$.
- $N_1$ transitions to accept state $f_1$ on $\varepsilon$.
Example construction

\[(\varepsilon+0)(1+10)^*\]
$\varepsilon 0$ $\varepsilon 0$ $\varepsilon 0$ $\varepsilon 0$

$(1+10)$ $10$ $1$ $10$ $1$
Oh, just draw it already